Semiparametric regression in R (some notes)

Data Mining
Master Degree in Computer Science
University of Padova

a.y. 2017/2018

Annamaria Guolo

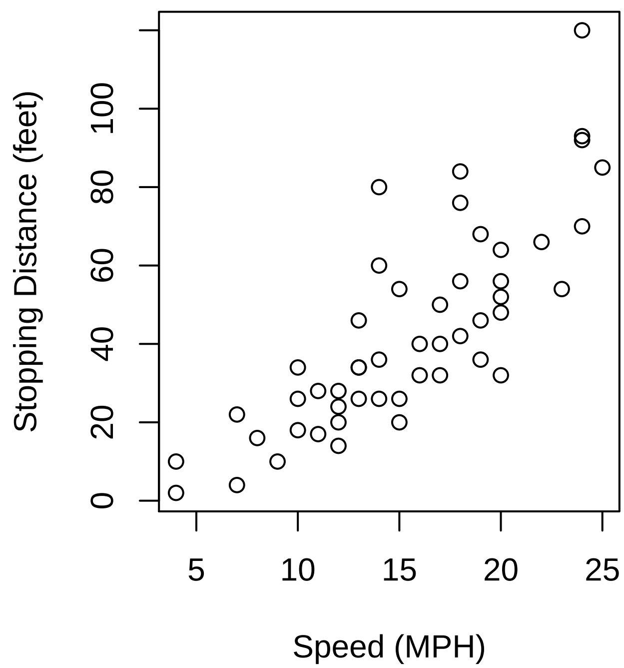
1 cars dataset

Consider dataset cars that contains the information about speed (miles per hour) and stopping distance (in feet) of 50 cars. Data were recorded in the 1920s. We will use this (small) data set to evaluate non-linear relationships between the variables.

```
data(cars)
dim(cars)

## [1] 50  2
```

Graphical evaluation of the relationship between the variables

```
plot(cars$speed, cars$dist, ylab = "Stopping Distance (feet)",
       xlab= "Speed (MPH)")
```



Comments?

Start with a linear regression model

```
m.lm <- lm(dist ~ speed, data=cars)
summary(m.lm)

##
## Call:
## lm(formula = dist ~ speed, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.069  -9.525  -2.272   9.215  43.201
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -17.5791     6.7584  -2.601   0.0123 *
## speed         3.9324     0.4155   9.464 1.49e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.38 on 48 degrees of freedom
## Multiple R-squared:  0.6511,Adjusted R-squared:  0.6438
## F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

and a polynomial with second degree

```
m.poly <- lm(dist ~ poly(speed, 2), data=cars)
summary(m.poly)

##
## Call:
## lm(formula = dist ~ poly(speed, 2), data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.720  -9.184  -3.188   4.628  45.152
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)        42.980      2.146  20.026  < 2e-16 ***
## poly(speed, 2)1  145.552     15.176   9.591 1.21e-12 ***
## poly(speed, 2)2   22.996     15.176   1.515    0.136
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.18 on 47 degrees of freedom
## Multiple R-squared:  0.6673,Adjusted R-squared:  0.6532
## F-statistic: 47.14 on 2 and 47 DF,  p-value: 5.852e-12
```

Estimate a non-linear relationship using a regression spline. We can use the most common choice, that is, a natural spline of order 3, after loading library `splines`

```
library(splines)
m.ns <- lm(dist ~ ns(speed, 3), data=cars)
summary(m.ns)

##
## Call:
## lm(formula = dist ~ ns(speed, 3), data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -26.789  -9.750  -2.421   7.326  44.203
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)       2.594      9.057   0.286 0.775875
## ns(speed, 3)1   35.959      8.731   4.118 0.000157 ***
## ns(speed, 3)2   96.444     20.561   4.691 2.46e-05 ***
```

```
## ns(speed, 3)3   72.986      8.021   9.099 7.49e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.23 on 46 degrees of freedom
## Multiple R-squared:  0.6722,Adjusted R-squared:  0.6508
## F-statistic: 31.44 on 3 and 46 DF,  p-value: 3.298e-11
```

The summary is similar to that from a linear model.
Estimate a smoothing spline, after choosing the degrees of freedom through cross validation.

```
fit.sp.cv <- smooth.spline(x= cars$speed, y=cars$dist, cv=TRUE)

## Warning in smooth.spline(x = cars$speed, y = cars$dist, cv = TRUE): cross-validation
with non-unique 'x' values seems doubtful

names(fit.sp.cv)

##  [1] "x"          "y"          "w"          "yin"        "tol"        "data"
##  [7] "no.weights" "lev"        "cv.crit"    "pen.crit"   "crit"       "df"
## [13] "spar"       "ratio"      "lambda"     "iparms"     "auxM"       "fit"
## [19] "call"

## default LOOCV
```

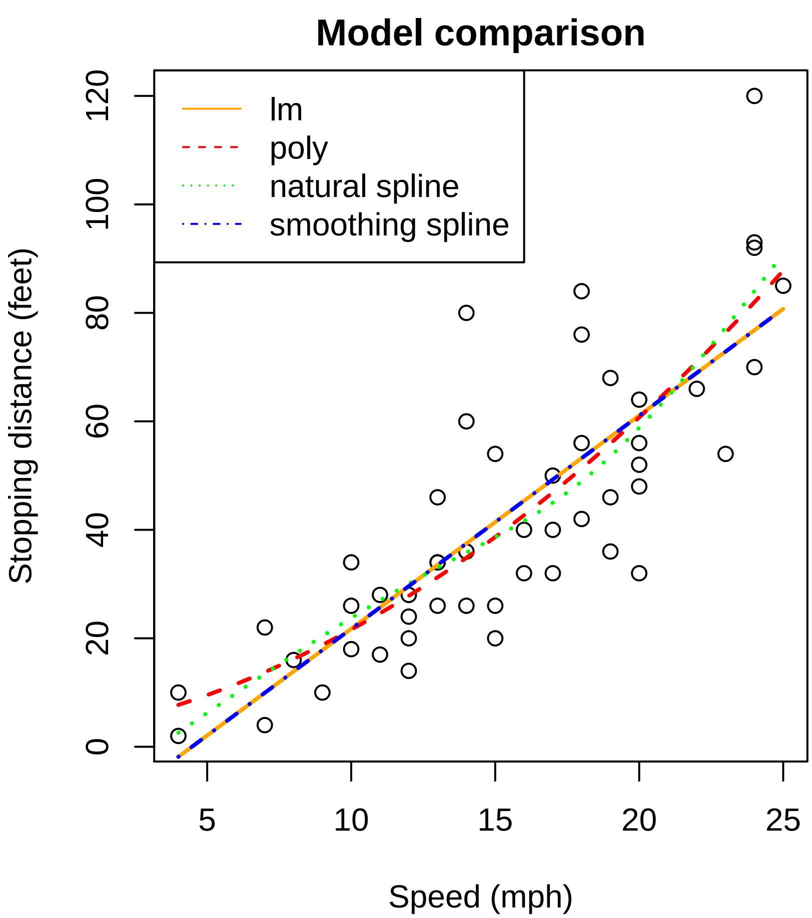
Among the quantities included in `fit.sp.cv`, it is useful to consider the values of λ or the value of the degrees of freedom `df`. Re-estimate the model with `df` chosen by the previous procedure

```
fit.sp <- smooth.spline(x= cars$speed, y=cars$dist, df=fit.sp.cv$df)
fit.sp

## Call:
## smooth.spline(x = cars$speed, y = cars$dist, df = fit.sp.cv$df)
##
## Smoothing Parameter  spar= 1.483527  lambda= 13428.63 (38 iterations)
## Equivalent Degrees of Freedom (Df): 2.000009
## Penalized Criterion (RSS): 4588.73
## GCV: 246.3871
```

Plot the estimated curves

```
## select a grid of speed
new.speed = seq(min(cars$speed),max(cars$speed), length.out=100)
```

