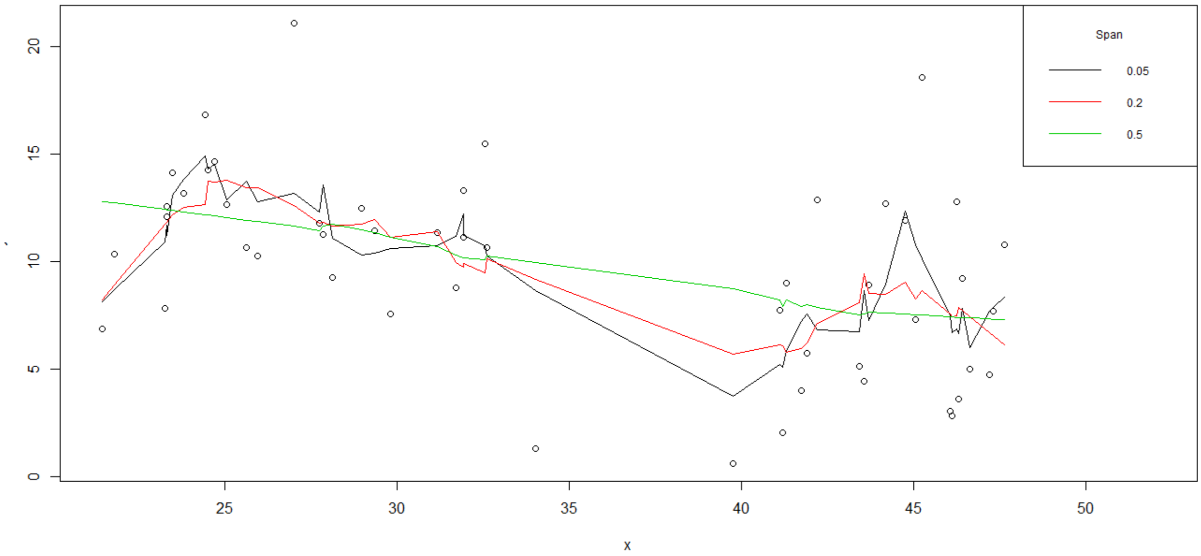
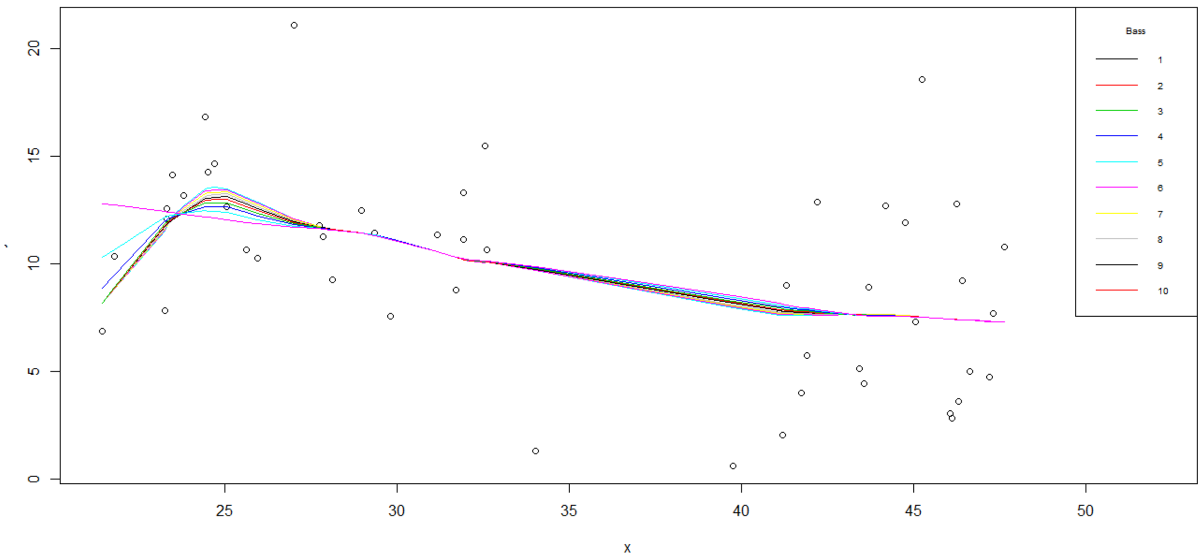
```r
install.packages("fANCOVA")
library(fANCOVA)
install.packages("np")
library(np)
install.packages("scatterplot3d")
library(scatterplot3d)

LifeCycleSavings
sr <- LifeCycleSavings$sr
pop15 <- LifeCycleSavings$pop15
pop75 <- LifeCycleSavings$pop75
dpi <- LifeCycleSavings$dpi
ddpi <- LifeCycleSavings$ddpi



# i)
spanvec = c(0.05, 0.2, 0.5)
bassvec = seq(1, 10)

#GRAPHICS SUPSMU


gr_supsmu <- function(x, y) {
  plot(x,y, xlim = c(min(x), max(x)+(max(x)-min(x))/6))
  for (j in (1:length(spanvec))){
    lines(supsmu(x, y, span=spanvec[j]),col=j)
  }
legend("topright", legend=spanvec, col=seq(spanvec),lty=1,cex=.75, title =
"Span")
}
gr_supsmu(pop15, sr)
```




```
gr_supsmu_bass <- function(x, y) {
  plot(x,y, xlim = c(min(x), max(x)+(max(x)-min(x))/6)) #на отдельном графике
  for (j in (1:length(bassvec))) {
    lines(supsmu(x, y, bass=bassvec[j]), col=j+4)
  }
legend("topright", legend=bassvec, col=seq(bassvec),lty=1,cex=.6, title = "Bass")
}
gr_supsmu_bass(pop15, sr)
```




```
#function reordering y vector according to vector x
#ВСПОМОГАТЕЛЬНАЯ ФУНКЦИЯ, РАНЖИРУЕТ ИГРЕК В СООТВЕТСТВИИ С РАНЖИРОВАННЫМ ИКСОМ
reord_y <- function(x,y) {
```

```
  x1 <- sort(x)

  y1 <- rep(0, length(y))

  for(i in 1:length(y)) {

    for(j in 1:length(x)) {

      if(x[i] == x1[j]) y1[j] <- y[i]

    }

  }

  return(cbind(x1, y1))

}



all_possible_mse <- function(x,y) {

  reord <- reord_y(x,y)
  x <- reord[,1]
  y <- reord[,2]

  M=cbind(x,y) #УБИРАЕМ ДУБЛИКАТЫ X, ЕСЛИ ОНИ ВДРУГ ЕСТЬ, ПОТОМУ ЧТО ПО УМОЛЧ
АНИЮ SUPSMU ЭТО ДЕЛАЕТ
  if(sum(duplicated(x)) == 0) x <- x
  else  {
    M=M[-which(duplicated(x)),]
  x <- M[,1]
  y <- M[,2]
  }
  #M=M[-which(duplicated(M[,2])),]

  m <- rep(0, (length(bassvec) + length(spanvec)))

  #all possible span
  for (j in (1:length(spanvec))){
    model = supsmu(x, y, span = spanvec[j])
    m[j] = mean((model$y-y)^2)
  }

  #all possible bass and span choosed by cross-validation (by default)
  for (i in (1:length(bassvec))){
    model = supsmu(x,y, bass=bassvec[i])
    m[length(spanvec)+ i] = mean((model$y-y)^2)
  }
  mse1 <- m # vector of mse for c(all possible span, all possible bass)
  res1 <- c(min(mse1), which(mse1 == min(mse1)))
  if(res1[2] >= 4) res2 <- c("bass = ", bassvec[res1[2]-3])
  else res2 <- c("span = ", spanvec[res1[2]])

  list(mse1, res1, res2)
}

all_possible_mse(pop15, sr)

> all_possible_mse(pop15, sr)
[[1]]
 [1] 10.21983 12.86419 15.50151 13.84201 13.89489 13.95830 14.02693 14.09797
14.16678
[10] 14.24971 14.37727 14.66417 15.45314

[[2]]
[1] 10.21983  1.00000

[[3]]
[1] "span = " "0.05"
```

