Local linear regression

Group 1: Pariente Antonio, Bosch Guillem, Ebner Lena

14/11/2023

Estimating the conditional variance by local linear regression

---

Aircraft Data

We are using `Aircraft data`, from the R library `sm`. These data record six characteristics of aircraft designs which appeared during the twentieth century.

- **Yr**: year of first manufacture
- **Period**: a code to indicate one of three broad time periods
- **Power**: total engine power (kW)
- **Span**: wing span (m)
- **Length**: length (m)
- **Weight**: maximum take-off weight (kg)
- **Speed**: maximum speed (km/h)
- **Range**: range (km)

We transform data taken logs (except `Yr` and `Period`): `lgPower`,…, `lgRange`. Go to R and charge the library `sm`:

```
library(sm)
```

```
## Package 'sm', version 2.2-5.7: type help(sm) for summary information
```

Now upload the data:

```
data(aircraft)
# help(aircraft)
attach(aircraft)
lgPower <- log(Power)
lgSpan <- log(Span)
lgLength <- log(Length)
lgWeight <- log(Weight)
lgSpeed <- log(Speed)
lgRange <- log(Range)
```

Estimating the conditional variance

Consider the heteroscedastic regression model

$$Y = m(x) + \sigma(x)\varepsilon = m(x) + \epsilon$$

,

where $E(\varepsilon) = 0$, $V(\varepsilon) = 1$ and $\sigma^2(x)$ is an unknown function that gives the conditional variance of Y given that the explanatory variable is equal to x. Let us define $Z = \log((Y - m(x))^2) = \log \epsilon^2$ and $\delta = \log(\varepsilon^2)$. Then

$$Z = \log \sigma^2(x) + \delta$$

,

and $\delta = \log \varepsilon^2$ is a random variable with expected value close to 0 (observe that $E(\log \varepsilon^2) \approx \log E(\varepsilon^2) = \log V(\varepsilon) = \log 1 = 0$) taking the role of *noise* in the regression of Z against x (that is, Z is the response variable and x is the predicting variable). Given that the values of ϵ_i^2 are not observable, a way to estimate the function $\sigma^2(x)$ is as follows

1. Fit a nonparametric regression to data (xi, yi) and save the estimated values $\hat{m}(x_i)$.
2. Transform the estimated residuals $\hat{\epsilon}_i = y_i - \hat{m}(x_i)$:

$$z_i = \log \epsilon_i^2 = \log((y_i - \hat{m}(x_i))^2)$$

.

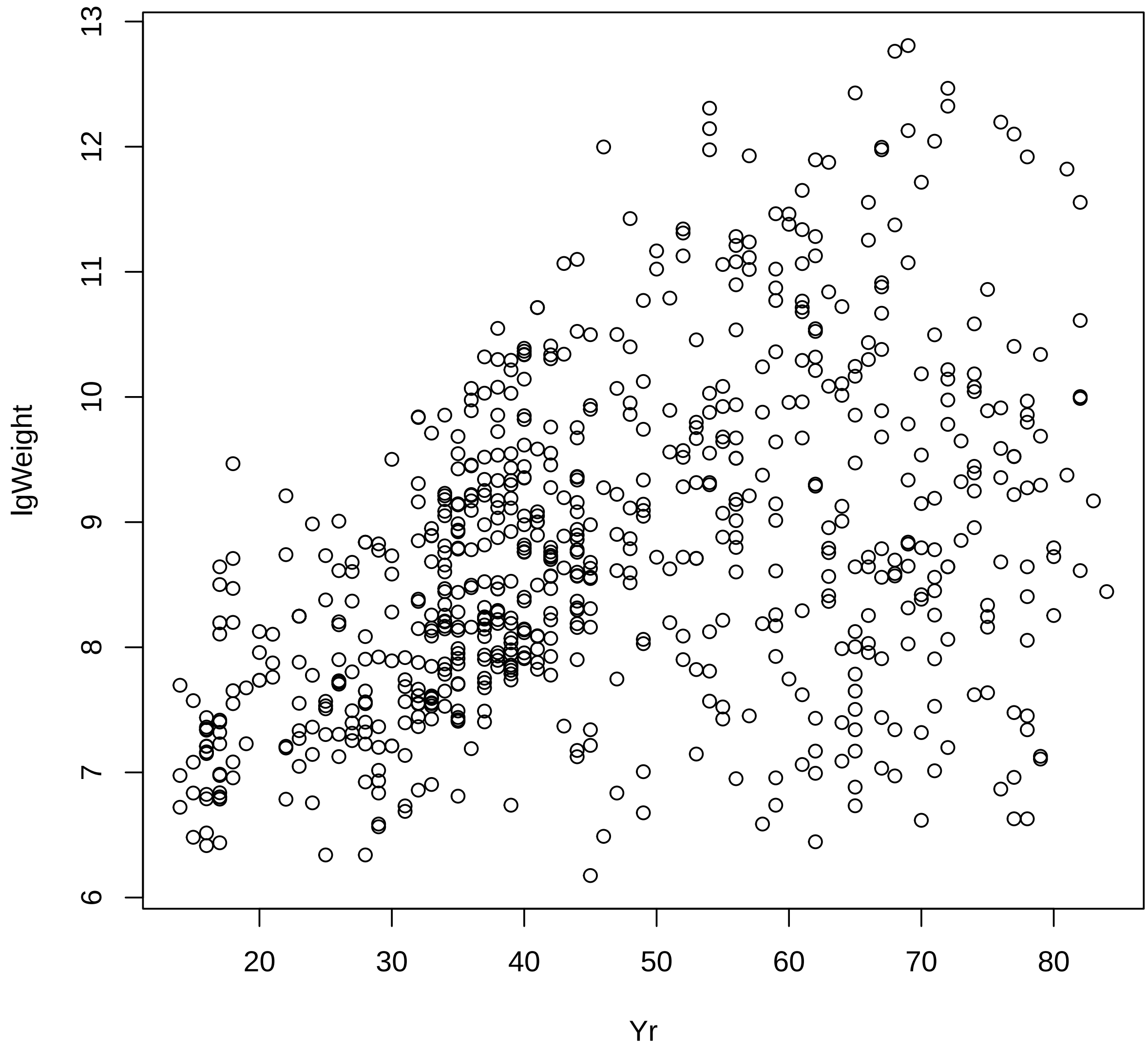
3. Fit a nonparametric regression to data (xi, zi) and call the estimated function $\hat{q}(x)$. Observe that $\hat{q}(x)$ is an estimate of $\log \sigma^2(x)$.
4. Estimate $\sigma^2(x)$ by $\hat{\sigma}^2(x) = e^{\hat{q}(x)}$.

