Semiparametric regression

Homework 1

Makowski Michał

31 march 2017

Contents

Exercise I

At the begging of this exercise we will do some linear algebra - basis, determinants etc... Then we will use OLS to find regresion coefficient for some sample.

Let's define following functions on $[1, 0]$

$$T_1(x) = 1, \qquad T_2(x) = x, \qquad T_3(x) = \left(x - \frac{1}{2}\right)_+$$

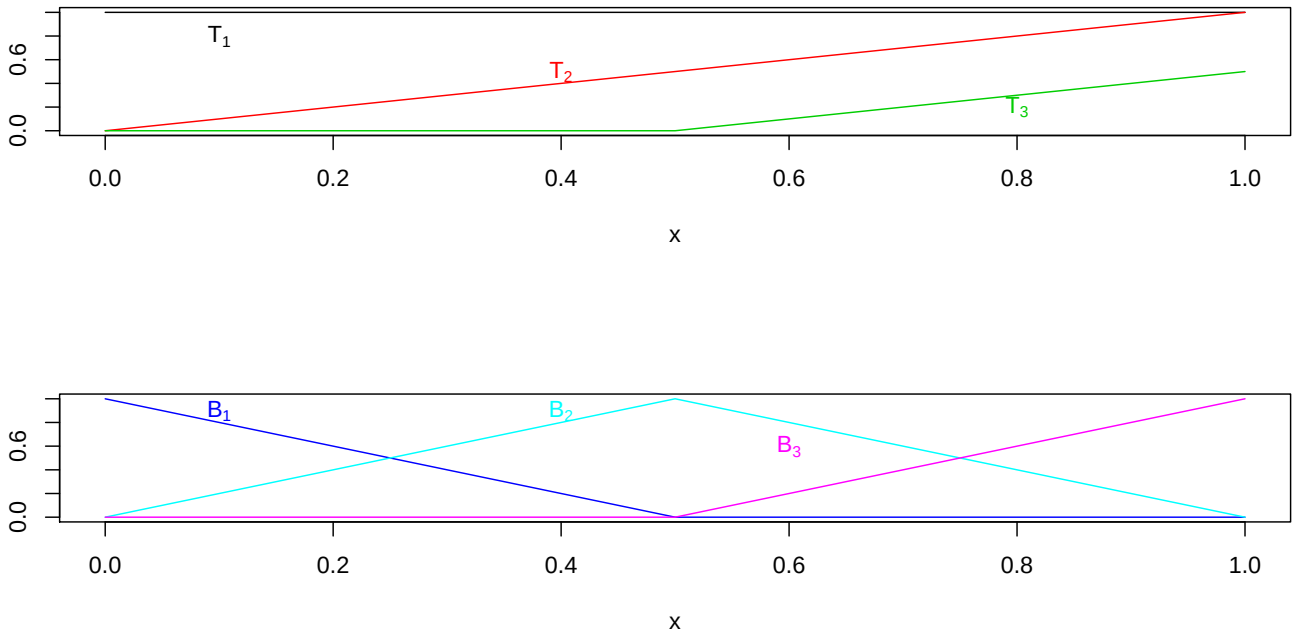
and

$$B_1(x) = (1 - 2x)_+, \qquad B_2(x) = 1 - \mid 2x - 1 \mid_+, \qquad B_3(x) = (2x - 1)_+$$

a)

At first, we have to plot functions defined above by running folowing code:

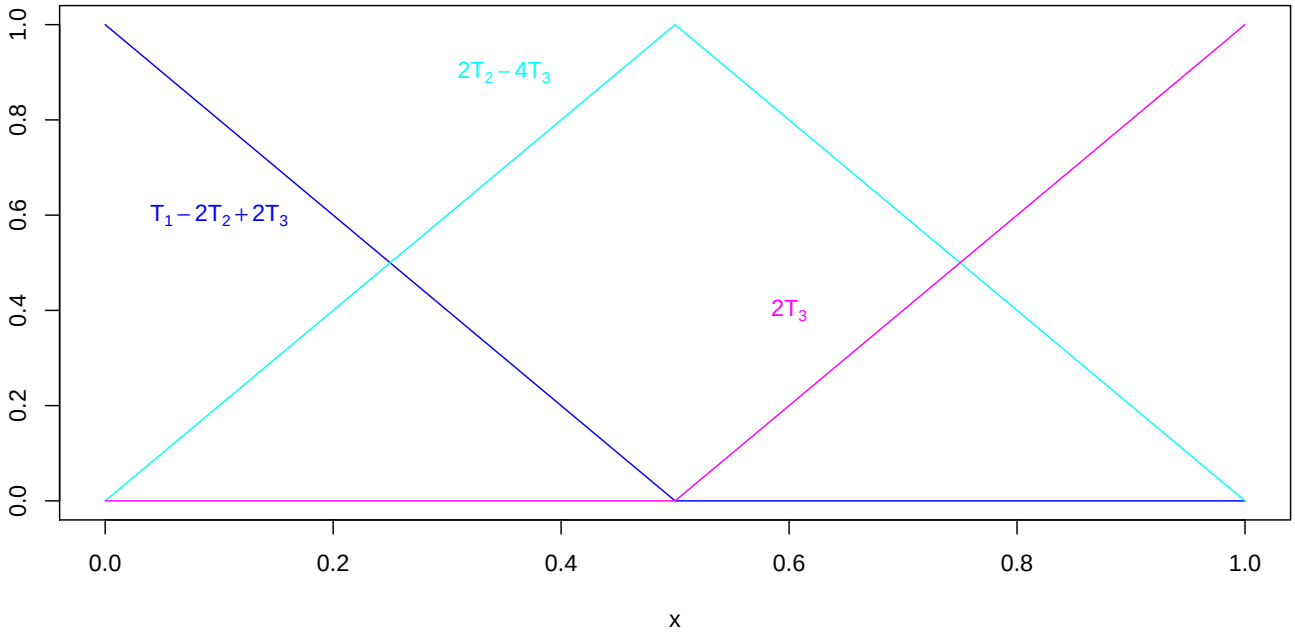
```
ng = 101
xg = seq(0, 1, length=ng)
T1g = rep(1, ng)
T2g = xg
T3g = (xg - .5)*(xg - .5>0)
B1g = (1 - 2*xg)*(1 - 2*xg>0)
B2g = 1 - abs(2*xg - 1)
B3g = 2*T3g
par(mfrow=c(2,1))
plot(0, type = "n", xlim=c(0,1), ylim=c(0,1), xlab="x", ylab="") #, bty="1")
lines(xg, T1g, col=1)
lines(xg, T2g, col=2)
lines(xg, T3g, col=3)
text(0.1, 0.8, expression(T[1]), col=1)
text(0.4, 0.5, expression(T[2]), col=2)
text(0.8, 0.2, expression(T[3]), col=3)
plot (0, type = "n", xlim=c(0,1), ylim=c(0,1), xlab="x", ylab="") #, bty="1")
lines(xg, B1g, col=4)
lines(xg, B2g, col=5)
lines(xg, B3g, col=6)
text(0.1, 0.9, expression(B[1]), col=4)
text(0.4, 0.9, expression(B[2]), col=5)
text(0.6, 0.6, expression(B[3]), col=6)
```