ЛУЧШАЯ МОДЕЛЬ СРЕДИ МОДЕЛЕЙ С РАЗНЫМИ SPAN И BASS (в смысле MSE) С ПАРАМЕТРОМ SPAN = 0.05

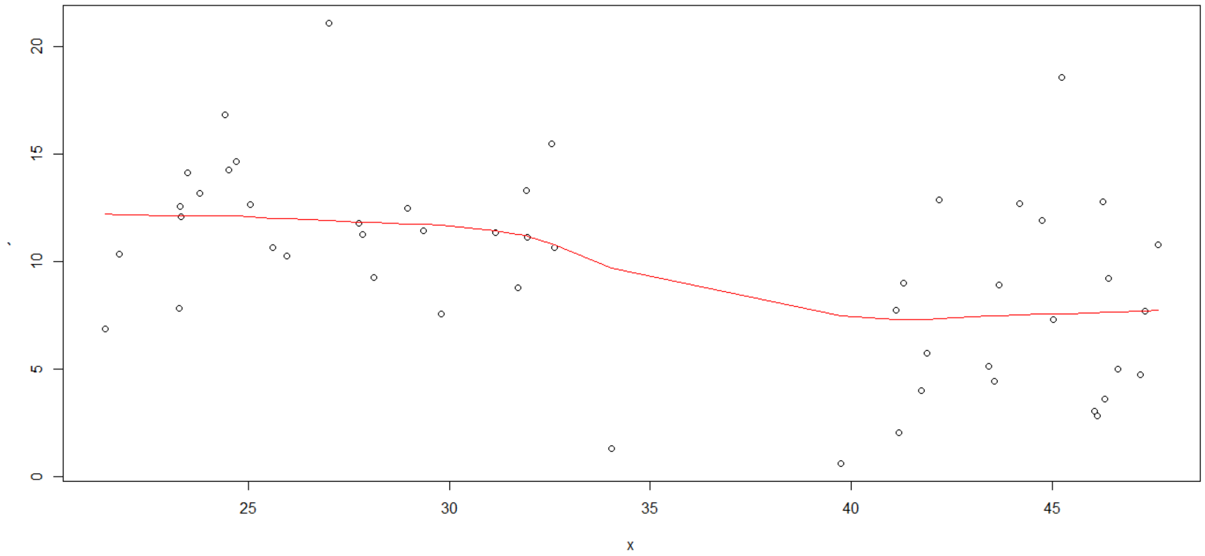
```
##ii)

bvec = c("cv.aic", "cv.ls")

kervec = c("epanechnikov","gaussian")


all_possible_mse_kernel <- function(x,y) {


  reord <- reord_y(x,y)

  x <- reord[,1]

  y <- reord[,2]


  M=cbind(x,y) #УБИРАЕМ ДУБЛИКАТЫ X, ЕСЛИ ОНИ ВДРУГ ЕСТЬ, ПОТОМУ ЧТО ПО УМОЛЧАНИЮ SUPSMU ЭТО ДЕЛАЕТ

  if(sum(duplicated(x)) == 0) x <- x

  else  {

    M=M[-which(duplicated(x)),]

    x <- M[,1]

    y <- M[,2]

  }


  res = 1000

  m <- rep(0, 4)

for (i in (1:2)){

  for (k in (1:2)){

    model=npreg(txdat=x, tydat=y,

                ckertype=kervec[i],

                bwmethod=bvec[k])

    if(i ==1) { m[i+k-1] = mean((fitted(model)-y)^2); m1 = m[i+k-1]}

    else { m[i+k] = mean((fitted(model)-y)^2); m1 = m[i+k] }

    if (m1 < res){

      res = m1

      vec = c(i,k)

    }

  }

}

  list(m, res, vec, c(kervec[vec[1]],bvec[vec[2]]))

}
```

```
all_possible_mse_kernel(pop15, sr)
```

```
> all_possible_mse_kernel(pop15, sr)
[[1]]
[1] 14.57089 14.57089 14.62272 14.60803

[[2]]
[1] 14.57089

[[3]]
[1] 1 1

[[4]]
[1] "epanechnikov" "cv.aic"
```

