7 Non-Linear Models

By: Udit (based on ISLR)

Setup

```
library(ISLR2)
library(splines)   # using splines
library(gam)       # using GAM models
```

```
## Loading required package: foreach
```

```
## Loaded gam 1.20
```

```
library(akima)     # surface plots

attach(Wage)
dim(Wage)
```

```
## [1] 3000   11
```

Polynomial Regression

Regression of Wage ~ Age, upto polynomial of power 4. Polynomial vs. Spline - most important drawback of spline being non-locality. That is the fitted function at a given value x0 depends on data values far from that point.

```
fit.poly <- lm(wage~poly(age,4), data=Wage)
summary(fit.poly)
```

```
##
## Call:
## lm(formula = wage ~ poly(age, 4), data = Wage)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -98.707 -24.626  -4.993  15.217 203.693
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    111.7036     0.7287 153.283  < 2e-16 ***
## poly(age, 4)1  447.0679    39.9148  11.201  < 2e-16 ***
## poly(age, 4)2 -478.3158    39.9148 -11.983  < 2e-16 ***
## poly(age, 4)3  125.5217    39.9148   3.145  0.00168 **
```

```
## poly(age, 4)4  -77.9112    39.9148  -1.952  0.05104 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.91 on 2995 degrees of freedom
## Multiple R-squared:  0.08626,    Adjusted R-squared:  0.08504
## F-statistic: 70.69 on 4 and 2995 DF,  p-value: < 2.2e-16
```

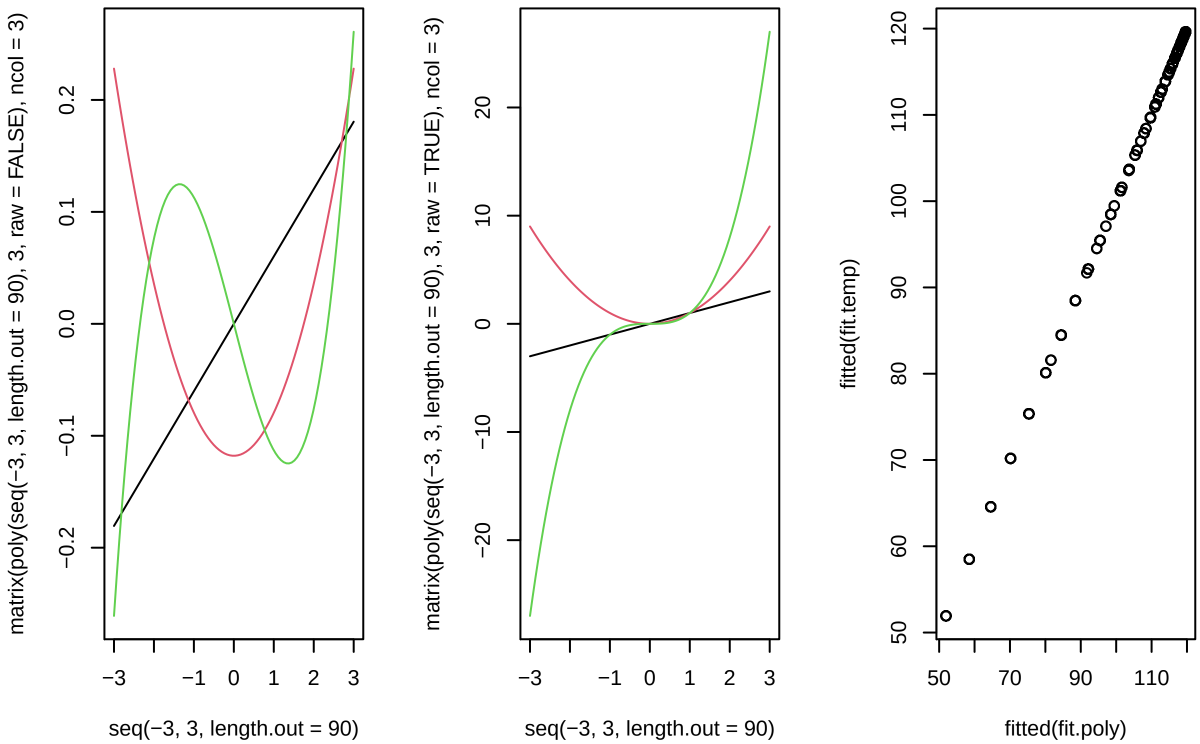
```
#Raw polynomials .. poly(age,4, raw=TRUE) or:
fit.temp = lm(wage~age+I(age^2)+I(age^3)+I(age^4),data=Wage) #I - treat it as-is
summary(fit.temp)
```

```
##
## Call:
## lm(formula = wage ~ age + I(age^2) + I(age^3) + I(age^4), data = Wage)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -98.707 -24.626  -4.993  15.217 203.693
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.842e+02  6.004e+01  -3.067 0.002180 **
## age          2.125e+01  5.887e+00   3.609 0.000312 ***
## I(age^2)    -5.639e-01  2.061e-01  -2.736 0.006261 **
## I(age^3)     6.811e-03  3.066e-03   2.221 0.026398 *
## I(age^4)    -3.204e-05  1.641e-05  -1.952 0.051039 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.91 on 2995 degrees of freedom
## Multiple R-squared:  0.08626,    Adjusted R-squared:  0.08504
## F-statistic: 70.69 on 4 and 2995 DF,  p-value: < 2.2e-16
```

Function **poly()** generates a basis of *orthogonal polynomials*, which is preferred. With orthogonal polynomials we can separately test each coefficient. In this case power-4 coefficient is not significant.