```r
plot(cars$speed, cars$dist, ylab = "Stopping distance (feet)",
        xlab= "Speed (mph)", main="Model comparison")
lines(new.speed, predict(m.lm, newdata=data.frame(speed=new.speed)),
        col='orange', lty=1, lwd=2)
lines(new.speed, predict(m.poly, newdata=data.frame(speed=new.speed)),
        col='red', lty=2, lwd=2)
lines(new.speed, predict(m.ns, newdata=data.frame(speed=new.speed)),
        col='green', lty=3, lwd=2)
lines(fit.sp, col='blue', lty=4, lwd=2)
legend('topleft', col=c('orange','red','green','blue'), lty=1:4,
        legend=c('lm','poly','natural spline','smoothing spline'))
```



The grid of values `new.speed` actually is needed to plot the predictions from the splines. Can we increase the knots of the natural spline? We can choose the knots through validation. In order to speed up the procedure, we can consider the following commands (otherwise examine the number of knots one-by-one).

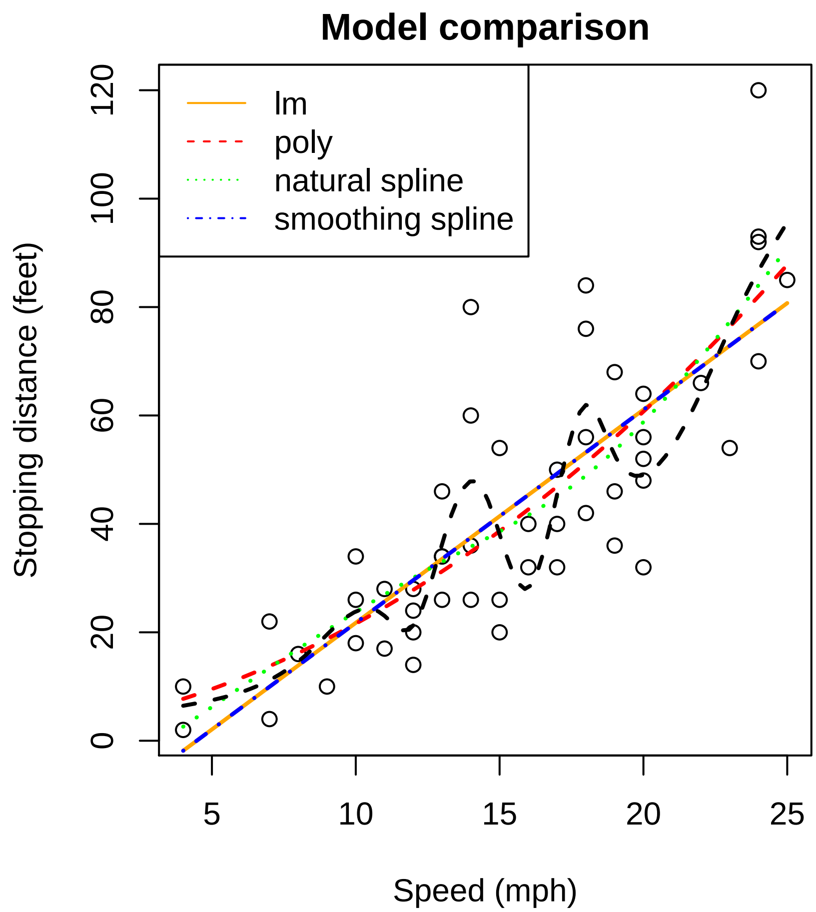
```r
## choose K in ns() by CV
set.seed(2906)
```

```
n <- NROW(cars)
## subdivide the sample of data into training set and test set
id.test <- sample(n, n*0.1)
cars.train <- cars[-id.test,]
cars.test <- cars[id.test,]
## choose the number of knots from 1 to 20
K <- 1:20
## for each knot estimate the natural spline of order k and
## save the corresponding RSS into object rss
rss <- rep(0.0, length(K))
for(i in 1:length(K)){ ## cycle examining each k from 1 to 20
    ## estimate
    m.ns.k <- lm(dist ~ ns(speed, K[i]), data=cars.train)
    ## prediction
    pred <- predict(m.ns.k, newdata=data.frame(speed=cars.test$speed))
    ## save the value of RSS
    rss[i] <- sum((cars.test$dist-pred)^2)
}

## Warning in predict.lm(m.ns.k, newdata = data.frame(speed = cars.test$speed)):
prediction from a rank-deficient fit may be misleading
## Warning in predict.lm(m.ns.k, newdata = data.frame(speed = cars.test$speed)):
prediction from a rank-deficient fit may be misleading

## choose k giving the smoothing spline with the smallest RSS
id <- which.min(rss)
k.min <- K[id]
k.min

## [1] 11

## estimate the model with the chosen k
m.ns.min <- lm(dist ~ ns(speed, k.min), data=cars)
```

Plot the previous graph adding on the prediction with the natural spline of order 11

```
plot(cars$speed, cars$dist, ylab = "Stopping distance (feet)",
        xlab= "Speed (mph)", main="Model comparison")
lines(new.speed, predict(m.lm, newdata=data.frame(speed=new.speed)),
        col='orange', lty=1, lwd=2)
lines(new.speed, predict(m.poly, newdata=data.frame(speed=new.speed)),
        col='red', lty=2, lwd=2)
lines(new.speed, predict(m.ns, newdata=data.frame(speed=new.speed)),
        col='green', lty=3, lwd=2)
lines(fit.sp, col='blue', lty=4, lwd=2)
```

```r
## or lines(predict(fit.sp, new.speed), col='blue', lty=4, lwd=2)
lines(new.speed, predict(m.ns.min, newdata=data.frame(speed=new.speed)),
        col='black', lty=2, lwd=2)
legend('topleft', col=c('orange','red','green','blue'), lty=1:4,
        legend=c('lm','poly','natural spline','smoothing spline'))
```



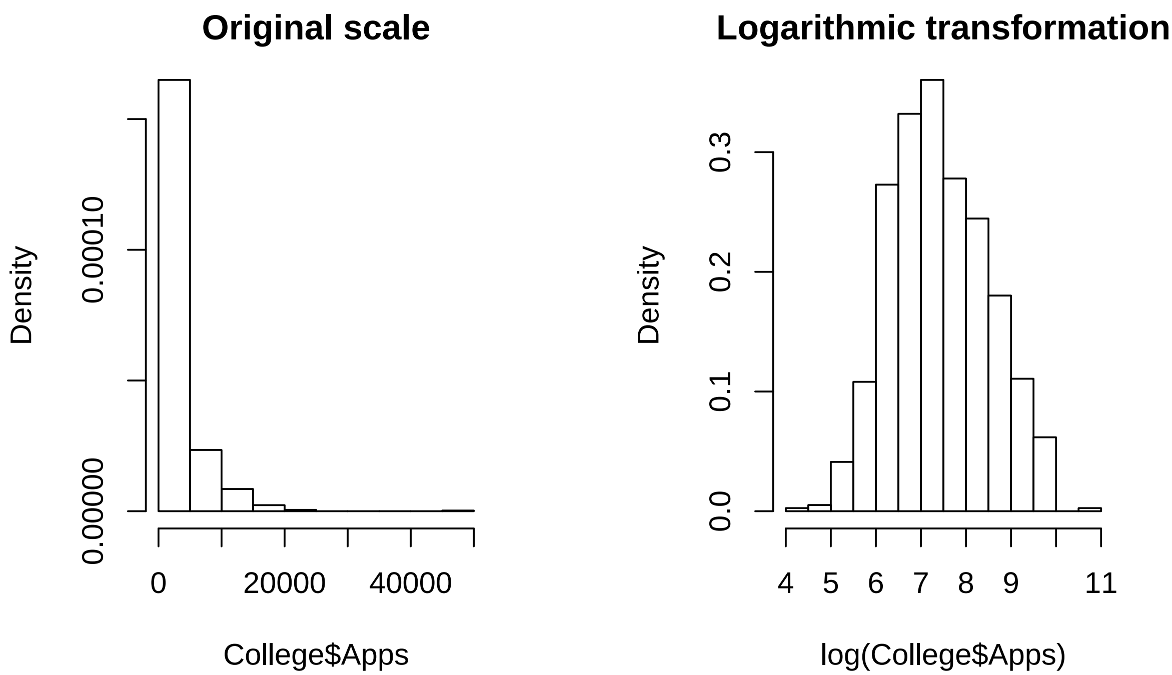
2 College dataset

Consider College dataset already illustrated during a class. The dataset contains the information about 777 US colleges.

