Assignment 4: Local Linear Regression

Ian Wallgren, Wanchang Zhang, Lavinia Hriscu, Victor Jimenez

In this assignment we will use non-parametric regression to study the relation between some variables in the aircraft dataset. For each relation, we should provide an estimator $\hat{m}(t)$ of the regression function $m(t)$, $\forall t \in \mathbb{R}$ and estimate the residual variance σ^2 as well.

In order to provide an estimation at a given value t, local polynomial regression uses data (x_i, y_i) such that x_i is in an interval centered at t, and fits a polynomial model for the interval minimizing the squared error. In order to obtain a smooth result, a weights function $w(x_i, t)$ usually symmetric and unimodal is applied to each datapoint. For instance, using a kernel function K, the weight of (x_i, y_i) when estimating $m(t)$ is:

$$w_i = w(t, x_i) = \frac{K\left(\frac{x_i - t}{h}\right)}{\sum_{j=1}^{n}\left(\frac{x_j - t}{h}\right)}$$

where h is the smoothing parameter or bandwidth. Intuitively, the bandwidth value is associated with the locality of the regression, since the larger the value, the higher the weight of more distant points for the estimation of the interval centered at x_i. The final estimate is significantly affected by changes in the choice of h, making it a crucial task in nonparametric estimation.

Finally, once the weights w_i have been calculated, the following weighted least squares problem is solved:

$$\min_{a,b \in \mathbb{R}} \sum_{i=1}^{n} w_i [y_i - (a + b(x_i - t))]^2$$

Since the weights depend on t, the optimal parameters a and b do as well. The regression model around t and the subsequent estimated value at $x = t$, for a local linear model, is the following:

$$l_t(x) = a(t) + b(t)(x - t) \Longrightarrow \hat{m}(t) = l_t(t) = a(t)$$

We will implement a local linear model on a dataset containing six characteristics of aircraft designs which appeared during the twentieth century. In particular, we will provide an estimation of the conditional variance of `lgWeight` (the logarithm of `Weight`) given `Yr`, the explanatory variable that represents the year.

We will transform the data taking the logarithm of the variables:

```
data(aircraft)
lgWeight = log(aircraft$Weight)
Yr = aircraft$Yr
```

Estimating the conditional variance

We will work with the following heteroscedastic regression model:

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

where $\mathbb{E}(\varepsilon) = 0$ and $\mathbb{V}(\varepsilon) = 1$ and $\sigma^2(x)$ is an unknown function that gives the conditional variance of Y. If we apply the transformations $Z = \log[(Y - m(x))^2] = \log(\varepsilon^2)$ and $\delta = log(\varepsilon^2)$ then:

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

with δ being a random variable with expected value close to zero, as $\mathbb{E}(\log(\varepsilon^2))) \approx \log \mathbb{E}(\varepsilon^2) = \log \mathbb{V}(\varepsilon) = 0$.

Given that the values of ϵ_i^2 are not observable, we will estimate the function $\sigma^2(x)$ by fitting a non-parametric regression to the data (x_i, y_i), applying the previous transformation, and then fitting again to (x_i, z_i) such that an estimation of $\log \sigma^2(x)$ can be obtained. We will perform this procedure twice, first by means of the function `loc.pol.reg`, which has been provided by the professor and allows for the choice of bandwidth using LOO-CV, and then by means of the function `sm.regression` of the library `sm`.

1. Manual local regression

We will start by loading the `loc.pol.reg` function that performs the fit, whose most significant arguments are the following:

- `x` and `y`: the observed data (in form of $n \times 1$ vectors)
- `h`: the smoothing parameter. By default we will take the value of the first quantile of the distance between x values.
- `q`: the degree of the local polynomial to be fitted (default `q=1`).
- `tg`: grid of values t where the estimated regression function is evaluated. By default, `tg=x`.
- `type.kernel`: "normal" (default), "epan", "rs.epan" or "unif".