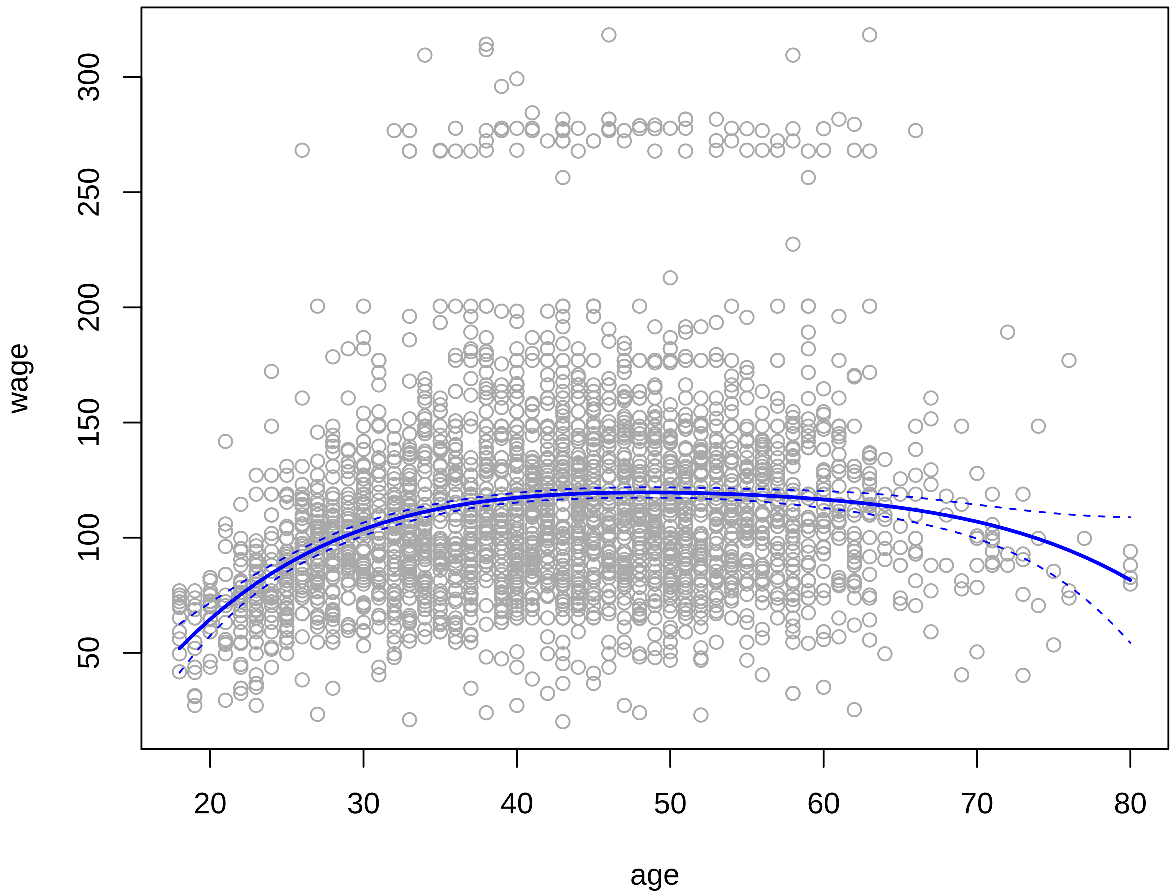
References:
* Visualizing orthogonal polynomials link
* Raw vs. Orthogonal link

```
#Poly() produces an orthogonal set of basis functions. For example:
par(mfrow=c(1,3))
matplot(seq(-3,3,length.out=90),
        matrix(poly(seq(-3,3,length.out=90),3, raw=FALSE), ncol=3),
        lty = 1, pch = 1, type="l")

matplot(seq(-3,3,length.out=90),
        matrix(poly(seq(-3,3,length.out=90),3, raw=TRUE), ncol=3),
        lty = 1, pch = 1, type="l")

# However the fitted values would be consistent.
plot(fitted(fit.poly), fitted(fit.temp))
```



Plotting the fitted function.

```
agelims=range(age)
age.grid = seq(from =agelims[1], to=agelims[2])
preds = predict(fit.poly, newdata=list(age=age.grid), se=TRUE)
se.bands = cbind(preds$fit+2*preds$se, preds$fit-2*preds$se)

#Plotting
plot(age,wage, col="darkgrey")
lines(age.grid, preds$fit, lwd=2, col="blue")   #line-width
matlines(age.grid, se.bands, col="blue", lty=2)  #line-type
```



Using **anova()** and **F-test** to test for significance of different variables in a series of nested models.

