Estimation non paramétrique

2023-02-26

0. Chargement des données

Les données contiennent 2 variables et 5.000 observations sans valeurs manquantes.

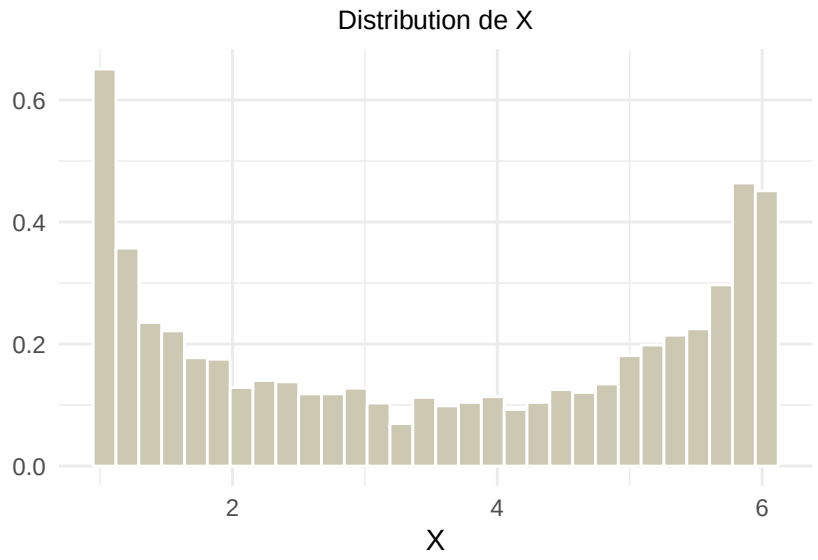
```
##        X                Y         
##  Min.   :1.000   Min.   :-1.656  
##  1st Qu.:1.622   1st Qu.: 3.695  
##  Median :3.573   Median : 4.484  
##  Mean   :3.538   Mean   : 4.926  
##  3rd Qu.:5.424   3rd Qu.: 6.241  
##  Max.   :6.000   Max.   :12.348
```

1. Estimation de la densité

1.1. Visualisation de densité

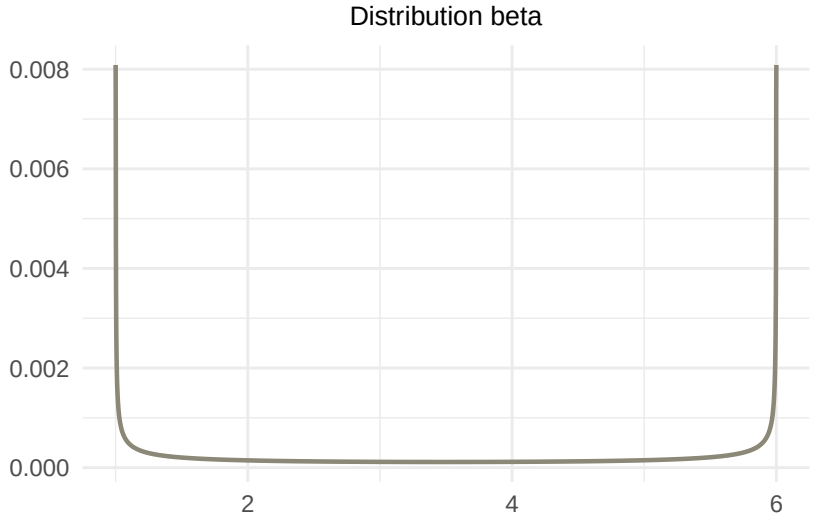
Nous pouvons approcher la densité g par une histogramme. L'histogramme approche facilement une fonction de densité en considérant constante la densité dans chacune des intervalles. Donc avec un bon nombre d'intervalles, elle peut décrire sans difficulté les différentes fonctions de densité ayant de formes variées malgré qu'elle reste discontinue.

Nous voyons que l'histogramme présente une forme convexe assez symétrique où nous observons de plus en plus d'observations vers les extrêmités et moins au centre.

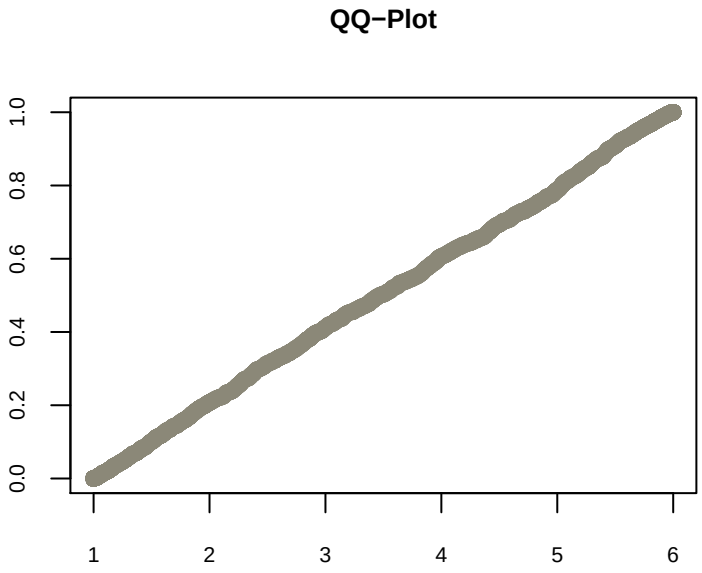


Une distribution beta peut avoir une telle densité. La dsitribution beta est une loi de probabilité continue avec deux paramètres α et β. Admettant une diversité de formes, elle permet de modéliser de nombresuses distributions.

Le graphique ci-dessous montre la densité beta obtenue avec nos données. Avec une forme qui ressemble beaucoup à l'histogramme précédante, nous pouvons supposer que la densité des X_i peut correspondre à une loi beta.



Nous pouvons vérifier cela avec un QQ-plot. Graphiquement, la comparaison entre la distribution observée et la loi beta montre que les points sont bien alignéees suivant la première bissectrice. Nous pouvons penser que la distribution suit bien la loi beta.