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

1.1 Fitting the model to (x_i, y_i)

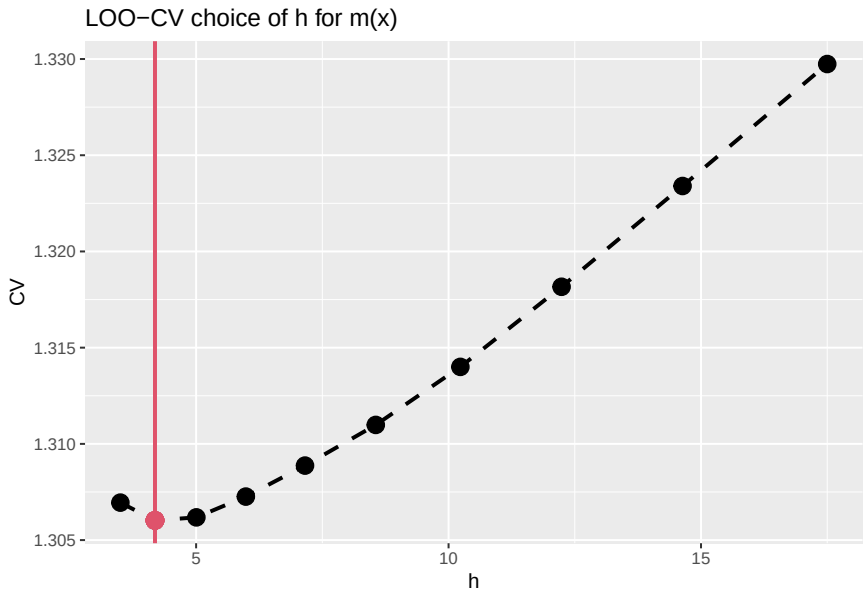
We will use `locpolreg` with bandwidth selection given by LOO-CV (that is, k-fold CV with $k = n$). The function that performs the bandwidth choice can be defined as follows:

```
h.k.fold.cv <- function(x,y,h.v = exp(seq(log(diff(range(x))/20),
                                          log(diff(range(x))/4),l=10)),
                        k=10,p=1,type.kernel="normal"){
  n <- length(x)
  perm <- sample(1:n)
  xperm <- x[perm]
  yperm <- y[perm]

  k.cv <- h.v*0
  for (i in (1:length(h.v))){
    h <- h.v[i]
    k.cv[i] <- k.fold.cv(x=xperm,y=yperm,k=k,h=h,p=p,
                         type.kernel=type.kernel)
  }
  return(list(k=k,h.v=h.v,k.cv=k.cv))
}
```

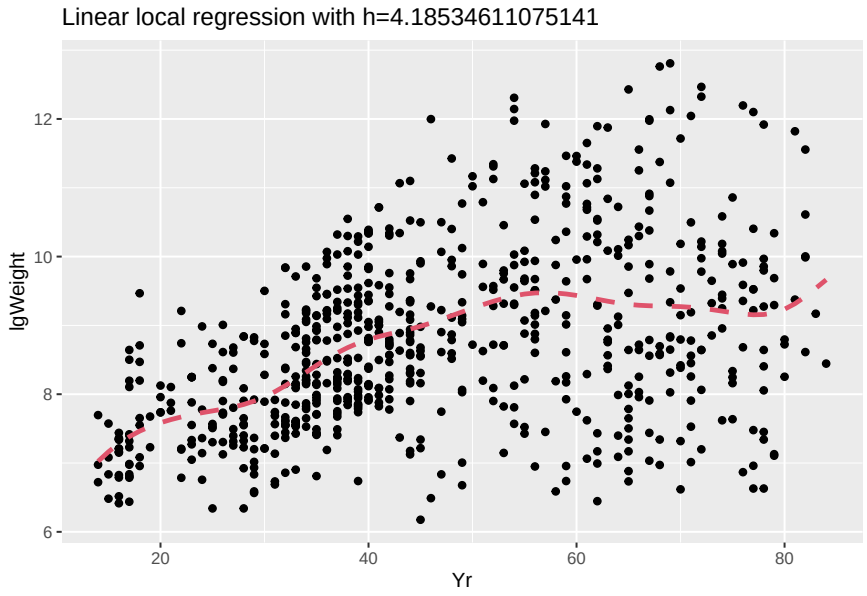
This function provides, according to the range of the explanatory variable `Yr`, the values of h considered `h.v` and the cross-validation error `k.cv`. We can plot these two to obtain the optimum value.

```
h.LOO.xy = h.k.fold.cv(Yr, lgWeight, k=length(lgWeight))
```