Fit models ranging from linear to a degree-5 polynomial and seek to determine the simplest model which is sufficient to explain the relationship between wage and age.

Upto Age^3 appears to be significant.

```
fit.1 <- lm(wage~age, data=Wage)
fit.2 <- lm(wage~poly(age,2), data=Wage)
fit.3 <- lm(wage~poly(age,3), data=Wage)
fit.4 <- lm(wage~poly(age,4), data=Wage)
fit.5 <- lm(wage~poly(age,5), data=Wage)
anova(fit.1, fit.2, fit.3, fit.4, fit.5)
```

```
## Analysis of Variance Table
##
## Model 1: wage ~ age
## Model 2: wage ~ poly(age, 2)
## Model 3: wage ~ poly(age, 3)
## Model 4: wage ~ poly(age, 4)
```

```
## Model 5: wage ~ poly(age, 5)
##   Res.Df     RSS Df Sum of Sq        F    Pr(>F)
## 1   2998 5022216
## 2   2997 4793430  1    228786 143.5931 < 2.2e-16 ***
## 3   2996 4777674  1     15756   9.8888  0.001679 **
## 4   2995 4771604  1      6070   3.8098  0.051046 .
## 5   2994 4770322  1      1283   0.8050  0.369682
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Since polynomials are orthogonal, we could have simply used the p-values from degree-5 fit to review the results. F-stat is equal to $t - stat^2$

```
coef ( summary (fit.5))
```

```
##                   Estimate Std. Error     t value     Pr(>|t|)
## (Intercept)      111.70361  0.7287647 153.2780243 0.000000e+00
## poly(age, 5)1   447.06785 39.9160847  11.2001930 1.491111e-28
## poly(age, 5)2 -478.31581 39.9160847 -11.9830341 2.367734e-32
## poly(age, 5)3  125.52169 39.9160847   3.1446392 1.679213e-03
## poly(age, 5)4  -77.91118 39.9160847  -1.9518743 5.104623e-02
## poly(age, 5)5  -35.81289 39.9160847  -0.8972045 3.696820e-01
```

```
(-11.9830341)^2
```

```
## [1] 143.5931
```

ANOVA method works whether or not we used orthogonal polynomials; it also works when we have other terms in the model as well.

Age^3 is not-significant when other variables are included.

```
fit.a <- lm(wage~education, data=Wage)
fit.b <- lm(wage~education+age, data=Wage)
fit.c <- lm(wage~education+poly(age,2), data=Wage)
fit.d <- lm(wage~education+poly(age,3), data=Wage)
anova(fit.a, fit.b, fit.c, fit.d)
```

```
## Analysis of Variance Table
##
## Model 1: wage ~ education
## Model 2: wage ~ education + age
## Model 3: wage ~ education + poly(age, 2)
## Model 4: wage ~ education + poly(age, 3)
##   Res.Df     RSS Df Sum of Sq        F Pr(>F)
## 1   2995 3995721
## 2   2994 3867992  1    127729 102.7378 <2e-16 ***
## 3   2993 3725395  1    142597 114.6969 <2e-16 ***
## 4   2992 3719809  1      5587   4.4936 0.0341 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(fit.a)
```

```
## Analysis of Variance Table
##
## Response: wage
##             Df  Sum Sq Mean Sq F value    Pr(>F)
## education    4 1226364  306591  229.81 < 2.2e-16 ***
## Residuals 2995 3995721    1334
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Polynomial - LOGISTIC regression

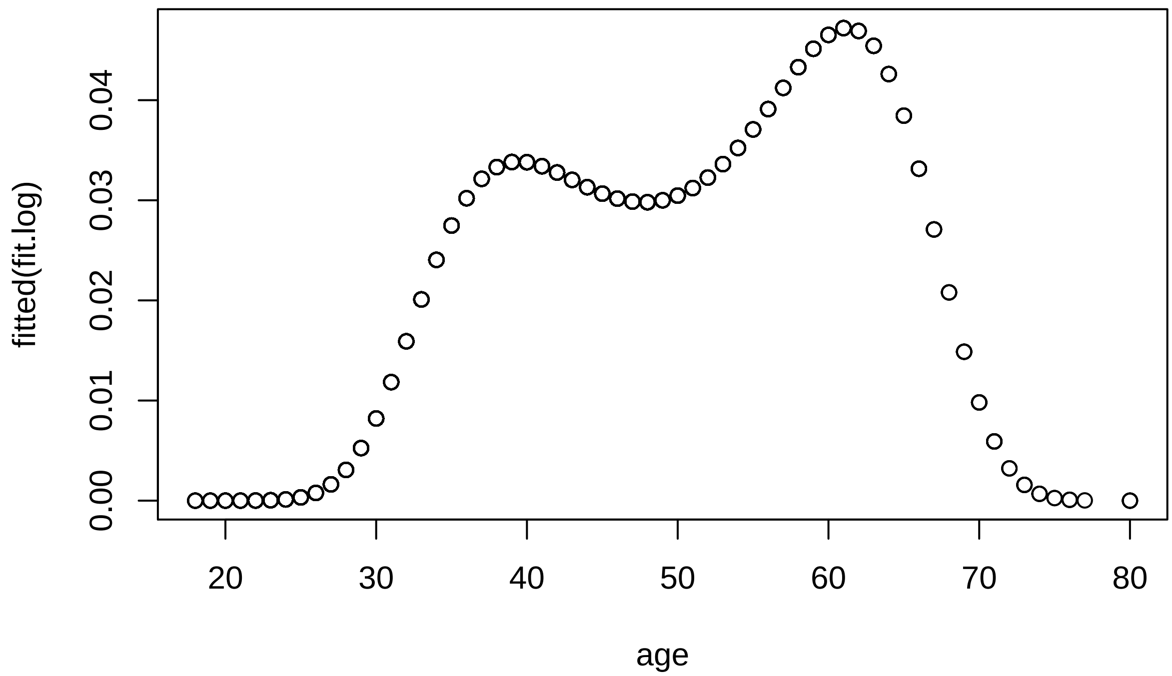
Change Wage output variable to 0/1, with 1 for >$250k earners. In GLM due to the way it functions, some of the orthogonality of coefficients is lost, therefore to decide inclusion/exclusion of variable, we'll need to rely on F-test.