b)

Next, we have to find expressionf for B_1, B_2, B_3 in terms of T_1, T_2, T_3. That is easy exercise, quite similiar to "gluing" payoffs of options in financial mathematics:

```
plot (0, type = "n", xlim=c(0,1), ylim=c(0,1), xlab="x", ylab="") #, bty="1")
lines(xg, T1g-2*T2g+2*T3g, col=4)
lines(xg, 2*T2g-4*T3g, col=5)
lines(xg, 2*T3g, col=6)
text(0.1, 0.6, expression(T[1]-2*T[2]+2*T[3]), col=4)
text(0.35, 0.9, expression(2*T[2]-4*T[3]), col=5)
text(0.6, 0.4, expression(2*T[3]), col=6)
```



We came up to following representations:

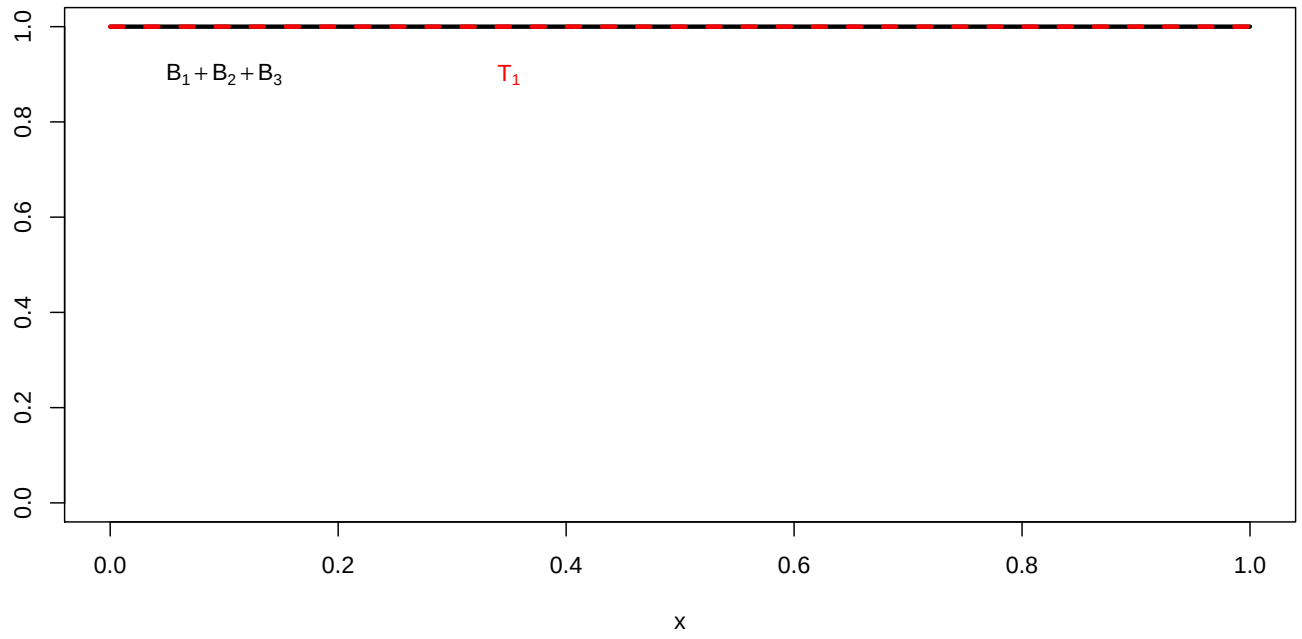
$$B_1 = T_1 - 2(T_2 - T_3)$$
$$B_2 = 2(T_2 - 2T_3)$$
$$B_3 = 2T_3$$

Last question: What is $B_1 + B_2 + B_3$? Let's see:

$$\begin{aligned} B_1 + B_2 + B_3 &= T_1 - 2T_2 + 2T_3 + 2(T_2 - 2T_3) + 2T_3 \\ &= T_1 = 1 \end{aligned} \tag{1}$$