```r
library(ISLR)
data(College)
dim(College)

## [1] 777  18
```

The analysis aims at explaining the number of applications `Apps` ai colleges. Choose a logarithmic transformation of variable `Apps` which gives rise to a distribution closer to the normal distribution

```
par(mfrow=c(1,2))
hist(College$Apps, prob=TRUE, main='Original scale')
hist(log(College$Apps), prob=TRUE, main='Logarithmic transformation')
College$Apps <- log(College$Apps)
```

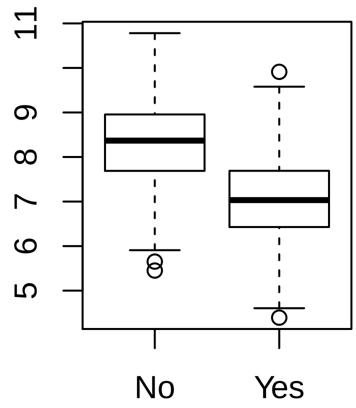


Consider a subset of the covariates

```
college <- College[,c('Apps', 'Private', 'PhD', 'S.F.Ratio', 'Accept')]
summary(college)

##       Apps        Private        PhD           S.F.Ratio         Accept
##  Min.   : 4.394   No :212   Min.   :  8.00   Min.   : 2.50   Min.   :   72
##  1st Qu.: 6.654   Yes:565   1st Qu.: 62.00   1st Qu.:11.50   1st Qu.:  604
##  Median : 7.351             Median : 75.00   Median :13.60   Median : 1110
##  Mean   : 7.427             Mean   : 72.66   Mean   :14.09   Mean   : 2019
##  3rd Qu.: 8.195             3rd Qu.: 85.00   3rd Qu.:16.50   3rd Qu.: 2424
##  Max.   :10.781             Max.   :103.00   Max.   :39.80   Max.   :26330
```

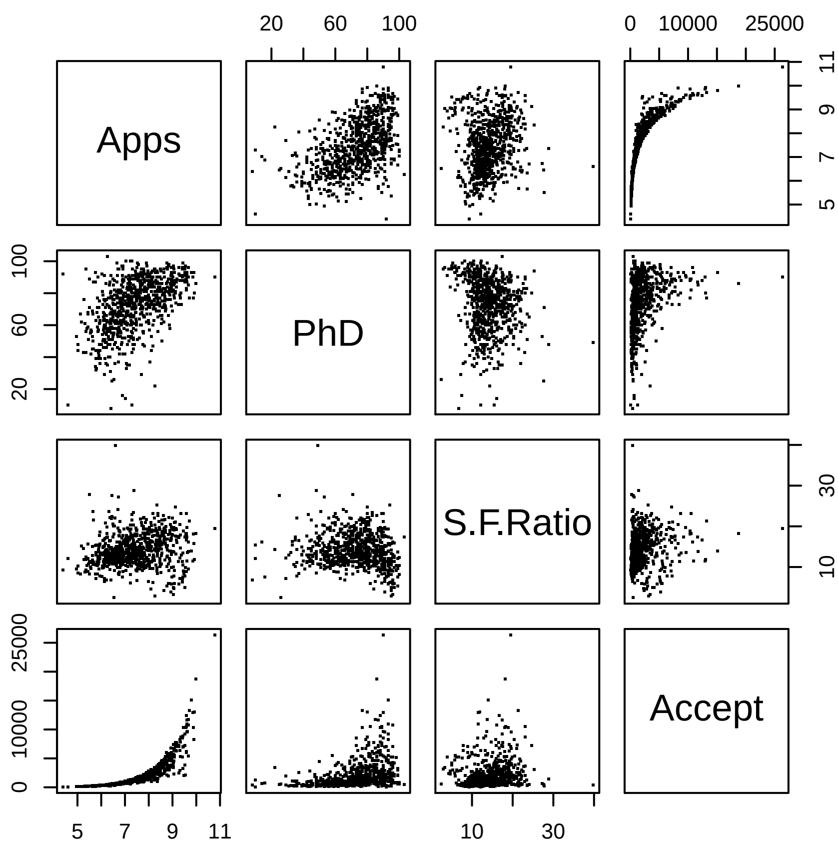
Relationship between `Apps` and `Private`

```
boxplot(college$Apps ~ college$Private)
```



Dispersion plot of the quantitative covariates

```
pairs(college[,-2], pch='.')
```



Do you see any non-linear relationships?

Start with a linear model in the covariates with a second degree polynomial in `Accept`

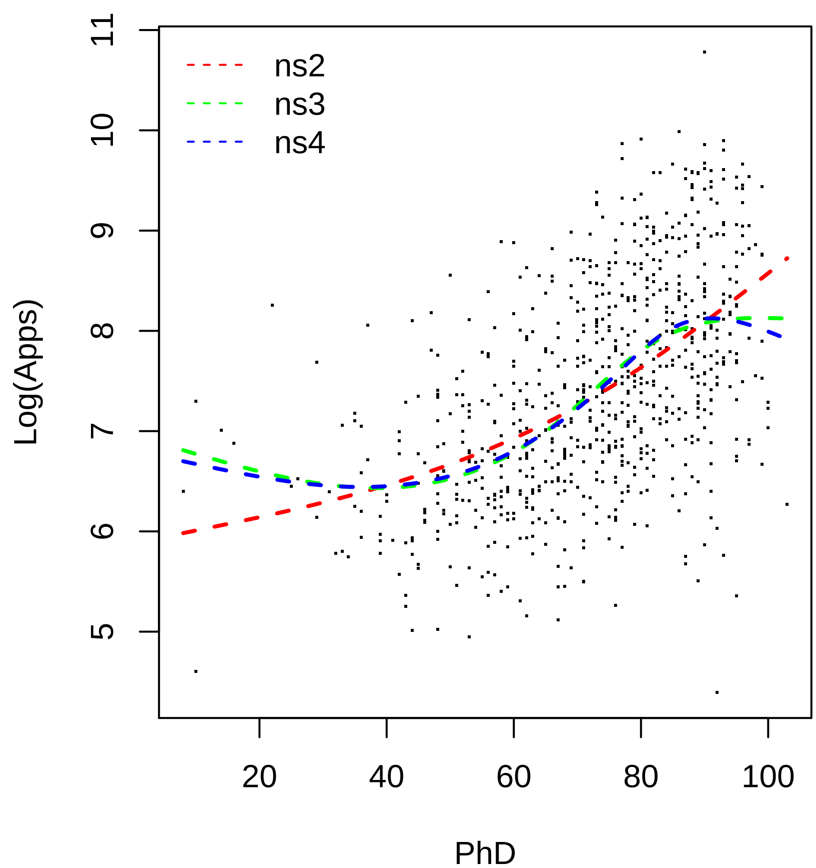
```
m <- lm(Apps ~ Private + PhD + S.F.Ratio + poly(Accept,2), data=college)
summary(m)

##
## Call:
## lm(formula = Apps ~ Private + PhD + S.F.Ratio + poly(Accept,
##     2), data = college)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3222 -0.2472  0.0369  0.2726  3.2141
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          6.461184   0.135771  47.589   <2e-16 ***
## PrivateYes          -0.053282   0.049272  -1.081    0.280
## PhD                  0.012268   0.001156  10.617   <2e-16 ***
## S.F.Ratio            0.008005   0.004936   1.622    0.105
## poly(Accept, 2)1    21.353337   0.570247  37.446   <2e-16 ***
## poly(Accept, 2)2   -10.289682   0.493707 -20.842   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4663 on 771 degrees of freedom
## Multiple R-squared:  0.8126,Adjusted R-squared:  0.8114
## F-statistic: 668.8 on 5 and 771 DF,  p-value: < 2.2e-16
```

Evaluate whether and how to insert the covariates in a non-linear way.
Start with a regression spline, choosing between 2, 3, 4 knots, using AIC