Apply this procedure to estimate the conditional variance of `lgWeigth` (variable Y) given `Yr` (variable x). Draw a graphic of $\hat{\epsilon}_i^2$ against x_i and superimpose the estimated function $\hat{\sigma}^2(x)$. Lastly draw the function $\hat{m}(x)$ and superimpose the bands $\hat{m}(x) \pm 1,96\hat{\sigma}(x)$.

Attention: Do the work twice: - First, use the function `loc.pol.reg` that you can find in ATENEA and choose all the bandwidth values you need by leave-one-out cross-validation (you have not to program it again! Just look for the right function in the `*.Rmd` files you can find in ATENEA) - Second, use the function `sm.regression` from library `sm` and choose all the bandwidth values you need by *direct plug-in* (use the function `dpill` from the same library `KernSmooth`).

Implementation

```
plot(Yr,lgWeight)
```



1. loc.pol.reg & LOOCV

Using the function `locpolreg`.

```
source("./locpolreg.R")
```

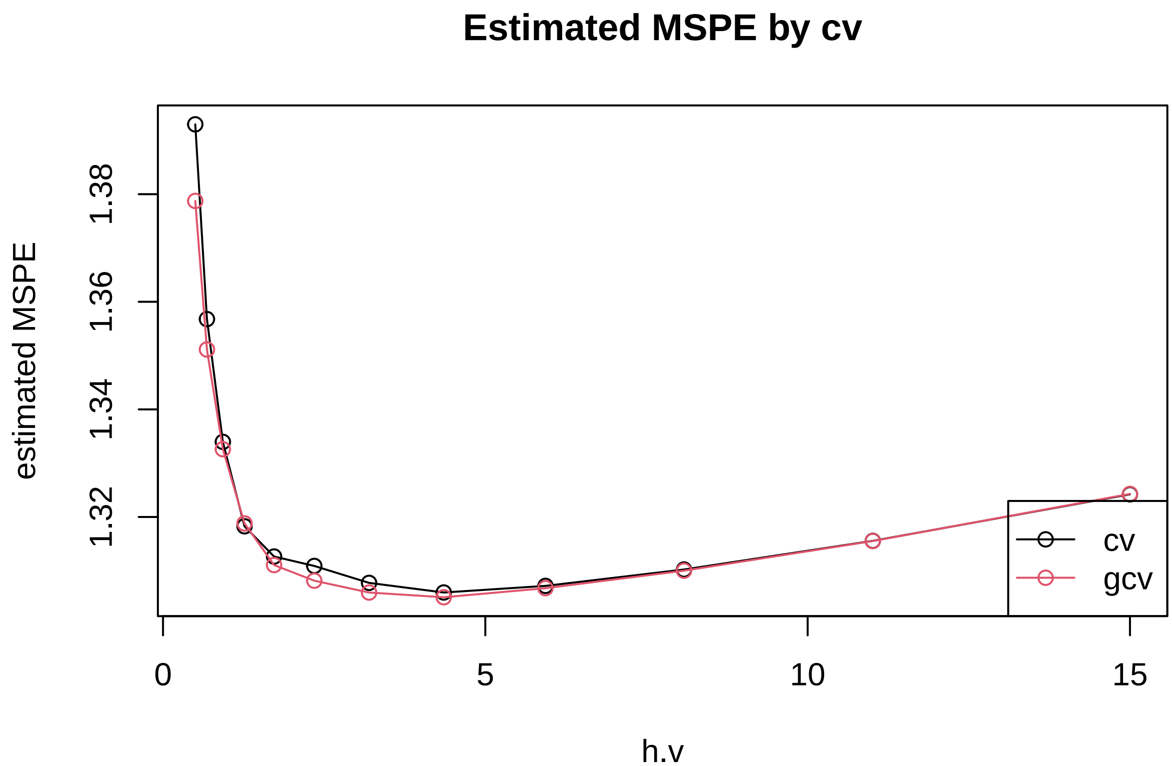
Ordinary and Generalized Cross-Validation Function

```
h.cv.gcv <- function(x,y,h.v = exp(seq(log(diff(range(x))/20),
                                       log(diff(range(x))/4),l=10)),
                     p=1,type.kernel="normal"){
  n <- length(x)
```

```
  cv <- h.v*0
  gcv <- h.v*0
  for (i in (1:length(h.v))){
    h <- h.v[i]
    aux <- locpolreg(x=x,y=y,h=h,q=p,tg=x,
                     type.kernel=type.kernel, doing.plot=FALSE)
    S <- aux$S
    h.y <- aux$mtgr
    hii <- diag(S)
    av.hii <- mean(hii)
    cv[i] <- sum(((y-h.y)/(1-hii))^2)/n
    gcv[i] <- sum(((y-h.y)/(1-av.hii))^2)/n
  }
  return(list(h.v=h.v,cv=cv,gcv=gcv))
}
```