1.2. Construction d'estimateur à noyau

L'estimateur à noyaux généralise les caractéristiques de l'histogramme et permet d'estimer la densité à partir des données en tant qu'un estimateur continu. Son expression mathématique est donnée par :

$$\hat{g}_{n,h}(x) = \frac{1}{Nh} \sum_{i=1}^{N} K(\frac{x-x_i}{h})$$

$$\hat{g}_{n,h}(x) = \frac{1}{N} \sum_{i=1}^{N} K_h(x-x_i)$$

où K correspond au noyau et h à la fenêtre.

Le noyau est une fonction de poids qui sert comme une mesure de moyenne pondérée des observations autour de point X que nous voudrons étudier.

Quant à la fenêtre h, elle permet de choisir jusqu'où, à partir de notre point évalué, seront les observations pertinentes. Autrement dit, les observations qui se trouvent en dehors de la fenêtre ne sont pas comptées car elles reçoivent un poids null.

Dans l'estimateur à noyaux, le noyau et la fenêtre sont les deux éléments importants que nous devons chercher à bien choisir.

Choix de noyau

Il existe différents noyaux :

- Le noyau **uniform** donne même poids à toutes les observations.
- Le noyau **triangular** donne plus d'importance aux observations qui sont plus proches au point que nous évaluons et moins aux observations qui sont éloignées.
- Le noyau **epanechnikov** suit le même principe que le noyau triangular mais l'importance donnée aux observations diminue marginalement avec l'éloignement des observations du point étudié et pas linéairement comme le noyau triangular
- Le noyau **guassien** et autres noyaux en forme de cloche accordent des poids différents selon la distance d'observations par rapport au point étudié. L'importance diminue de façon dramatique avec la distance.

Malgré tout, ils accomplissent tous la même mission : représenter une façon d'aggréger les observations autour du point cible pour estimer sa densité. La différence repose sur le choix de comment incorporer ces informations en estimation : certains accordent une importance égale mais la plupart choisit de considérer les observations plus proches plus pertinents en leurs attribuant un poids plus important.

De ce fait, le choix de noyau affecte peu le résultat et les différents noyaux donnent de résultats presque similaires pour une fenêtre donnée.

Choix de fenêtre

Dans le choix de fenêtre, il y a un grand compromis biais - variance. Une fenêtre large permet à l'estimateur d'être plus résistant aux bruits et une petite fenêtre augmente sa sensibilité à la variance. Contrairement au noyau, le choix de fenêtre affecte beaucoup les résultats.

Pour notre étude, quatre estimateurs à noyau gaussien sont construits avec quatres fenêtres différentes: 0,1, 0,3, 0,5, et 1.

```r
# Création des estimateurs à noyau guassien avec différentes fenêtres

density_1 = density(X, bw = 0.1, kernel = "gaussian")
density_2 = density(X, bw = 0.3, kernel = "gaussian")
density_3 = density(X, bw = 0.5, kernel = "gaussian")
density_4 = density(X, bw = 1, kernel = "gaussian")
```

1.3. Représentation graphique

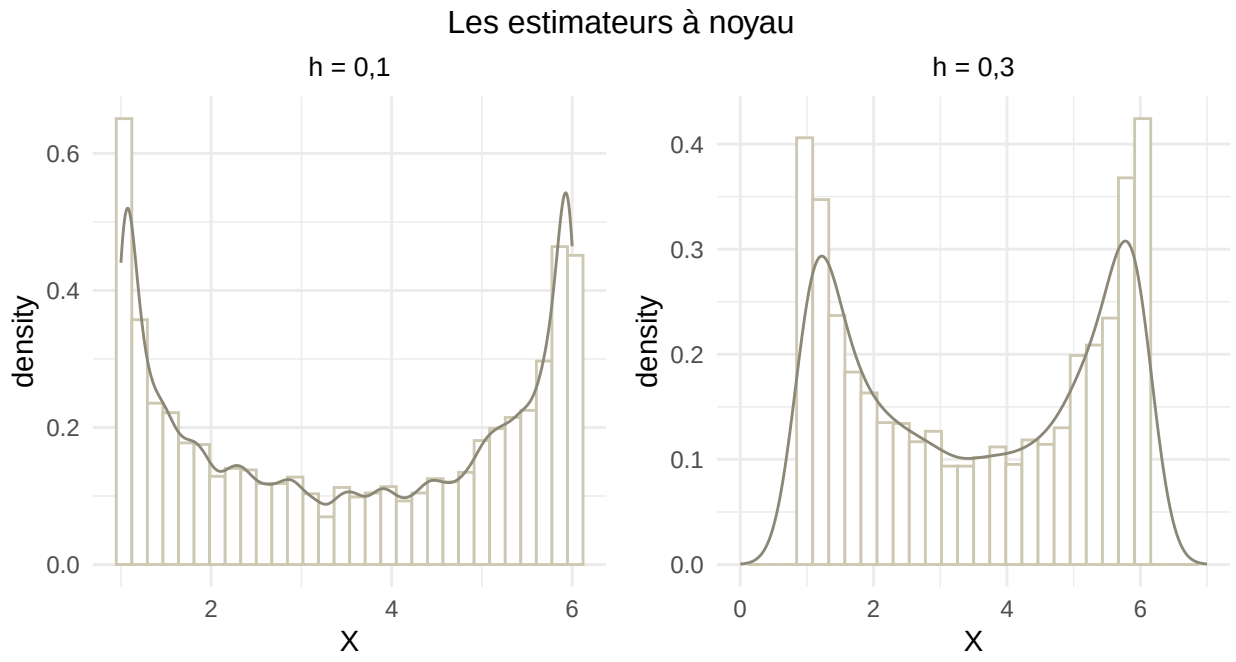
Lorsque la fenêtre h est petite, l'estimateur devient sensible à la variabilité et résistante aux bruits. Il y a un phénomène de surajustement.