Predict() function also provides probabilities, using **type = "response"** option, however that would make the standard-errors/ confidence internal non-sensical.

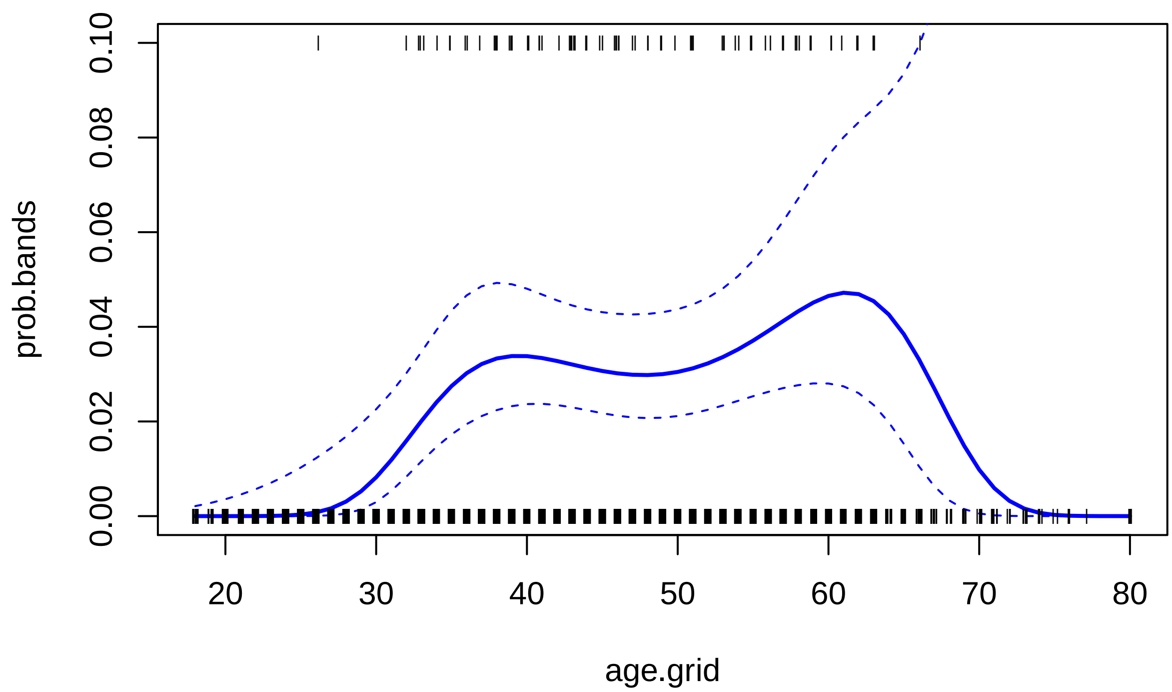
```r
fit.log <- glm(I(wage>250)~poly(age,4), data=Wage, family=binomial)
summary(fit.log)
```

```
##
## Call:
## glm(formula = I(wage > 250) ~ poly(age, 4), family = binomial,
##     data = Wage)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.3110  -0.2607  -0.2488  -0.1791   3.7859
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -4.3012     0.3451 -12.465  < 2e-16 ***
## poly(age, 4)1   71.9642    26.1176   2.755  0.00586 **
## poly(age, 4)2  -85.7729    35.9043  -2.389  0.01690 *
## poly(age, 4)3   34.1626    19.6890   1.735  0.08272 .
## poly(age, 4)4  -47.4008    24.0909  -1.968  0.04912 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 730.53  on 2999  degrees of freedom
## Residual deviance: 701.22  on 2995  degrees of freedom
## AIC: 711.22
##
## Number of Fisher Scoring iterations: 9
```

```r
plot(age,fitted(fit.log))
```




```r
#Calculate Predicted values + Standard Error band for LOGIT!!
preds = predict(fit.log, list(age=age.grid), se=T)
se.bands = preds$fit + cbind(fit=0, lower=-2*preds$se, upper=+2*preds$se)

#Converting from Log-Odds/LOGIT to Probability
prob.bands = exp(se.bands)/(1+exp(se.bands))

#Plotting
matplot(age.grid,prob.bands, col="blue", lwd=c(2,1,1), lty=c(1,2,2), type="l", ylim=c(0,0.1))
points(jitter(age), I(wage>250)/10, pch="|", cex=0.5)
```




```
# fit.log.a <- glm(I(wage>250)~poly(age,2), data=Wage, family=binomial)
# fit.log.b <- glm(I(wage>250)~poly(age,3), data=Wage, family=binomial)
# anova(fit.log.a, fit.log.b)
```