We will use the minimum value of the sequence `h.v`:

```
h.LOO.xy.min = h.LOO.xy$h.v[which.min(h.LOO.xy$k.cv)]
m.xy.1 = locpolreg(Yr, lgWeight, h=h.LOO.xy.min, doing.plot=FALSE)
```



1.2 Transforming the estimated residuals

For each point, we will calculate the estimated residuals $\hat{\epsilon}_i = y_i - \hat{m}(x_i)$ and transform them according to the previous relation to obtain z_i. Since we chose $t = x$, we can simply compute the subtraction with the output of the regression function:

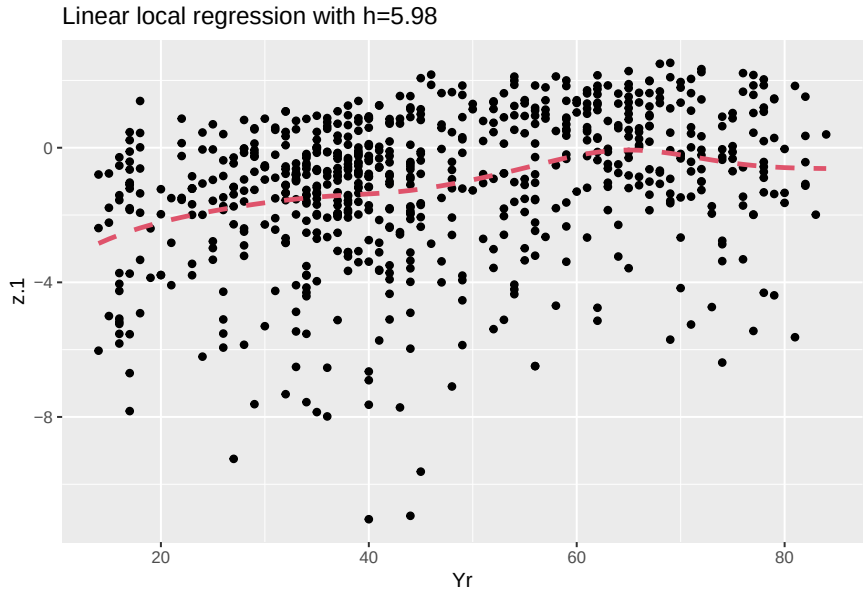
```
eps.1 = lgWeight - m.xy.1$mtg
z.1 = log(eps.1^2)
```

1.3 Fitting the model to (x_i, z_i)

Now we can obtain an estiamate of $\log \sigma^2(x)$ by applying the linear regression model to the transformated values. Again, we will use LOO-CV to select the best bandwidth value.

```
h.LOO.xz = h.k.fold.cv(Yr, z.1, k=length(lgWeight))
```