```
phd.ns2 <- lm(Apps ~ ns(PhD, 2), data=college)
phd.ns3 <- lm(Apps ~ ns(PhD, 3), data=college)
phd.ns4 <- lm(Apps ~ ns(PhD, 4), data=college)
extractAIC(phd.ns2)

## [1]    3.0000 -130.1426

extractAIC(phd.ns3)

## [1]    4.0000 -148.3151

extractAIC(phd.ns4)

## [1]    5.0000 -147.5264

## winner: ns 3
```

Graphical inspection

```
plot(college$PhD, college$Apps, xlab='PhD', ylab='Log(Apps)', pch='.')
new.PhD <- seq(min(college$PhD), max(college$PhD), length.out=100)
lines(new.PhD, predict(phd.ns2, newdata=data.frame(PhD=new.PhD)),
        col='red', lty=2, lwd=2)
lines(new.PhD, predict(phd.ns3, newdata=data.frame(PhD=new.PhD)),
        col='green', lty=2, lwd=2)
lines(new.PhD, predict(phd.ns4, newdata=data.frame(PhD=new.PhD)),
        col='blue', lty=2, lwd=2)
legend('topleft', legend=c('ns2','ns3','ns4'),
        col=c('red','green','blue'), lty=c(2,2,2), bty='n')
```



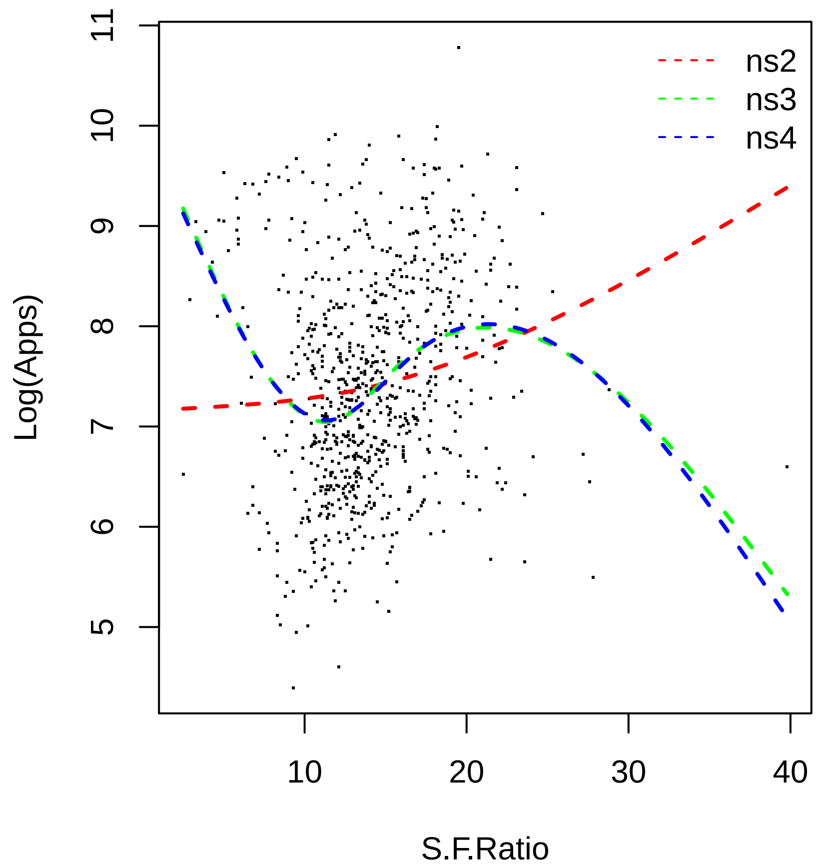
Now move to variable `S.F.Ratio`

```
## natural splines
sf.ns2 <- lm(Apps ~ ns(S.F.Ratio, 2), data=college)
sf.ns3 <- lm(Apps ~ ns(S.F.Ratio, 3), data=college)
sf.ns4 <- lm(Apps ~ ns(S.F.Ratio, 4), data=college)
extractAIC(sf.ns2)
```

```
## [1]  3.00000 93.92657
extractAIC(sf.ns3)
## [1]  4.00000 23.16954
extractAIC(sf.ns4)
## [1]  5.00000 25.09598
## winner: ns 3
```

Graphical inspection

```
plot(college$S.F.Ratio, college$Apps, xlab='S.F.Ratio', ylab='Log(Apps)', pch='.')
new.sfratio <- seq(min(college$S.F.Ratio), max(college$S.F.Ratio), length.out=100)
lines(new.sfratio, predict(sf.ns2,
        newdata=data.frame(S.F.Ratio=new.sfratio)), col='red', lty=2, lwd=2)
lines(new.sfratio, predict(sf.ns3,
        newdata=data.frame(S.F.Ratio=new.sfratio)), col='green', lty=2, lwd=2)
lines(new.sfratio, predict(sf.ns4,
        newdata=data.frame(S.F.Ratio=new.sfratio)), col='blue', lty=2, lwd=2)
legend('topright', legend=c('ns2','ns3','ns4'),
        col=c('red','green','blue'), lty=c(2,2,2), bty='n')
```



Estimate the additive model using the natural splines for `PhD` and `S.F.Ratio` and a quadratic term for `Accept`: this means we estimate a GAM. As we are using natural splines, the reference function is still `lm()`

```
m.ns <- lm(Apps ~ Private + ns(PhD, 3) + ns(S.F.Ratio, 3) + poly(Accept, 2),
        data=college)
summary(m.ns)

##
## Call:
## lm(formula = Apps ~ Private + ns(PhD, 3) + ns(S.F.Ratio, 3) +
##     poly(Accept, 2), data = college)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.35755 -0.24562  0.03378  0.26517  3.07669
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          7.23114    0.21551  33.554  < 2e-16 ***
```

```
## PrivateYes        -0.01831    0.04994  -0.367   0.7140
## ns(PhD, 3)1        0.62037    0.10084   6.152 1.23e-09 ***
## ns(PhD, 3)2        0.95419    0.36224   2.634   0.0086 **
## ns(PhD, 3)3        0.85483    0.11152   7.665 5.41e-14 ***
## ns(S.F.Ratio, 3)1  0.30515    0.12299   2.481   0.0133 *
## ns(S.F.Ratio, 3)2 -0.81222    0.36281  -2.239   0.0255 *
## ns(S.F.Ratio, 3)3 -0.50849    0.36466  -1.394   0.1636
## poly(Accept, 2)1  20.85228    0.57889  36.021  < 2e-16 ***
## poly(Accept, 2)2  -9.99342    0.49444 -20.212  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4614 on 767 degrees of freedom
## Multiple R-squared:  0.8174,Adjusted R-squared:  0.8153
## F-statistic: 381.6 on 9 and 767 DF,  p-value: < 2.2e-16
```

Can we eliminate `Private`?