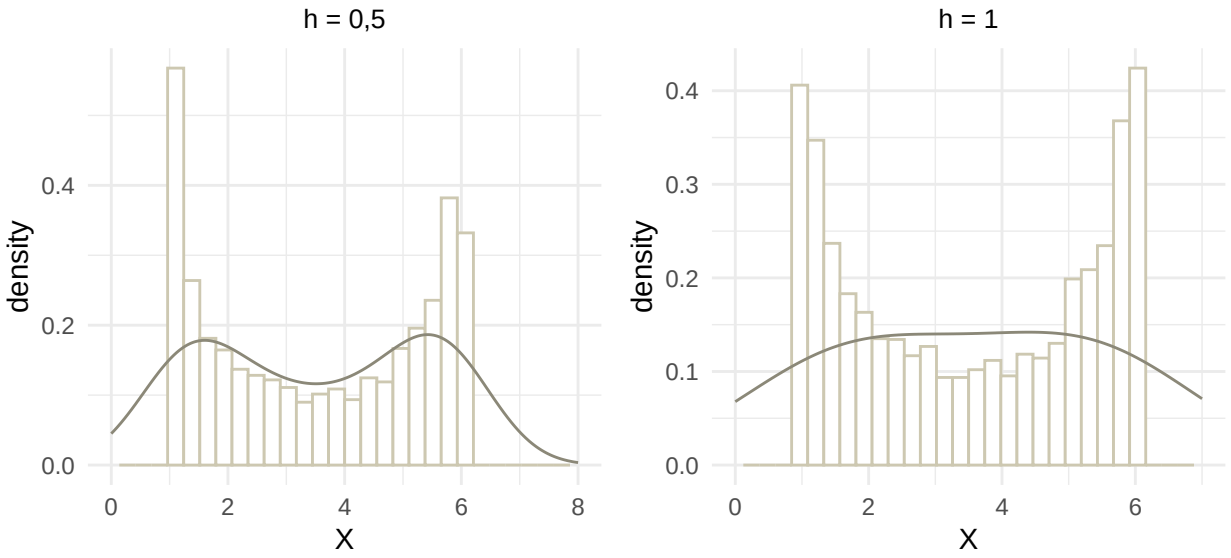
Lorsque la fenêtre h est grande, l'estimateur ne capte plus des variations importantes et présente un biais important.

Ce que nous voudrons avoir est une fenêtre suiffisamment petite pour pouvoir capturer les variations pertinentes mais pas trop petite pour que les bruits n'interviennent pas excessivement.

Parmi nos estimateurs,

- Pour un $h = 0,1$, nous observons que l'estimateur capte toutes les variabilités et surajuste aux observations.
- $h = 0,3$ semble la fenêtre optimale parmi les 4 estimateurs avec une variabilité et un biais corrects.
- Pour un $h = 0,5$ ou 1, les estimateurs deviennents très lisses perdant beaucoup d'information liée à la variabilité d'autant plus que h soit élevé. Ces deux estimateurs ont moins de biais.



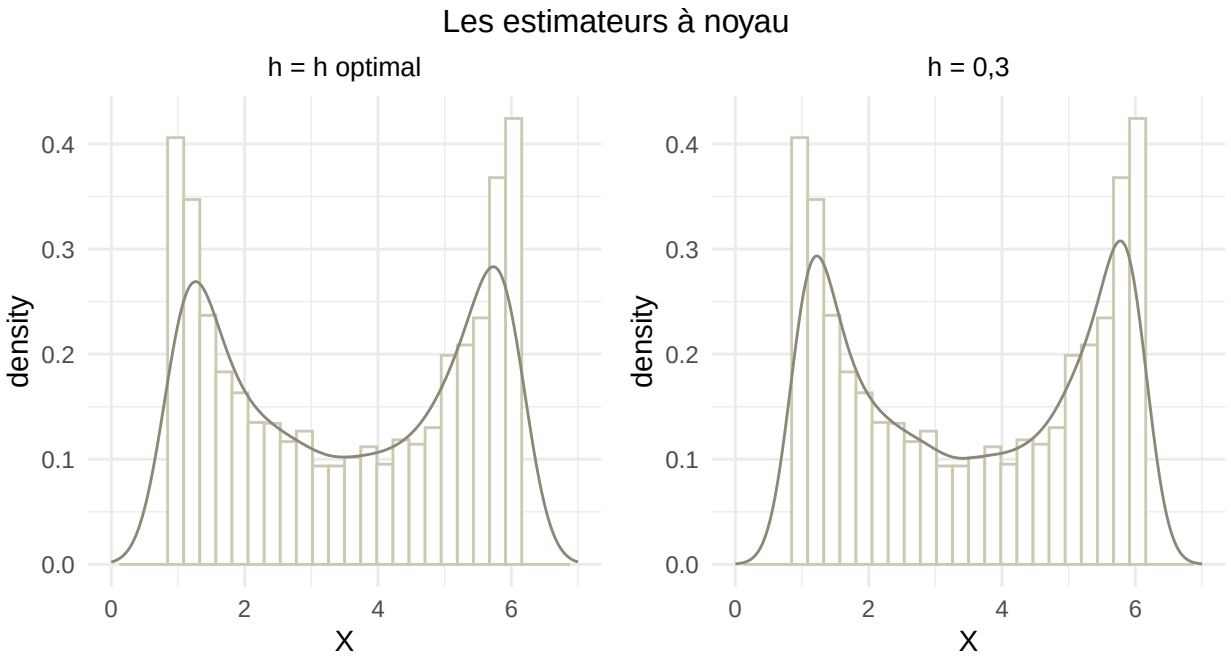


Nous pouvons comparer notre meilleur estimateur avec un estimateur construit avec un h optimal proposé par la fonction de selecteur de fenêtre $bw.nrd()$. h proposé par la fonction est 0,3559641. Très proche de notre h choisi, il donne une estimation semblable.

```
# Création de h
h = bw.nrd(X)
h
```

```
## [1] 0.3559641
```

```
# Ajustement du modèle avec h proposé
density_0 = density(X, bw = h, kernel = "gaussian")
```