Plot confirms:

```
plot (0, type = "n", xlim=c(0,1), ylim=c(0,1), xlab="x", ylab="") #, bty="1")
lines(xg, B1g+B2g+B3g, col=1, lwd=3)
lines(xg, T1g, col=2, lty=2, lwd=3)
text(0.1, 0.9, expression(B[1]+B[2]+B[3]), col=1)
text(0.35, 0.9, expression(T[1]), col=2)
```



c)

To find matrix L_{TB} such that

$$\begin{bmatrix} B_1 & B_2 & B_3 \end{bmatrix} = \begin{bmatrix} T_1 & T_2 & T_3 \end{bmatrix} L_{TB}$$

we have to solve:

$$\begin{bmatrix} T_1 - 2(T_2 - T_3) & 2(T_2 - 2T_3) & 2T_3 \end{bmatrix} = \begin{bmatrix} T_1 & T_2 & T_3 \end{bmatrix} L_{TB}$$

One of the solutions of this eqution is following matrix:

$$\begin{bmatrix} 1 & 0 & 0 \\ -2 & 2 & 0 \\ 2 & -4 & 2 \end{bmatrix}$$

```
LTB = matrix(c(-1,2,0,
               -2,2,0,
                2,4,2), byrow = T, ncol = 3)
```

$(L_{TB})^T$ is transformation matrix from space $Lin\{T_1, T_2, T_3\}$ to $Lin\{B_1, B_2, B_3\}$.

d)

Now we have to compute determinant of L_{TB}, if it differs from zero, which is true, matrix is invertible (because we are "living" in the world, or rather field, of real numbers).

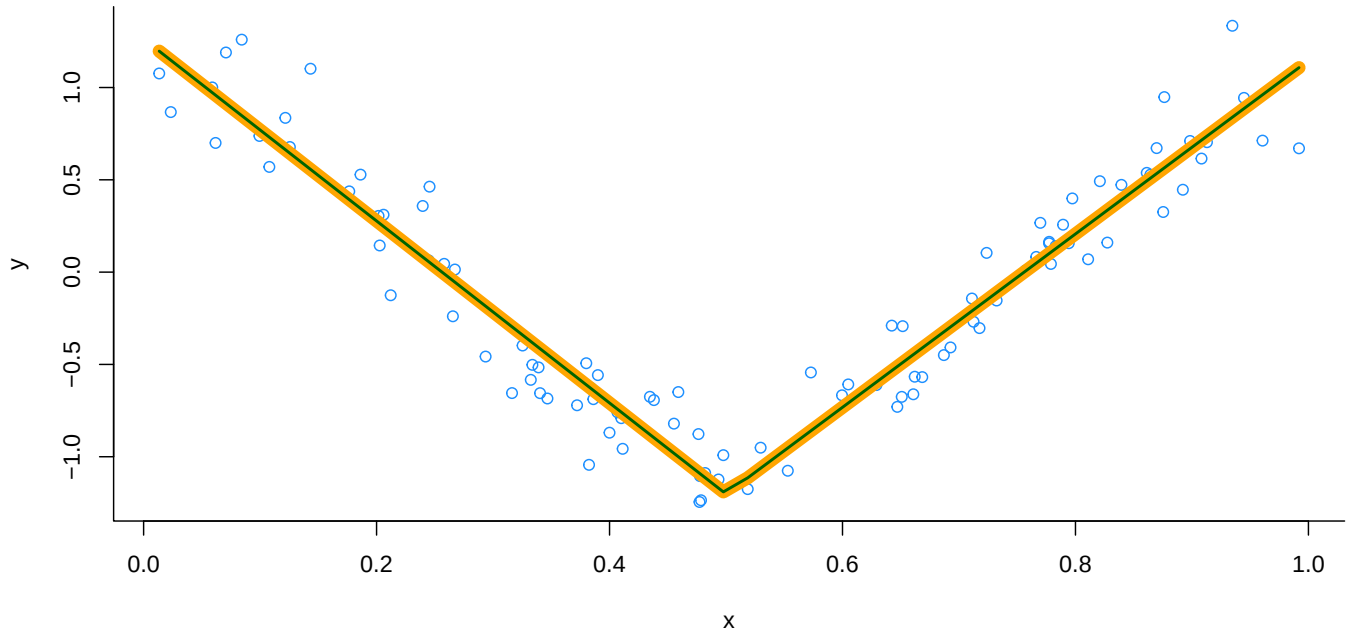
```
det(LTB)
```

```
[1] 4
```

e)

Below we compare two models, first build using basis $\{T_1, T_2, T_3\}$, another using $\{B_1, B_2, B_3\}$. We will see that they are visually undifferentiated.

```
par(mfrow = c(1,1))
set.seed(1)
n = 100
x = sort(runif(100))
y = cos(2*pi*x)  + 0.2*rnorm(n)
plot(x, y, col = "dodgerblue", bty = "l")
XT = cbind(rep(1,n),
            x,
            (2*x-1)*(2*x-1>0))

XB = cbind((1-2*x)*(1-2*x>0),
            1 - abs(2*x-1),
            (2*x-1)*(2*x-1>0))

fitT = lm(y~ -1+XT)
fitB = lm(y~ -1+XB)

lines(x, fitted(fitT), col = "orange", lwd = 9)
lines(x, fitted(fitB), col = "darkgreen", lwd = 2)
```



Above we can see that fitted lines overlap each other.