Leave-One-Out Cross Validation

```
h.v <-  exp(seq(from=log(.5), to = log(15), length=12))

out.cv.gcv <- h.cv.gcv(x=Yr, y=lgWeight, h.v=h.v)

y.max <- max(c(out.cv.gcv$cv,out.cv.gcv$gcv))
y.min <- min(c(out.cv.gcv$cv,out.cv.gcv$gcv))

plot(h.v,out.cv.gcv$cv,ylab="estimated MSPE",ylim=c(y.min,y.max), main="Estimated MSPE by cv")

lines(h.v,out.cv.gcv$cv)
points(h.v,out.cv.gcv$gcv,col=2)

lines(h.v,out.cv.gcv$gcv,col=2)
legend("bottomright",c("cv","gcv"), col=1:4,lty=1,pch=1)
```

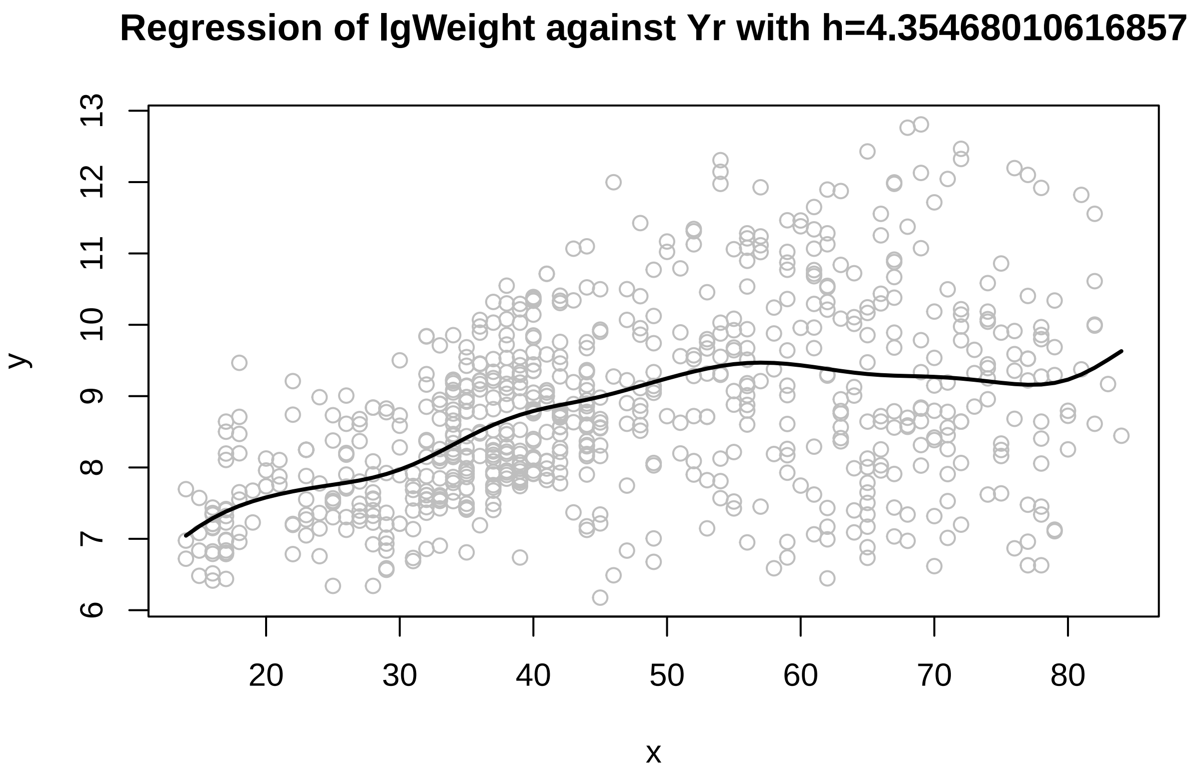



```
opt.h.cv <- h.v[which.min(out.cv.gcv$cv)]

print(paste0("Best h choosen by LOOCV: ", opt.h.cv))
```

```
## [1] "Best h choosen by LOOCV: 4.35468010616857"
```