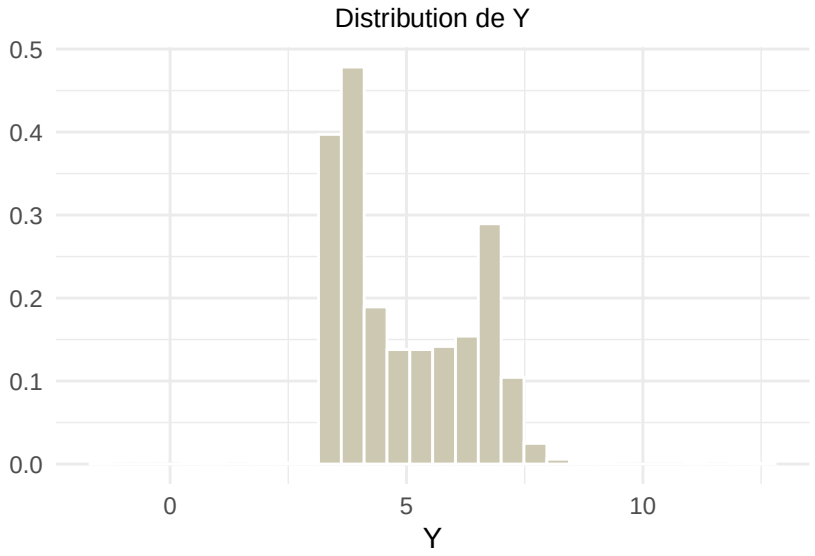


2. Estimation de la fonction de régression

2.1. Relation entre les X et Y

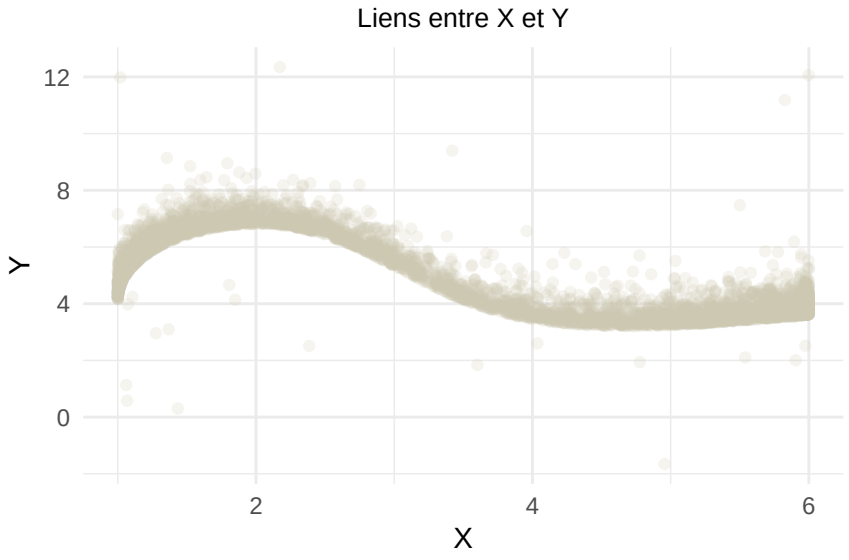
Distribution de Y

Nous observons que les Y_i sont concentrés entre 3 et 8,5 avec de faibles observations aux extrémités de -2,5 et de 13,5. La distribution est irrégulière avec de grandes intervalles vides d'observations dans le reste.



Lien X et Y

Nous pouvons étudier le lien entre les X_i et Y_i avec un nuage de points. Nous observons que la relation entre les deux variables présente une petite oscillation. Y_i augmente en fonction de X_i pour $X_i < 2$, puis prend une tendance décroissante jusqu'à $X_i < 4{,}5$, pour prendre un comportement constant, voire légèrement croissant, par la suite. Les observations sont plus importantes aux deux extrêmités de X_i qu'au centre.



2.2. Linéarité

Bien que nous puissions observer une forme du lien apparent entre les X_i et Y_i sur le nuage de points, le lien n'est visiblement pas linéaire.

En effet, les Y_i présentent un chagement de comportement selon les différentes intervalles de X_i. La régression linéaire ne pourra pas expliquer cette relation. Il n'est donc pas plausible de penser que la fonction r est linéaire.

Une régression non paramétrique serait plus pertinente par rapport à la régresson linéaire.

2.3. Construction d'estimateur non paramétrique

Nous pouvons donc utiliser l'estimateur Nadaraya-Watson pour décrire la relation entre les X_i et Y_i.

L'estimateur Nadaraya-Watson est une moyenne pondérée des Y_i dans une fenêtre donnée. C'est-à-dire, la somme pondérée de toutes les valeurs de Y_i, où leur poids sont noyaux, est divisée par la somme des noyaux.

En effet, dans l'objectif d'estimer une fonction de regression $r : \mathbb{R} \to \mathbb{R}$ de façon non paramétrique, nous pouvons écrire :

$$r(x) = E(Y|X = x)$$

$$r(x) = \int y f_{Y|X=x}(y)dy$$

$$r(x) = \frac{\int y f(x,y)dy}{f_X(x)}$$

En remplaçant f et f_X par leur estimateur à noyau,

$$\frac{\int y \hat{f}(x,y)dy}{\hat{f}_X(x)} = \frac{\int y \frac{1}{N} \sum_{i=1}^{N} K_{h1}(x-x_i)K_{h2}(y-y_i)dy}{\sum_{i=1}^{N} K_{h1}(x-x_i)}$$

$$\frac{\int y \hat{f}(x,y)dy}{\hat{f}_X(x)} = \frac{\frac{1}{N} \sum_{i=1}^{N} K_{h1}(x-x_i) \int y K_{h2}(y-y_i)dy}{\sum_{i=1}^{N} K_{h1}(x-x_i)}$$

$$\frac{\int y \hat{f}(x,y)dy}{\hat{f}_X(x)} = \frac{\frac{1}{N} \sum_{i=1}^{N} K_{h1}(x-x_i).y_i}{\frac{1}{N} \sum_{i=1}^{N} K_{h1}(x-x_i)}$$

$$\frac{\int y \hat{f}(x,y)dy}{\hat{f}_X(x)} = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{K_{h1}(x-x_i)}{\frac{1}{N} \sum_{i=1}^{N} K_{h1}(x-x_i)}\right).y_i$$

Nous obtenons l'estimateur Nadaraya-Watson :

$$\hat{r}_{n,h}(x) = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{K_h(x-x_i)}{\frac{1}{N} \sum_{j=1}^{N} K_h(x-x_j)}\right).y_i$$

$$\hat{r}_{n,h}(x) = \frac{1}{N} \sum_{i=1}^{N} W_{h,i}(x).y_i$$

où $W_{h,i}(x) = \frac{K_h(x-x_i)}{\frac{1}{N} \sum_{j=1}^{N} K_h(x-x_j)}$

En considérent $W_{h,i}(x)$ comme le poids, l'estimateur Nadaraya-Watson peut être vu comme la somme pondérée de toutes les variables dépendantes observées dans la fenêtre choisie.

