Entrega 5 - Regresión no paramétrica

Coudet & Czarnievicz

Diciembre 2018

El código para esta entrega puede encontrarse haciendo click aquí.

Ejercicio 1

```
r <- function(x){10 * sin(x) + x^2}
set.seed(1234)
x <- runif(n <- 100, 0, 6)
epsilon <- rnorm(100, 0, 3)
y <- r(x) + epsilon
as_tibble(cbind(x, y)) %>%
   ggplot() +
   geom_line(aes(x, y=r(x)), col="red", alpha=0.25) +
   geom_point(aes(x,y), color="blue") +
   labs(x="X", y="Y") +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

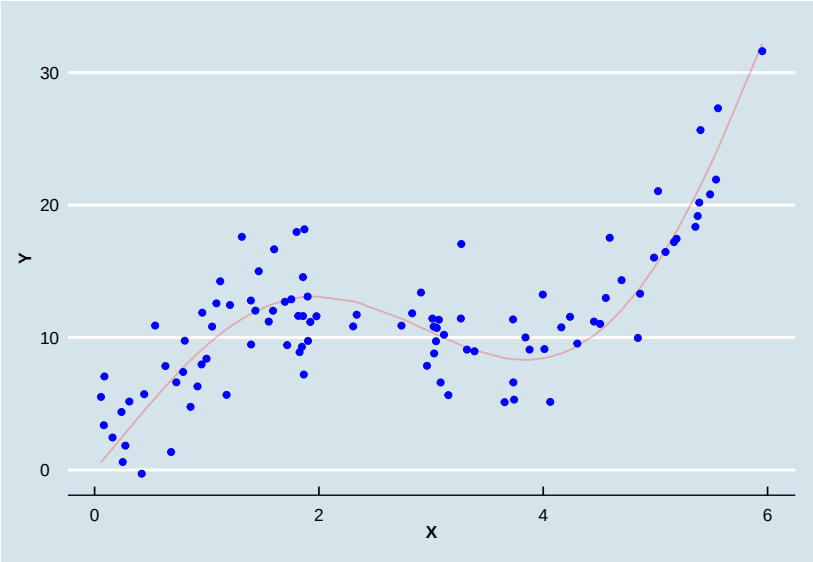

Figure 1: Scatter plot of X and Y (blue) and function $r(x)$ (red)

Linear Smoother: Regressogram

Consiste en tomar $a \leq x_i \leq b$ $i = 1, \ldots, n$ y dividir $(a; b)$ en m intervalos de igual longitud denominados $B_1; \ldots; B_m$. Sea k_j el número de $x_i \in B_j$, se define el *linear smoother*:

$$\hat{r}_n(x) = \frac{1}{k_j} \sum_{i/x_i \in B_j} Y_i = \sum_{i=1}^{n} Y_i \, \ell_i(x)$$

donde

$$\ell_i(x) = \begin{cases} \dfrac{1}{k_j} & \text{si } x_i \in B_j \\ \\ 0 & \text{otherwise} \end{cases}$$

```
m <- 10
B_j <- NULL
for (i in 1:m) {
   B_j <- c(B_j,
            rep(as.numeric(names(table(cut(sort(x), m, labels=1:m))))[i],
                as.numeric(table(cut(sort(x), m, labels=1:m)))[i]))
   }
as_tibble(cbind(x, y)) %>%
   arrange(x) %>%
   mutate(B_j = B_j) %>%
   group_by(B_j) %>%
   summarise(k_j = n(), r_n = sum(y/k_j)) %>%
   bind_cols(
      as_tibble(levels(cut(sort(x), m))) %>%
      separate(value, into=c("A", "B"), sep=",") %>%
      transmute(x_min = as.numeric(gsub("(", "", A, fixed=TRUE)),
                x_max = as.numeric(gsub("]", "", B, fixed=TRUE)))
   ) %>%
   ggplot() +
   geom_segment(aes(x=x_min, xend=x_max, y=r_n, yend=r_n), color="red") +
   geom_step(aes(x=x_min, y=r_n), color="red") +
   geom_point(data=as_tibble(cbind(x, y)), aes(x,y), color="blue", alpha=0.25) +
   labs(x="X", y="Y") +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

```
## Warning: Calling `as_tibble()` on a vector is discouraged, because the behavior is li
## This warning is displayed once per session.
```

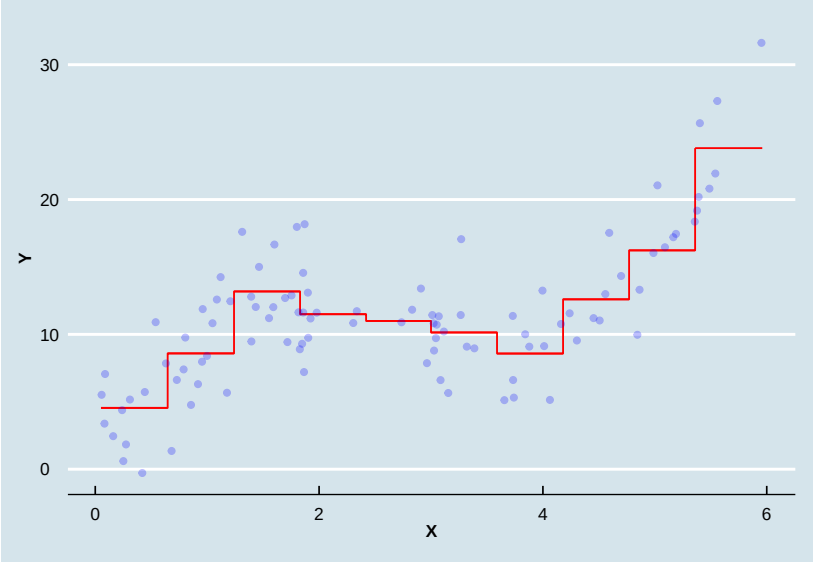

Figure 2: Regressogram of Y given X

Linear Smoother: Local Averages

Consiste en definir los conjuntos $B_x = \{i \,/\, |x_i - x| \leq h\}$, dado $h > 0$. Sea n_x el cardinal del conjunto B_x. Para aquellos valores de x tales que $n_x > 0$ se define el *linear smoother*:

$$\hat{r}_n(x) = \frac{1}{n_x} \sum_{i \in B_x} Y_i = \sum_{i=1}^{n} Y_i \, \ell_i(x)$$

donde

$$\ell_i(x) = \begin{cases} \dfrac{1}{n_x} & \text{si } |x_i - x| \leq h \\ \\ 0 & \text{otherwise} \end{cases}$$

```
h <- (max(x) - min(x)) / m
# Calcula las distancias entre todos los valores de X
distancias <- as.matrix(dist(sort(x), diag=TRUE, upper=TRUE))
# Asigna TRUE a aquellas distances que sean menores que h
distancias <- distancias <= h
# Sumamos con filas para obetener los n_x
n_x <- rowSums(distancias)
names(n_x) <- NULL
# Construimos la matriz L
L <- distancias * (1/n_x)
# Ordenamos y en función de x
y_ordered <- as_tibble(cbind(x, y)) %>% arrange(x)
# Calculamos r_n
```

```
r_n <- L %*% as.numeric(y_ordered$y)
# Graficamos
y_ordered %>%
   mutate(r_n = r_n) %>%
   ggplot() +
   geom_step(aes(x, r_n), col="red", na.rm=TRUE) +
   geom_segment(aes(x=x, xend=lead(x), y=r_n, yend=r_n), col="red", na.rm=TRUE) +
   geom_segment(aes(x=max(x), xend=max(x)+.2, y=r_n[length(r_n)], yend=r_n[length(r_n)]
                col="red", na.rm=TRUE) +
   geom_point(aes(x,y), col="blue", alpha=0.25)  +
   labs(x='X', y='Y') +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

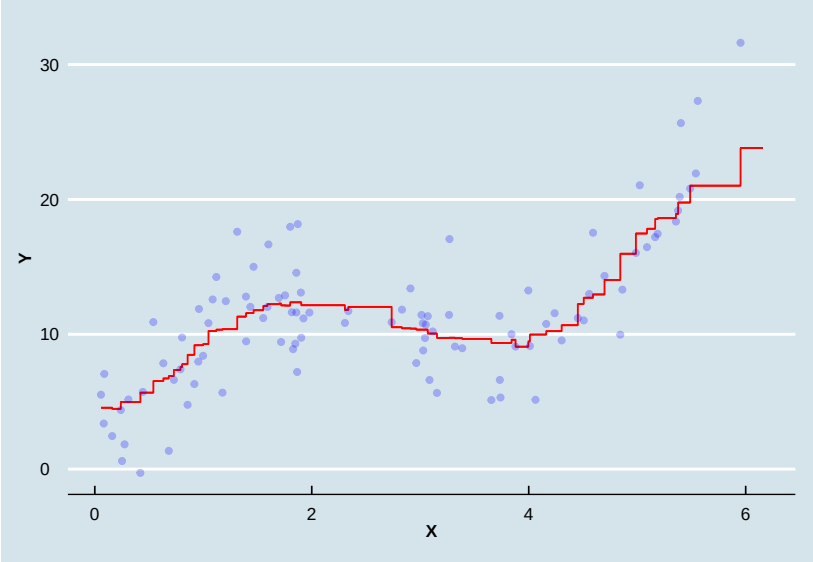

Figure 3: Local Average of Y given X

Kernel Regression

El objetivo es estimar r a través de promedios ponerados de las Y_i. En particular, buscamos que dicho sistema de ponderadors otorgue mayor peso a los valores más cercanos a x. Por lo tanto, el estimador sigue siendo de la forma:

$$\hat{r}_n(x) = \sum_{i=1}^{n} \ell_i(x)\, Y_i$$

donde ahora $\ell_i(x)$ serán los pesos dados por:

$$\ell_i(x) = \frac{K\left(\frac{x-x_i}{h}\right)}{\sum_{j=1}^{n} K\left(\frac{x-x_j}{h}\right)}$$

Dicho estimador es conocido como *Nadaraya-Watson kernel estimator*.