Exercise II

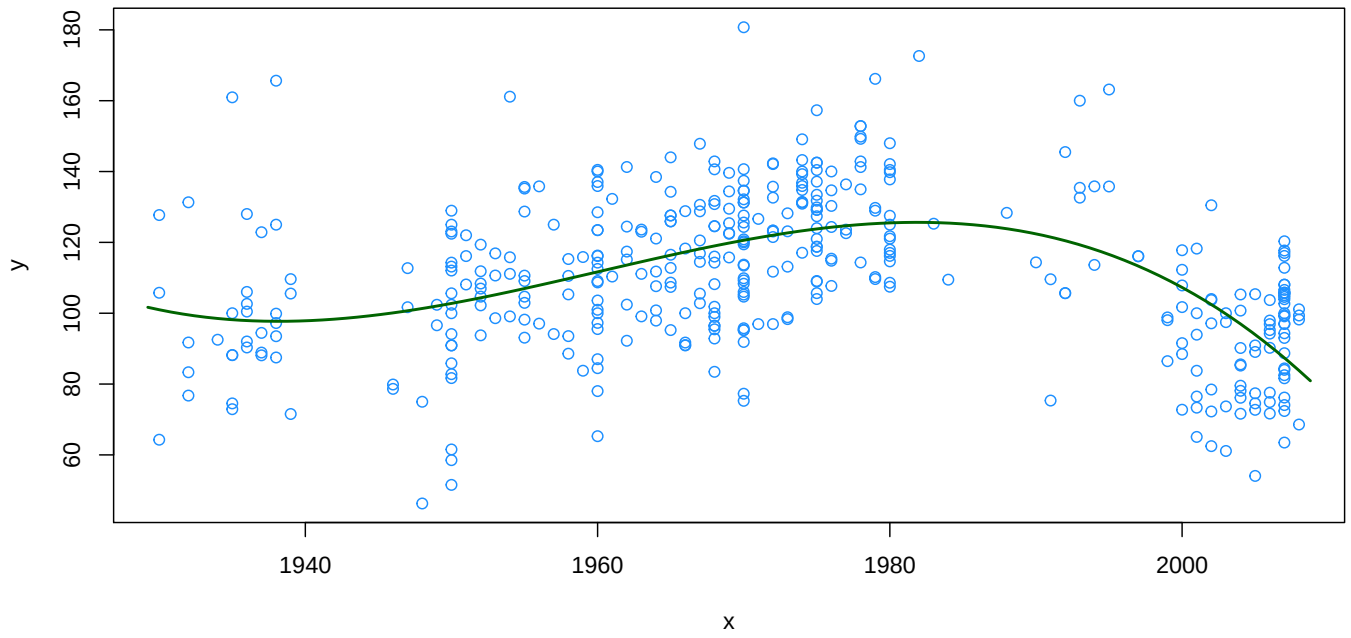
a)

Firstly, we create cubic regression model, based on data about apartaments price in Warsaw:

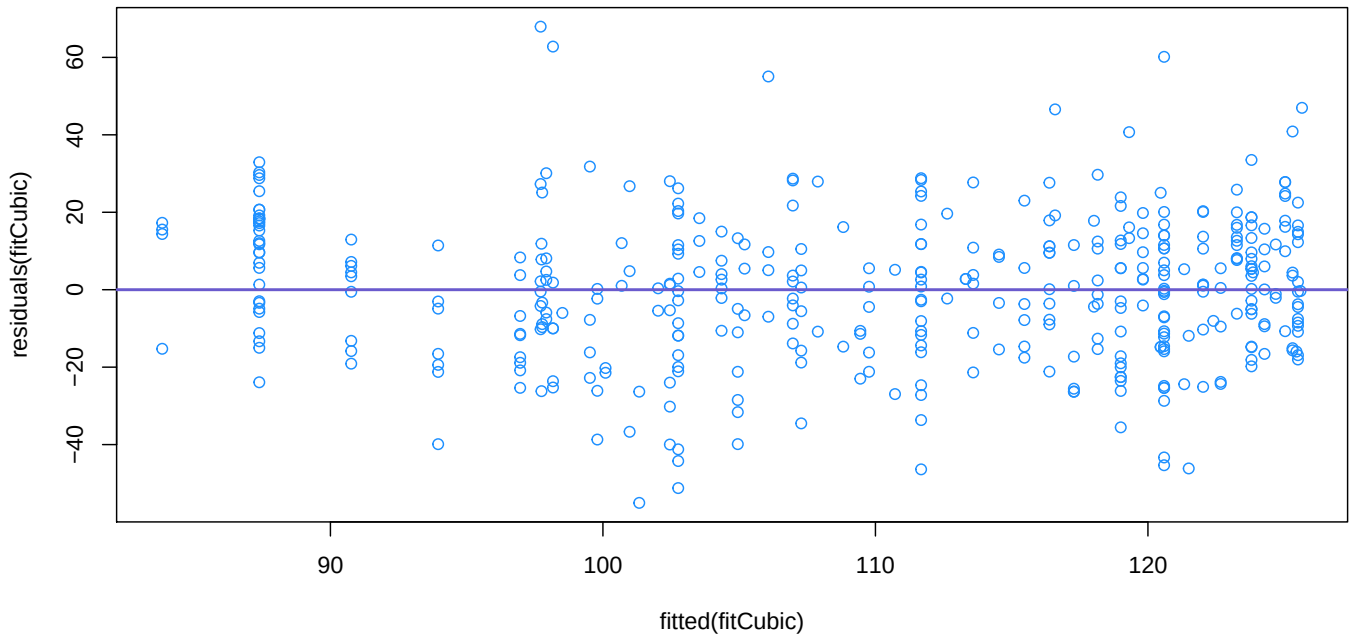
$$y_i = \beta_0 + \beta_1 + \beta_2 x_i^2 + \beta_3 x_i^3 + \epsilon_i$$