Le poids est composé de valeurs d'une fonction de noyau. Comme présenté précedemment, il existe different noyau que nous pouvons choisir. En général, le noyau donne plus grand poids aux observations qui sont proches de la cible et plus petit poids aux observations qui sont éloignées. Le poids est null pour les observations qui se trouvent en dehors de la fenêtre. Ainsi, l'estimateur s'adapte bien aux différentes formes générées par les données.

Comme pour l'estimateur de la densité, le noyau attribue de poids différents aux observations selon leur distance par rapport au point étudié et la fenêtre arbitre le biais et la variance de l'estimateur.

Quatre estimateurs Nadaraya-Watson avec le noyau gaussien et quatre fenêtres différentes 0,1, 0,3, 0,7, et 1,5 sont construits.

```
# Création des estimateurs non paramétriques guassiens avec différentes fenêtres
reg1 = locpoly(X, Y, kernel ='normal', bandwidth = 0.1)
reg2 = locpoly(X, Y, kernel ='normal', bandwidth = 0.3)
reg3 = locpoly(X, Y, kernel ='normal', bandwidth = 0.7)
reg4 = locpoly(X, Y, kernel ='normal', bandwidth = 1.5)
```

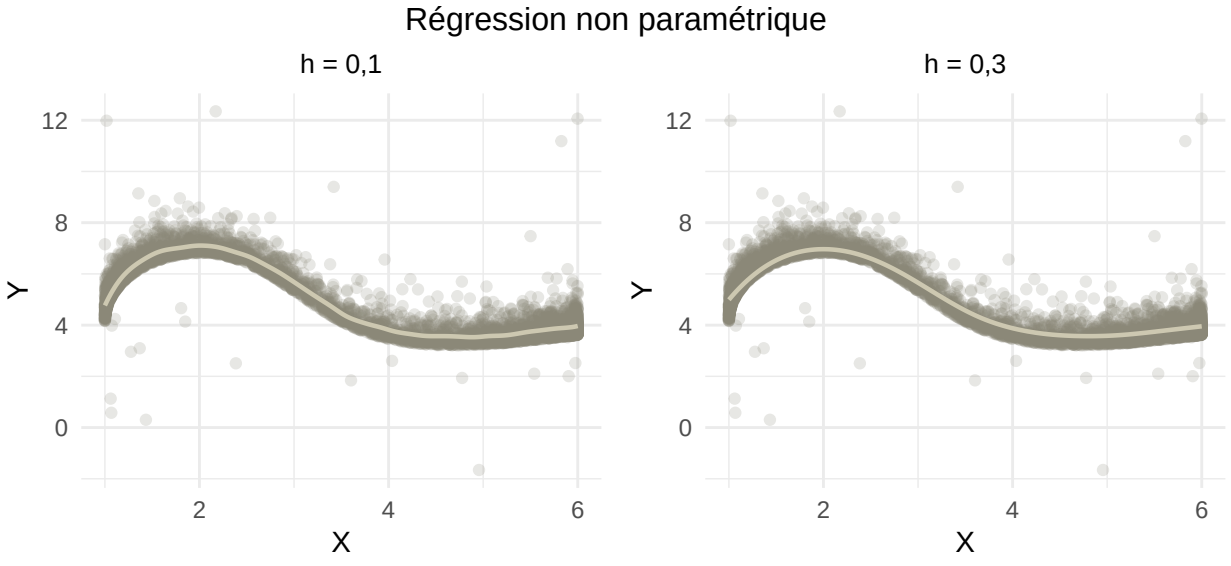
2.4. Représentation graphique

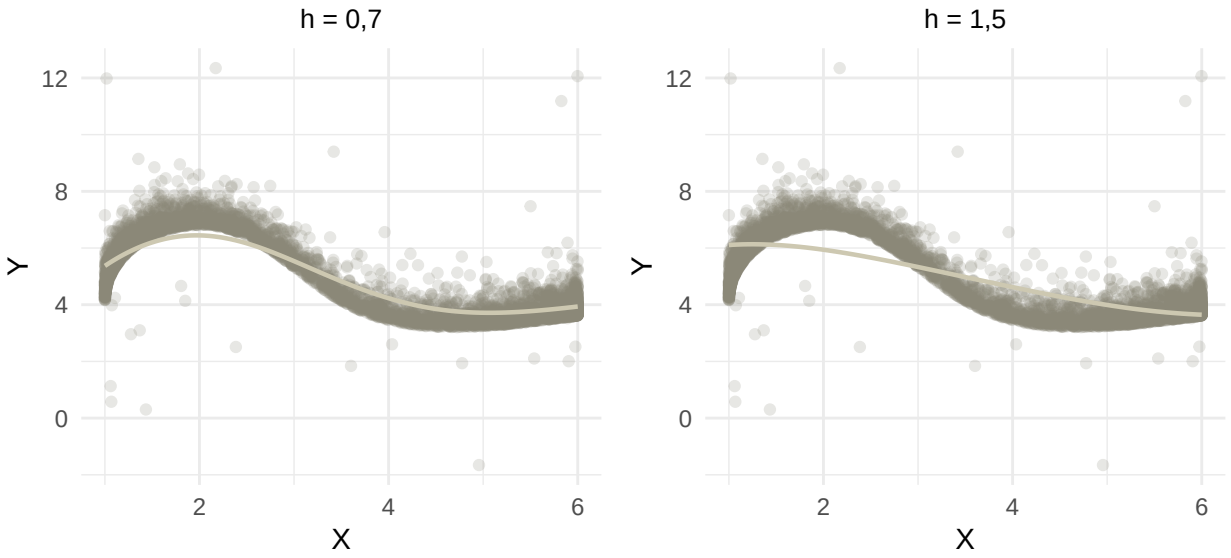
Comme nous avons vu dans le cas de l'estimateur à noyau, l'estimateur devient sensible à la variabilité et résistant aux bruits lorsque la fenêtre est petite. Lorsqu'elle est grande, l'estimateur ne capte plus de variations importantes et présente un biais important.