```
h <- (max(x) - min(x)) / m
x_rep <- NULL
indices <- NULL
for (i in 1:length(x)) {
   x_rep <- c(x_rep, rep(x[i], length(x)))
   indices <- c(indices, rep(i, length(x)))
}
denominadores <- cbind(indices, x_rep, x) %>%
   as_tibble() %>%
   rename(x_j = x, x = x_rep, indice = indices) %>%
   mutate(kernel = dnorm(x=((x - x_j)/h))) %>%
   group_by(indice) %>%
   summarise(denom = sum(kernel))
elles <- dnorm(as.matrix(dist(x)) / h) * (1/as.numeric(denominadores$denom))
r_n <- as.numeric(elles %*% y)
tibble(x=x, y=y, r_n=r_n) %>%
   ggplot() +
   geom_line(aes(x, r_n), col="red") +
   geom_point(aes(x, y), col="blue", alpha=0.25) +
   labs(x="X", y="Y") +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

k-nearest neighbours regression

La presente sección se pasa en James et al. (2013). El método *knn* se basa en identificar las k observaciones más cercanas a valor muestral x_0 y promediar los valores de y correspondientes a cada vecindad $\mathcal{N}_0$. De esta forma, se obtiene el estimador de r:

$$\hat{r}_n(x) = \frac{1}{k} \sum_{x_i \in \mathcal{N}_0} y_i = \sum_{i=1}^{n} \ell_i(x)\, y_i$$

donde

$$\ell_i(x) = \begin{cases} \frac{1}{k} & \text{if } x_i \in \mathcal{N}_0 \\ 0 & \text{otherwise} \end{cases}$$

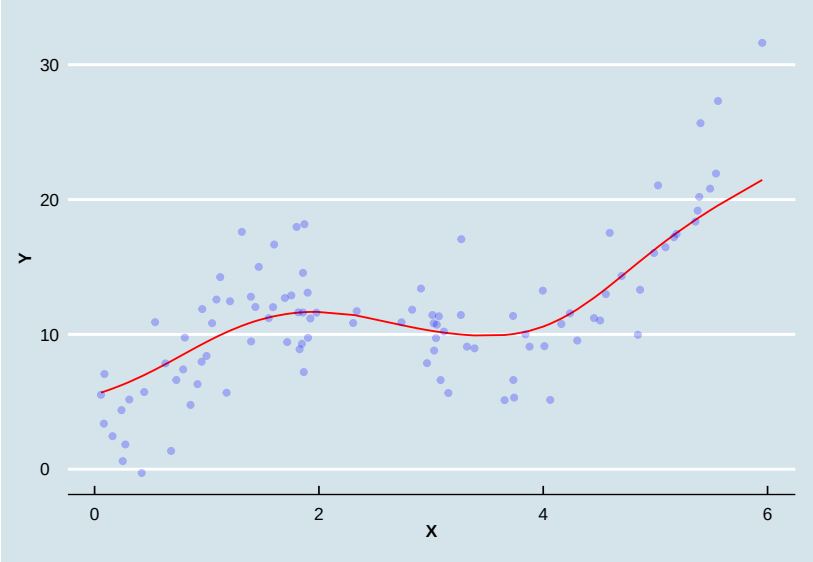

Figure 4: Nadayara-Watson estimator of $r(x)$ using Gaussian kernel

```
library(FNN)
as_tibble(x, y) %>%
   rename(x = value) %>%
   mutate(k_3 = knn.reg(train=x, y=y, k=3)$pred,
          k_10 = knn.reg(train=x, y=y, k=10)$pred,
          k_20 = knn.reg(train=x, y=y, k=20)$pred) %>%
   gather(key=key, value=predicteds, -x) %>%
   mutate(key = gsub("_", " = ", key)) %>%
   ggplot() +
   geom_line(aes(x, predicteds, color=key)) +
   geom_point(data=tibble(x, y), aes(x, y), col="blue", alpha=0.25) +
   scale_color_manual(values=c("red","darkgreen","orange"),
                      breaks=c("k = 3", "k = 10", "k = 20")) +
   labs(x="X", y="Y") +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"),
         legend.position="bottom",
         legend.title=element_blank(),
         legend.spacing.x=unit(0.5, "cm"))
```

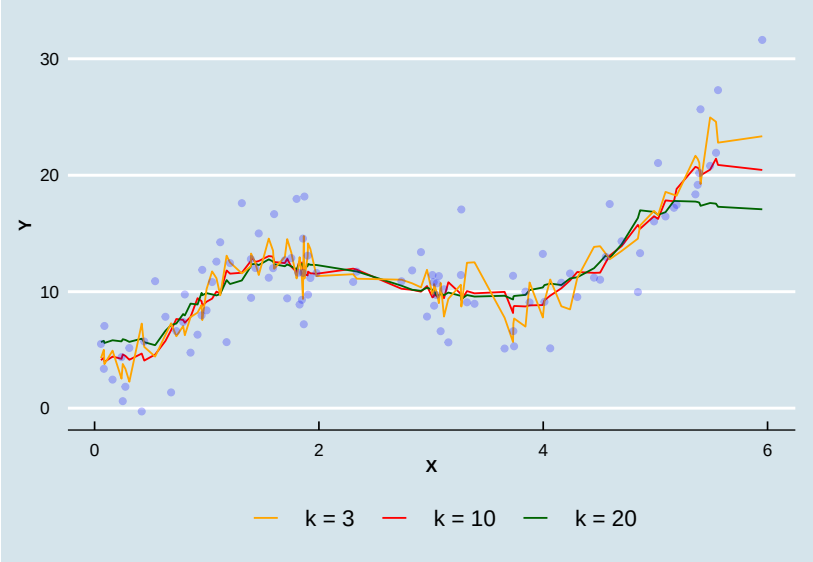

Figure 5: KNN regression of Y given X for different values of k

```
kas <- seq(1, n-1, 1)
r2pred <- NULL
press <- NULL
for (i in 1:length(kas)) {
   r2pred <- c(r2pred, knn.reg(train=x, y=y, k=i)$R2Pred)
   press <- c(press, knn.reg(train=x, y=y, k=i)$PRESS)
}
facets <- c('press'='Sums of squares \n of the predicted residuals',
            'r2pred'='Predicted R-square')
optimal_k <- tibble(kas = kas, r2pred = r2pred, press = press)
gather(optimal_k, key=key, value=value, -kas) %>%
   ggplot() +
   geom_line(aes(kas, value, color=key)) +
   geom_point(data=tibble(
      x = rep(kas[which.max(optimal_k$r2pred)], 2),
      y = c(max(optimal_k$r2pred), min(optimal_k$press)),
      key = c("r2pred", "press")), aes(x, y, color=key)) +
   facet_wrap(~key, scales="free_y", labeller = as_labeller(facets)) +
   labs(x='k')  +
   ggthemes::theme_economist() +
   theme(legend.position='none',
         axis.title.y= element_blank(),
         axis.title=element_text(face="bold"))
```