Fit nonparametric regression $\hat{m}$ with optimal h of `lgWeight` against `Yr`

```
m_hat <- locpolreg(x=Yr,y=lgWeight,h=opt.h.cv,q=1,r=0,main=paste0("Regression of lgWeight against Yr wi
```

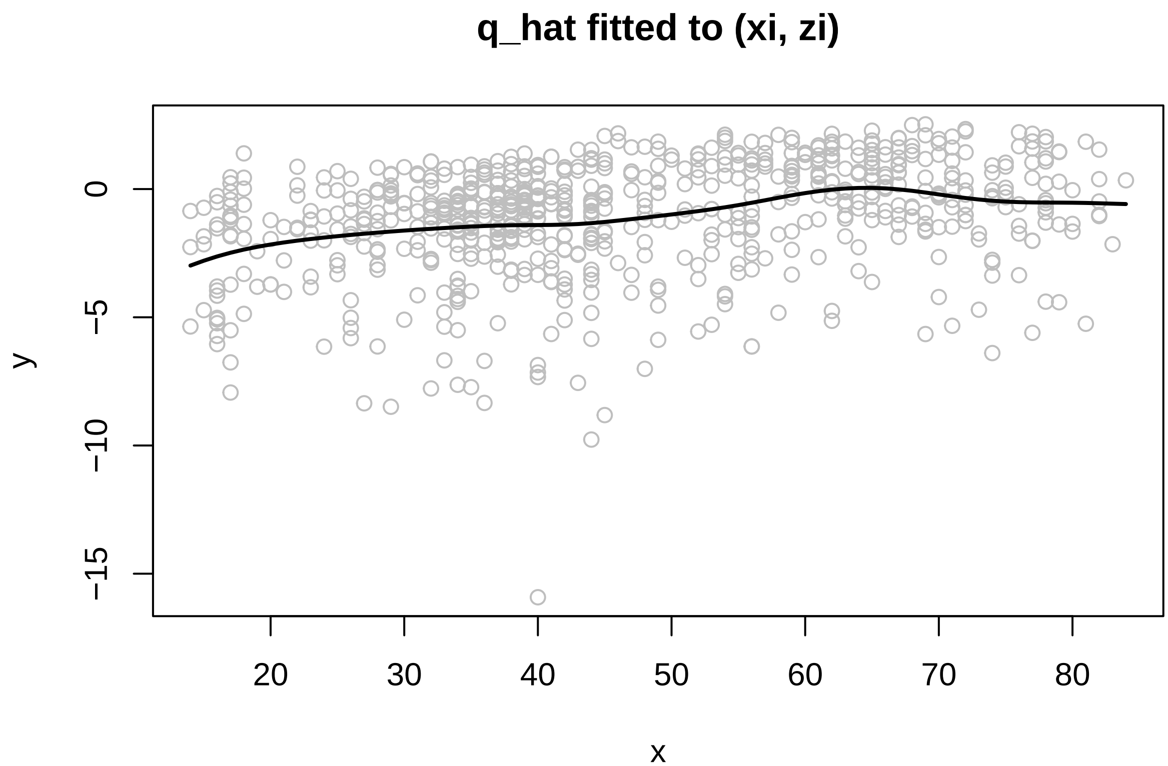


Transform estimated residuals to z_i

```
residuals <- lgWeight - m_hat$mtgr
squared_residuals <- residuals^2
zi <- log(squared_residuals)
```

Fit nonparametric regression $\hat{q}$ to data (x_i, z_i)

```
q_hat <- locpolreg(x=Yr,y=zi,h=opt.h.cv,q=1,r=0,main="q_hat fitted to (xi, zi)")
```