Comme dans l'estimateur de densité, nous voudrons avoir une fenêtre suiffisamment petite pour pouvoir capturer les variations pertinentes mais pas trop petite pour que les bruits n'interviennent pas excessivement.

Parmi nos estimateurs,

- Pour un $h = 0,1$, la fenêtre semble optimale pour l'estimateur avec une variabilité et un biais corrects.
- Quant aux estimateurs avec un h supérieur ou égal à $0,3$, les estimateurs commencent à perdre d'information liée à la variabilité, autant plus que h soit élevé. La relation devient presque linéaire pour un $h = 1,5$. En effet, lorsque h devient grand, toutes les observations sont prises en compte avec un poids constant comme en modèle linéaire.





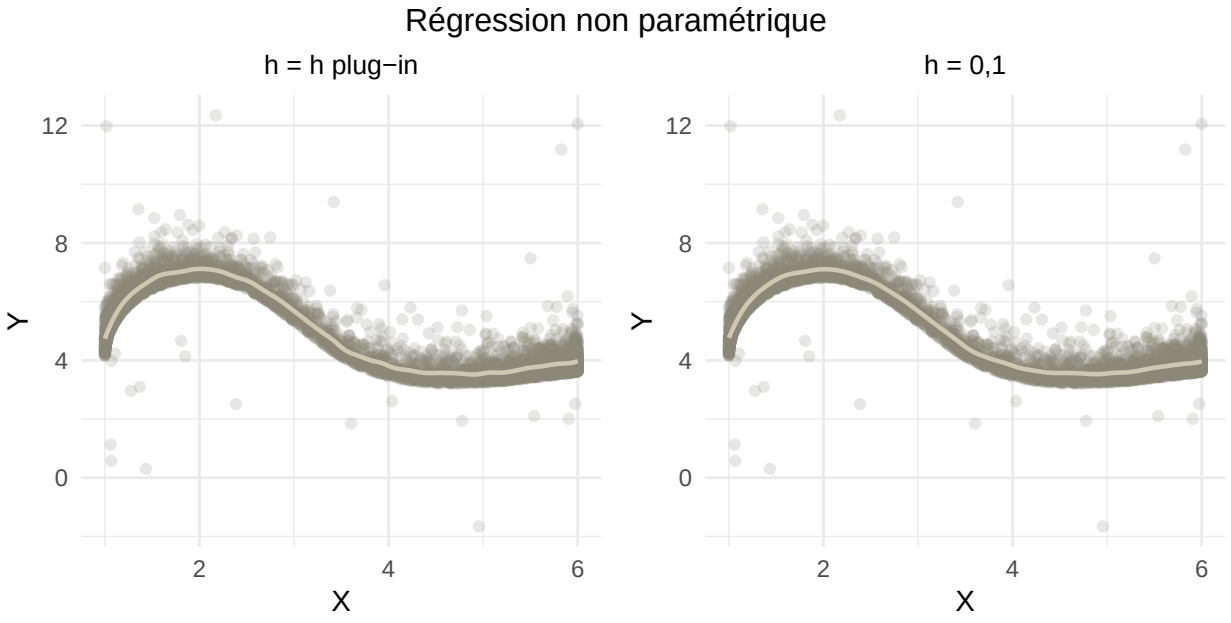
Nous pouvons également appliquer un plug-in pour construire un estimateur et le comparer avec notre estimateur choisi. h obtenu avec la méthode de plug-in est égal à 0,07736951.

```
# Création de h avec plug-in
h = dpill(X,Y)
h
```

```
## [1] 0.07736951
```

```
# Régression non paramétrique avec h créé
reg_0 = locpoly(X, Y, kernel ='normal', bandwidth = h)
```

Sachant que h de plug-in est plus petit que h de notre estimateur choisi, nous observons plus de variabilité sur l'estimateur plug-in. Il semble qu'une fenêtre inférieur à 0,1 est trop petite et engendre un effet de surajustement.



2.5. Nadaraya-Watson

$$\hat{r}_{n,h}(x) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{K_h(x-x_i)}{\frac{1}{N}\sum_{j=1}^{N}K_h(x-x_j)}\right).y_i$$