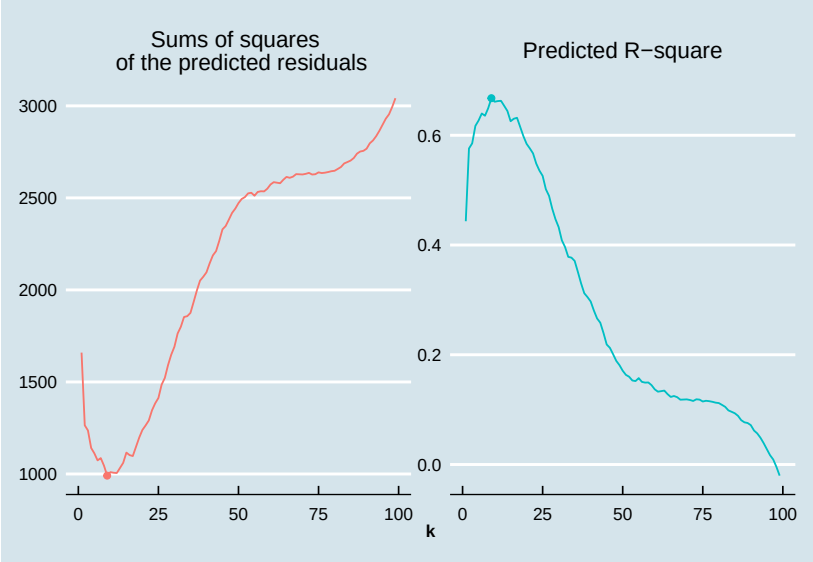

Figure 6: Sums of squares of the predicted residuals and Predicted R-square for different values of k

```
as_tibble(x, y) %>%
   ggplot() +
   geom_line(aes(x, knn.reg(train=x, y=y, k=9)$pred), color="red") +
   geom_point(data=tibble(x, y), aes(x, y), col="blue", alpha=0.25) +
   labs(x="X", y="Y") +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

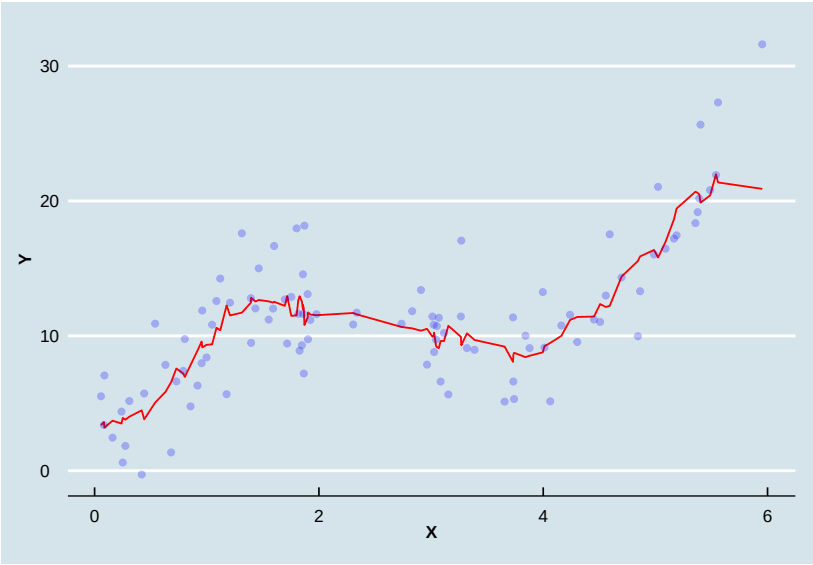

Figure 7: KNN regression of Y given X ($k = 9$)

Local polynomial regression

La estimación de $r(x)$ por polinomios locales se basa en suponer que la misma es un polinomio de grado p para valores u en una vencindad en torno a x. Es decir:

$$P_x(u;a) = a_0 + a_1(u-x) + \frac{a_2}{2!}(x-u)^2 + \ldots + \frac{a_p}{p!}(x-u)^p$$

Por lo tanto:

$$\hat{r_n}(u) = P_x(u;\hat{a})$$

Para el caso particular en que $u = x$:

$$\hat{r}_n(x) = P_x(x;\hat{a}) = \hat{a}_0$$

Ahora bien, las estimaciones de los coeficientes a_j, $j = 0, \ldots, p$ se obtienen minimizando la suma de cuadrados ponderados:

$$\min_a \left\{ \sum_{i=1}^{n} w_i(x)(Y_i - \hat{r}_n(x))^2 \right\} = \min_a \left\{ \sum_{i=1}^{n} w_i(x)(Y_i - P_x(x;\hat{a}))^2 \right\}$$

donde $w_i(x) = K\left(\frac{x_i - x}{h}\right)$

Este problema puede resolverse más sencillamente si es expresado en forma matricial, utilizando las siguientes matrices:

$$\boldsymbol{X}_x = \begin{pmatrix} 1 & x_1 - x & \cdots & \frac{(x_1-x)^p}{p!} \\ 1 & x_2 - x & \cdots & \frac{(x_2-x)^p}{p!} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n - x & \cdots & \frac{(x_n-x)^p}{p!} \end{pmatrix}_{n\times(p+1)}$$

$$\boldsymbol{W}_x = \begin{pmatrix} w_1(x) & 0 & \cdots & 0 \\ 0 & w_2(x) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_n(x) \end{pmatrix}_{n\times n} = \begin{pmatrix} K\left(\frac{x_1-x}{h}\right) & 0 & \cdots & 0 \\ 0 & K\left(\frac{x_2-x}{h}\right) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & K\left(\frac{x_n-x}{h}\right) \end{pmatrix}_{n\times n}$$

$$\min_a \left\{ (Y - \boldsymbol{X}_x a)' \boldsymbol{W}_x (Y - \boldsymbol{X}_x a) \right\} =$$

$$= \min_a \left\{ Y' \boldsymbol{W}_x Y - Y' \boldsymbol{W}_x \boldsymbol{X}_x a - a' \boldsymbol{X}'_x \boldsymbol{W}_x Y + a' \boldsymbol{X}'_x \boldsymbol{W}_x \boldsymbol{X}_x a \right\}$$

$$\frac{\partial}{\partial a'} = -2\boldsymbol{X}'_x \boldsymbol{W}_x Y + 2\boldsymbol{X}'_x \boldsymbol{W}_x \boldsymbol{X}_x a = 0 \Rightarrow \boldsymbol{X}'_x \boldsymbol{W}_x Y = \boldsymbol{X}'_x \boldsymbol{W}_x \boldsymbol{X}_x a \Rightarrow$$

$$\Rightarrow \boxed{a = (\boldsymbol{X}'_x \boldsymbol{W}_x \boldsymbol{X}_x)^{-1} (\boldsymbol{X}'_x \boldsymbol{W}_x Y)}$$

Por lo tanto:

$$\hat{r}_n(x) = \sum_{i=1}^{n} \ell_i(x) Y_i$$

donde

$$\ell(x)' = (\boldsymbol{X}'_x \boldsymbol{W}_x \boldsymbol{X}_x)^{-1} \boldsymbol{X}'_x \boldsymbol{W}_x$$

Para el caso particular en que $p = 1$ se tiene el estimador conocido como *Local Linear Smoothing*:

$$\ell_i(x) = \frac{b_i(x)}{\sum_{j=1}^{n} b_j(x)}$$

donde

$$b_i(x) = K\left(\frac{x_i - x}{h}\right)(S_{n,2} - (x_i - x)S_{n,1}(x))$$

$$S_{n,j}(x) = \sum_{i=1}^{n} K\left(\frac{x_i - x}{h}\right)(x_i - x)^j, \ j = 1, 2$$

```
local_poly <- loess(y ~ x, tibble(x, y), degree=2, family="gaussian", span=0.75)
tibble(x, y, local_poly$fitted) %>%
   ggplot() +
   geom_line(aes(x, local_poly$fitted), col="red") +
   geom_point(aes(x, y), color="blue", alpha=0.25) +
   labs(x='X', y='Y') +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

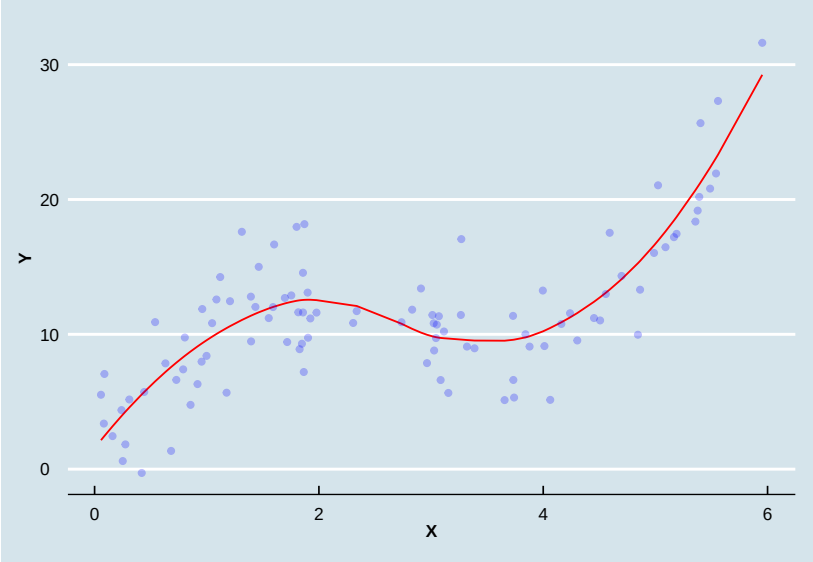

Figure 8: Local polynomial regression of Y given X (Gaussian family, h=0.75, degree=2)

Ejercicio 2

Parte a

```
# CMB data from WMAP probe
# first column is wavenumber (x variable)
# second column is the estimated specturm
# third column is estimated standard deviation
cmb <- read_csv("cmb.csv")
n <- dim(filter(cmb, ell <= 401))[1]
filter(cmb, ell <=401) %>%
   ggplot() +
   geom_point(aes(ell, Cl), col="blue") +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face = "bold"))
```

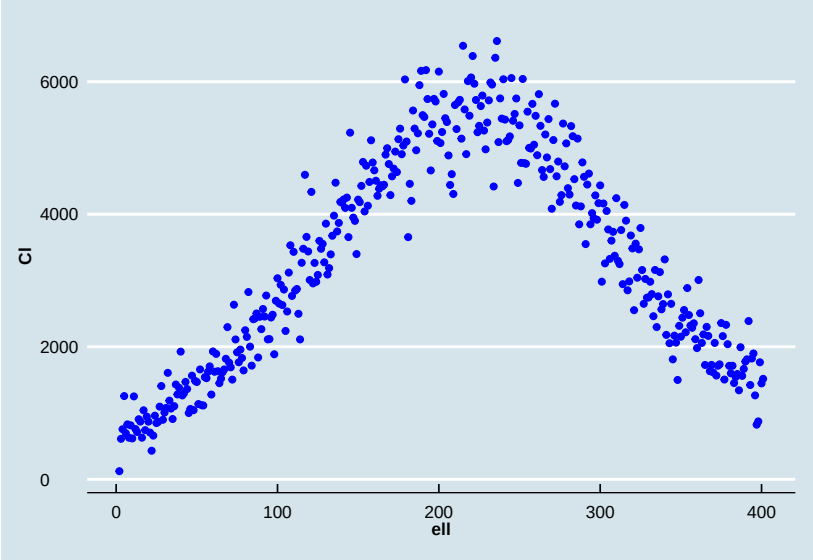

Figure 9: Scatter plot of ell (x axis) and Cl (y axis)