```
m.ns2 <- lm(Apps ~ ns(PhD, 3) + ns(S.F.Ratio, 3) + poly(Accept, 2), data=college)
anova(m.ns, m.ns2)

## Analysis of Variance Table
##
## Model 1: Apps ~ Private + ns(PhD, 3) + ns(S.F.Ratio, 3) + poly(Accept,
##     2)
## Model 2: Apps ~ ns(PhD, 3) + ns(S.F.Ratio, 3) + poly(Accept, 2)
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    767 163.31
## 2    768 163.34 -1 -0.028627 0.1344  0.714

## yes, we can
summary(m.ns2)

##
## Call:
## lm(formula = Apps ~ ns(PhD, 3) + ns(S.F.Ratio, 3) + poly(Accept,
##     2), data = college)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.35478 -0.24648  0.03743  0.26953  3.08279
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         7.2135     0.2099  34.361  < 2e-16 ***
```

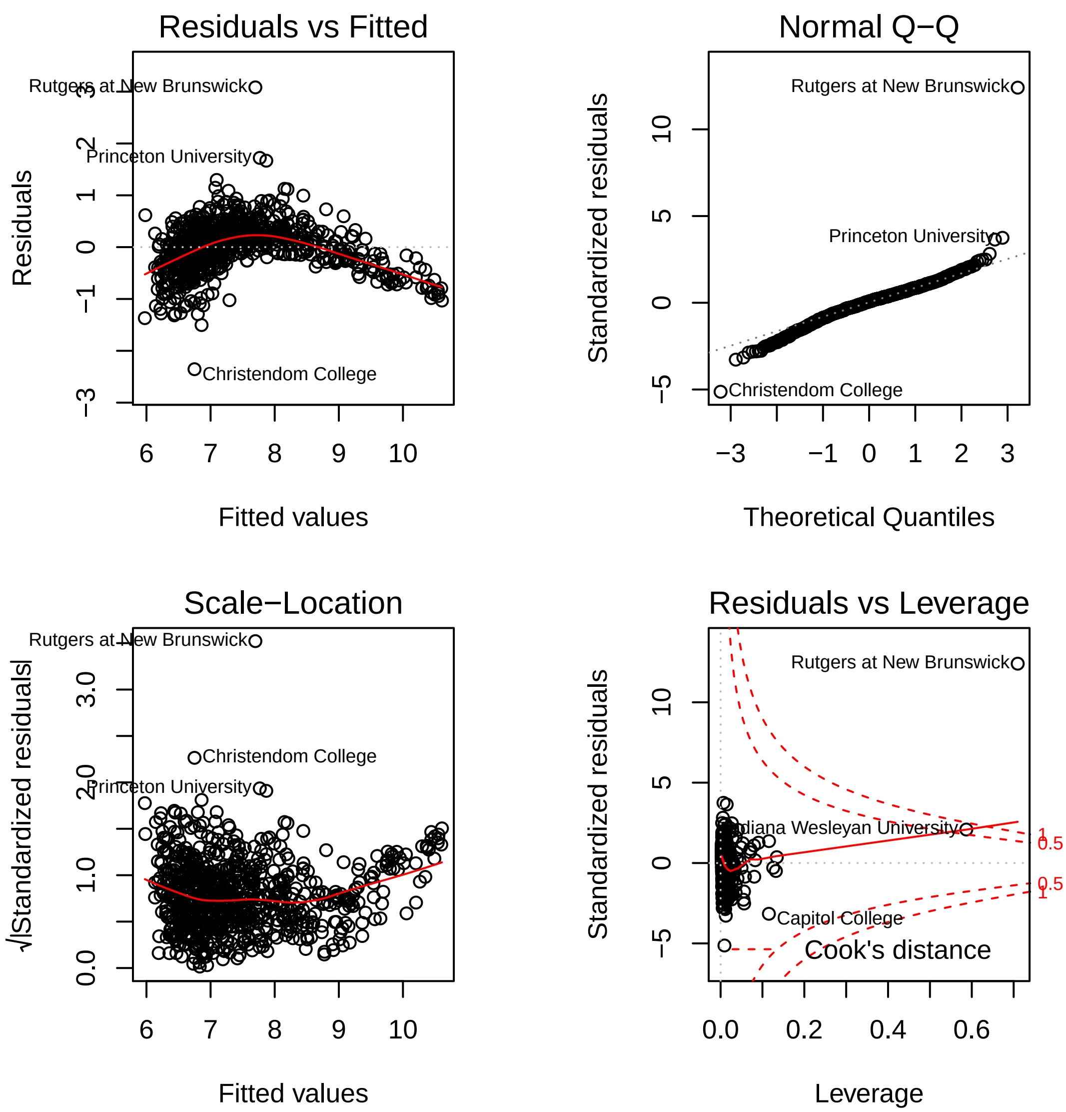
```
## ns(PhD, 3)1         0.6223     0.1006   6.184 1.02e-09 ***
## ns(PhD, 3)2         0.9564     0.3620   2.642  0.00841 **
## ns(PhD, 3)3         0.8551     0.1115   7.672 5.15e-14 ***
## ns(S.F.Ratio, 3)1   0.3226     0.1134   2.846  0.00455 **
## ns(S.F.Ratio, 3)2  -0.8074     0.3624  -2.228  0.02616 *
## ns(S.F.Ratio, 3)3  -0.5054     0.3644  -1.387  0.16580
## poly(Accept, 2)1   20.9340     0.5340  39.203  < 2e-16 ***
## poly(Accept, 2)2  -10.0300     0.4840 -20.724  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4612 on 768 degrees of freedom
## Multiple R-squared:  0.8174,Adjusted R-squared:  0.8155
## F-statistic: 429.8 on 8 and 768 DF,  p-value: < 2.2e-16
```

Can we reduce the order of the natural spline for `S.F.Ratio`?

```
m.ns3 <- lm(Apps ~ ns(PhD, 3) + ns(S.F.Ratio, 2) + poly(Accept, 2), data=college)
anova(m.ns2, m.ns3)

## Analysis of Variance Table
##
## Model 1: Apps ~ ns(PhD, 3) + ns(S.F.Ratio, 3) + poly(Accept, 2)
## Model 2: Apps ~ ns(PhD, 3) + ns(S.F.Ratio, 2) + poly(Accept, 2)
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1    768 163.34
## 2    769 166.67 -1   -3.3275 15.645 8.349e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## no, we can't
```

Residual analysis `m.ns`

```
par(mfrow=c(2,2))
plot(m.ns2)
```



Comments? Try to improve the model by inserting a natural spline for `Accept`

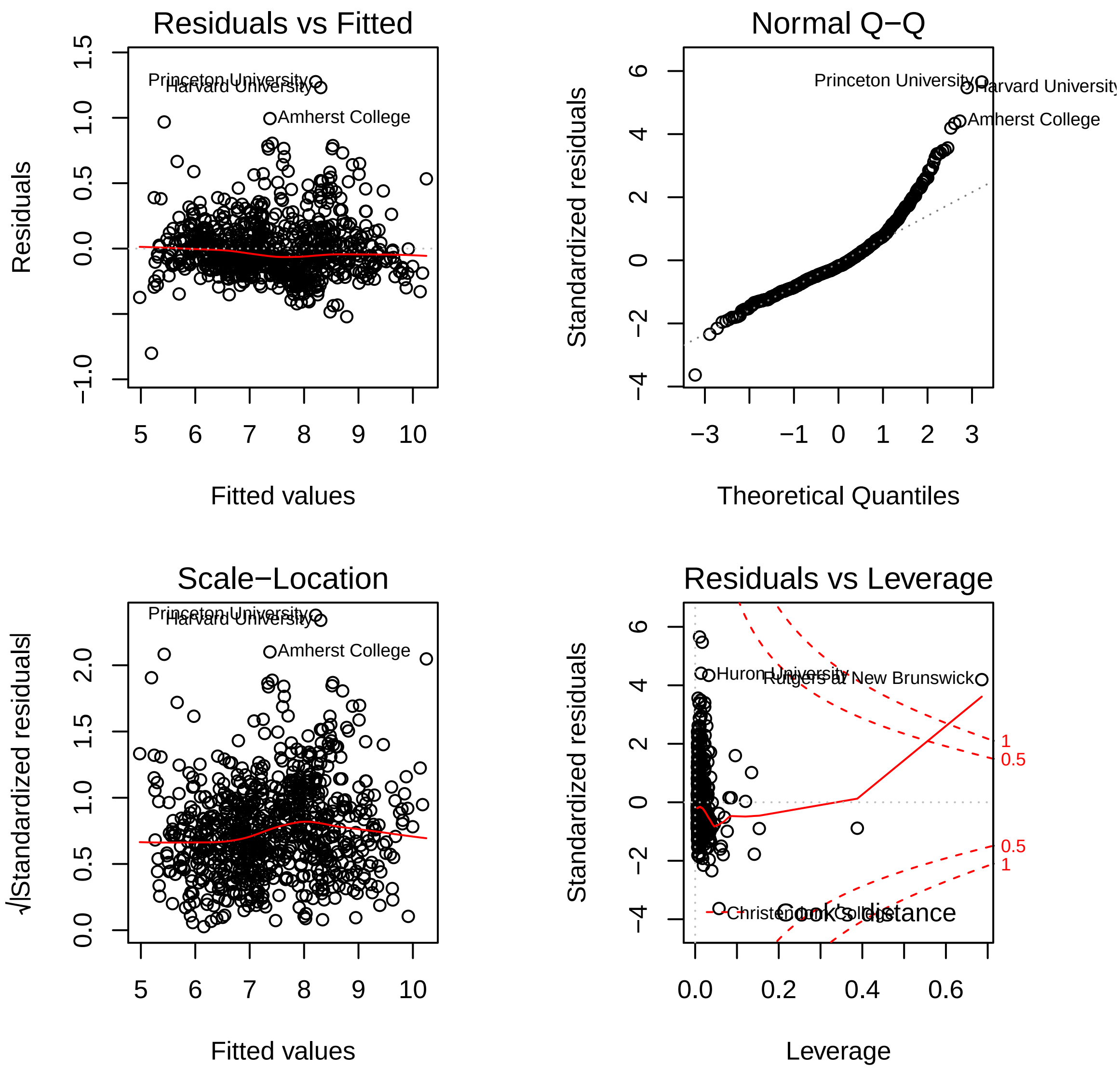
```
## natural splines
accept.ns3 <- lm(Apps ~ ns(Accept, 3), data=college)
accept.ns4 <- lm(Apps ~ ns(Accept, 4), data=college)
accept.ns5 <- lm(Apps ~ ns(Accept, 5), data=college)
extractAIC(accept.ns3)