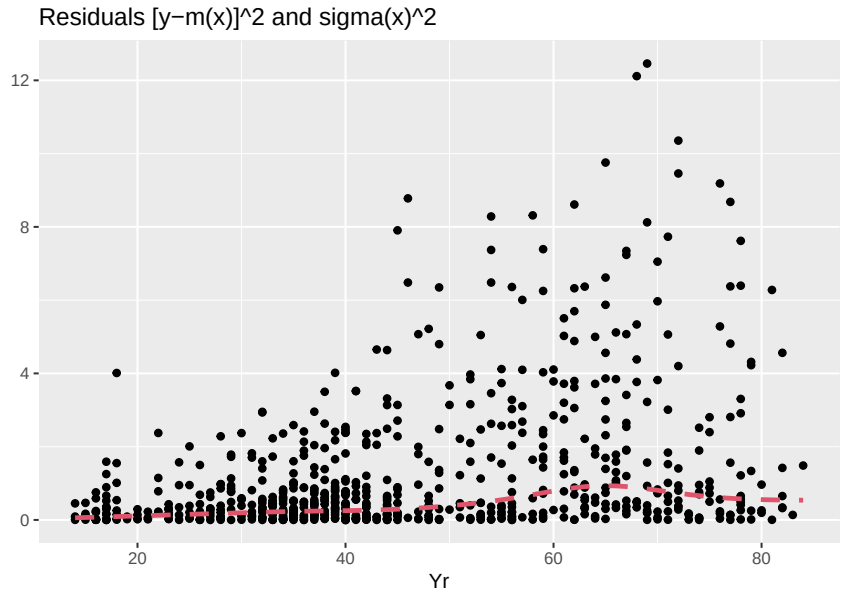

```
h.LOO.xz.min = h.LOO.xz$h.v[which.min(h.LOO.xz$k.cv)]
m.xz.1 = locpolreg(Yr,z.1,h=h.LOO.xz.min, doing.plot=FALSE)
```

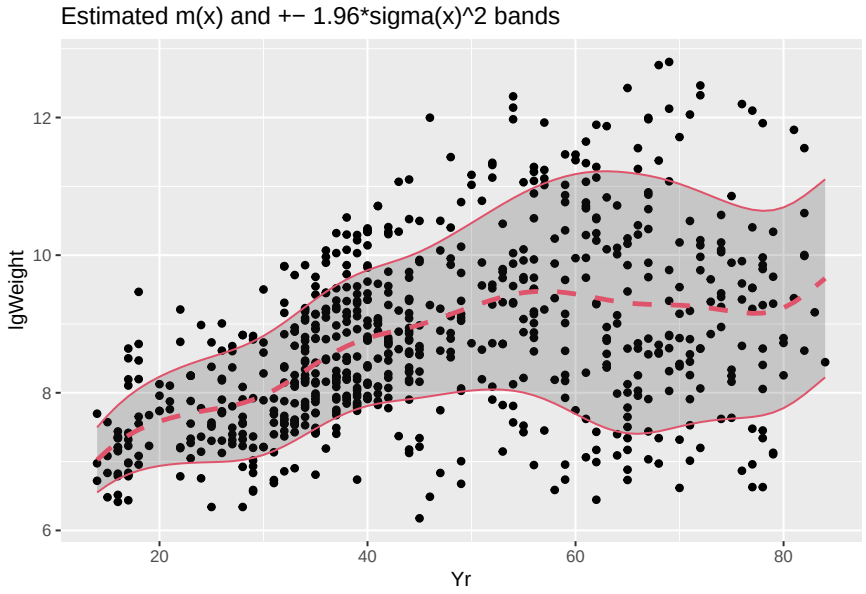


1.4 Estimation of $\sigma^2(x)$

The estimator can be obtained as $\hat{\sigma}^2(x) = e^{\hat{q}(x)}$, where $\hat{q}(x)$ is the estimated regression function of the previous section.

```
s2.1 = exp(m.xz.1$mtgr)
```