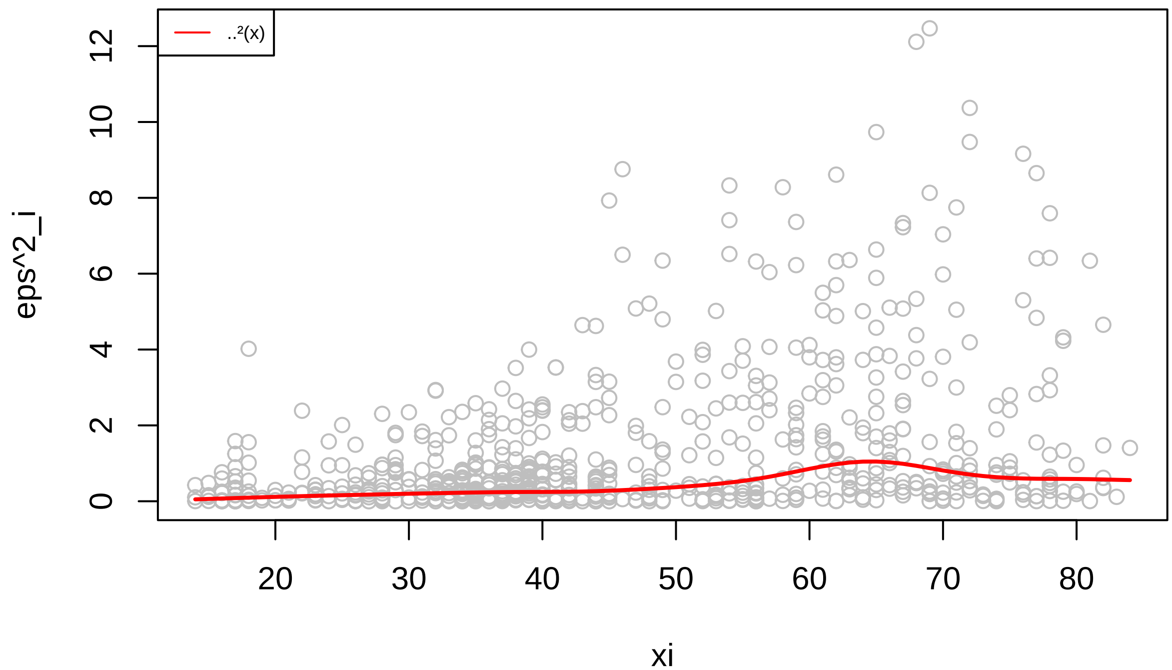


Estimate conditional variance $\hat{\sigma}^2$ of `lgWeight` $\sigma^2(x) = e^{\hat{q}(x)}$

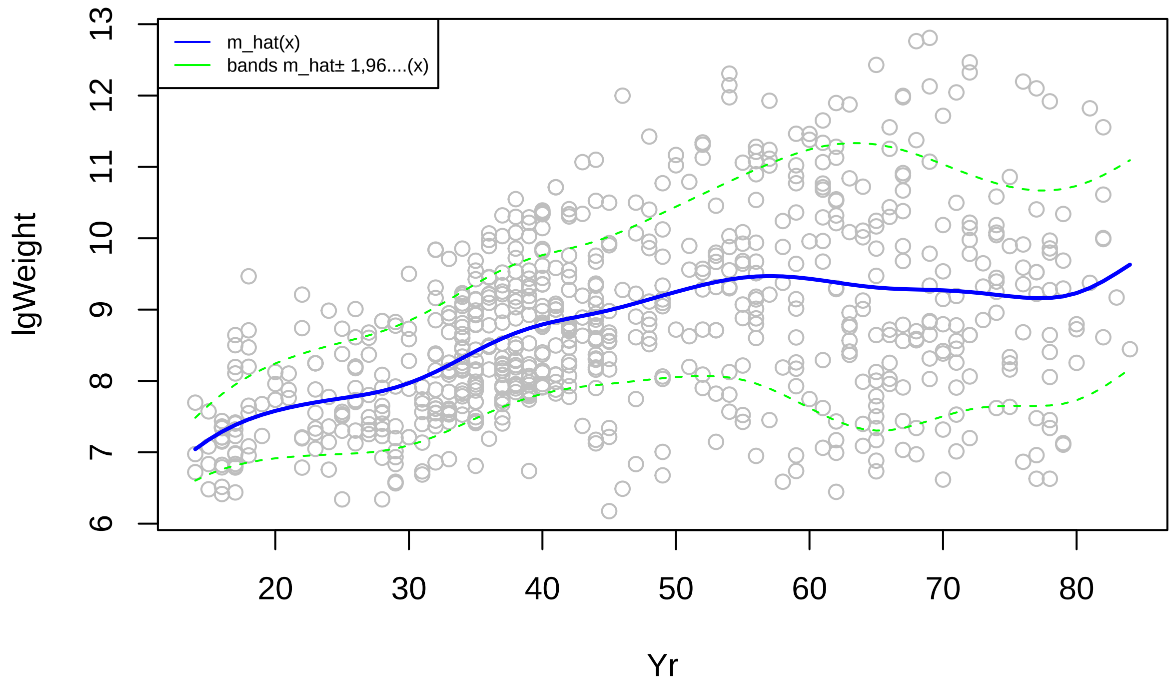
```
sigma_hat_squared <- exp(q_hat$mtgr)
```