We will present belowe three different plots presenting our model:

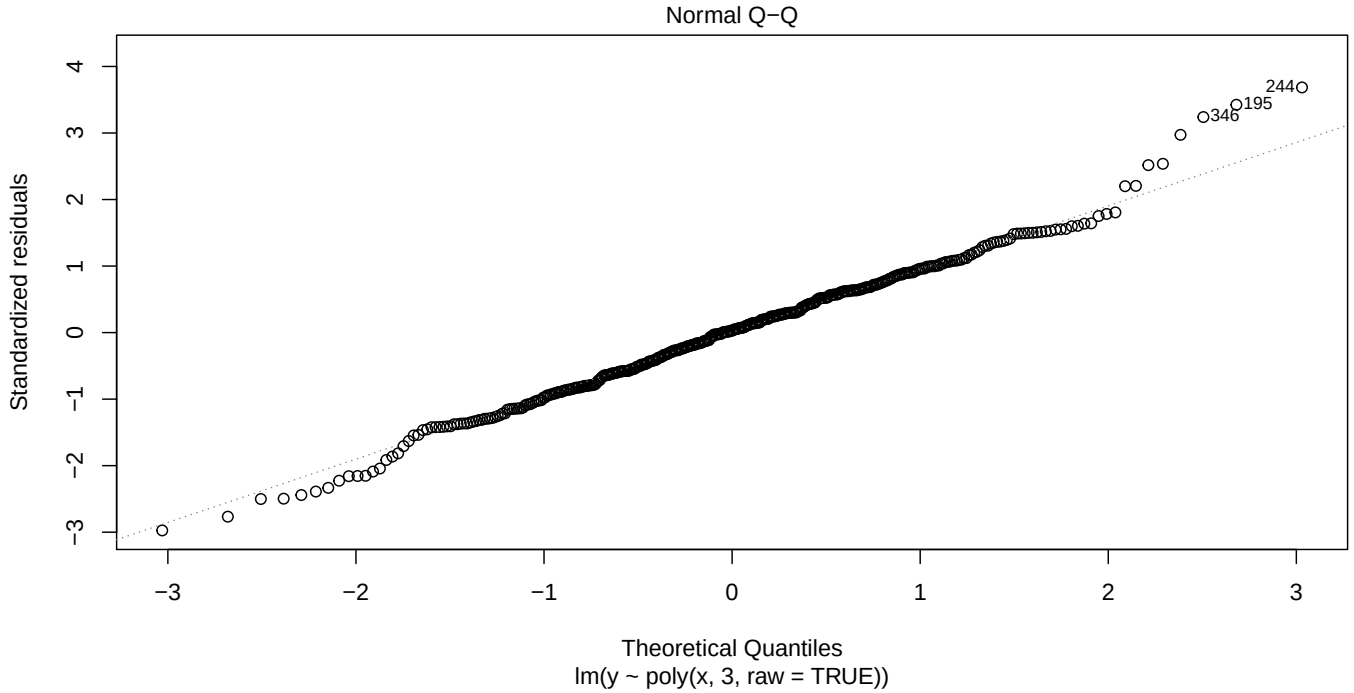
```
library(HRW)
data(WarsawApts)
x = WarsawApts$construction.date
y = WarsawApts$areaPerMzloty
fitCubic = lm(y ~ poly(x, 3, raw = TRUE))
ng = 101
xg = seq( 1.01*min(x) - 0.01*max(x),
          1.01*max(x) - 0.01*min(x), length = ng)

fHatCubicg = as.vector( cbind(rep(1, ng), xg, xg^2, xg^3)%*%fitCubic$coef)

plot(x,y, col="dodgerblue")
lines(xg, fHatCubicg, col = "darkgreen", lwd = 2)
```




```
plot( fitted(fitCubic), residuals(fitCubic),
      col = "dodgerblue")
abline(0,0, col = "slateblue", lwd = 2)
```




```
plot(fitCubic,2)
```



On the first plot above we can see that smooth curve is enough good fit to data. Second plot prosent no dependecy between residuals and fitted values, what confirm homoscedasticity. Last, diagnostic plot, shows that there is few "heavy" observations from the tail, whose residuals don't fit to normal distribution.

b)

Now we define the truncated line function with a knot at kappa (κ):

$$(x-\kappa)_+ = (x-\kappa)\cdot 1_{\{x>\kappa\}}$$

```
trLin = function(x,kappa) return( (x-kappa)*(x>kappa))
```