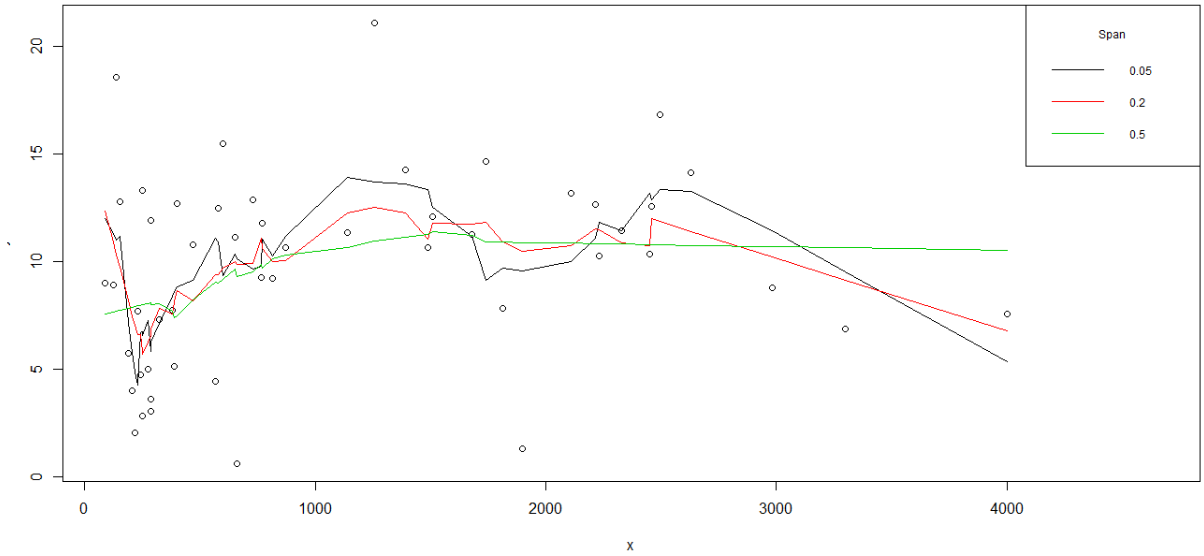
ЛУЧШАЯ МОДЕЛЬ epanechnikov, aic

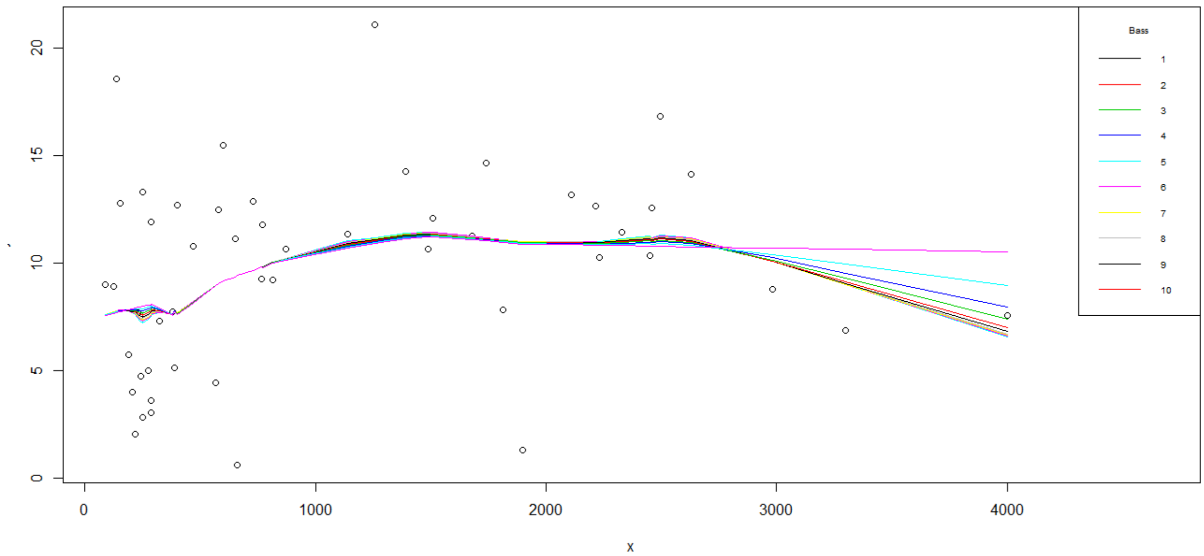
```
best_kernel_model <- function(x,y) {
  reord <- reord_y(x,y)
  x <- reord[,1]
  y <- reord[,2]

  M=cbind(x,y) #УБИРАЕМ ДУБЛИКАТЫ X, ЕСЛИ ОНИ ВДРУГ ЕСТЬ, ПОТОМУ ЧТО ПО УМОЛЧАНИЮ SUPSMU ЭТО ДЕЛАЕТ
  if(sum(duplicated(x)) == 0) x <- x
  else  {
    M=M[-which(duplicated(x)),]
    x <- M[,1]
    y <- M[,2]
  }

  mod = all_possible_mse_kernel(x, y)
model=npreg(txdat=x, tydat=y,
            ckertype=kervec[mod[[3]][1]],
            bwmethod=bvec[mod[[3]][2]])
plot(x,y)
yy <- fitted(model)
points(x, yy, type="l", col="red")
}

best_kernel_model(pop15, sr)
```

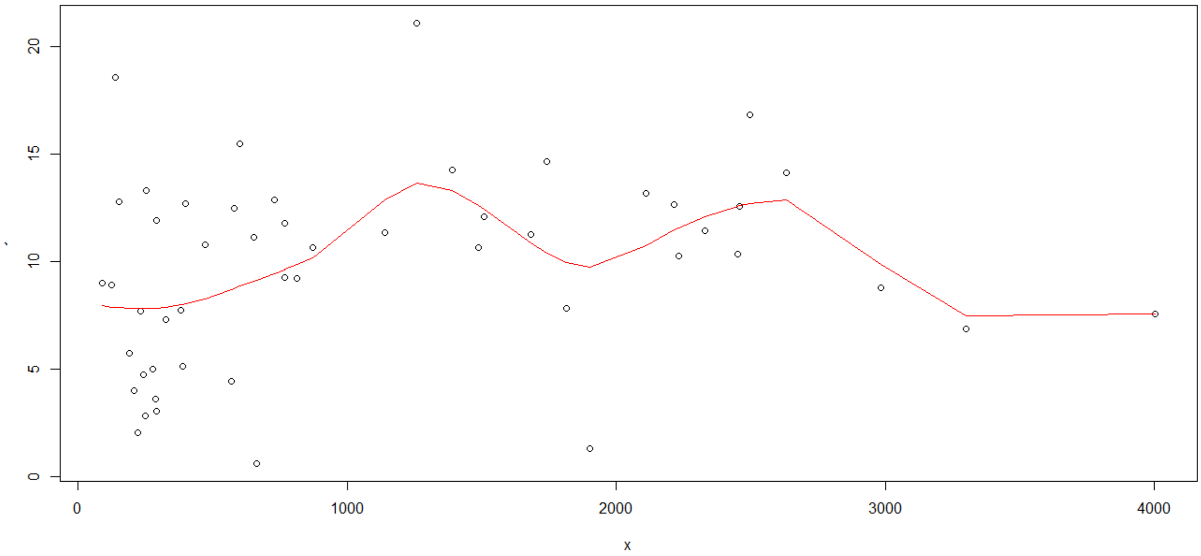



```
### iii)

#ONLY GRAPHS
main_grafs <- function(x,y) {
  gr_supsmu(x,y)
  gr_supsmu_bass(x,y)
  best_kernel_model(x,y)
}
#main_grafs(pop75,sr)
main_grafs(dpi,sr)
#main_grafs(ddpi,sr)
```

НАПРИМЕР, ДЛЯ DPI


Span
0.05
0.2
0.5
0
1000
2000
3000
4000
x
0
5
10
15
20


Bass
1
2
3
4
5
6
7
8
9
10
0
1000
2000
3000
4000
x
0
5
10
15
20




```
#WITHOUT GRAPHS
main_f <- function(x,y) {
  list(all_possible_mse(x,y), all_possible_mse_kernel(x,y))
}
p15 = main_f(pop15, sr)
p75 = main_f(pop75, sr)
dp = main_f(dpi, sr)
ddp = main_f(ddpi, sr)

res_table <- cbind(pop15 = p15[[1]][[1]], pop75 = p75[[1]][[1]], pdi =
dp[[1]][[1]], ddpi = ddp[[1]][[1]])
row.names(res_table) <- c(paste0("Span = ", spanvec[1:3]),
                          paste0("Bass = ", bassvec[1:10]))
res_table1 <- rbind(res_table,
                    min_span = apply(res_table[1:3,], 2, min),
                    min_bass = apply(res_table[4:10,], 2, min),
                    min_span_bass = apply(res_table, 2, min),
                    mean_span_bass = apply(res_table, 2, mean))
res_table1
```

	pop15	pop75	pdi	ddpi
Span = 0.05	10.21983	14.96054	13.32605	11.14913
Span = 0.2	12.86419	15.53902	13.86446	15.27046
Span = 0.5	15.50151	18.90698	17.23007	15.98470
Bass = 1	13.84201	18.34425	16.19331	15.40993
Bass = 2	13.89489	18.36885	16.23625	15.44151
Bass = 3	13.95830	18.39707	16.28457	15.47883
Bass = 4	14.02693	18.42976	16.33938	15.52343
Bass = 5	14.09797	18.46796	16.40233	15.57740
Bass = 6	14.16678	18.51306	16.47622	15.64364
Bass = 7	14.24971	18.56682	16.56900	15.72623
Bass = 8	14.37727	18.63151	16.69689	15.83106
Bass = 9	14.66417	18.71009	16.89925	15.96684
Bass = 10	15.45314	18.80644	17.27655	16.14684
min_span	10.21983	14.96054	13.32605	11.14913
min_bass	13.84201	18.34425	16.19331	15.40993
min_span_bass	10.21983	14.96054	13.32605	11.14913
mean_span_bass	13.94744	18.04941	16.13802	15.31923

```
#ПО ТАБЛИЦЕ ВЫШЕ ДЛЯ КАЖДОЙ ПЕРЕМЕННОЙ МОЖНО ВЫБРАТЬ ЛУЧШУЮ МОДЕЛЬ (В СМЫСЛЕ MSE), ВЫБИРАЯ ТОТ ПАРАМЕТР SPAN ИЛИ BASS, ПРИ КОТОРОМ MSE ДОСТИГАЕТ МИНИМУМА

#ПОСЛЕДНЯЯ СТРОКА ПОЗВОЛЯЕТ РАССМОТРЕТЬ СРЕДНИЕ ЗНАЧЕНИЯ ПОЛУЧЕННЫХ MSE ВСЕХ ПЕРЕМЕННЫХ, А ЗНАЧИТ ВЫБРАТЬ ТЕ ДВЕ ПЕРЕМЕННЫЕ, ДЛЯ КОТОРЫХ ЭТО ЗНАЧЕНИЕ НАИМЕНЬШЕЕ, ОДНАКО, СТОИТ ВКЛЮЧИТЬ В РАССМОТРЕНИЕ РЕЗУЛЬТАТЫ ЯДЕРНОЙ РЕГРЕССИИ И ПОДРОБНЕЕ ИЗУЧИТЬ ГРАФИКИ, СДЕЛАТЬ ЭМПИРИЧЕСКИЙ ВЫВОД

res_table_kernel <- cbind(pop15 = p15[[2]][[1]], pop75 = p75[[2]][[1]], dpi = dp[[2]][[1]], ddpi = ddp[[2]][[1]])

row.names(res_table_kernel) <- c(paste0(kervec[1],", ", bvec[1]),
                                  paste0(kervec[1],", ", bvec[2]),
                                  paste0(kervec[2],", ", bvec[1]),
                                  paste0(kervec[2],", ", bvec[2]))

res_table_kernel1 <- rbind(res_table_kernel,
                           mean_ker = apply(res_table_kernel, 2, mean),
                           min_ker = apply(res_table_kernel, 2, min))
```