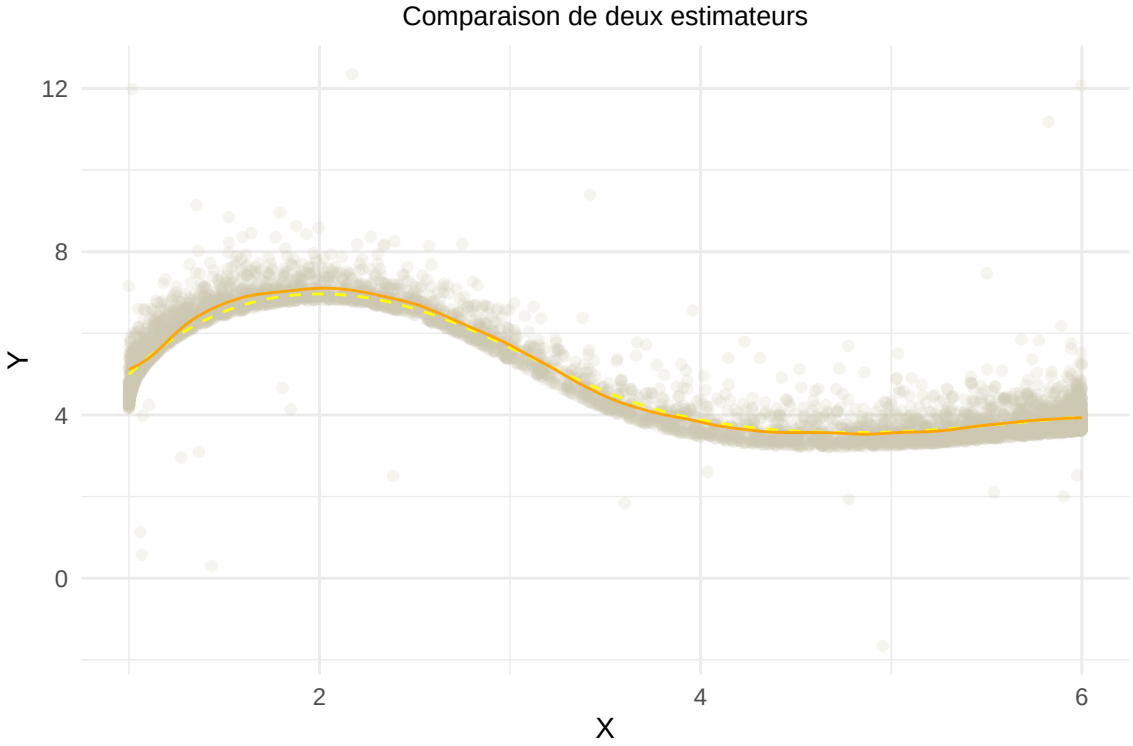
Nous pouvons coder l'estimateur Nadaraya-Watson comme ci-dessus :

```
# Fonction Nadaraya - Watson

NW = function (x, X, Y, h, K = dnorm()){
  # x - point d'intérêt ; X - variable explicative; Y - la variable dépendante;
  # h - fenêtre; K - noyau

  Kx = sapply(X, function(X) dnorm((x-X)/h)) # Calcul de noyau gaussien

  W = Kx / rowSums(Kx) # Calcul de poids en divisant le noyau par la somme des noyaux

  return(W%*%Y) # Poids*Y
}
```

Sur le graphique ci-dessous,

- la ligne jaune en pointillé correspond à notre estimation obtenue précedemment
- la ligne orange correspond à notre estimation codée

Nous observons les deux estimateurs se superposent et sont presque identiques.



2.6. Comparaison

Le résultat de la régression linéaire est semblable à notre estimateur non paramétrique avec une grande fenêtre, c'est-à-dire un modèle non paramétrique sans variabilité.

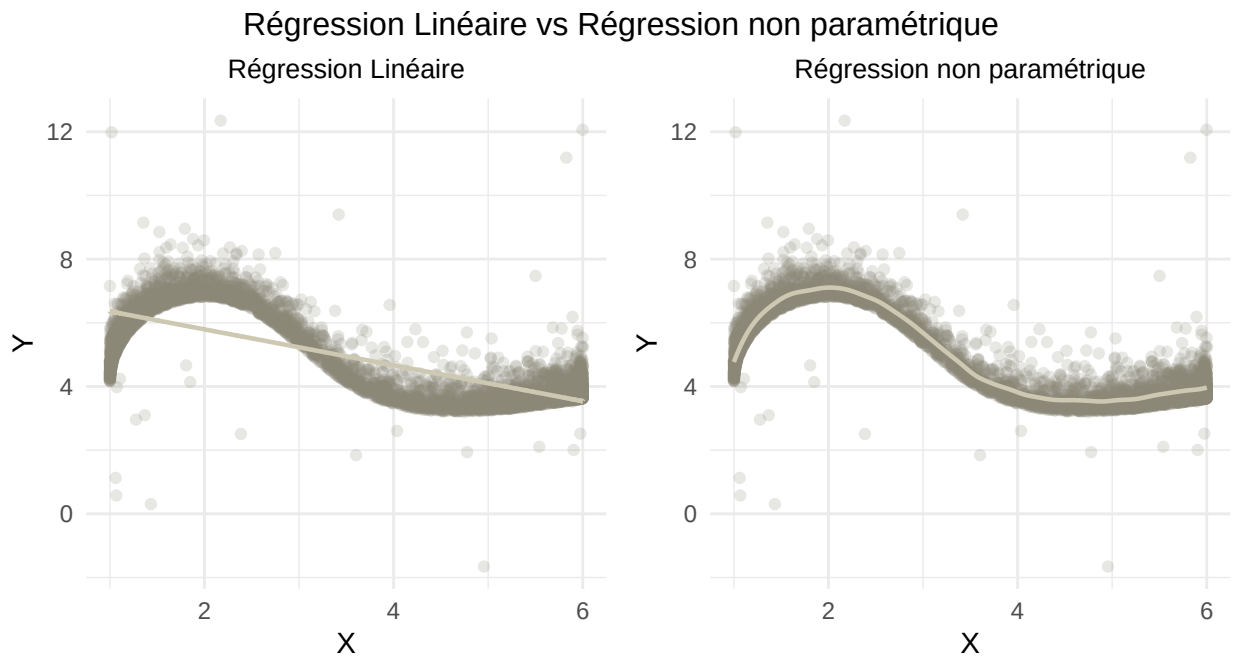
En effet, il y a un compromis biais - variance dans le choix du modèle paramétrique et non-paramétrique. Parce que les résultats statistiques reposent sur l'information contenue dans les données et dans les hypothèses d'à priori, le choix entre les modèles paramétriques et non paramétriques dépendent de quelles hypothèes que nous voulons prendre.

Le modèle paramétrique suppose plus d'hypothèses et ces hypothèses sont plus fortes que le modèle non paramétrique. Ces hypothèses prises sont retournées dans les résultats. Si les hypothèses sont correctes, l'utilisation du modèle paramétrique est bénéfique, plus efficace et précis que le modèle non paramétrique. Si nous estimons qu'il y a une grande probabilité que le modèle paramétrique correspond aux données, le modèle paramétrique est à privilégier. Il donne une bonne information aux résultats donc une estimation précise.

En passant au modèle non parmétrique, l'estimateur devient plus flexible pour s'adapter aux données. Le biais est réduit mais la variance augmente dans l'estimation.