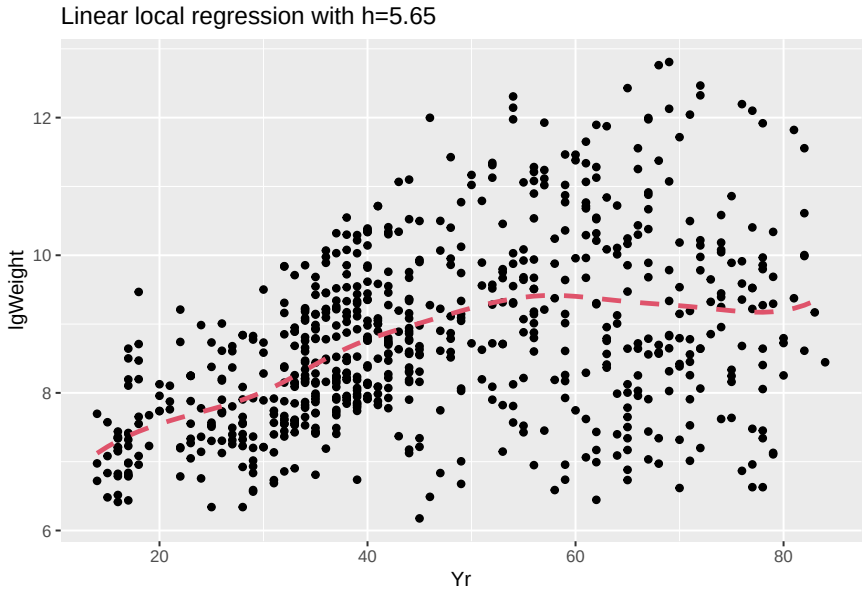


2. Local regression using `sm`

We will repeat the same procedure by means of the function `sm.regression`. The bandwidth choice will be done by direct plug-in with the function `dpill` from the library `KernSmooth`.

2.1 Fitting the model to (x_i, y_i)

```
h.plug.xy = dpill(Yr, lgWeight)
m.xy.2 = sm.regression(Yr, lgWeight, h=h.plug.xy, eval.points=Yr, display="none")
```