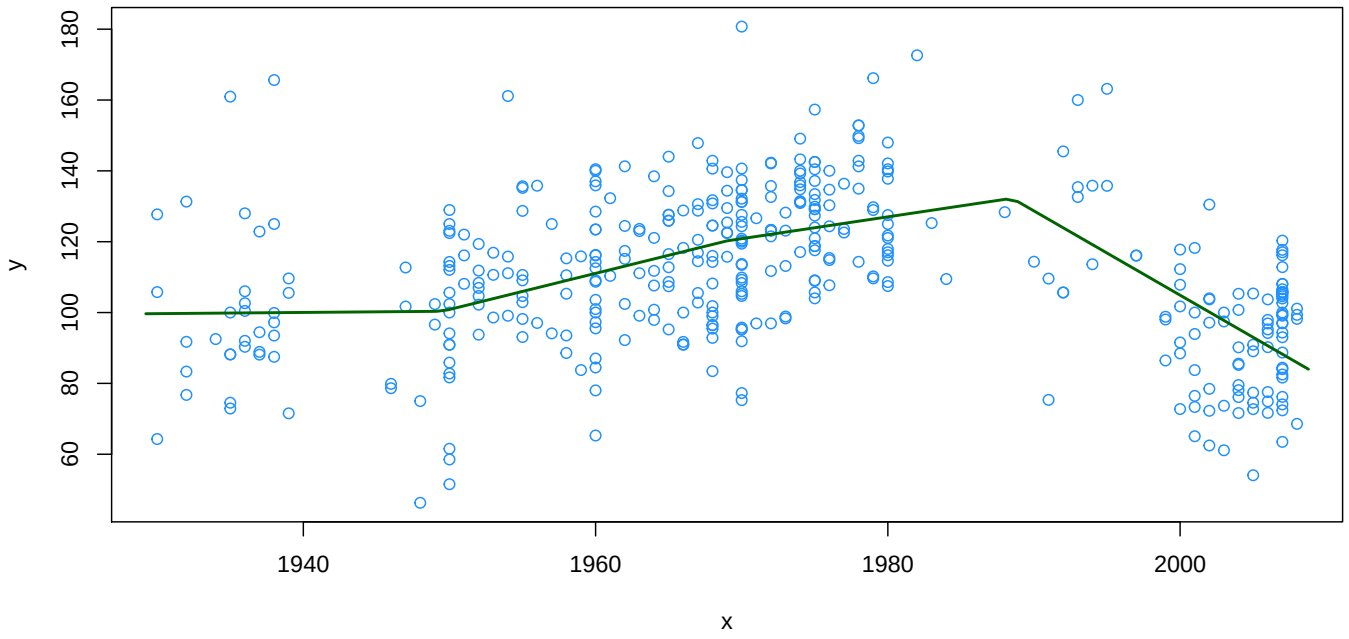
c)

Now we fit the spline regression model to data from **a)**:

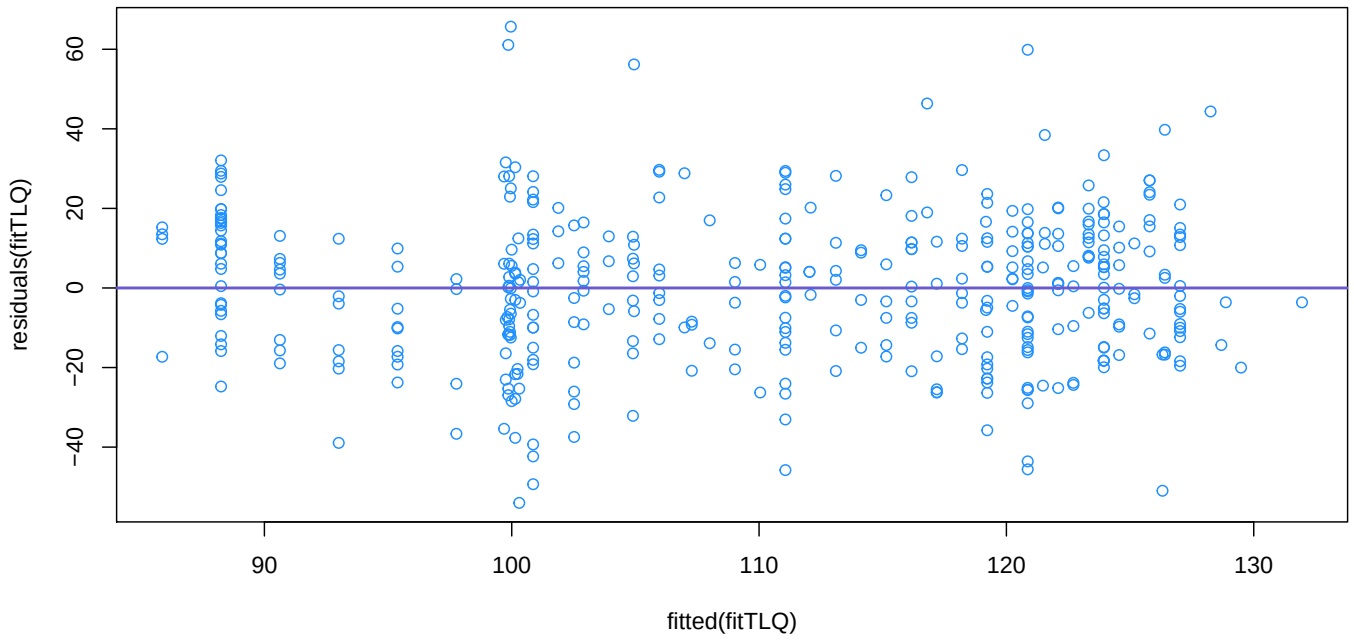
$$y_i = \beta_0 + \beta_1 x_i + u_1(x_i-\kappa_1)_+ + u_2(x_i-\kappa_2)_+ + u_3(x_i-\kappa_3)_+ + \epsilon_i$$

R code responsible for this operation (as in **a)** plots will be also presented):