## [1]     4.000 -2100.384

extractAIC(accept.ns4)

## [1]     5.000 -2153.596

extractAIC(accept.ns5)

## [1]     6.000 -2205.942

## we choose ns 5
```

Re-estimate the model with all natural splines

```
m.ns4 <- lm(Apps ~ ns(PhD, 3) + ns(S.F.Ratio, 2) + ns(Accept, 5), data=college)
summary(m.ns4)

##
## Call:
## lm(formula = Apps ~ ns(PhD, 3) + ns(S.F.Ratio, 2) + ns(Accept,
##     5), data = college)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.80045 -0.13396 -0.04132  0.09428  1.27619
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)         5.17426    0.09633  53.713  < 2e-16 ***
## ns(PhD, 3)1         0.05842    0.05066   1.153  0.24919
## ns(PhD, 3)2         0.39190    0.17902   2.189  0.02889 *
## ns(PhD, 3)3         0.30365    0.05604   5.419 8.04e-08 ***
## ns(S.F.Ratio, 2)1 -0.41417    0.09700  -4.270 2.20e-05 ***
## ns(S.F.Ratio, 2)2  0.45043    0.13988   3.220  0.00134 **
## ns(Accept, 5)1      2.11807    0.05185  40.852  < 2e-16 ***
## ns(Accept, 5)2      2.99507    0.05927  50.529  < 2e-16 ***
## ns(Accept, 5)3      4.59606    0.08164  56.298  < 2e-16 ***
## ns(Accept, 5)4      6.59798    0.14430  45.725  < 2e-16 ***
## ns(Accept, 5)5      4.46085    0.19169  23.271  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2269 on 766 degrees of freedom
## Multiple R-squared:  0.9559,Adjusted R-squared:  0.9553
## F-statistic:  1661 on 10 and 766 DF,  p-value: < 2.2e-16

anova(m.ns2, m.ns4)

## Analysis of Variance Table
##
## Model 1: Apps ~ ns(PhD, 3) + ns(S.F.Ratio, 3) + poly(Accept, 2)
## Model 2: Apps ~ ns(PhD, 3) + ns(S.F.Ratio, 2) + ns(Accept, 5)
##   Res.Df     RSS Df Sum of Sq    F    Pr(>F)
## 1    768 163.344
## 2    766  39.445  2     123.9 1203 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

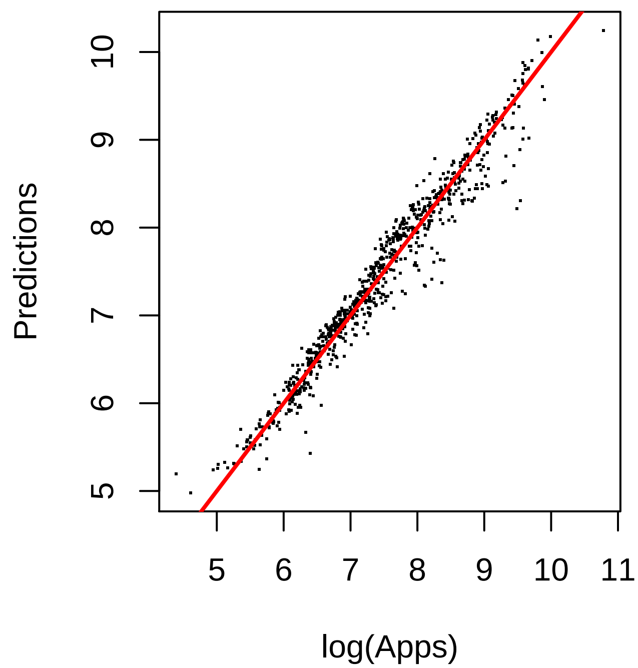
Residual analysis

```
par(mfrow=c(2,2))
plot(m.ns4)
```



Comments?
Predictions on the training set

```
plot(college$Apps, predict(m.ns4), xlab='log(Apps)', ylab='Predictions', pch='.')
abline(0, 1, col='red', lwd=2)
```



Try to estimate a model using the smoothing splines. Start with a smoothing spline for `PhD`, choosing λ (or the degrees of freedom) through cross validation

```
phd.cv = smooth.spline(x= college$PhD, y=college$Apps, cv=TRUE)

## Warning in smooth.spline(x = college$PhD, y = college$Apps, cv = TRUE): cross-validat
with non-unique 'x' values seems doubtful

phd.cv

## Call:
## smooth.spline(x = college$PhD, y = college$Apps, cv = TRUE)
##
## Smoothing Parameter  spar= 0.7964489  lambda= 0.020413 (14 iterations)
## Equivalent Degrees of Freedom (Df): 5.257
## Penalized Criterion (RSS): 66.28162
## PRESS(l.o.o. CV): 0.8298997

## estimate of the spline
phd.fit <- smooth.spline(x= college$PhD, y=college$Apps, df=phd.cv$df)
```

Smoothing spline for `S.F.Ratio`

```
set.seed(111)
sf.cv = smooth.spline(x= college$S.F.Ratio, y=college$Apps, cv=TRUE)
```

```
## Warning in smooth.spline(x = college$S.F.Ratio, y = college$Apps, cv = TRUE):
cross-validation with non-unique 'x' values seems doubtful

sf.cv

## Call:
## smooth.spline(x = college$S.F.Ratio, y = college$Apps, cv = TRUE)
##
## Smoothing Parameter  spar= 0.9363268  lambda= 0.007357639 (16 iterations)
## Equivalent Degrees of Freedom (Df): 5.712937
## Penalized Criterion (RSS): 200.9153
## PRESS(l.o.o. CV): 1.029901

sf.fit <- smooth.spline(x= college$S.F.Ratio, y=college$Apps, df=sf.cv$df)
```

Using the smoothing splines, the GAM is estimated using functionalities in library `gam`.

```
library(gam)
```