	pop15	pop75	dpi	ddpi
epanechnikov, cv.aic	14.57089	16.35921	18.14779	13.96645
epanechnikov, cv.ls	14.57089	16.35551	15.87978	13.51248
gaussian, cv.aic	14.62272	18.97862	17.90112	13.65991
gaussian, cv.ls	14.60803	18.82517	14.45897	14.39146
mean_ker	14.59313	17.62963	16.59691	13.88257
min_ker	14.57089	16.35551	14.45897	13.51248

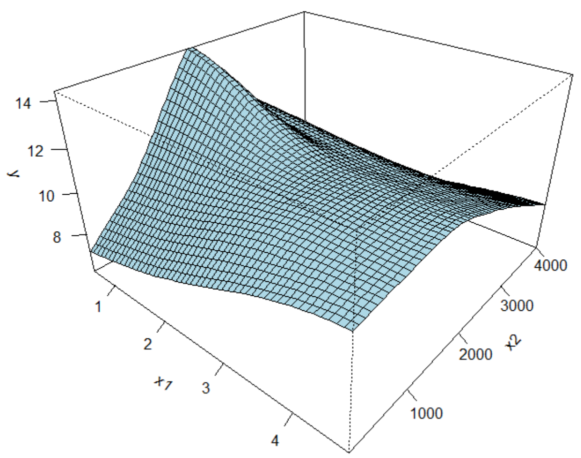
```
table <- rbind(res_table, res_table_kernel)
```

```
rbind(table,
      mean = apply(table, 2, mean),
      min = apply(table, 2, min))

rbind(res_table1, res_table_kernel1)
#ИТОГОВАЯ ТАБЛИЦА
```

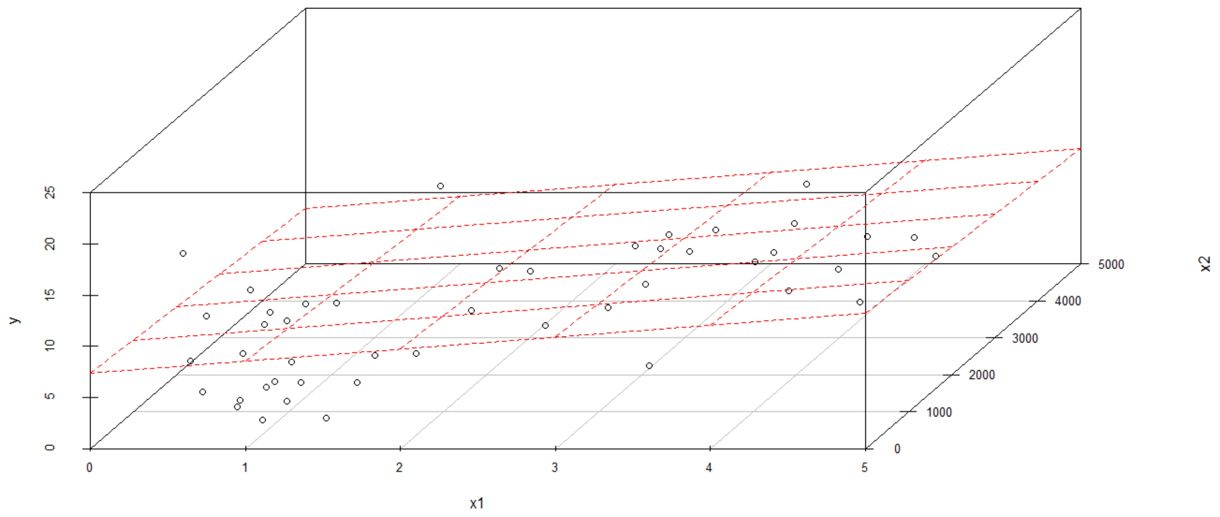
	pop15	pop75	pdi	ddpi
Span = 0.05	10.21983	14.96054	13.32605	11.14913
Span = 0.2	12.86419	15.53902	13.86446	15.27046
Span = 0.5	15.50151	18.90698	17.23007	15.98470
Bass = 1	13.84201	18.34425	16.19331	15.40993
Bass = 2	13.89489	18.36885	16.23625	15.44151
Bass = 3	13.95830	18.39707	16.28457	15.47883
Bass = 4	14.02693	18.42976	16.33938	15.52343
Bass = 5	14.09797	18.46796	16.40233	15.57740
Bass = 6	14.16678	18.51306	16.47622	15.64364
Bass = 7	14.24971	18.56682	16.56900	15.72623
Bass = 8	14.37727	18.63151	16.69689	15.83106
Bass = 9	14.66417	18.71009	16.89925	15.96684
Bass = 10	15.45314	18.80644	17.27655	16.14684
min_span	10.21983	14.96054	13.32605	11.14913
min_bass	13.84201	18.34425	16.19331	15.40993
min_span_bass	10.21983	14.96054	13.32605	11.14913
mean_span_bass	13.94744	18.04941	16.13802	15.31923
epanechnikov, cv.aic	14.57089	16.35921	18.14779	13.96645
epanechnikov, cv.ls	14.57089	16.35551	15.87978	13.51248
gaussian, cv.aic	14.62272	18.97862	17.90112	13.65991
gaussian, cv.ls	14.60803	18.82517	14.45897	14.39146
mean_ker	14.59313	17.62963	16.59691	13.88257
min_ker	14.57089	16.35551	14.45897	13.51248

```
#ПОСЛЕ ТЩАТЕЛЬНОГО ИЗУЧЕНИЯ ГРАФИКОВ БЫЛО РЕШЕНО ВЗЯТЬ В КАЧЕСТВЕ ДВУХ
ПЕРЕМЕННЫХ - POP75 И DPI, ПОСКОЛЬКУ, НАПРИМЕР, ВТОРАЯ ПЕРЕМЕННАЯ ПОЛУЧИЛА
НЕБОЛЬШИЕ ЗНАЧЕНИЯ ОШИБКИ В СЛУЧАЕ ЯДЕРНОЙ РЕГРЕССИИ, КОТОРАЯ БОЛЕЕ ГЛАДКО
ОПИСЫВАЕТ ПОВЕДЕНИЕ ТОЧЕК В ОТЛИЧИЕ ОТ SUPSMU, А ДЛЯ ГРАФИКОВ ПЕРЕМЕННОЙ
POP75 ВСЕ МОДЕЛИ ОКАЗАЛИСЬ КРАЙНЕ ПОХОЖИМИ, ЧТО НЕ СКАЗАТЬ О МОДЕЛЯХ ДРУГИХ
ПЕРЕМЕННЫХ.ТОГДА


#### iv)
x1 <- pop75
x2 <- dpi
y <- sr
```