Step Functions

Using **cut()**. Breaks can be manually assigned using **breaks option.

The age < 33.5 category is left out, so the intercept coefficient of 94 can be interpreted as the average salary for those under 33.5 years of age, and the other coefficients are average additional salary for those other age groups.

```
table(cut(age,4))
```

```
##
## (17.9,33.5]   (33.5,49]   (49,64.5] (64.5,80.1]
##         750        1399         779          72
```

```
fit.step = lm(wage~cut(age,4), data=Wage)
coef(summary(fit.step))
```

```
##                          Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)             94.158392   1.476069 63.789970 0.000000e+00
## cut(age, 4)(33.5,49]    24.053491   1.829431 13.148074 1.982315e-38
## cut(age, 4)(49,64.5]    23.664559   2.067958 11.443444 1.040750e-29
## cut(age, 4)(64.5,80.1]   7.640592   4.987424  1.531972 1.256350e-01
```

Splines - Fixed-knot Cubic Spline and Smooth-spline

bs() generates the B-spline basis matrix for a polynomial spline (cubic by default.) **ns()** generates natural spline. More explanation here.

```r
# 3 knots will lead to 7 DFs (K+4) = 1 intercept + 6 basis functions
# We can either specify knots or DFs
fit.spline = lm(wage~bs(age, knots=c(25, 40, 60)),data=Wage)
summary(fit.spline)
```