Regressogram

```
cmb <- select(cmb, ell, Cl) %>% rename(x = ell, y = Cl) %>% filter(x <= 401)
x <- cmb$x; y <- cmb$y; m <- 10
B_j <- NULL
for (i in 1:m) {
   B_j <- c(B_j,
            rep(as.numeric(names(table(cut(sort(x), m, labels=1:m))))[i],
                as.numeric(table(cut(sort(x), m, labels=1:m)))[i]))
   }
as_tibble(cbind(x, y)) %>%
   arrange(x) %>%
   mutate(B_j = B_j) %>%
   group_by(B_j) %>%
   summarise(k_j = n(), r_n = sum(y/k_j)) %>%
   bind_cols(as_tibble(levels(cut(sort(x), m))) %>%
             separate(value, into=c("A", "B"), sep=",") %>%
             transmute(x_min = as.numeric(gsub("(", "", A, fixed=TRUE)),
                       x_max = as.numeric(gsub("]", "", B, fixed=TRUE)))) %>%
   ggplot() +
   geom_point(data=as_tibble(cbind(x, y)), aes(x,y), color="blue", alpha=0.25) +
   geom_segment(aes(x=x_min, xend=x_max, y=r_n, yend=r_n), color="red") +
   geom_step(aes(x=x_min, y=r_n), color="red") +
   labs(x="ell", y="Cl") +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

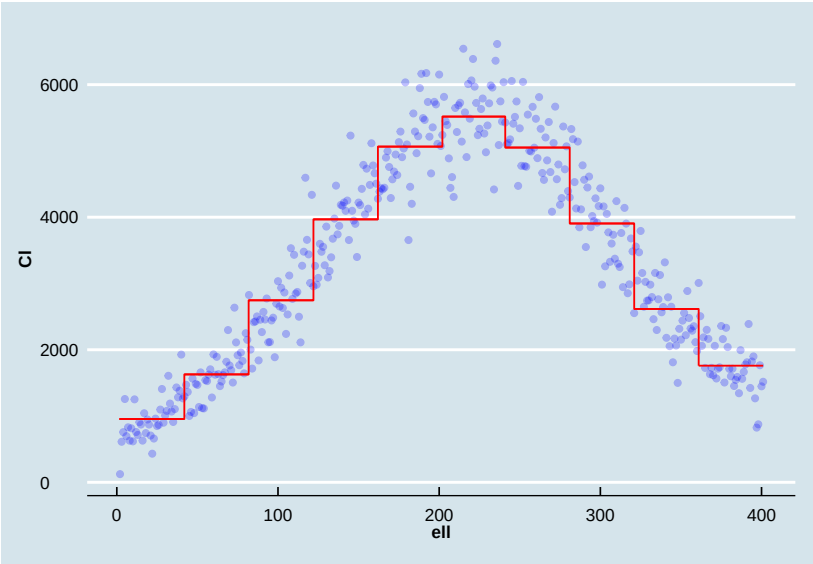

Figure 10: Regressogram of Cl given ell

Local Averages

```
h <- (max(x) - min(x)) / m
distancias <- as.matrix(dist(sort(x), diag=TRUE, upper=TRUE))
distancias <- distancias <= h
n_x <- rowSums(distancias)
names(n_x) <- NULL
L <- distancias * (1/n_x)
y_ordered <- as_tibble(cbind(x, y)) %>% arrange(x)
r_n <- L %*% as.numeric(y_ordered$y)
y_ordered %>%
   mutate(r_n = r_n) %>%
   ggplot() +
   geom_point(aes(x,y), col="blue", alpha=0.25) +
   geom_step(aes(x, r_n), col="red", na.rm=TRUE) +
   geom_segment(aes(x=x, xend=lead(x), y=r_n, yend=r_n), col="red", na.rm=TRUE) +
   geom_segment(aes(x=max(x), xend=max(x)+.2, y=r_n[length(r_n)], yend=r_n[length(r_n)]
                col="red", na.rm=TRUE) +
   labs(x='ell', y='Cl') +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

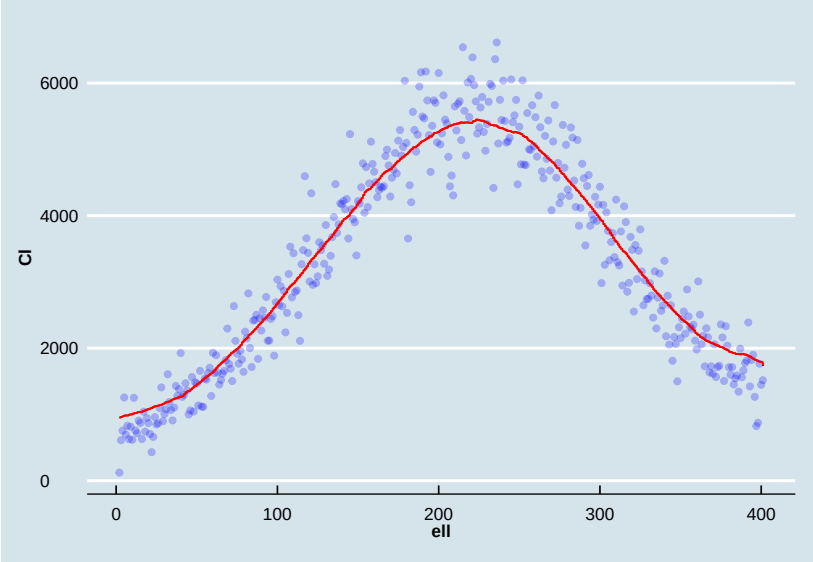

Figure 11: Local average of Cl given ell

Kernel Regression

```
h <- (max(x) - min(x)) / m
x_rep <- NULL
indices <- NULL
for (i in 1:length(x)) {
   x_rep <- c(x_rep, rep(x[i], length(x)))
   indices <- c(indices, rep(i, length(x)))
}
denominadores <- cbind(indices, x_rep, x) %>%
   as_tibble() %>%
   rename(x_j = x, x = x_rep, indice = indices) %>%
   mutate(kernel = dnorm(x=((x - x_j)/h))) %>%
   group_by(indice) %>%
   summarise(denom = sum(kernel))
elles <- dnorm(as.matrix(dist(x)) / h) * (1/as.numeric(denominadores$denom))
r_n <- as.numeric(elles %*% y)
tibble(x=x, y=y, r_n=r_n) %>%
   ggplot() +
   geom_point(aes(x, y), col="blue", alpha=0.25) +
   geom_line(aes(x, r_n), col="red") +
   labs(x="ell", y="Cl") +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

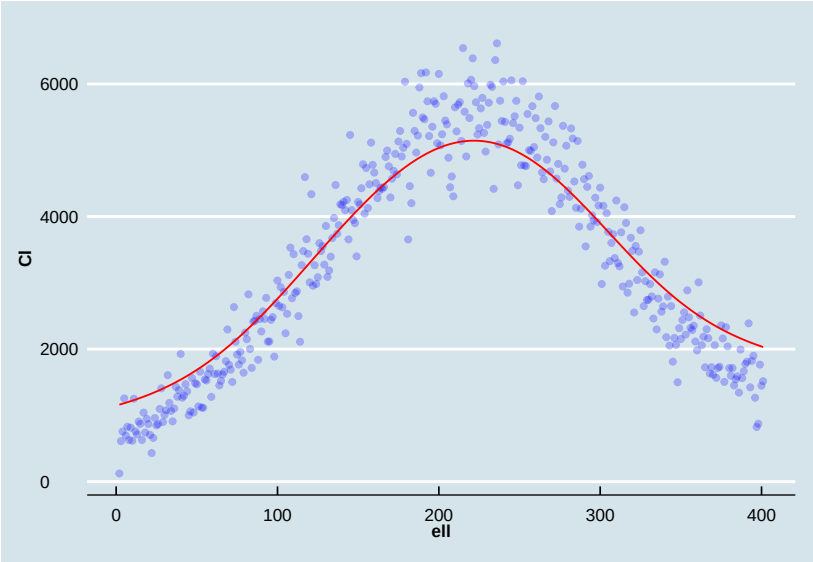

Figure 12: Nadayara-Watson estimator of $r(x)$ using Gaussian kernel

k-nearest neighbours regression

```
kas <- seq(1, n-1, 1)
r2pred <- NULL
press <- NULL
for (i in 1:length(kas)) {
   r2pred <- c(r2pred, knn.reg(train=x, y=y, k=i)$R2Pred)
   press <- c(press, knn.reg(train=x, y=y, k=i)$PRESS)
}
facets <- c('press'='Sums of squares \n of the predicted residuals',
            'r2pred'='Predicted R-square')
optimal_k <- tibble(kas = kas, r2pred = r2pred, press = press)
gather(optimal_k, key=key, value=value, -kas) %>%
   ggplot() +
   geom_line(aes(kas, value, color=key)) +
   geom_point(data=tibble(
      x = rep(kas[which.max(optimal_k$r2pred)], 2),
      y = c(max(optimal_k$r2pred), min(optimal_k$press)),
      key = c("r2pred", "press")), aes(x, y, color=key)) +
   facet_wrap(~key, scales="free_y", labeller = as_labeller(facets)) +
   labs(x='k')  +
   ggthemes::theme_economist() +
   theme(legend.position='none',
         axis.title.y= element_blank(),
         axis.title=element_text(face="bold"))
```

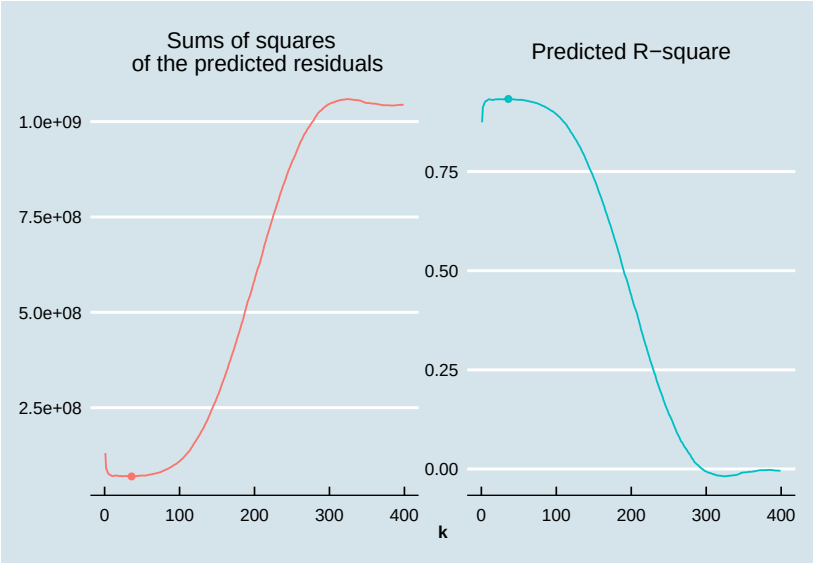

Figure 13: Sums of squares of the predicted residuals and Predicted R-square for different values of k