Plot ϵ_i^2 against x_i with $\sigma^2(x)$, $\hat{m}(x)$ and bands $\hat{m} \pm 1,96\hat{\sigma}(x)$‘.

```
plot(Yr, squared_residuals, col = "grey", xlab = "xi", ylab = "eps^2_i")

lines(Yr, sigma_hat_squared, col = "red", lwd = 2)

legend("topleft",
       c(" ²(x)"),
       lty=c(1),
       col=c("red"),cex=.6)
```




```
# Calculate m(x) ± 1.96*sigma(x)
upper_band <- m_hat$mtgr + 1.96 * sqrt(sigma_hat_squared)
lower_band <- m_hat$mtgr - 1.96 * sqrt(sigma_hat_squared)

plot(Yr, lgWeight, col = "grey", xlab = "Yr", ylab = "lgWeight")
lines(Yr, m_hat$mtgr, col = "blue", lwd = 2)

# bands
lines(Yr, upper_band, col = "green", lty = 2)
lines(Yr, lower_band, col = "green", lty = 2)

legend("topleft",
       c("m_hat(x)", "bands m_hat± 1,96^ (x)"),
       lty=c(1,1),
       col=c("blue", "green"),cex=.6)
```



2. sm.regression & direct plug-in

Using direct plugin `dpill` for choice of h

```
require(KernSmooth) # for function "dpill"
```

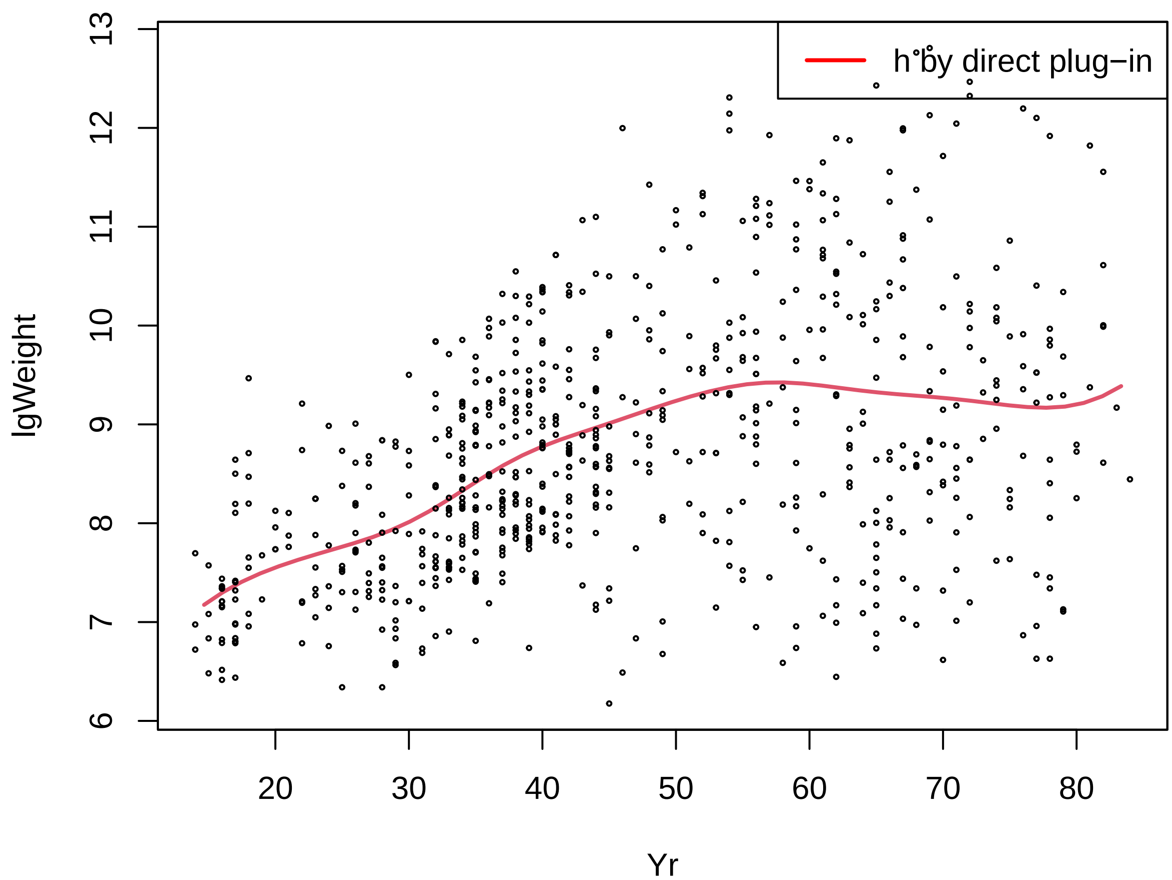
```
## Loading required package: KernSmooth
```

```
## KernSmooth 2.23 loaded
## Copyright M. P. Wand 1997-2009
```

```
h.dpi <- dpill(x=Yr, y=lgWeight,gridsize=101,range.x=range(Yr))
```

Nonparametric regression fit of `Yr` against `lgWeight`

```
m_hat.sm <- sm.regression(x=Yr,y=lgWeight,h=h.dpi,col=2,lwd=2)
m_hat.sm_all_estimates <- sm.regression(x=Yr,y=lgWeight,h=h.dpi,col=2,lwd=2, eval.points = Yr, display=
legend("topright",c("h by direct plug-in"),col=c("red"),lty=1,lwd=2)
```