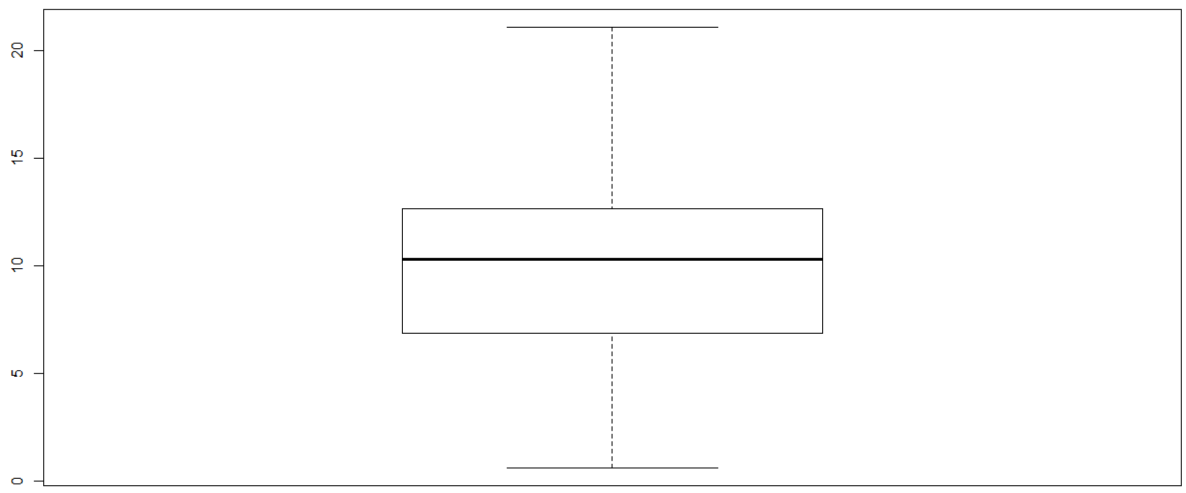
```
M=cbind(x1,x2,y)
if(sum(duplicated(x1)) != 0){
  M=M[-which(duplicated(x1)),]
  x1 <- M[,1]
  x2 <- M[,2]
  y <- M[,3]
}
if(sum(duplicated(x2)) != 0) {
  M=M[-which(duplicated(x2)),]
  x1 <- M[,1]
  x2 <- M[,2]
  y <- M[,3]
}


s2=loess.as(cbind(x1,x2),y,plot=TRUE)
#mean((s2$fitted-y)^2)
```



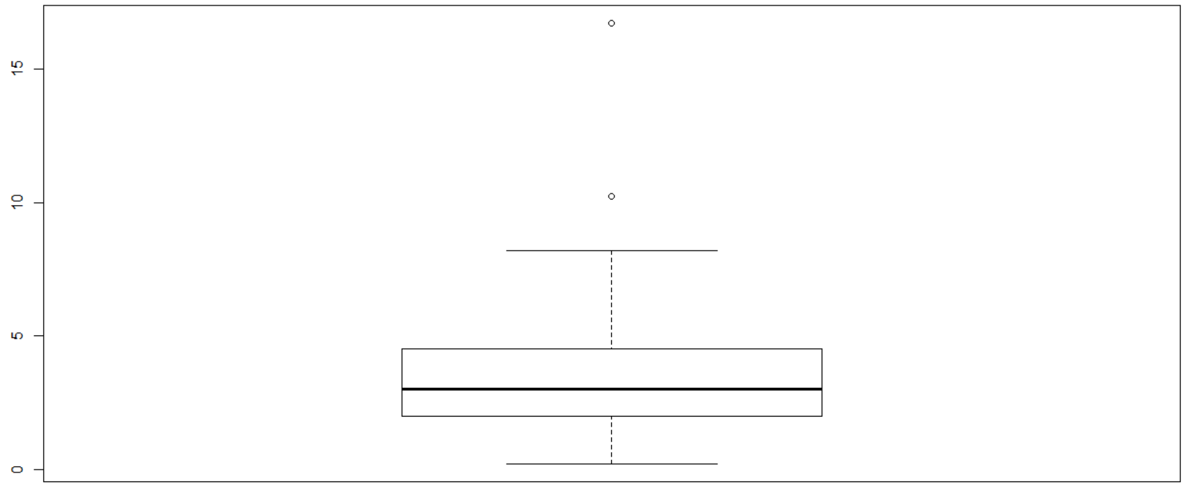

```
##### v)
s=scatterplot3d(x1,x2,y,angle=65)
l=lm(y~x1+x2)
s$plane3d(l, col="red")
#mean((l$fitted.values-y)^2)
```



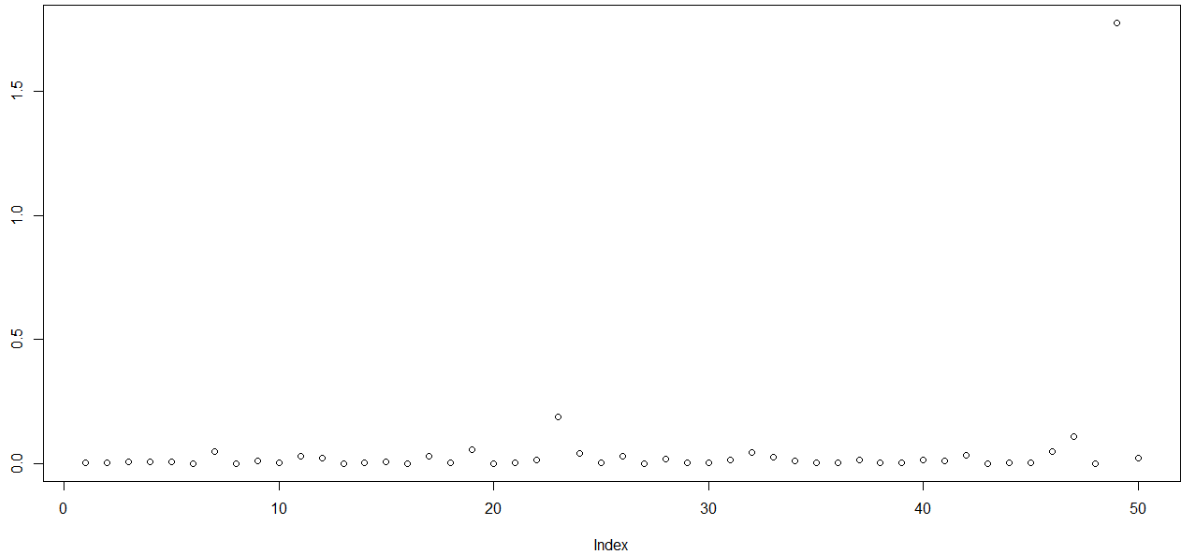

```
###### vi)
boxplot(sr)
```




```
boxplot(ddpi)
```



plot(cooks.distance(lm(sr~ddpi)))