```
as_tibble(x, y) %>%
   ggplot() +
   geom_point(data=tibble(x, y), aes(x, y), col="blue", alpha=0.25) +
   geom_line(aes(x, knn.reg(train=x, y=y, k=36)$pred), color="red") +
   labs(x='ell', y='Cl') +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

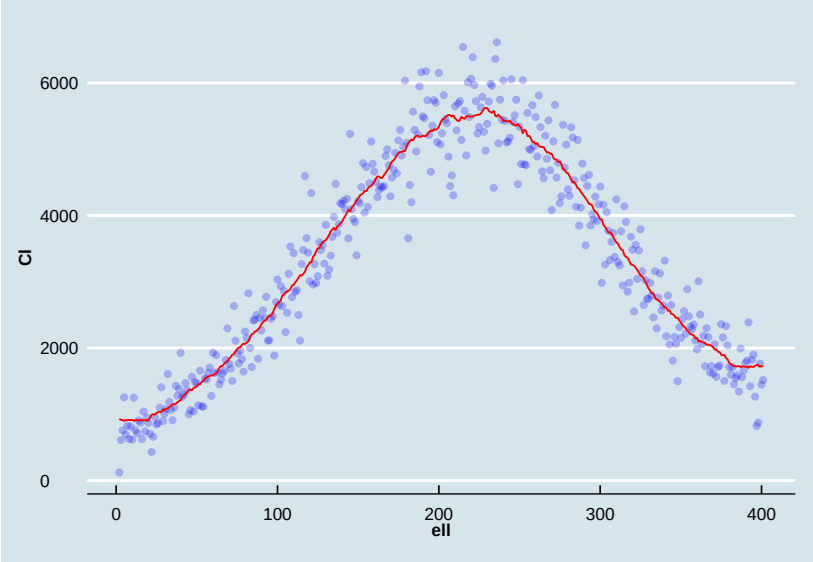

Figure 14: KNN regression of Cl given ell ($k = 36$)

Local polynomial regression

```
local_poly <- loess(y ~ x, tibble(x, y), degree=2, family="gaussian", span=0.75)
tibble(x, y, local_poly$fitted) %>%
   ggplot() +
   geom_point(aes(x, y), color="blue", alpha=0.25) +
   geom_line(aes(x, local_poly$fitted), col="red") +
   labs(x='ell', y='Cl') +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

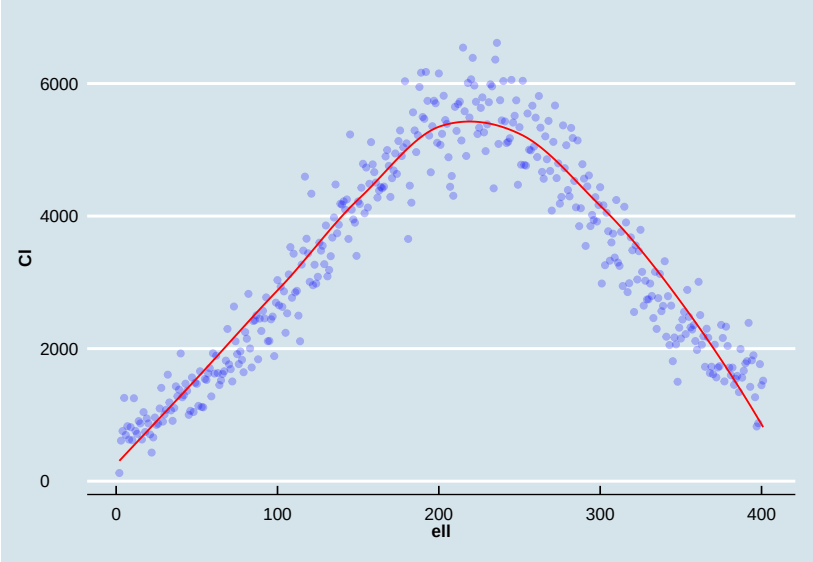

Figure 15: Local polynomial regression of Cl given ell (Gaussian family, h=0.75, degree=2)

Parte b

"The function estimate $\hat{r}_n(x)$ is relatively insensitive to heteroscedasticity. However, when it comes to making confindence bands for $r(x)$ we must take into account the nonconstant variance." Wasserman (2007). Por lo tanto, debemos estimar la varianza para construir intervalos de confianza, para esto seguiremos la estrategia planteada por el autor, la cual consiste en:

- Estimar $r(x)$ mediante cualquier método no paramétrico.
- Definir $Z_i = \log(Y_i - \hat{r}_n(x_i))^2$
- Regresar Z_i contra x_i mediante cualquier método no paramétricO para conseguir una estimación $\hat{q}(x)$ de $\log(\sigma^2)$.
- Entonces $\hat{\sigma}^2(x) = \exp \hat{q}(x)$

El inervalo de confianza para $\hat{r}_n(x)$ queda determinado por:

$$\hat{r}_n(x) \pm c\,\hat{\sigma}(x)$$

donde c es una constante.

Si $\sigma^2(x) = \sigma^2 = V(\epsilon_i)$ entonces:

$V(\hat{r}_n(x)) = \sigma^2 ||\ell(x)||^2$.

Entonces se considera el intervalo de confianza $I(x)$ para $E(\hat{r}_n(x)) = \bar{r}(x)$:

$$\bar{r}(x) \pm c\sigma^2 ||\ell(x)||^2$$

donde c se obtiene a partir de calcular:

$$P(\bar{r}(x) \notin I(x) \text{ para algún } x \in [a,b]) = P\left(\max_{x\in[a,b]} \frac{|\hat{r}_n(x) - \bar{r}(x)|}{\sigma||\ell(x)||} > c\right) =$$

$$= P\left(\max_{x\in[a,b]} \frac{|\sum_i \epsilon_i \ell_i(x)|}{\sigma||\ell(x)||} > c\right) = P\left(\max_{x\in[a,b]} |\sum_{i=1}^{n} Z_i\, T_i(x)| > c\right)$$

donde $Z_i = \frac{\epsilon_i}{\sigma} \sim N(0,1)$ y $T_i(x) = \frac{\ell_i(x)}{||l(x)||}$.

Lo anterior implica entonces calcular la distribución del máximo de un *proceso gaussiano*, el cual afortunadamente es un problema estudiado.

En el problema del presente ejercicio, Scott (2015) obtiene un valor para c de 3.33.

```
cmb <- read_csv("cmb.csv") %>% select(ell, Cl) %>% rename(x = ell, y = Cl)
n <- dim(cmb)[1]
x <- cmb$x
y <- cmb$y
cmb %>%
   ggplot() +
   geom_point(aes(x, y), col="blue", alpha=0.15) +
   labs(x="ell", y="Cl") +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face = "bold"))
```

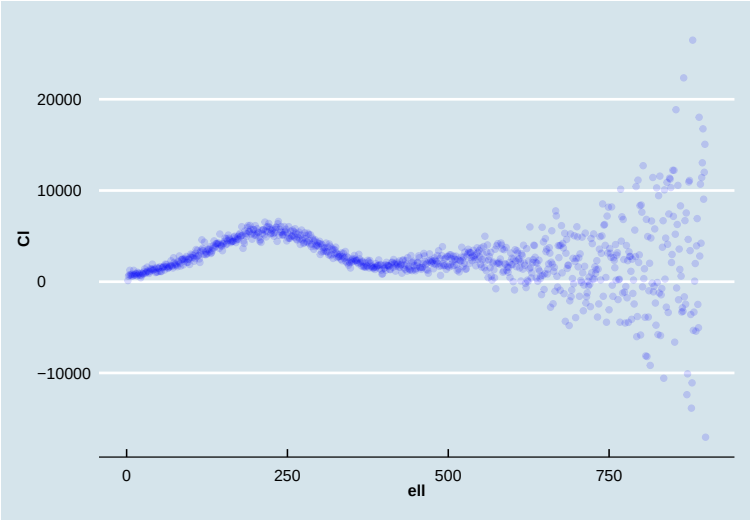

Figure 16: Scatter plot of ell (x axis) and Cl (y axis)

Local polynomial regression

```
local_poly <- loess(y ~ x, tibble(x, y), degree=2, family="gaussian", span=0.75)
tibble(x, y, r_n=local_poly$fitted) %>%
   ggplot() +
   geom_point(aes(x, y), color="blue", alpha=0.15) +
   geom_line(aes(x, r_n), col="red") +
   labs(x='ell', y='Cl') +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

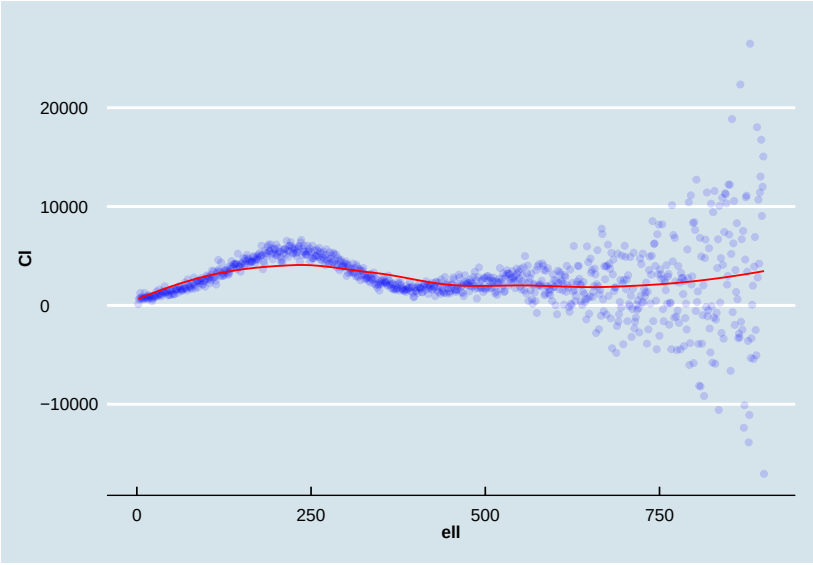

Figure 17: Local polynomial regression of Cl given ell (Gaussian family, h=0.75, degree=2)

```
z <- log(local_poly$residuals^2)
z_poly <- loess(z ~ x, tibble(x, z), degree=2, family="gaussian", span=0.75)
tibble(x, z, z_n=z_poly$fitted) %>%
   ggplot() +
   geom_point(aes(x, z), color="darkgreen", alpha=0.25) +
   geom_line(aes(x, z_n), col="magenta") +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold")) +
   labs(x='X', y=expression(log(hat(epsilon)^2)))
```