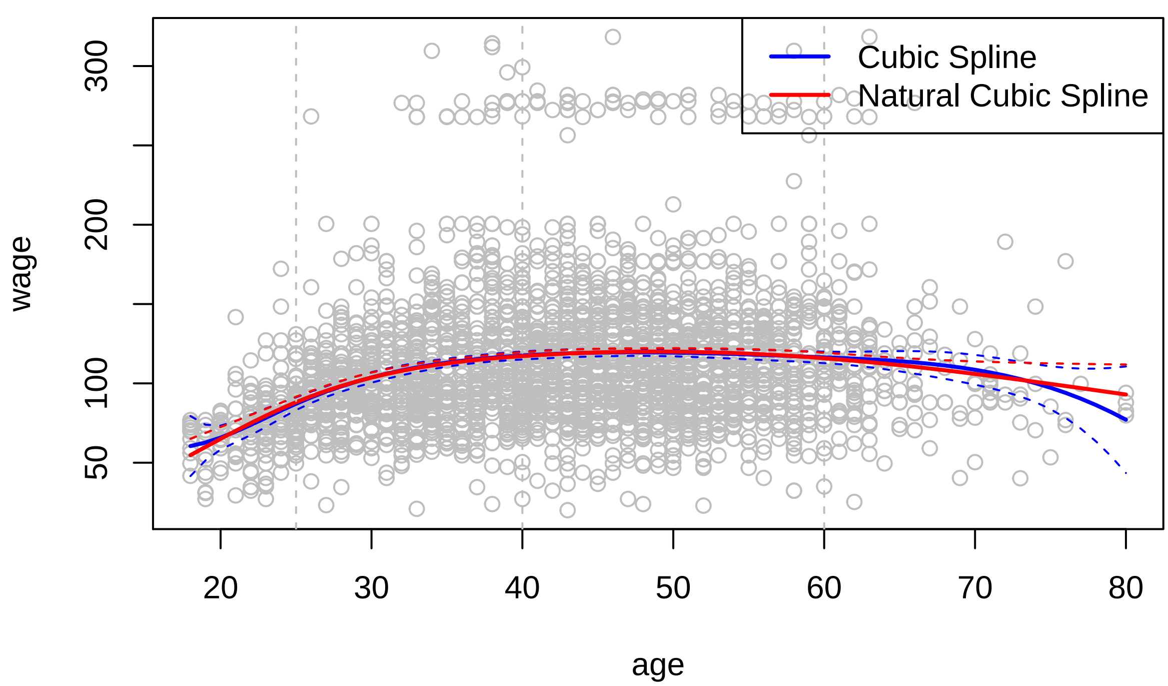
```
##
## Call:
## lm(formula = wage ~ bs(age, knots = c(25, 40, 60)), data = Wage)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -98.832 -24.537  -5.049  15.209 203.207
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       60.494      9.460   6.394 1.86e-10 ***
## bs(age, knots = c(25, 40, 60))1    3.980     12.538   0.317 0.750899
## bs(age, knots = c(25, 40, 60))2   44.631      9.626   4.636 3.70e-06 ***
## bs(age, knots = c(25, 40, 60))3   62.839     10.755   5.843 5.69e-09 ***
## bs(age, knots = c(25, 40, 60))4   55.991     10.706   5.230 1.81e-07 ***
## bs(age, knots = c(25, 40, 60))5   50.688     14.402   3.520 0.000439 ***
## bs(age, knots = c(25, 40, 60))6   16.606     19.126   0.868 0.385338
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.92 on 2993 degrees of freedom
## Multiple R-squared:  0.08642,    Adjusted R-squared:  0.08459
## F-statistic: 47.19 on 6 and 2993 DF,  p-value: < 2.2e-16
```

```r
pred.spline = predict(fit.spline, list(age=age.grid), se=TRUE)

fit.nspline = lm(wage~ns(age, knots=c(25, 40, 60)),data=Wage)
summary(fit.spline)
```

```
##
## Call:
## lm(formula = wage ~ bs(age, knots = c(25, 40, 60)), data = Wage)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -98.832 -24.537  -5.049  15.209 203.207
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       60.494      9.460   6.394 1.86e-10 ***
## bs(age, knots = c(25, 40, 60))1    3.980     12.538   0.317 0.750899
## bs(age, knots = c(25, 40, 60))2   44.631      9.626   4.636 3.70e-06 ***
## bs(age, knots = c(25, 40, 60))3   62.839     10.755   5.843 5.69e-09 ***
```

```
## bs(age, knots = c(25, 40, 60))4   55.991     10.706   5.230 1.81e-07 ***
## bs(age, knots = c(25, 40, 60))5   50.688     14.402   3.520 0.000439 ***
## bs(age, knots = c(25, 40, 60))6   16.606     19.126   0.868 0.385338
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.92 on 2993 degrees of freedom
## Multiple R-squared:  0.08642,    Adjusted R-squared:  0.08459
## F-statistic: 47.19 on 6 and 2993 DF,  p-value: < 2.2e-16
```

```
pred.nspline = predict(fit.nspline, list(age=age.grid), se=TRUE)

plot(age, wage, col="gray")
lines(age.grid, pred.spline$fit , col="blue", lwd=2)
lines(age.grid, pred.nspline$fit, col="red", lwd=2)
abline(v=c(25, 40, 60), lty=2, col="grey")
matlines(age.grid, cbind(pred.spline$fit + 2*pred.spline$se,
                         pred.spline$fit - 2*pred.spline$se,
                         pred.nspline$fit + 2*pred.nspline$se,
                         pred.nspline$fit + 2*pred.nspline$se), lty="dashed", col=c("blue", "blue", "re
legend("topright", legend=c("Cubic Spline", "Natural Cubic Spline"), col= c("blue", "red"), lty=1, lwd=
```