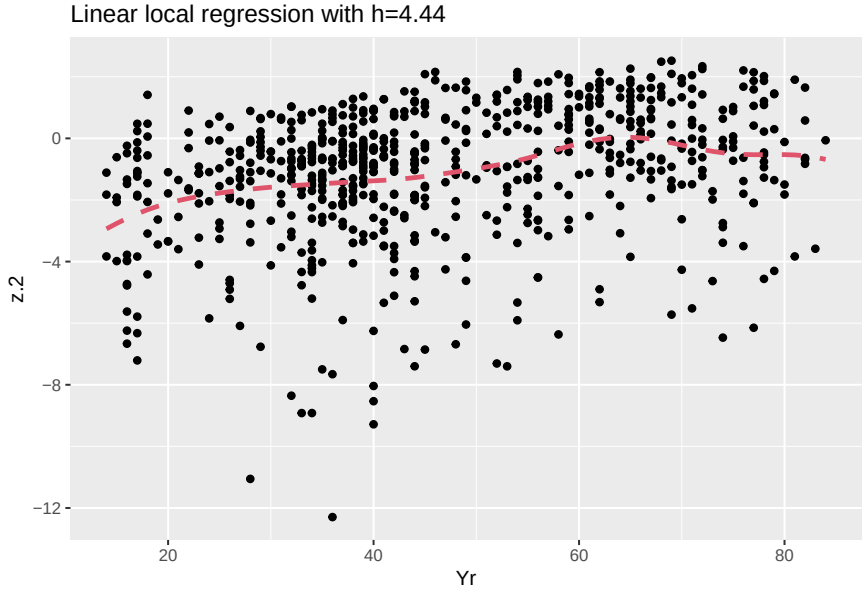


2.2 Transforming the estimated residuals

```
eps.2 = lgWeight - m.xy.2$estimate
z.2 = log(eps.2^2)
```

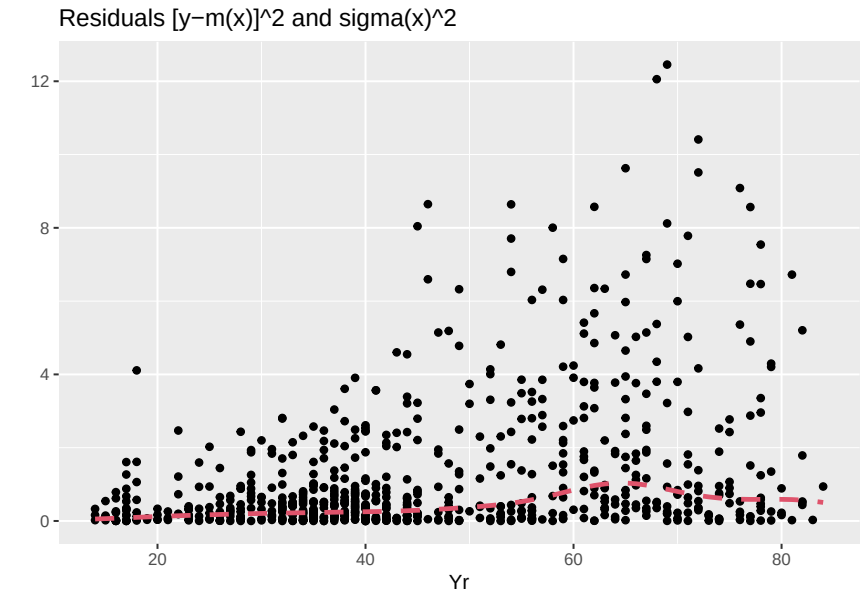
2.3 Fitting the model to (x_i, z_i)

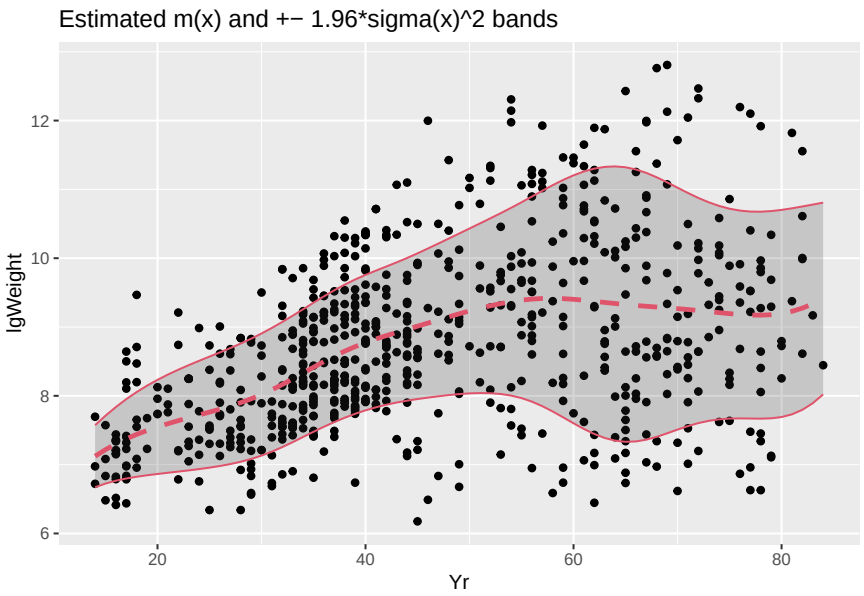
```
h.plug.xz = dpill(Yr, z.2)
m.xz.2 = sm.regression(Yr, z.2, h=h.plug.xz, eval.points=Yr, display="none")
```