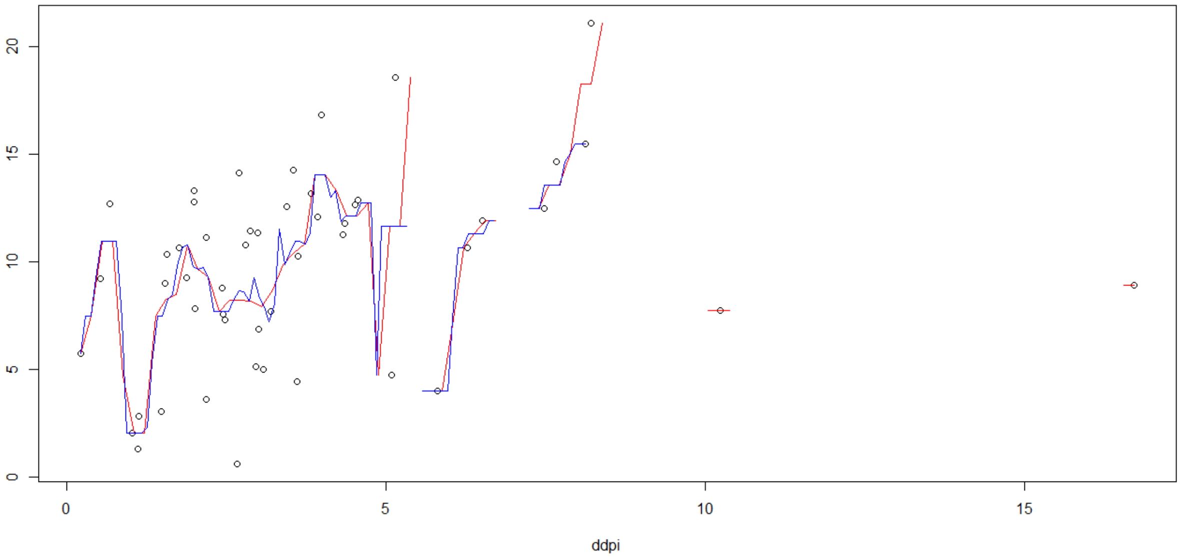
which(cooks.distance(lm(sr~ddpi)) > 4/50)

```
> which(cooks.distance(lm(sr~ddpi)) > 4/50 )
23 47 49
```

plot(ddpi, sr)

lines(ksmooth(ddpi, sr), col = "red")

lines(ksmooth(ddpi[-c(23,47,49)], sr[-which(cooks.distance(lm(sr~ddpi)) > 4/50)]), col = "blue")



#КРАСНАЯ КРИВАЯ ВКЛЮЧЕТ ВЫБРОСЫ, ЗНАЧИТ ОЦЕНКА МЕНЕЕ ТОЧНА

N2

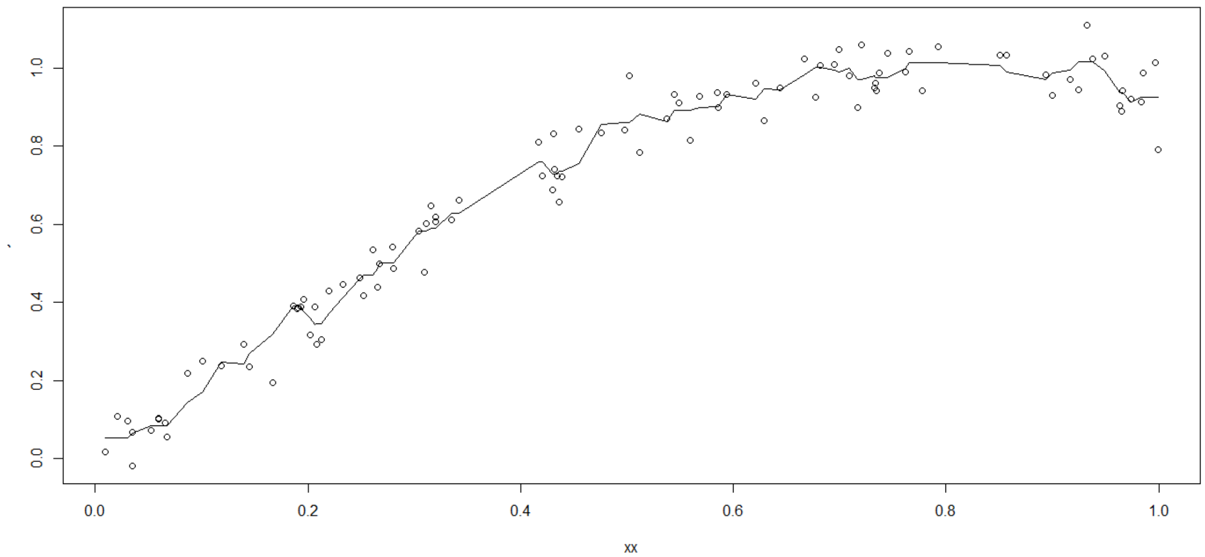
```
N=100
xx=runif(N, min=0, max=1)
e=rnorm(N, sd= 0.05)
y=sin(2*xx)+e
#1
k = 5
#2
xx_diff <- matrix(0, N, N)
abs_xx_diff <- matrix(0, N, N)
sort_a_x_d <- matrix(0, N, N)
xs_give_the_dif <- matrix(0, N, N)
ys <- matrix(0, N, N)

for (i in 1:N) {
  xx_diff[i, ] <- xx - xx[i]
  abs_xx_diff[i, ] <- abs(xx_diff[i, ])
  sort_a_x_d[i, ] <- sort(abs_xx_diff[i, ])
  rownames(xx_diff) <- as.vector(paste0("x", 1:N))
}

for(i in 1:N) {
```

```
  for(j in 1:N) {
    xs_give_the_dif[i,j] <- xx[which(abs_xx_diff[i, ] == sort_a_x_d[i, j])]
    ys[i,j]  <- y[which(xx ==  xs_give_the_dif[i,j])]
  }
}
```