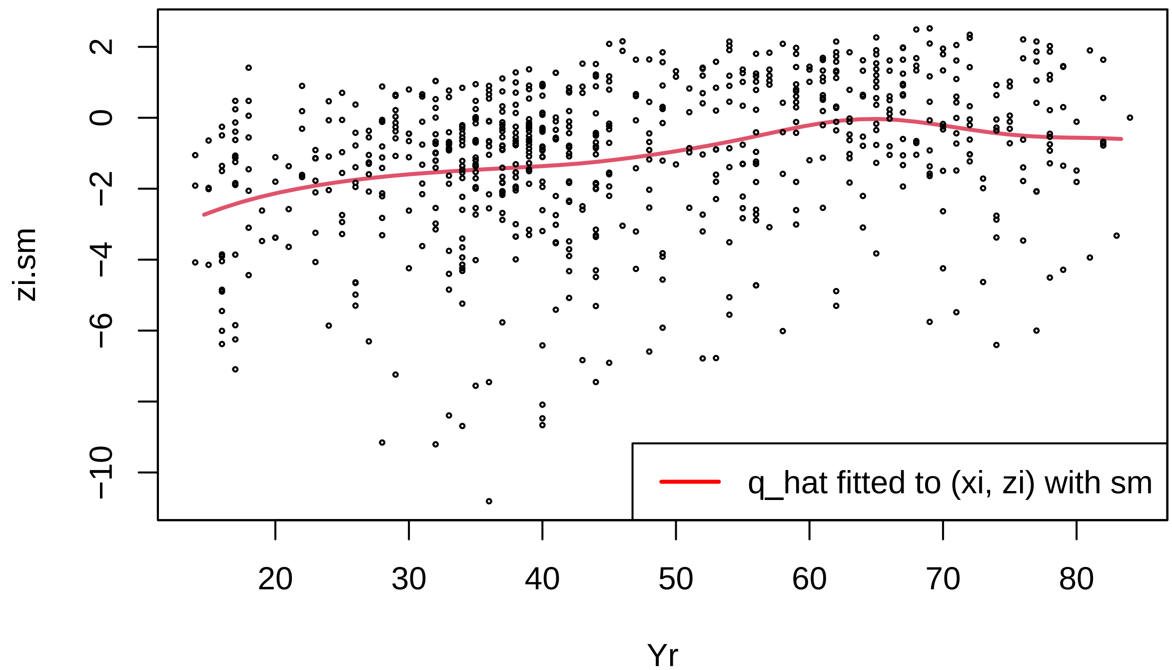


Transform estimated residuals to z_i

```
residuals.sm <- lgWeight - m_hat.sm_all_estimates$estimate
squared_residuals.sm <- residuals.sm^2
zi.sm <- log(squared_residuals.sm)
```

Fit nonparametric regression $\hat{q}$ to data (x_i, z_i)

```
q_hat.sm <- sm.regression(x=Yr,y=zi.sm,h=h.dpi,col=2,lwd=2)
q_hat.sm_all_estimates <- sm.regression(x=Yr,y=zi.sm,h=h.dpi,col=2,lwd=2, eval.points=Yr, display="none

legend("bottomright",c("q_hat fitted to (xi, zi) with sm"),col=c("red"),lty=1,lwd=2)
```