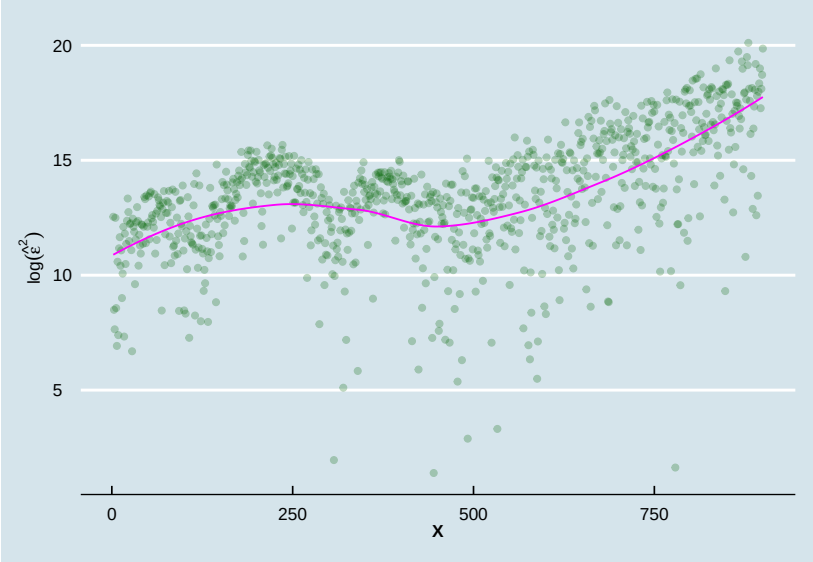

Figure 18: Local polynomial regression of the logaritm of the squared residuals given ell (Gaussian family, h=0.75, degree=2)

```
c <- 3.33
sigma_gorro <- sqrt(exp(z_poly$fitted))
upper_band <- local_poly$fitted + c * sigma_gorro
lower_band <- local_poly$fitted - c * sigma_gorro
tibble(x, y, r_n=local_poly$fitted, lower_band, upper_band) %>%
   ggplot() +
   geom_point(aes(x, y), color="blue", alpha=0.05) +
   geom_line(aes(x, r_n), col="red") +
   geom_line(aes(x, lower_band), col = "red", linetype="dotted") +
   geom_line(aes(x, upper_band), col = "red", linetype="dotted") +
   labs(x='ell', y='Cl') +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold")) +
   coord_cartesian(ylim=c(-10000,15000))
```

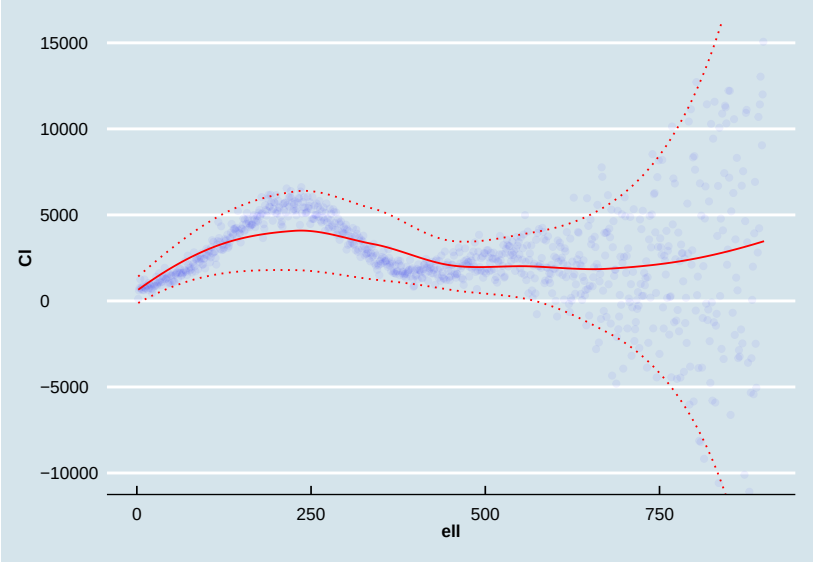

Figure 19: Local polynomial regression of Cl given ell (solid line) and confidence bands (dotted lines).

Ejercicio 3

```
r <- function(x){sqrt(x*(1 - x)) * sin((2.1 * pi)/(x + 0.05))}
n <- 1000
i <- seq(1, n, 1)
x <- i/n
sigma <- c(1/10, 1, 3)
set.seed(1234)
y_01 <- r(x) + sigma[1] * rnorm(n)
y_1 <- r(x) + sigma[2] * rnorm(n)
y_3 <- r(x) + sigma[3] * rnorm(n)
tibble(x, y_01, y_1, y_3) %>%
   gather(key=ys, value=value, -x) %>%
   mutate(ys = if_else(ys == "y_01", "sigma == 0.1",
               if_else(ys == "y_1", "sigma == 1", "sigma == 3"))) %>%
   ggplot() +
   geom_point(aes(x, value), col="blue", alpha=0.25) +
   geom_line(aes(x, r(x)), col="red") +
   facet_wrap(~ys, scales="free_y", ncol=2, labeller="label_parsed") +
   labs(x="X", y="Y") +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

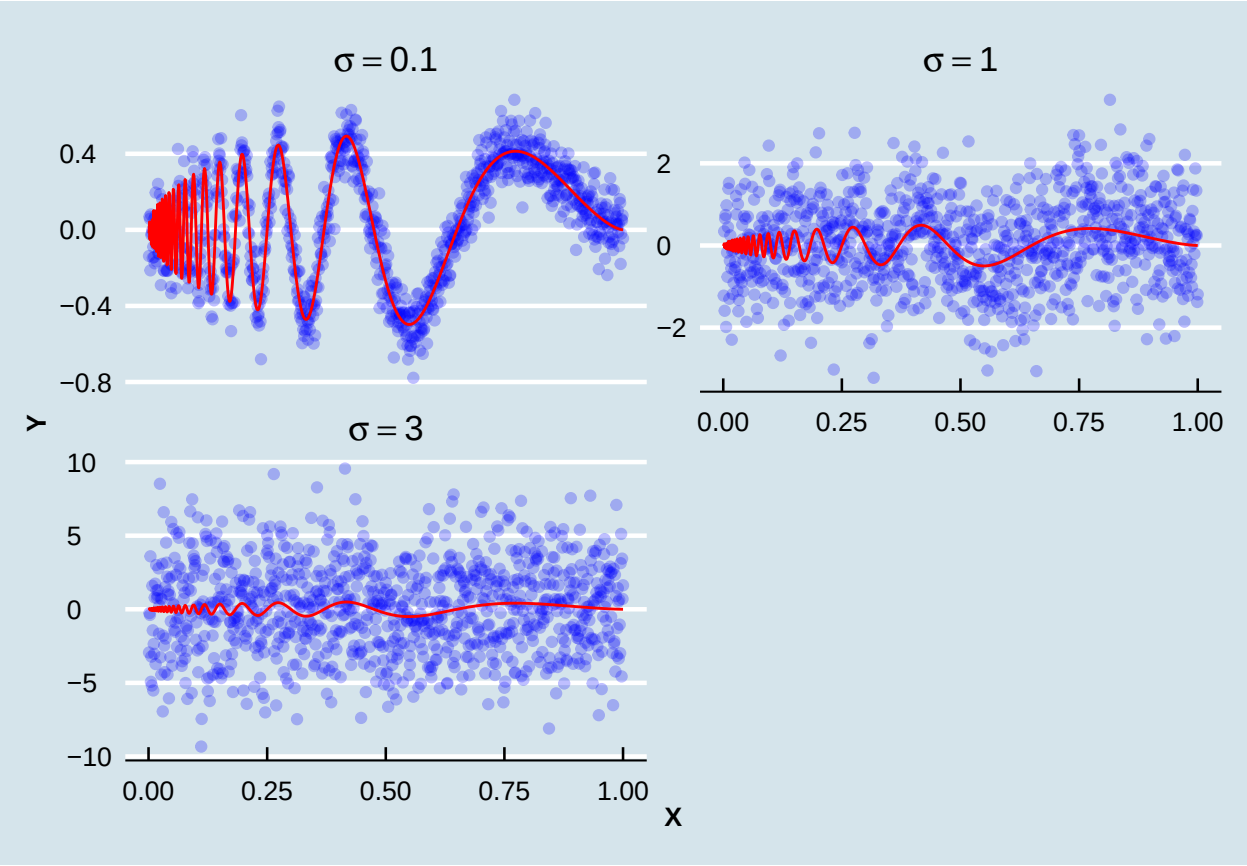

Figure 20: The Doppler Effect

knn estimation

```
kas <- seq(1, 50, 1)
r2pred <- NULL
press <- NULL
y <- cbind(y_01, y_1, y_3)
for (j in 1:dim(y)[2]) {
   r2pred_a <- NULL
   press_a <- NULL
   for (i in 1:length(kas)) {
      r2pred_a <- c(r2pred_a, knn.reg(train=x, y=y[,j], k=i)$R2Pred)
      press_a <- c(press_a, knn.reg(train=x, y=y[,j], k=i)$PRESS)
   }
   r2pred <- cbind(r2pred, r2pred_a)
   press <- cbind(press, press_a)
}
colnames(r2pred) <- colnames(y)
```