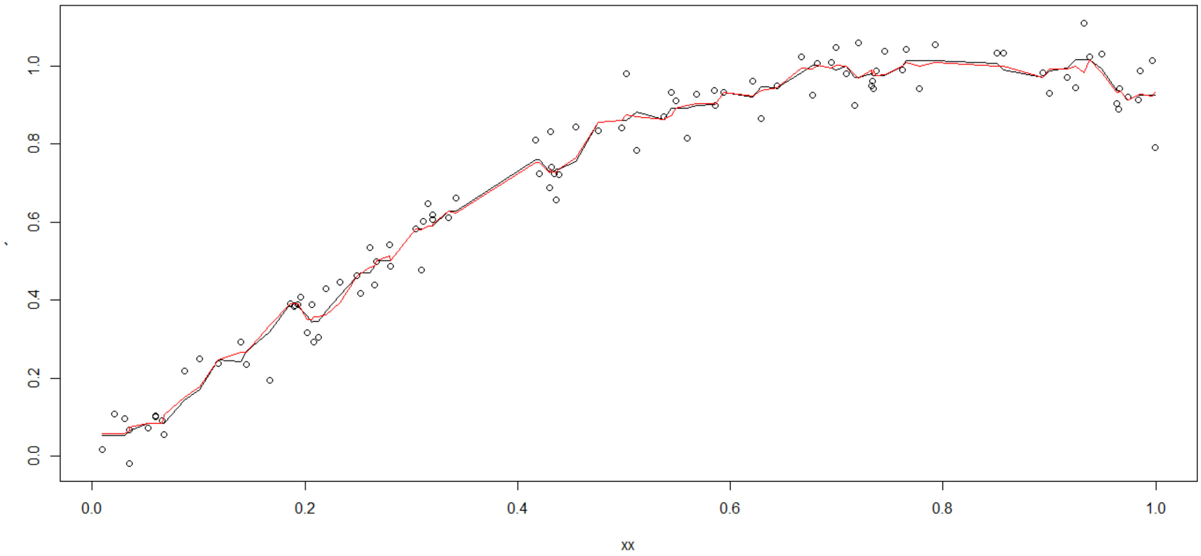
```
> (xx_diff)[1:5, 1:5]
         [,1]         [,2]        [,3]         [,4]        [,5]
x1 0.00000000 -0.446048069 -0.48839813 -0.442428713 -0.01720777
x2 0.44604807  0.000000000 -0.04235007  0.003619356  0.42884030
x3 0.48839813  0.042350066  0.00000000  0.045969422  0.47119036
x4 0.44242871 -0.003619356 -0.04596942  0.000000000  0.42522094
x5 0.01720777 -0.428840298 -0.47119036 -0.425220942  0.00000000
> (abs_xx_diff)[1:5, 1:5]
           [,1]        [,2]       [,3]        [,4]       [,5]
[1,] 0.00000000 0.446048069 0.48839813 0.442428713 0.01720777
[2,] 0.44604807 0.000000000 0.04235007 0.003619356 0.42884030
[3,] 0.48839813 0.042350066 0.00000000 0.045969422 0.47119036
[4,] 0.44242871 0.003619356 0.04596942 0.000000000 0.42522094
[5,] 0.01720777 0.428840298 0.47119036 0.425220942 0.00000000
> (sort_a_x_d)[1:5, 1:5]
     [,1]        [,2]        [,3]        [,4]       [,5]
[1,]    0 0.005000905 0.013096496 0.014521261 0.01720777
[2,]    0 0.003619356 0.012347572 0.016063256 0.01729852
[3,]    0 0.001901238 0.004154007 0.006267448 0.01014526
[4,]    0 0.003619356 0.008728216 0.013679166 0.01488570
[5,]    0 0.004111275 0.010296694 0.017207771 0.02220868
```

```
> xs_give_the_dif[1:10,1:6]
            [,1]       [,2]       [,3]       [,4]       [,5]       [,6]
 [1,] 0.69464224 0.69964315 0.68154575 0.70916350 0.67743447 0.71707723
 [2,] 0.24859417 0.25221353 0.26094174 0.23253092 0.26589269 0.26709923
 [3,] 0.20624411 0.20814534 0.20209010 0.21251156 0.19609885 0.19307428
 [4,] 0.25221353 0.24859417 0.26094174 0.26589269 0.26709923 0.23253092
 [5,] 0.67743447 0.68154575 0.66713778 0.69464224 0.69964315 0.70916350
 [6,] 0.30389245 0.30955651 0.31158603 0.31597478 0.32008080 0.32026735
 [7,] 0.96465915 0.96602593 0.96302952 0.97371860 0.94899109 0.98364125
 [8,] 0.45513778 0.43915604 0.43662272 0.43432534 0.47622206 0.43141592
 [9,] 0.32026735 0.32008080 0.31597478 0.31158603 0.30955651 0.33544335
[10,] 0.05254462 0.05975815 0.05995755 0.06609007 0.06791805 0.03535953
> ys[1:10,1:6]
            [,1]      [,2]      [,3]       [,4]       [,5]        [,6]
 [1,] 1.00936837 1.0476671 1.0070448 0.98059627 0.92495730  0.89808129
 [2,] 0.46357760 0.4185005 0.5354203 0.44529745 0.43814518  0.49886958
 [3,] 0.38966970 0.2923901 0.3164290 0.30398239 0.40750487  0.38973451
 [4,] 0.41850047 0.4635776 0.5354203 0.43814518 0.49886958  0.44529745
 [5,] 0.92495730 1.0070448 1.0229359 1.00936837 1.04766707  0.98059627
 [6,] 0.58354132 0.4774047 0.6020900 0.64727977 0.60570742  0.61962666
 [7,] 0.88934449 0.9425140 0.9040839 0.92010901 1.02930846  0.91217417
 [8,] 0.84437380 0.7204286 0.6562671 0.72441789 0.83294755  0.74036440
 [9,] 0.61962666 0.6057074 0.6472798 0.60209001 0.47740472  0.61051552
[10,] 0.07197216 0.1037404 0.1021555 0.09062141 0.05532943 -0.01776695
```

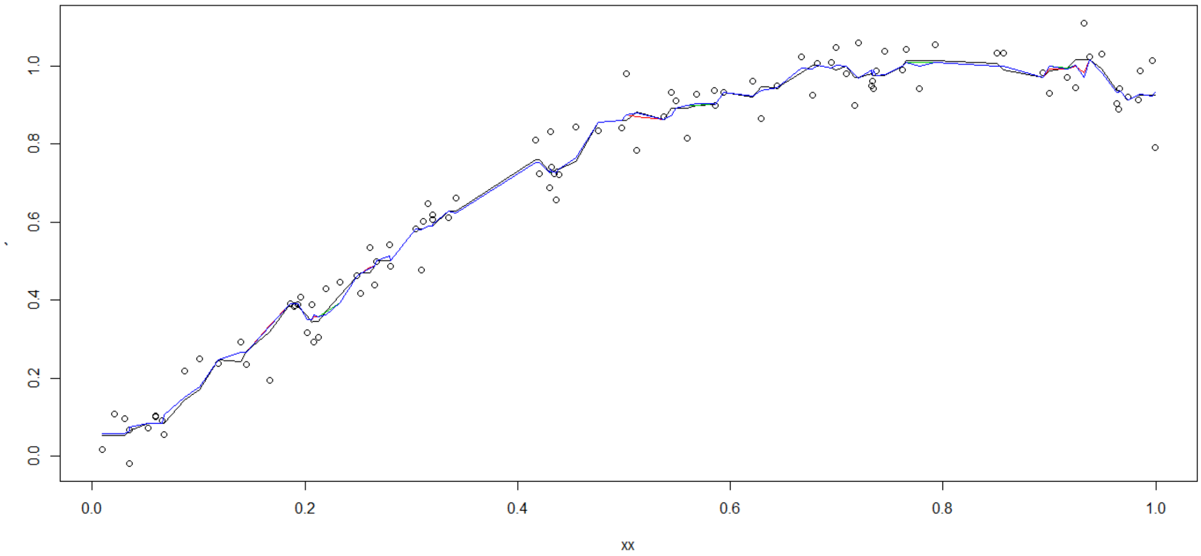
```
#3
fn <- rep(0, N)
for(i in 1:N) {
  fn[i] <- 1/k*sum(ys[i, 1:k])
}
#fn
plot(xx,y)
reor <- reord_y(xx,fn)
lines(reor[,1], reor[,2], col = 1)
```




```
#4
ei <- fn - y
deltaei <- ifelse(abs(ei) <= 1, (1-abs(ei))^4, 0)
#5
fn1 <- rep(0, N)
kk <- rep(0, N)
for(i in 1:N) {
  kk[i] <- round(k/deltaei[i])
  fn1[i] <- 1/(kk[i])*sum(ys[i, 1:kk[i]])
}
fn1
reor <- reord_y(xx,fn1)
lines(reor[,1], reor[,2], col = 2)
```




```
#6
for(j in 1:10) {
  ei1 <- fn1 - y
  deltaei1 <- ifelse(abs(ei1) <= 1, (1-abs(ei1))^4, 0)
  fn1 <- rep(0, N)
  kk <- rep(0, N)
  for(i in 1:N) {
    kk[i] <- round(k/deltaei1[i])
    fn1[i] <- 1/(kk[i])*sum(ys[i, 1:kk[i]])
  }
  reor <- reord_y(xx,fn1)
lines(reor[,1], reor[,2], col = j+2)
}
```



#БОЛЬШАЯ ЧАСТЬ КРИВЫХ СОВПАЛА. КАЧЕСТВО ОЦЕНКИ K-NEAREST NEIGHBOURS ПОДТВЕРДИЛОСЬ.