The syntax is the same used to fit the linear model, the only difference being we need function `s() to estimate the smoothing splines`

```
m.gam <- gam(Apps ~ Private + s(PhD, 5) + s(S.F.Ratio, 6) + poly(Accept, 2),
        data=college)
summary(m.gam)

##
## Call: gam(formula = Apps ~ Private + s(PhD, 5) + s(S.F.Ratio, 6) +
##     poly(Accept, 2), data = college)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.30245 -0.24215  0.03238  0.26289  3.06226
##
## (Dispersion Parameter for gaussian family taken to be 0.2103)
##
##     Null Deviance: 894.5761 on 776 degrees of freedom
## Residual Deviance: 160.2744 on 762.0002 degrees of freedom
## AIC: 1010.494
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                   Df Sum Sq Mean Sq F value    Pr(>F)
## Private            1 193.40 193.396 919.471 < 2.2e-16 ***
## s(PhD, 5)          1 171.84 171.839 816.985 < 2.2e-16 ***
```

```
## s(S.F.Ratio, 6)   1   1.56   1.555   7.395  0.006689 **
## poly(Accept, 2)   2 333.07 166.535 791.765 < 2.2e-16 ***
## Residuals       762 160.27   0.210
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                 Npar Df Npar F     Pr(F)
## (Intercept)
## Private
## s(PhD, 5)             4 2.9293 0.0202094 *
## s(S.F.Ratio, 6)       5 4.1686 0.0009626 ***
## poly(Accept, 2)
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The output of `summary()` provides some well-known quantities. The other components are the results of the hypothesis tests to evaluate whether each component is significant (`Anova for Parametric Effects`) and to evaluate whether a linear relationship is acceptable for components non-linearly inserted `Anova for Nonparametric Effects`. Results are always reported in terms of p-value. In our example, there is no need to eliminate any variable, neither to linearly insert variables.
Try a smoothing spline for `Accept`

```
accept.cv <- smooth.spline(x= college$Accept, y=college$Apps, cv=TRUE)

## Warning in smooth.spline(x = college$Accept, y = college$Apps, cv = TRUE):
cross-validation with non-unique 'x' values seems doubtful

accept.fit <- smooth.spline(x= college$Accept, y=college$Apps, df=accept.cv$df)
accept.fit

## Call:
## smooth.spline(x = college$Accept, y = college$Apps, df = accept.cv$df)
##
## Smoothing Parameter  spar= 0.8755007  lambda= 3.312272e-06 (12 iterations)
## Equivalent Degrees of Freedom (Df): 21.38246
## Penalized Criterion (RSS): 38.73522
## GCV: 0.05700537
```

Many degrees of freedom

```
m.gam2 <- gam(Apps ~ Private + s(PhD, 5) + s(S.F.Ratio, 6) + s(Accept, 21),
        data=college)
summary(m.gam2)
```

```
##
## Call: gam(formula = Apps ~ Private + s(PhD, 5) + s(S.F.Ratio, 6) +
##     s(Accept, 21), data = college)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -0.79635 -0.13145 -0.03762  0.09315  1.27763
##
## (Dispersion Parameter for gaussian family taken to be 0.0475)
##
##     Null Deviance: 894.5761 on 776 degrees of freedom
## Residual Deviance: 35.2717 on 743 degrees of freedom
## AIC: -127.733
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                  Df Sum Sq Mean Sq   F value  Pr(>F)
## Private           1 120.87  120.87 2546.0849 < 2e-16 ***
## s(PhD, 5)         1  65.29   65.29 1375.3054 < 2e-16 ***
## s(S.F.Ratio, 6)   1   0.15    0.15    3.1936 0.07434 .
## s(Accept, 21)     1 376.98  376.98 7941.1205 < 2e-16 ***
## Residuals       743  35.27    0.05
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                 Npar Df  Npar F     Pr(F)
## (Intercept)
## Private
## s(PhD, 5)             4   5.347 0.0003003 ***
## s(S.F.Ratio, 6)       5   9.001 2.563e-08 ***
## s(Accept, 21)        20 237.255 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Model comparison

```
anova(m.gam, m.gam2)

## Analysis of Deviance Table
##
## Model 1: Apps ~ Private + s(PhD, 5) + s(S.F.Ratio, 6) + poly(Accept, 2)
## Model 2: Apps ~ Private + s(PhD, 5) + s(S.F.Ratio, 6) + s(Accept, 21)
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       762    160.274
```

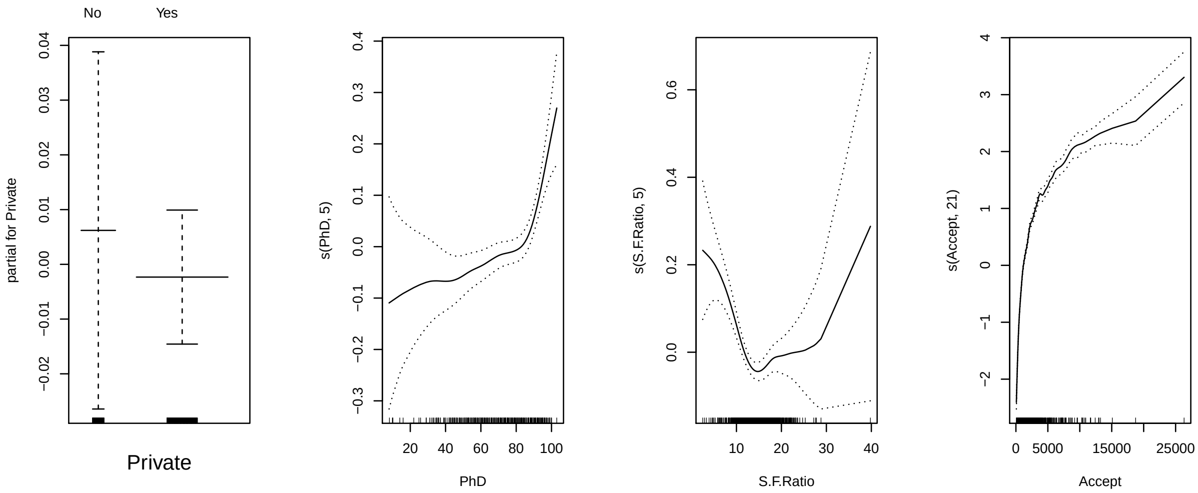
```
## 2       743     35.272 19      125 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We maintain model `m.gam2`. Try to reduce the order of the spline in `S.F.Ratio`

```
m.gam3 <- gam(Apps ~ Private + s(PhD, 5) + s(S.F.Ratio, 5) + s(Accept, 21),
        data=college)
summary(m.gam3)

##
## Call: gam(formula = Apps ~ Private + s(PhD, 5) + s(S.F.Ratio, 5) +
##     s(Accept, 21), data = college)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -0.79244 -0.13097 -0.03971  0.09032  1.28767
##
## (Dispersion Parameter for gaussian family taken to be 0.0476)
##
##     Null Deviance: 894.5761 on 776 degrees of freedom
## Residual Deviance: 35.3983 on 743.9995 degrees of freedom
## AIC: -126.9481
##
## Number of Local Scoring Iterations: 2
##
## Anova for Parametric Effects
##                  Df Sum Sq Mean Sq   F value  Pr(>F)
## Private           1 121.39  121.39 2551.4570 < 2e-16 ***
## s(PhD, 5)         1  65.49   65.49 1376.4648 < 2e-16 ***
## s(S.F.Ratio, 5)   1   0.16    0.16    3.2607 0.07137 .
## s(Accept, 21)     1 377.12  377.12 7926.3150 < 2e-16 ***
## Residuals       744  35.40    0.05
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                 Npar Df  Npar F     Pr(F)
## (Intercept)
## Private
## s(PhD, 5)             4   5.429 0.0002596 ***
## s(S.F.Ratio, 5)       4  10.480 2.902e-08 ***
## s(Accept, 21)        20 236.452 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