```
colnames(press) <- colnames(y)
r2pred <- as_tibble(r2pred) %>%
   mutate(indicador = "r2pred", kas = kas) %>%
   gather(key=key, value=value, -c(indicador, kas))
optimal_r2pred <- r2pred %>%
   group_by(key) %>%
   summarise(ks_star=which.max(value),
             value=max(value)) %>%
   mutate(indicador = "r2pred")
press <- as_tibble(press) %>%
   mutate(indicador = "press", kas = kas) %>%
   gather(key=key, value=value, -c(indicador, kas))
optimal_press <- press %>%
   group_by(key) %>%
   summarise(ks_star=which.min(value),
             value=min(value)) %>%
   mutate(indicador = "press")
facets <- c('press'='Sums of squares \n of the predicted residuals',
            'r2pred'='Predicted R-square')
bind_rows(press, r2pred) %>%
   ggplot() +
   geom_line(aes(kas, value, color=key)) +
   geom_point(data=bind_rows(optimal_r2pred, optimal_press),
              aes(ks_star, value, color=key)) +
   facet_wrap(~indicador, scales="free_y", labeller = as_labeller(facets)) +
   labs(x='k', color=NULL)  +
   scale_color_discrete(breaks=waiver(), labels=c(expression(sigma == 0.1),
                                                  expression(sigma == 1),
                                                  expression(sigma == 3))) +
   ggthemes::theme_economist() +
   theme(legend.position='bottom',
         axis.title.y=element_blank(),
         axis.title.x=element_text(face="bold"))
```

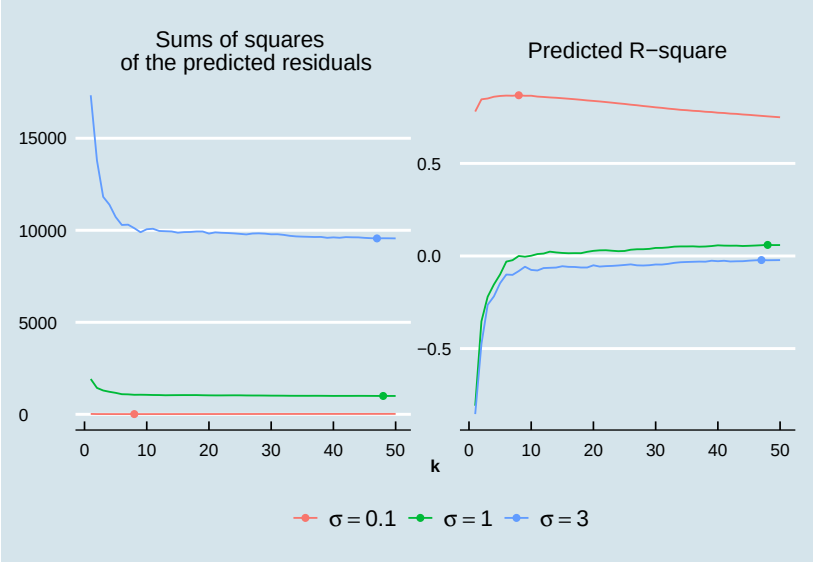

Figure 21: Sums of squares of the predicted residuals and Predicted R-square for different values of k

```
bind_cols(as_tibble(x), as_tibble(y)) %>%
   rename(x = value) %>%
   gather(key=key, value="ys", -x) %>%
   mutate(sigma = gsub("y_", "sigma_", key)) %>%
   dplyr::select(x, sigma, ys, -key) %>%
   group_by(sigma) %>%
   mutate(r_01 = knn.reg(train=x, y=ys, k=8)$pred,
          r_1 = knn.reg(train=x, y=ys, k=48)$pred,
          r_3 = knn.reg(train=x, y=ys, k=47)$pred) %>%
   ungroup() %>%
   gather(key=key, value="rs", -x, -sigma, -ys) %>%
   filter((sigma == "sigma_01" & key == "r_01") |
          (sigma == "sigma_1" & key == "r_1") |
          (sigma == "sigma_3" & key == "r_3")) %>%
   mutate(sigma = if_else(sigma == "sigma_01", "sigma == 0.1",
                  if_else(sigma == "sigma_1", "sigma == 1", "sigma == 3"))) %>%
   dplyr::select(x, sigma, ys, rs, -key) %>%
   ggplot() +
   geom_point(aes(x, ys), col="blue", alpha=0.15) +
   geom_line(aes(x, rs), color="red") +
   facet_wrap(~sigma, scales="free_y", ncol=2, labeller="label_parsed") +
   labs(x="X", y="Y") +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"))
```

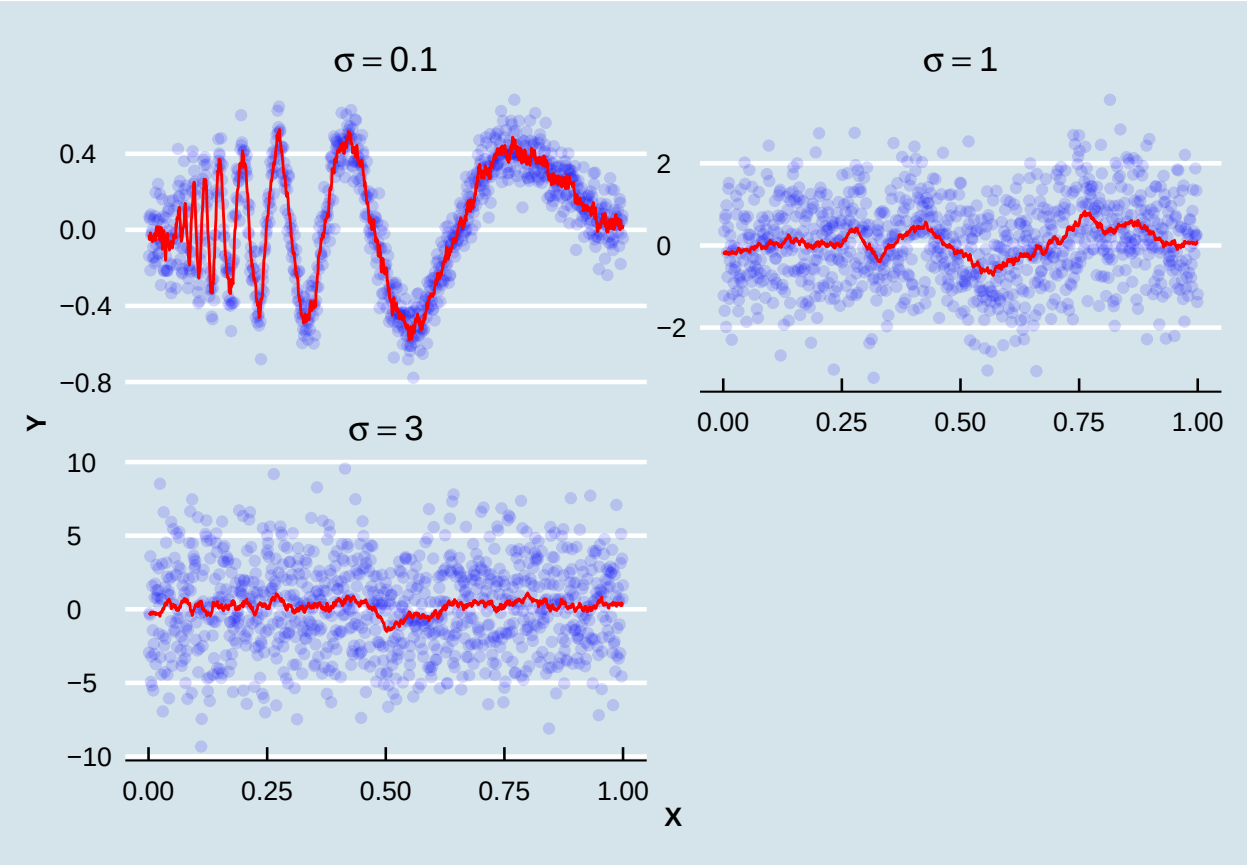

Figure 22: KNN regression of Y given X ($k = 9$)

Kernel Regression

```
h <- 0.01
x_rep <- NULL
indices <- NULL
for (i in 1:length(x)) {
   x_rep <- c(x_rep, rep(x[i], length(x)))
   indices <- c(indices, rep(i, length(x)))
}
denominadores <- cbind(indices, x_rep, x) %>%
   as_tibble() %>%
   rename(x_j = x, x = x_rep, indice = indices) %>%
   mutate(kernel = dnorm(x=((x - x_j)/h))) %>%
   group_by(indice) %>%
   summarise(denom = sum(kernel))
elles <- dnorm(as.matrix(dist(x)) / h) * (1/as.numeric(denominadores$denom))
p1 <- tibble(x=x, y=y_01) %>%
```