Task 2

Каломиан В. 1
мсм 181

Theoretical part

$\boxed{№ 1}$ $K(x) = (1-|x|)\cdot \mathbb{I}\{|x| \le 1\}$

$(x_i; y_i)$, $i = 1..6$ $\quad x_i = i$

$H - ?$ $m - ?$ $\quad$ i) $h = 1/2$ $\quad$ ii) $h = 3/2$

□ Nadaraya-Watson estimator:

$$\hat{r}(\bar{x}) = \frac{\sum_i K\left(\frac{x - x_i}{h}\right)\cdot y_i}{\sum_i K\left(\frac{x-x_i}{h}\right)}$$

i)

$x = 1$ $\quad K\left(\frac{1-1}{1/2}\right) = K(0) = 1 \quad K\left(\frac{1-2}{1/2}\right) = K(-2) = 0$

$K\left(\frac{1-3}{1/2}\right) = K(-4) = 0 \ldots = K\left(\frac{1-6}{1/2}\right)$

$x = 2$ $\quad K\left(\frac{2-1}{1/2}\right) = K(2) = 0 \quad K\left(\frac{2-2}{1/2}\right) = K(0) = 1$

$K\left(\frac{2-3}{1/2}\right) = K(-2) = 0 \ldots = K\left(\frac{2-6}{1/2}\right) = 0$

$x = 3$ $\quad K\left(\frac{3-1}{1/2}\right) = K(4) = 0 = \ldots = K\left(\frac{3-6}{1/2}\right)$, кроме

$K\left(\frac{3-3}{1/2}\right) = K(0) = 1 \Rightarrow$

аналогично для остальных x. Получаем, что

$\sum K\left(\frac{x-x_i}{h}\right) = 1 \quad \forall x$ и

$\hat{r}(\bar{x}) = 1\cdot y_1 + \ldots + 1\cdot y_6 \Rightarrow$

$$\hat{r}(\bar{x}) = \begin{pmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_6 \end{pmatrix} = H \begin{pmatrix} y_1 \\ \vdots \\ y_6 \end{pmatrix}, \text{ где } H = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \vdots & & \ddots & & & \vdots \\ 0 & & \cdots & & & 1 \end{pmatrix} = E$$

$\boxed{m = \operatorname{tr} H = 6}$

й)

$x=1$ $\quad K\left(\frac{1-1}{3/2}\right)=K(0)=1 \quad K\left(\frac{1-2}{3/2}\right)=K\left(-\frac{2}{3}\right)=\frac{1}{3}$

$K\left(\frac{1-3}{3/2}\right)=K\left(-\frac{4}{3}\right)=0 \quad \ldots \quad K\left(\frac{1-6}{3/2}\right)=0$

$\sum_i K\left(\frac{x-x_i}{h}\right)=1+\frac{1}{3}=4/3$

$x=2$ $\quad K\left(\frac{2-1}{3/2}\right)=K\left(\frac{2}{3}\right)=1/3 \quad K\left(\frac{2-2}{3/2}\right)=K(0)=1$

$K\left(\frac{2-3}{3/2}\right)=K\left(-\frac{2}{3}\right)=1/3 \quad K\left(\frac{2-4}{3/2}\right)=K\left(-4/3\right)=0\ldots$

$\sum K\left(\frac{2-x_i}{h}\right)=1+2\cdot 1/3=5/3$

$x=3$

$K\left(\frac{3-1}{3/2}\right)=K\left(4/3\right)=0 \quad K\left(\frac{3-2}{3/2}\right)=K\left(2/3\right)=1/3$

$K\left(\frac{3-3}{3/2}\right)=K(0)=1 \quad K\left(\frac{3-4}{3/2}\right)=K\left(-2/3\right)=1/3$

$K\left(\frac{3-5}{3/2}\right)=K\left(-4/3\right)=0 \quad \ldots$

$\sum K\left(\frac{3-x_i}{h}\right)=1/3\cdot 2+1=5/3$

$x=4$ $\quad K\left(\frac{4-1}{3/2}\right)=0 \quad K\left(\frac{4-2}{3/2}\right)=0 \quad K\left(\frac{4-3}{3/2}\right)=1/3$

$K\left(\frac{4-4}{3/2}\right)=1 \quad K\left(\frac{4-5}{3/2}\right)=1/3 \quad K\left(\frac{4-6}{3/2}\right)=0$

$\sum K=5/3$

$x=5$ $\quad K\left(\frac{5-4}{3/2}\right)=1/3 \quad K\left(\frac{5-5}{3/2}\right)=1 \quad K\left(\frac{5-6}{3/2}\right)=1/3$

ост. нули $\quad \sum K=5/3$

$x=6$ $\quad K\left(\frac{6-5}{3/2}\right)=1/3 \quad K\left(\frac{6-6}{3/2}\right)=1$, ост. нули $\Rightarrow$

$\sum K=4/3$

Итого,

~~\ldots~~ $\hat{r}(\vec{x}) = \hat{r}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix} = \hat{r}\begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{pmatrix} =$

$$= \underbrace{\begin{pmatrix} 3/4 & 1/4 & 0 & 0 & 0 & 0 \\ 1/5 & 3/5 & 1/5 & 0 & 0 & 0 \\ 0 & 1/5 & 3/5 & 1/5 & 0 & 0 \\ 0 & 0 & 1/5 & 3/5 & 1/5 & 0 \\ 0 & 0 & 0 & 1/5 & 3/5 & 1/5 \\ 0 & 0 & 0 & 0 & 1/4 & 3/4 \end{pmatrix}}_{H} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{pmatrix}$$

$m = tr H = \frac{3}{4_2} \cdot \cancel{2}^1 + 3/5 \cdot 4 = 3/2^{15} + \frac{12}{5}^{12} = \frac{39}{10} = \boxed{3,9}$

$\boxed{N2}$ $y_i = (X_i + 1)^2 + \varepsilon_i \quad ; i = 1..n$

$E(\varepsilon_1) = .. E(\varepsilon_n) = 0 \quad ; Var(\varepsilon_i) = \sigma^2$

$X_1 .. X_n \sim U[0;1]$

$K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \qquad h^*_{opt} - ?$

Согласно теории:

$\bar{X}_1 .. \bar{X}_n \sim iid \sim f(x)$ и $\{x : f(x) \neq 0\}$ - ограничено

то $MISE = \mathbb{E}\int(\hat{r}_n(x) - r(x))^2 dx \approx$ (с точн. до $\bar{o}(1)$)

$$\approx \frac{h^4}{4}\left(\int x^2 K(x) dx\right)^2 \cdot \int\left(r''(x) + 2r'(x)\frac{f'(x)}{f(x)}\right)^2 dx$$

$$+ \frac{1}{nh} \cdot \sigma^2 \int K^2(x) dx \cdot \int \frac{1}{f(x)} dx$$

$\Rightarrow$ для получения h_{opt} нужно $MISE'_h = 0$

$$r(x) = (x+1)^2 \Rightarrow r'(x) = 2(x+1) \Rightarrow r''(x) = 2$$

$$f(x) = 1 \cdot \mathbb{1}\{0 \le x \le 1\} \Rightarrow f'(x) = 0 \quad \forall x$$

Тогда, подставляя, получаем

$$MISE = \frac{h^4}{4} \cdot \left(\int_{-\infty}^{+\infty} x^2 \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx\right)^2 \cdot \left(\int_0^1 2^2 dx\right) +$$

$$+ \frac{1}{nh} \sigma^2 \int_{-\infty}^{+\infty} \left(\frac{1}{\sqrt{2\pi}} e^{-x^2/2}\right)^2 dx \cdot \int_0^1 1\, dx =$$

$$= \frac{h^4}{4} \cdot \frac{1}{2\pi} \left(\underbrace{\int_{-\infty}^{+\infty} x^2 e^{-x^2/2} dx}_{\sqrt{2\pi}}\right)^2 \cdot 4 + \frac{1}{nh} \sigma^2 \cdot \frac{1}{2\pi} \cdot \sqrt{\pi} \cdot 1 =$$

$$= h^4 + \frac{\sigma^2}{2nh\sqrt{\pi}} \Rightarrow MISE'_h = 0$$

$$4h^3 - \frac{\sigma^2}{2nh^2\sqrt{\pi}} = 0 \Rightarrow h^5 = \frac{\sigma^2}{8n\sqrt{\pi}} \Rightarrow$$

$$\boxed{h_{opt} = \left(\frac{\sigma^2}{8n\sqrt{\pi}}\right)^{1/5}}$$