2.4 Estimation of $\sigma^2(x)$

```
s2.2 = exp(m.xz.2$estimate)
```

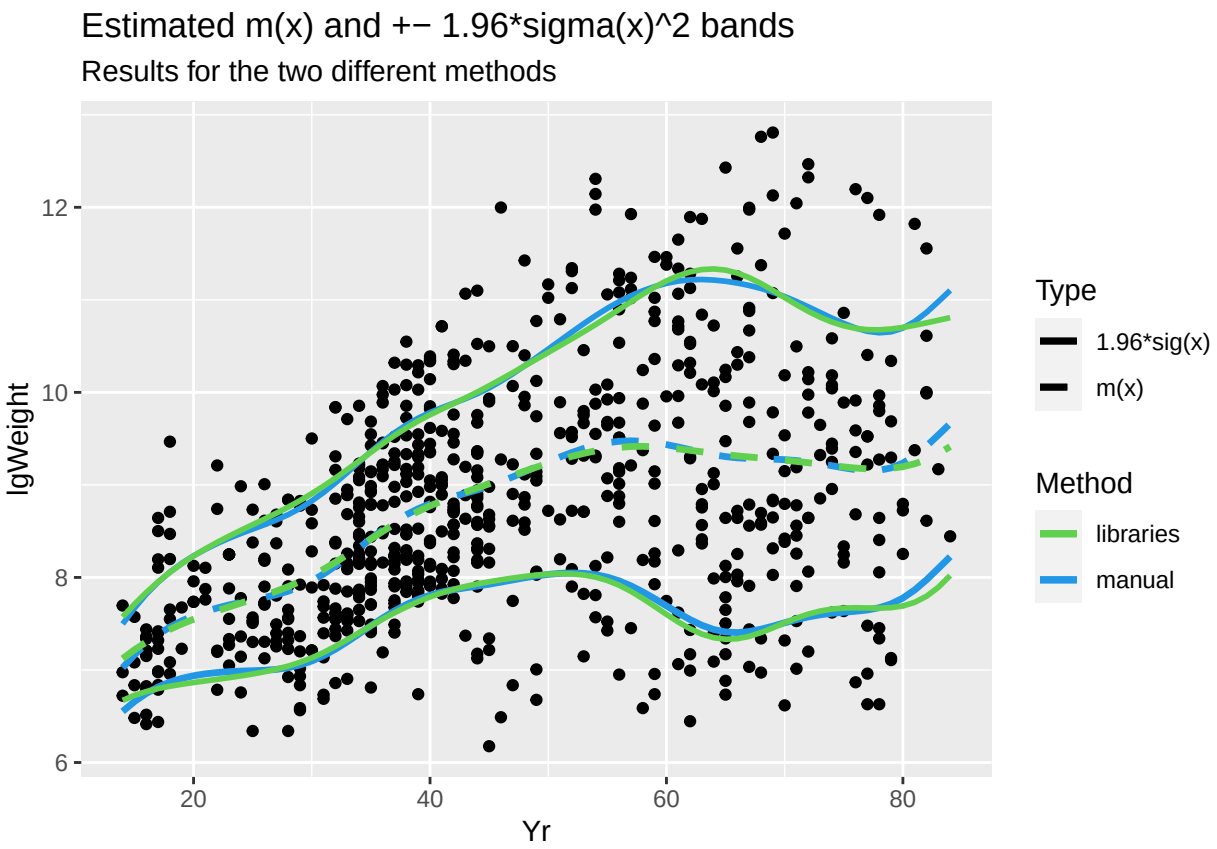




3. Conclusions

If we compare the results obtained by the two methods, we can see that the bandwidths are slightly different. In particular, for the (x_i, z_i) estimation, the bandwidth chosen for the LOO-CV method is superior, which could explain the slight differences in the result that we observe at the right end of the graph, where the absence of reference datapoints makes the bandwidth choice critical and its effects more noticeable.

```
##              X.Y      X.Z
## LOO-CV 4.185346 5.984916
## dpill  5.650869 4.440591
```


Estimated m(x) and +− 1.96*sigma(x)^2 bands
Results for the two different methods
lgWeight
Yr
12
10
8
6
20
40
60
80
Type
1.96*sig(x)
m(x)
Method
libraries
manual