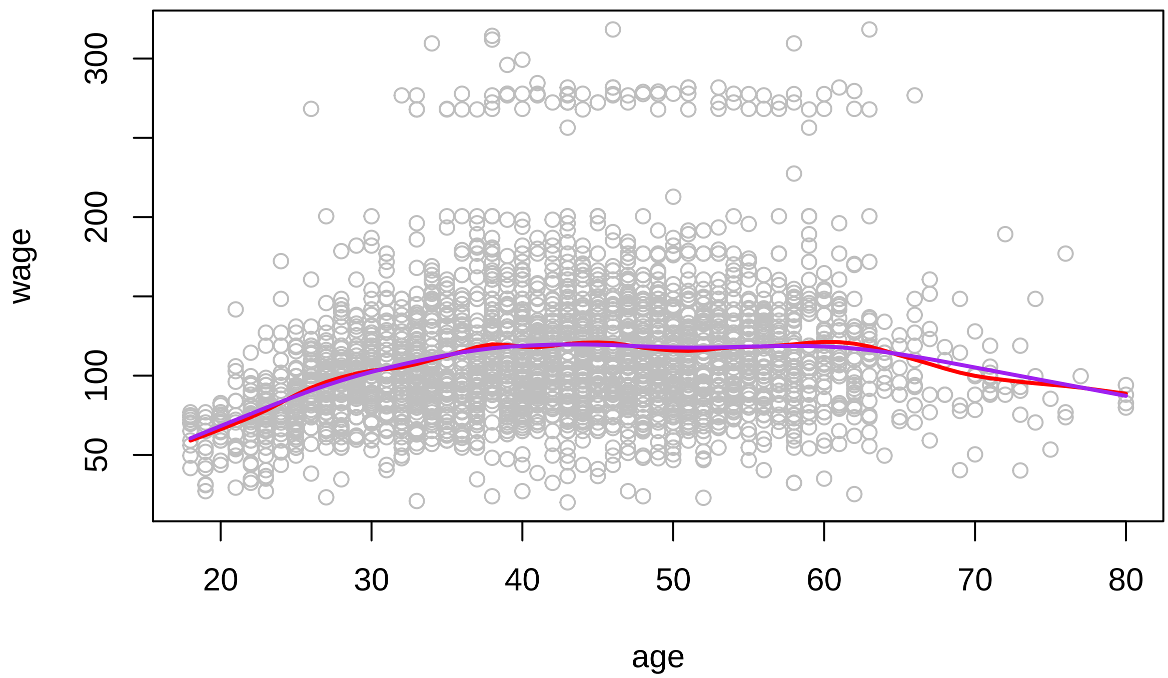


Smooth-spline does not require knot selection, as each point is a knot. But a smoothing parameter *lambda*.

```r
# Controlling smoothing parameter by (i) Effective degrees of freedom
fit.sm.spline = smooth.spline(age, wage, df=16)
plot(age, wage, col="gray")
lines(fit.sm.spline, col="red", lwd=2)

# ... or (ii) cross-validation
fit.sm.spline = smooth.spline(age, wage, cv=TRUE)
```

```
## Warning in smooth.spline(age, wage, cv = TRUE): cross-validation with non-unique
## 'x' values seems doubtful
```

```r
lines(fit.sm.spline, col="purple", lwd=2)
```



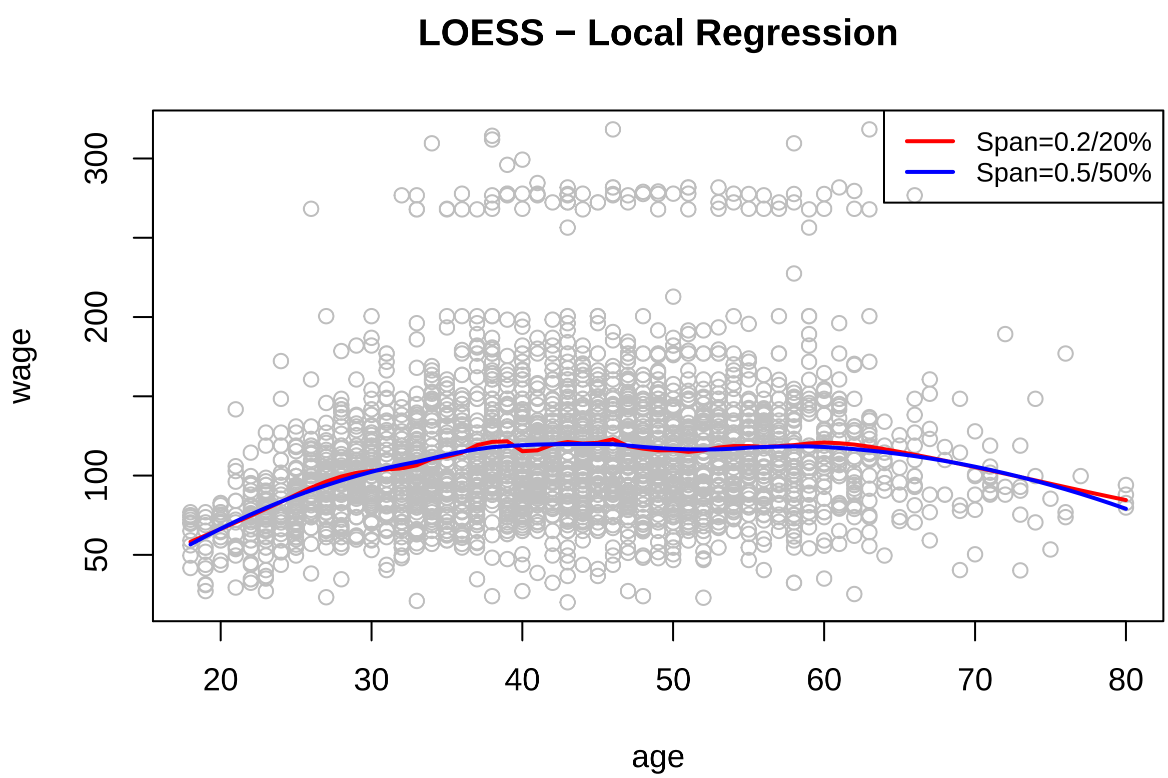

```r
fit.sm.spline
```

```
## Call:
## smooth.spline(x = age, y = wage, cv = TRUE)
##
## Smoothing Parameter  spar= 0.6988943  lambda= 0.02792303 (12 iterations)
## Equivalent Degrees of Freedom (Df): 6.794596
## Penalized Criterion (RSS): 75215.9
## PRESS(l.o.o. CV): 1593.383
```

LOESS - Local Regression

The larger the span, the smoother the fit.

```
plot(age, wage, xlim=agelims, col="gray")
title("LOESS - Local Regression")

fit1 <- loess(wage~age, span=0.2, data=Wage)
fit2 <- loess(wage~age, span=0.5, data=Wage)
lines(age.grid, predict(fit1, data.frame(age=age.grid)), col="red", lwd=2)
lines(age.grid, predict(fit2, data.frame(age=age.grid)), col="blue", lwd=2)
legend("topright", legend=c("Span=0.2/20%", "Span=0.5/50%"), col=c("Red", "Blue"), lty=1, lwd=2, cex=0.8
```