```
   mutate(r_n = as.numeric(elles %*% y_01)) %>%
   ggplot() +
   geom_point(aes(x, y), col="blue", alpha=0.15) +
   geom_line(aes(x, r_n), col="red") +
   labs(x=NULL, y="Cl", title=expression(sigma == 0.01)) +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"),
         plot.title=element_text(hjust=0.5))
p2 <- tibble(x=x, y=y_1) %>%
   mutate(r_n = as.numeric(elles %*% y_1)) %>%
   ggplot() +
   geom_point(aes(x, y), col="blue", alpha=0.15) +
   geom_line(aes(x, r_n), col="red") +
   labs(x="ell", y=NULL, title=expression(sigma == 1)) +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"),
         plot.title=element_text(hjust=0.5))
p3 <- tibble(x=x, y=y_3) %>%
   mutate(r_n = as.numeric(elles %*% y_3)) %>%
   ggplot() +
   geom_point(aes(x, y), col="blue", alpha=0.15) +
   geom_line(aes(x, r_n), col="red") +
   labs(x="ell", y="Cl", title=expression(sigma == 3)) +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"),
         plot.title=element_text(hjust=0.5))
grid.arrange(p1, p2, p3, ncol = 2)
```

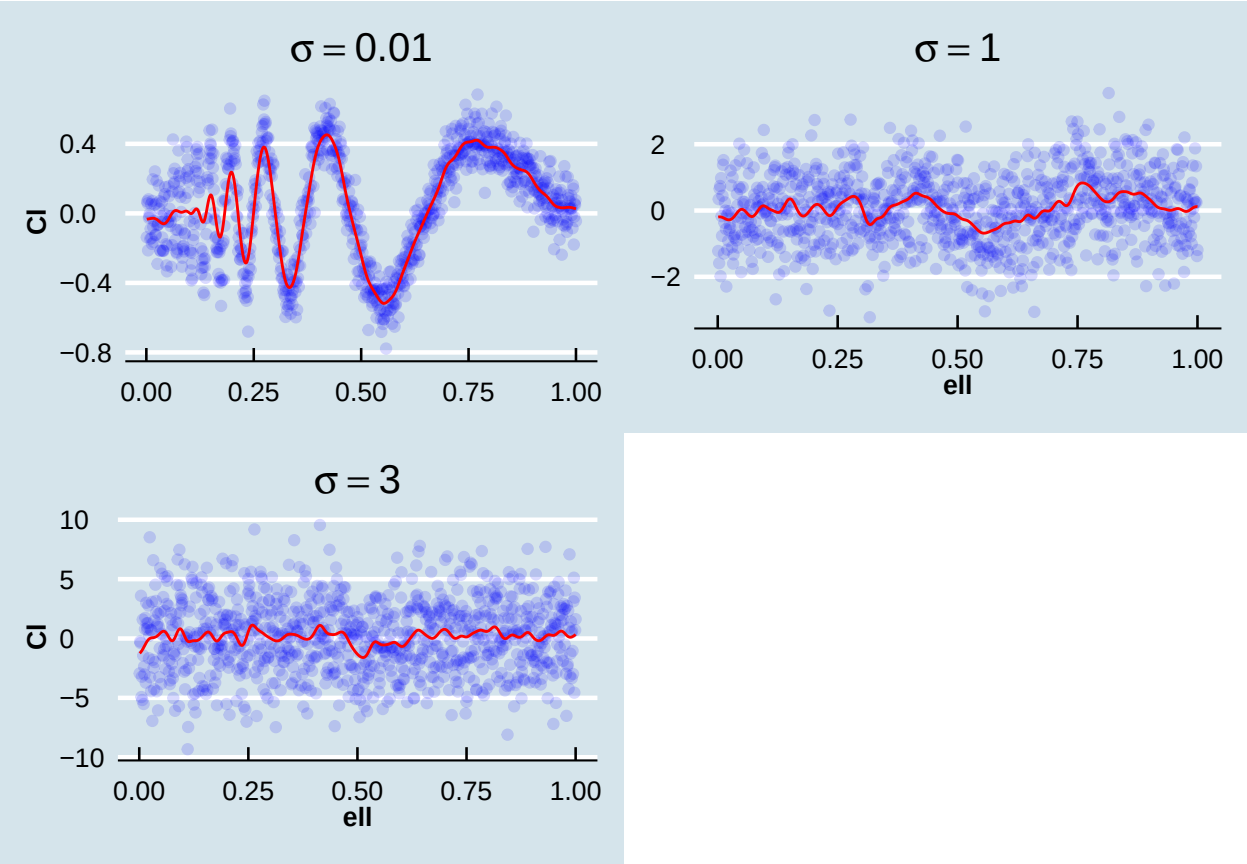

Figure 23: Nadayara-Watson estimator of $r(x)$ using Gaussian kernel

Local polynomial regression

```
local_poly <- loess(y_01 ~ x, tibble(x, y_01), degree=1, family="gaussian", span=0.05)
p1 <- tibble(x, y_01, r_n=local_poly$fitted) %>%
   ggplot() +
   geom_point(aes(x, y_01), color="blue", alpha=0.15) +
   geom_line(aes(x, r_n), col="red") +
   labs(x='ell', y='Cl', title = expression(sigma == 0.1)) +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"),
         plot.title = element_text(hjust=0.5))
local_poly <- loess(y_1 ~ x, tibble(x, y_1), degree=1, family="gaussian", span=0.05)
p2 <- tibble(x, y_1, r_n=local_poly$fitted) %>%
   ggplot() +
   geom_point(aes(x, y_1), color="blue", alpha=0.15) +
   geom_line(aes(x, r_n), col="red") +
   labs(x='ell', y='Cl', title = expression(sigma == 1)) +
   ggthemes::theme_economist() +
```

```
   theme(axis.title=element_text(face="bold"),
         plot.title = element_text(hjust=0.5))
local_poly <- loess(y_3 ~ x, tibble(x, y_3), degree=1, family="gaussian", span=0.05)
p3 <- tibble(x, y_3, r_n=local_poly$fitted) %>%
   ggplot() +
   geom_point(aes(x, y_3), color="blue", alpha=0.15) +
   geom_line(aes(x, r_n), col="red") +
   labs(x='ell', y='Cl', title = expression(sigma == 3)) +
   ggthemes::theme_economist() +
   theme(axis.title=element_text(face="bold"),
         plot.title = element_text(hjust=0.5))
grid.arrange(p1, p2, p3, ncol=2)
```

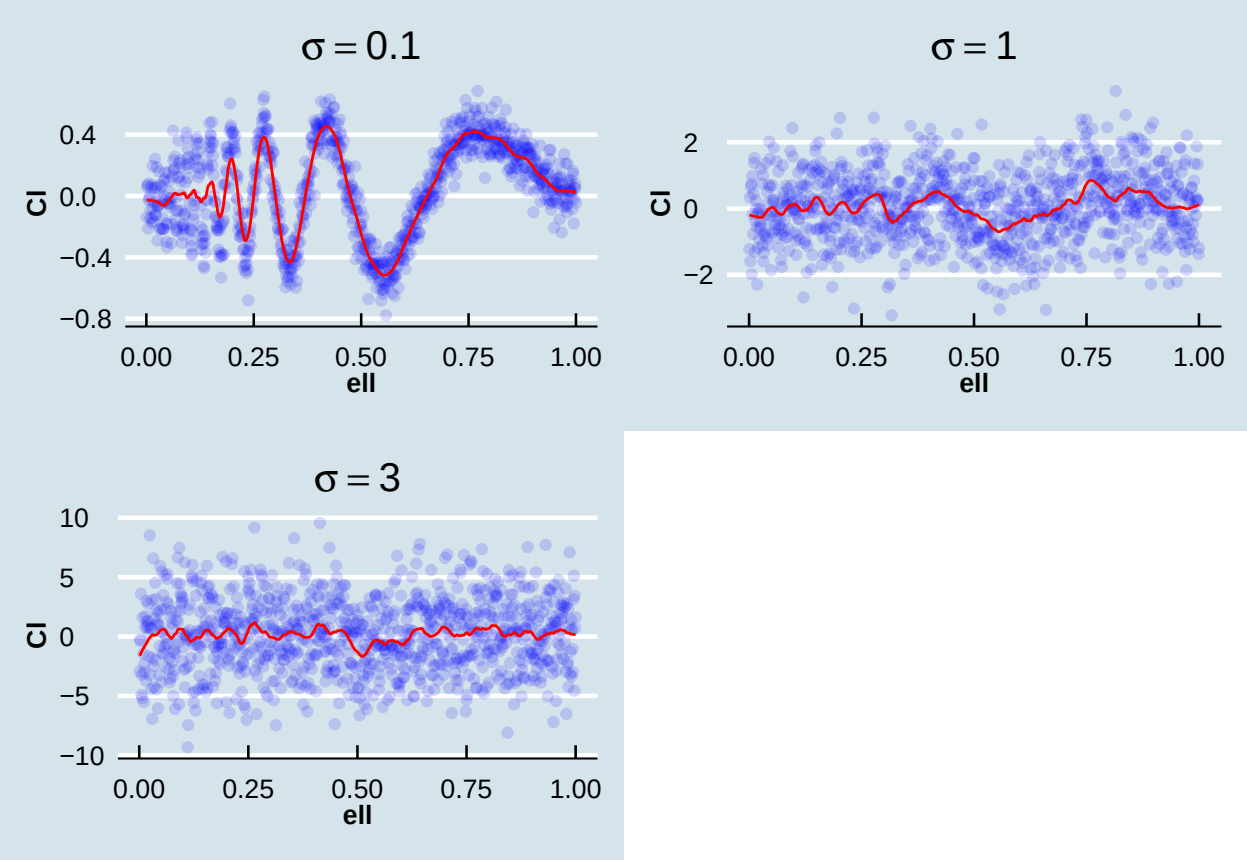

Figure 24: Local polynomial regression of Cl given ell (Gaussian family, h=0.05, degree=1)

References

Beygelzimer, Alina, Sham Kakadet, John Langford, Sunil Arya, David Mount, and Shengqiao Li. 2018. *FNN: Fast Nearest Neighbor Search Algorithms and Applications*. https://CRAN.R-project.org/package=FNN.

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An Introduction to Statistical Learning*. Vol. 112. Springer.

R Core Team. 2018. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Scott, David W. 2015. *Multivariate Density Estimation: Theory, Practice, and Visualization*. John Wiley & Sons.

Wasserman, Larry. 2007. *All of Nonparametric Statistics*. Springer, New York.

Wickham, Hadley. 2017. *Tidyverse: Easily Install and Load the 'Tidyverse'*. https://CRAN.R-project.org/package=tidyverse.