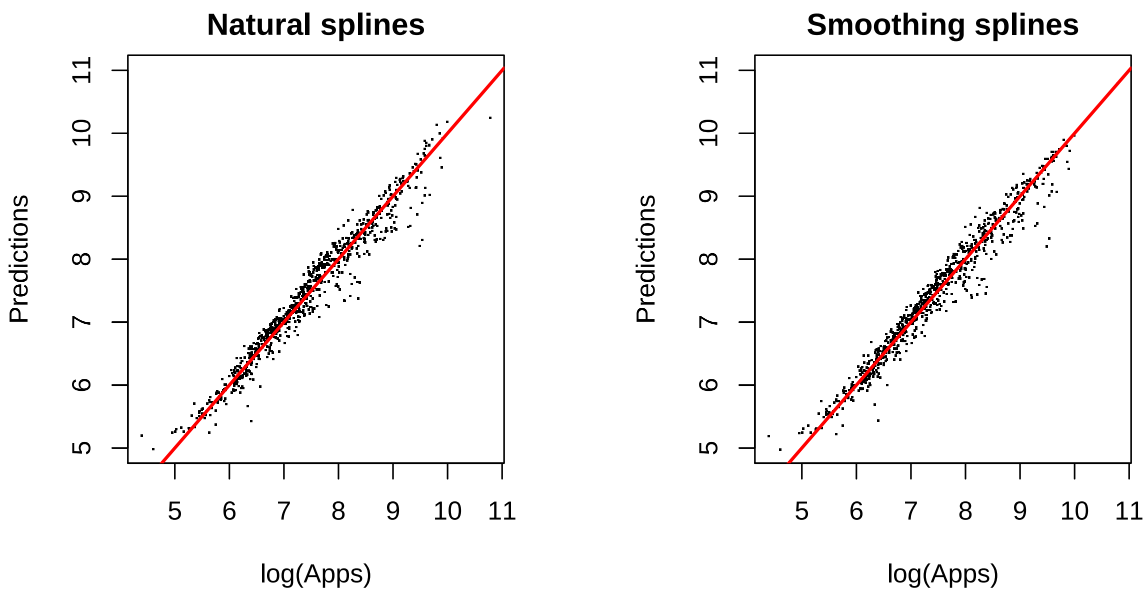
Comments?
Graphical analysis

```
par(mfrow=c(1,4))
plot(m.gam3, se=TRUE)
```



Option `se=TRUE` plots upper and lower point-wise twice-standard-error curves.
How can we comment on?
Predictions on the training set using the model with natural splines and the model with smoothing splines

```
par(mfrow=c(1,2))
plot(college$Apps, predict(m.ns4), xlab='log(Apps)', ylab='Predictions',
        pch='.', main='Natural splines', ylim=c(5, 11))
## ylim=c(5, 11) specifies lower and upper limits for the y-axis
## in order to compare the graphs
abline(0, 1, col='red', lwd=2)
plot(college$Apps, predict(m.gam3), xlab='log(Apps)', ylab='Predictions',
        pch='.', main='Smoothing splines', ylim=c(5, 11))
abline(0, 1, col='red', lwd=2)
```



How to apply the analysis in case of logistic regression model?
Construct variable `HighApps` distinguishing `Apps` above 8 or not (on the original scale).

```
HighApps <- College$Apps> 8
table(HighApps)

## HighApps
## FALSE  TRUE
##   544   233
```

Generalized additive model: just specify option `family='binomial'`

```
glm.gam <- gam(HighApps ~ Private + s(PhD, 5) + s(S.F.Ratio, 6) +
        s(Accept, 21), data=college, family='binomial')
summary(glm.gam)

##
## Call: gam(formula = HighApps ~ Private + s(PhD, 5) + s(S.F.Ratio, 6) +
##     s(Accept, 21), family = "binomial", data = college)
## Deviance Residuals:
##        Min         1Q     Median         3Q        Max
## -8.490e+00 -2.107e-08 -2.107e-08  2.107e-08  8.490e+00
##
## (Dispersion Parameter for binomial family taken to be 1)
##
##     Null Deviance: 949.1136 on 776 degrees of freedom
## Residual Deviance: 3676.453 on 743 degrees of freedom
## AIC: 3744.453
##
```

```
## Number of Local Scoring Iterations: 30
##
## Anova for Parametric Effects
##                    Df     Sum Sq    Mean Sq  F value    Pr(>F)
## Private             1 3.0711e+16 3.0711e+16  98.7134 < 2.2e-16 ***
## s(PhD, 5)           1 5.8141e+15 5.8141e+15  18.6879 1.749e-05 ***
## s(S.F.Ratio, 6)     1 2.4058e+13 2.4058e+13   0.0773     0.781
## s(Accept, 21)       1 1.0658e+17 1.0658e+17 342.5796 < 2.2e-16 ***
## Residuals         743 2.3116e+17 3.1111e+14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                  Npar Df Npar Chisq    P(Chi)
## (Intercept)
## Private
## s(PhD, 5)              4 3.8533e+15 < 2.2e-16 ***
## s(S.F.Ratio, 6)        5 5.9629e+15 < 2.2e-16 ***
## s(Accept, 21)         20 1.0947e+17 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

and go on with usual model selection, evaluation of the accuracy of the model, ... For example

```
glm.gam2 <- gam(HighApps ~ Private + s(PhD, 5) + s(S.F.Ratio, 5) +
        s(Accept, 21), data=college, family='binomial')
summary(glm.gam2)

##
## Call: gam(formula = HighApps ~ Private + s(PhD, 5) + s(S.F.Ratio, 5) +
##     s(Accept, 21), family = "binomial", data = college)
## Deviance Residuals:
##        Min         1Q     Median         3Q        Max
## -8.490e+00 -2.107e-08 -2.107e-08  2.107e-08  8.490e+00
##
## (Dispersion Parameter for binomial family taken to be 1)
##
##     Null Deviance: 949.1136 on 776 degrees of freedom
## Residual Deviance: 4901.937 on 743.9995 degrees of freedom
## AIC: 4967.938
##
## Number of Local Scoring Iterations: 30
##
## Anova for Parametric Effects
```

```
##                   Df     Sum Sq    Mean Sq  F value    Pr(>F)    
## Private            1 3.7680e+16 3.7680e+16 134.8720 < 2.2e-16 ***
## s(PhD, 5)          1 1.4398e+16 1.4398e+16  51.5346 1.707e-12 ***
## s(S.F.Ratio, 5)    1 1.7498e+14 1.7498e+14   0.6263     0.429    
## s(Accept, 21)      1 1.0764e+17 1.0764e+17 385.2887 < 2.2e-16 ***
## Residuals        744 2.0786e+17 2.7938e+14                       
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Anova for Nonparametric Effects
##                  Npar Df Npar Chisq    P(Chi)    
## (Intercept)                                      
## Private                                          
## s(PhD, 5)              4 1.4299e+15 < 2.2e-16 ***
## s(S.F.Ratio, 5)        4 1.6208e+15 < 2.2e-16 ***
## s(Accept, 21)         20 9.2314e+16 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## it seems reasonable to move to a smoothing spline with 5 df for S.F.Ratio
```

```
glm.gam3 <- gam(HighApps ~ Private + s(PhD, 5) + s(S.F.Ratio, 4) +
        s(Accept, 21), data=college, family='binomial')
summary(glm.gam3)
```

```
## 
## Call: gam(formula = HighApps ~ Private + s(PhD, 5) + s(S.F.Ratio, 4) + 
##     s(Accept, 21), family = "binomial", data = college)
## Deviance Residuals:
##        Min         1Q     Median         3Q        Max 
## -8.490e+00 -2.107e-08 -2.107e-08  2.107e-08  8.490e+00 
## 
## (Dispersion Parameter for binomial family taken to be 1)
## 
##     Null Deviance: 949.1136 on 776 degrees of freedom
## Residual Deviance: 3964.802 on 744.9994 degrees of freedom
## AIC: 4028.803 
## 
## Number of Local Scoring Iterations: 30 
## 
## Anova for Parametric Effects
##                   Df     Sum Sq    Mean Sq F value    Pr(>F)    
## Private            1 2.7270e+16 2.7270e+16 128.349 < 2.2e-16 ***
## s(PhD, 5)          1 1.5473e+16 1.5473e+16  72.823 < 2.2e-16 ***
```

```
## s(S.F.Ratio, 4)   1 5.3193e+15 5.3193e+15  25.036 7.027e-07 ***
## s(Accept, 21)     1 7.5310e+16 7.5310e+16 354.452 < 2.2e-16 ***
## Residuals       745 1.5829e+17 2.1247e+14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                 Npar Df Npar Chisq    P(Chi)
## (Intercept)
## Private
## s(PhD, 5)             4 1.1007e+15 < 2.2e-16 ***
## s(S.F.Ratio, 4)       3 4.3756e+15 < 2.2e-16 ***
## s(Accept, 21)        20 9.6816e+16 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## it seems reasonable to move to a smoothing spline with 4 df for S.F.Ratio
```