Néanmoins, dans notre cas, nous avons une relations X_i et Y_i qui n'est visiblement pas linéaire. Donc les hypothèses paramétriques de linéarité que nous avons pris pour appliquer une régression linéaire ne sont pas correctes. De ce fait le modèle prend de fausse information à partir des hypothèses qui n'étaient pas correctes d'où l'ajustement pauvre du modèle sur les données.

```
# Ajustement de régression linéaire
reg5 = lm(Y~X)
```



3. Validation croisée

3.1. Estimateur du risque quadratique

Soit $\hat{R}^2$ le risque quadratique de $\hat{r}_{n,h}(x)$

$$\hat{R}^2 = \frac{1}{N}\sum_{i=1}^{N}(Y_i - \hat{r}_{n,h}^{-}(x))^2$$

On cherche la fenêtre h qui minimise $\hat{R}^2$. La méthode erreur quadratique moyenne (MSE) par la validation croisée est utilisée pour sélectionner h.

En effet, si nous appliquons tout simplement la méthode d'erreur quadratique moyenne sans la validation croisée, nous aurons toujours un $h = 0$ ou le plus petit possible avec la convergence de $\hat{r}_{n,h}(x)$ vers Y_i pour minimiser $\hat{R}^2$. Avec un $h \simeq 0$, l'estimateur prendra en compte que les points extrêment près du point d'évaluation et surajutera sur les données.

C'est pour éviter ce problème que nous appliquons la validation croisée.

3.2. Découpage du jeu de données

En validation croisée, le découpage du jeu de données repose sur l'idée d'utiliser deux fois l'échantillon : un sous-ensemble J_+ pour ajuster le modèle et le reste du sous-ensemble J_- pour évaluer sa performance.

Cette méthode de validation croisée consiste à éviter la dépendance sur l'échantillon en utilisant des observations distinctes d'échantillon pour l'ajustement et pour l'évaluation.

3.3 Leave-one-out

Implémentons la leave-one-out validation croisée.

```
# Cross-validation
cv = function(X, Y, h, K = dnorm) { # La fonction de validation croisée donne
 sum( # la somme de
   (
   (Y - nw(X, X, Y, h, K))/ # différence de Y et Y ajusté par l'estimateur nadraya-watson
     (1 - K(0) / colSums(K(outer(X, X, "-")/h))) # sur l'échantillon d'ajustement
   )^2 # élevé au carré
   )
 }

# Cross-validation sur une grille de h
cv.grid = function(X, Y, h.grid = diff(range(X) * (seq(0.1, 1, l = 10))), K = dnorm, plot.cv = FALSE){
  obj = sapply(h.grid, function(h) cv(X = X, Y = Y, h = h, K = K)) # Application de validation croisée
  h = h.grid[which.min(obj)] # h correspond au h parmi la grille qui minimise le résultat
  if (plot_cv) {plot(h_grid, obj, type)} # Traçage de graphique lorsque plot.cv = TRUE
}
```

4. Discussion

La régression, en général, est un modèle donnant une espérance conditionnelle de la variable cible pour une variable explicative donnée.

À la différence de la régression paramétrique qui suppose une fonction à priori pour modéliser, la régression non paramétrique modélise purement à partir de l'information retirées des données. De ce fait, la régression non paramétrique donne plus de flexibilité et prédit assez bien des relations de forme complexe et variée.

Lorsque nous avons la difficulté d'attribuer une fonction de régression paramétrique à priori, la régression non paramétrique devient une bonne solution pour la modélisation.

Dans notre cas, la régression non paramétrique a assez bien réussi à décrire le lien entre les Y_i et X_i. Néanmoins, vu la forme du lien, nous aurions pu envisager une régression paramétrique, par exemple, une régression polynomiale, permettant de limiter la variance par rapport au modèle non paramétrique.

Bibliographie

Anders Munk-Nielson, superpronker, (2017). Kernel Density Estimation [Vidéo]. YouTube https://www.youtube.com/watch?v=gPWsDh59zdo&list=PLG8higEnlmAG-Vw33i9vaKCQ9MSUa3lUS

Anders Munk-Nielson, superpronker, (2017). Nonparametric Kernel regression [Vidéo]. YouTube https://www.youtube.com/watch?v=ncF7ArjJFqM&list=PLG8higEnlmAG-Vw33i9vaKCQ9MSUa3lUS&index=2

Brian Zaharatos, brianzaha (2021). Introduction to nonparametric regression models [Vidéo]. YouTube https://www.youtube.com/watch?v=XyhOXwvxqQ8

Brian Zaharatos, brianzaha (2021). Kernel estimation [Vidéo]. YouTube https://www.youtube.com/watch?v=pLYTbbZC4HM

Brian Zaharatos, brianzaha (2021). Motivating kernel estimation [Vidéo]. YouTube https://www.youtube.com/watch?v=yDK0jfEmZG8

Gabriel Turinici, gabrielturinici5220, (2017). Statistique Non paramétrique [Vidéo]. YouTube https://www.youtube.com/watch?v=rgn7xdpIl1M&t=759s

GeoDa Software, GeoDaCenter, (2017). Density Estimation and Kernel Regression [Vidéo]. YouTube https://www.youtube.com/watch?v=0kHy7dzyT90

Jochen Voss, JochenVoss, (2022) Cross Validation [Vidéo]. YouTube https://www.youtube.com/watch?v=EyHa_ZVvRX8&list=PL5uCU4p9aonZPNJP7H75dgkUBFTAjmjY-

Jochen Voss, JochenVoss, (2022). Histograms [Vidéo]. YouTube https://www.youtube.com/watch?v=EyHa_ZVvRX8&list=PL5uCU4p9aonZPNJP7H75dgkUBFTAjmjY-

Jochen Voss, JochenVoss, (2022). Kernel Density Estimation [Vidéo]. YouTube https://www.youtube.com/watch?v=IT90OYOyWN8&list=PL5uCU4p9aonZPNJP7H75dgkUBFTAjmjY-&index=2

Justin Esarey, jeesarey, Kernel Density Estimation and Kernel Regression (2013). [Vidéo]. YouTube https://www.youtube.com/watch?v=QSNN0no4dSI