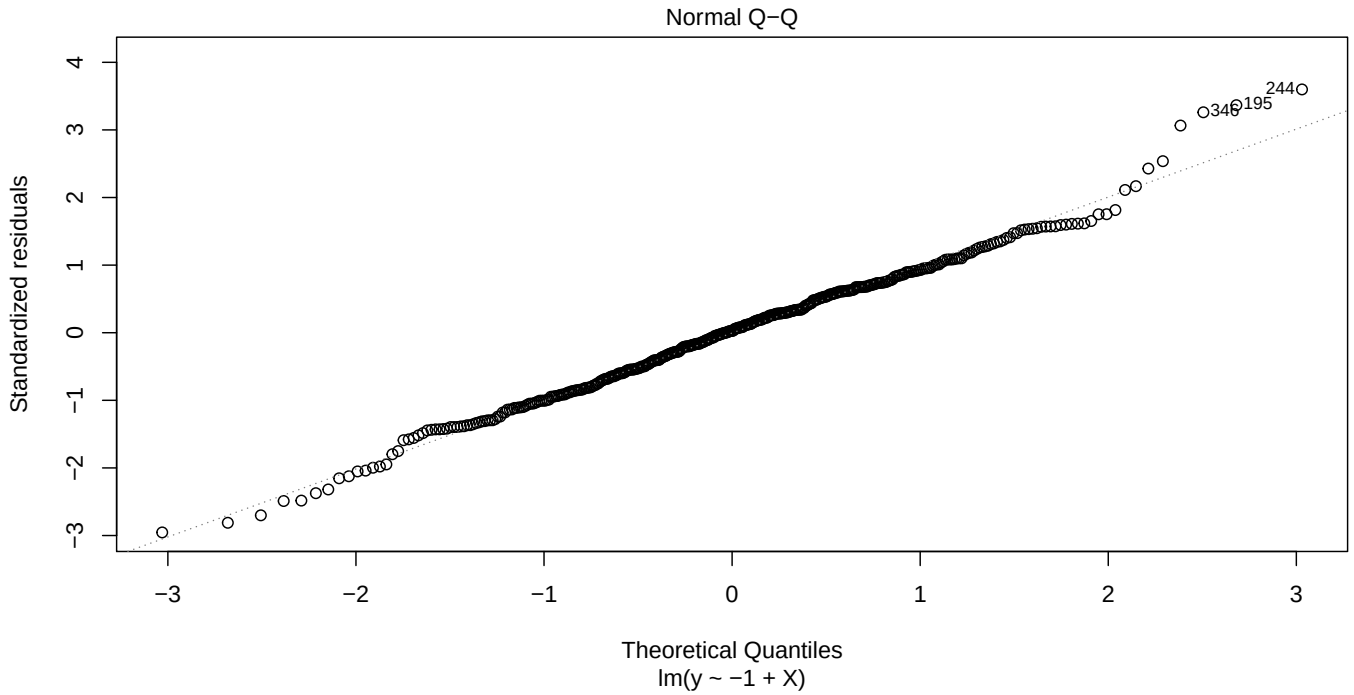
```
knots = seq(min(x), max(x), length = 5)[-c(1,5)]
X = cbind(1,x)
for(k in 1:3) X = cbind(X, trLin(x, knots[k]))
fitTLQ = lm(y ~ -1 + X)
Xg = cbind(1,xg)
for(k in 1:3) Xg = cbind(Xg,trLin(xg, knots[k]))
fHatTLQg = as.vector(Xg%*%fitTLQ$coef)
plot(x,y, col = "dodgerblue")
lines(xg, fHatTLQg, col = "darkgreen", lwd = 2)
```




```
plot(fitted(fitTLQ), residuals(fitTLQ), col = "dodgerblue")
abline(0,0,col="slateblue", lwd = 2)
```




```
plot(fitTLQ, 2)
```



As in **a)** we obtain curve which is quite good fitted to the data. It is not so smooth as in **a)**. Second plot resemble this from point **a)** - we see no dependency between residuals and fitted values, which mean good properities of model. At third diagnostic plot we can observe a lot of observations which standardized residuals do not come from standard normal distribution. Especially observations which quantiles are bigger than 2.2 do not meet our requirements.

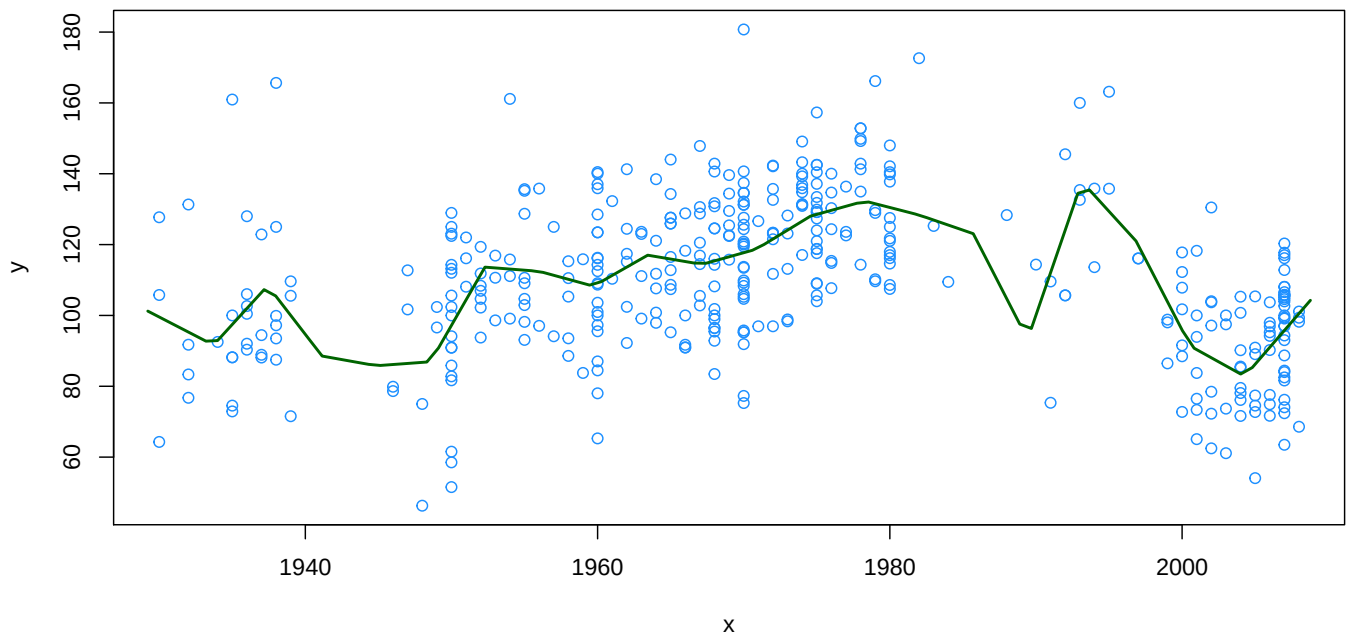
d)