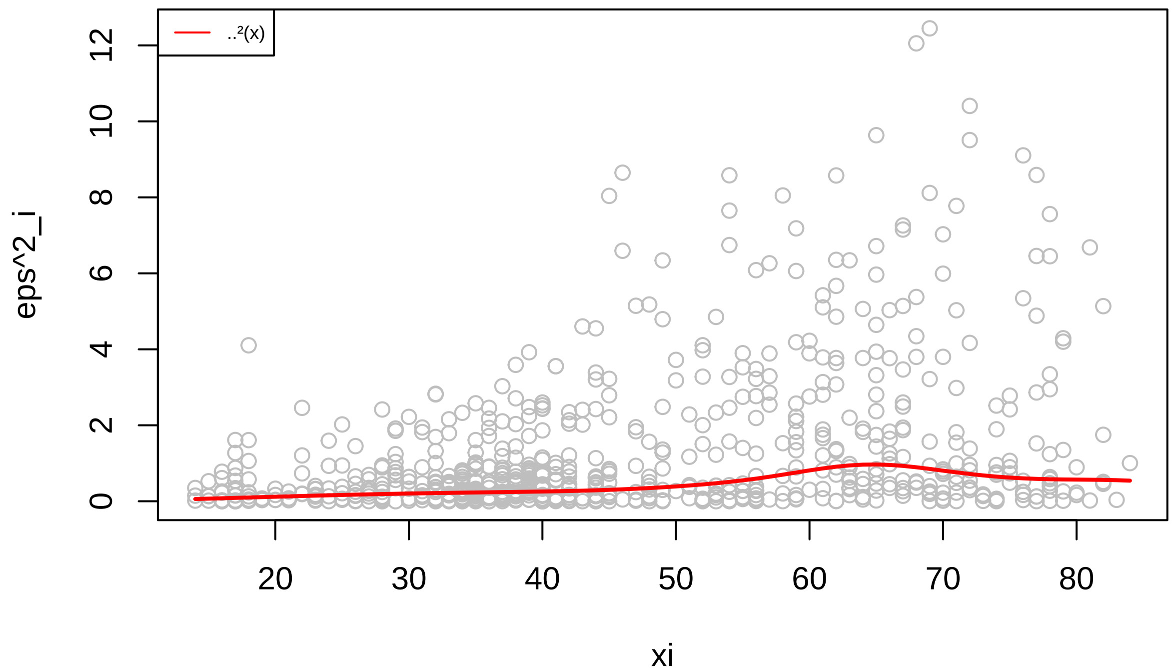


Estimate conditional variance $\sigma^2(x)$ of `lgWeight`

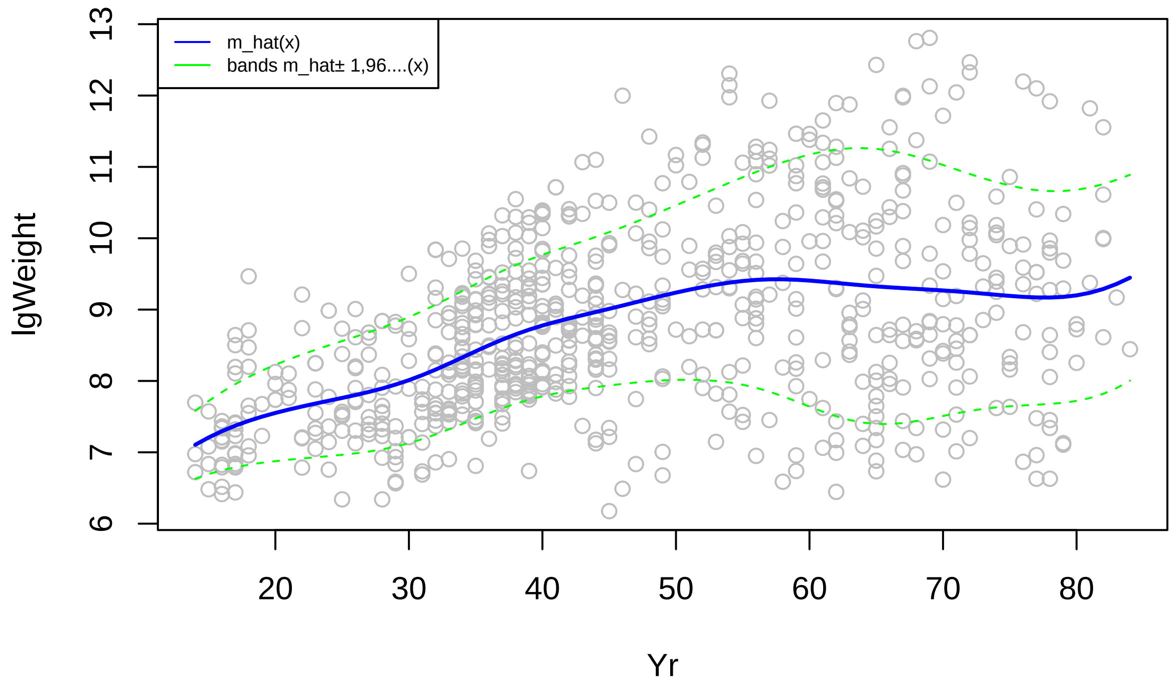
```
sigma_hat_squared.sm <- exp(q_hat.sm_all_estimates$estimate)
```

Plot ϵ_i^2 against x_i with $\sigma^2(x)$, $\hat{m}(x)$ and bands $\hat{m} \pm 1,96\hat{\sigma}(x)$.

```
plot(Yr, squared_residuals.sm, col = "grey", xlab = "xi", ylab = "eps^2_i")

lines(Yr, sigma_hat_squared.sm, col = "red", lwd = 2)

legend("topleft",
       c(" ²(x)"),
       lty=c(1),
       col=c("red"),cex=.6)
```




```r
# Calculate m(x) ± 1.96*sigma(x)
upper_band.sm <- m_hat.sm_all_estimates$estimate + (1.96 * sqrt(sigma_hat_squared.sm))
lower_band.sm <- m_hat.sm_all_estimates$estimate - (1.96 * sqrt(sigma_hat_squared.sm))

plot(Yr, lgWeight, col = "grey", xlab = "Yr", ylab = "lgWeight")
lines(Yr, m_hat.sm_all_estimates$estimate, col = "blue", lwd = 2)

# bands
lines(Yr, upper_band.sm, col = "green", lty = 2)
lines(Yr, lower_band.sm, col = "green", lty = 2)

legend("topleft",
       c("m_hat(x)", "bands m_hat± 1,96^ (x)"),
       lty=c(1,1),
       col=c("blue", "green"), cex=.6)
```


m_hat(x)
bands m_hat± 1,96....(x)
lgWeight
Yr
6
7
8
9
10
11
12
13
20
30
40
50
60
70
80