Now we run R script, which use OLS to fit the spline regression model with 20 knots:

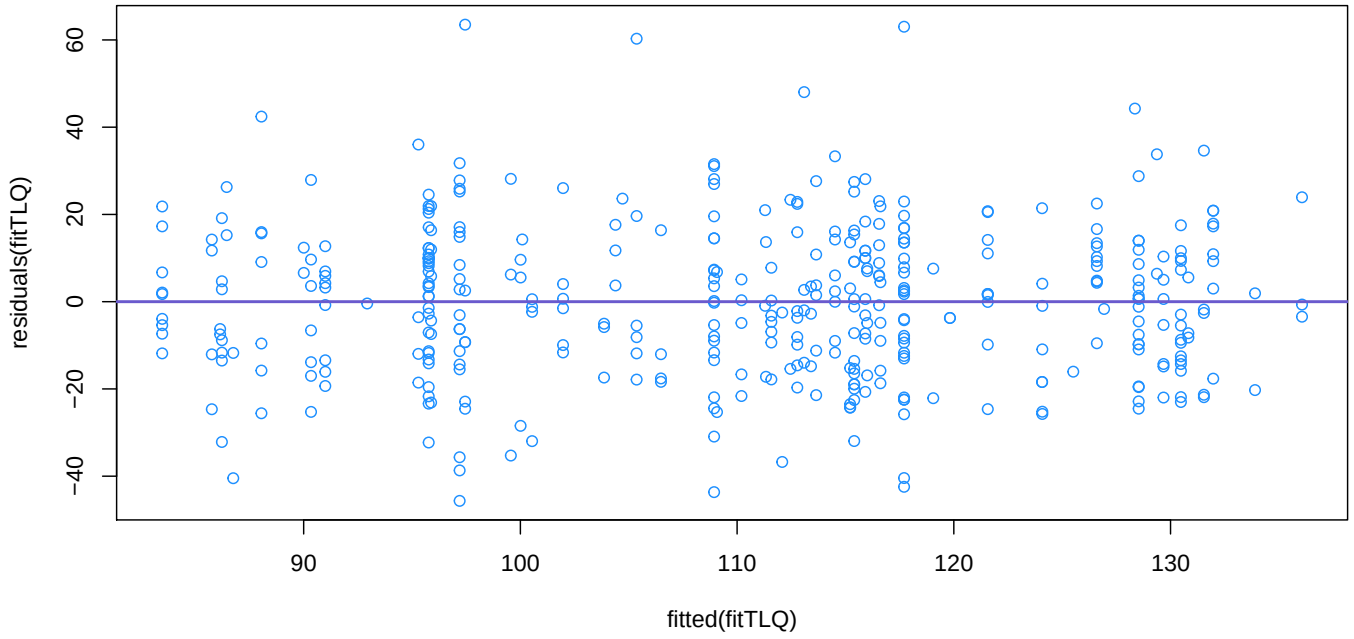
$$y_i = \beta_0 + \beta_1 x_i + \sum_{k=1}^{20} u_k (x_i - \kappa_k)_+ + \epsilon_i$$

R code (model fitting & 2 plots:

```
nods = seq(min(x), max(x), length = 22)[-c(1,22)]
X = cbind(1,x)
for( k in 1:20) X = cbind(X, trLin(x, nods[k]))
fitTLQ = lm(y~-1+X)
Xg = cbind(1,xg)
for(k in 1:20) Xg = cbind( Xg, trLin(xg, nods[k]))
fHatTLQg = as.vector(Xg%*%fitTLQ$coefficients)
plot(x,y, col = "dodgerblue")
lines(xg, fHatTLQg, col = "darkgreen", lwd = 2)
```




```
plot(fitted(fitTLQ), residuals(fitTLQ), col = "dodgerblue")
abline(0,0,col="slateblue", lwd = 2)
```



At the first plot we can see that, there is too many knots and curve curve is overfitted. There are to many fluctuations in fitted line, which do not necessary shows dependencies in real data. For expample - in years 1980-1998 there is exteme lack of data, we do have around 15 observations to build model on them, but fitted line shows a lot of dependencies (it has 4 extremas). On the other hand, second plot shows no dependency between residuals and fitted values. Model fullfiled one of the assumptions, but visual diagnostic of fitted line suggest it is not well build. We could use less number of knots to fix this model. We could also do better *data mining* to find some observations wich we could substract from our sample (i.e which residuals seems to be not standard normal).