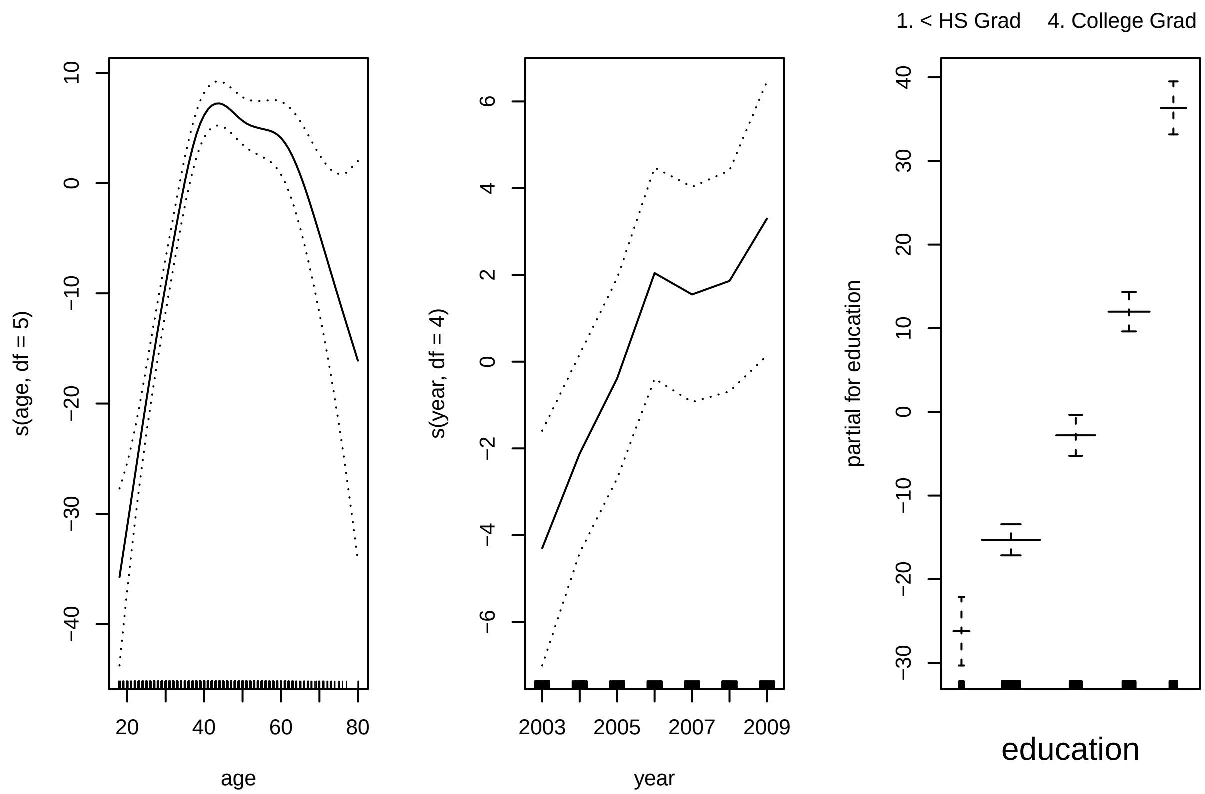
GAM: Generalized Additive Models

Mixing more than one predictors. Use **s()** to specify a **Smoothing Spline** fit in a GAM Formula.

The generic **plot()** function recognizes that gam.m3 is an object of class Gam, and invokes the appropriate **plot.Gam()** method.

Compelling evidence that a GAM with a linear function of year is better than a GAM that does not include year at all (p-value = 0.00014). However, there is no evidence that a non-linear function of year is needed (p-value = 0.349).

```r
# Gam with smoothing spline
gam.m3 = gam(wage~s(age,df=5)+s(year,df=4)+education, data=Wage)
par(mfrow=c(1,3))
plot(gam.m3,se=T)
```




```r
# Should YEAR be linear or non-linear?
gam.m1 <- gam(wage ~ s(age,5) + education, data=Wage)
gam.m2 <- gam(wage ~ year + s(age,5) + education, data=Wage)
anova(gam.m1, gam.m2, gam.m3, test="F")
```

```
## Analysis of Deviance Table
##
## Model 1: wage ~ s(age, 5) + education
## Model 2: wage ~ year + s(age, 5) + education
## Model 3: wage ~ s(age, df = 5) + s(year, df = 4) + education
##   Resid. Df Resid. Dev Df Deviance       F    Pr(>F)
## 1      2990    3711731
## 2      2989    3693842  1  17889.2 14.4771 0.0001447 ***
## 3      2986    3689770  3   4071.1  1.0982 0.3485661
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The “Anova for Parametric Effects” p-values clearly demonstrate that year, age, and education are all highly statistically significant, even when only assuming a linear relationship. Alternatively, the “Anova for

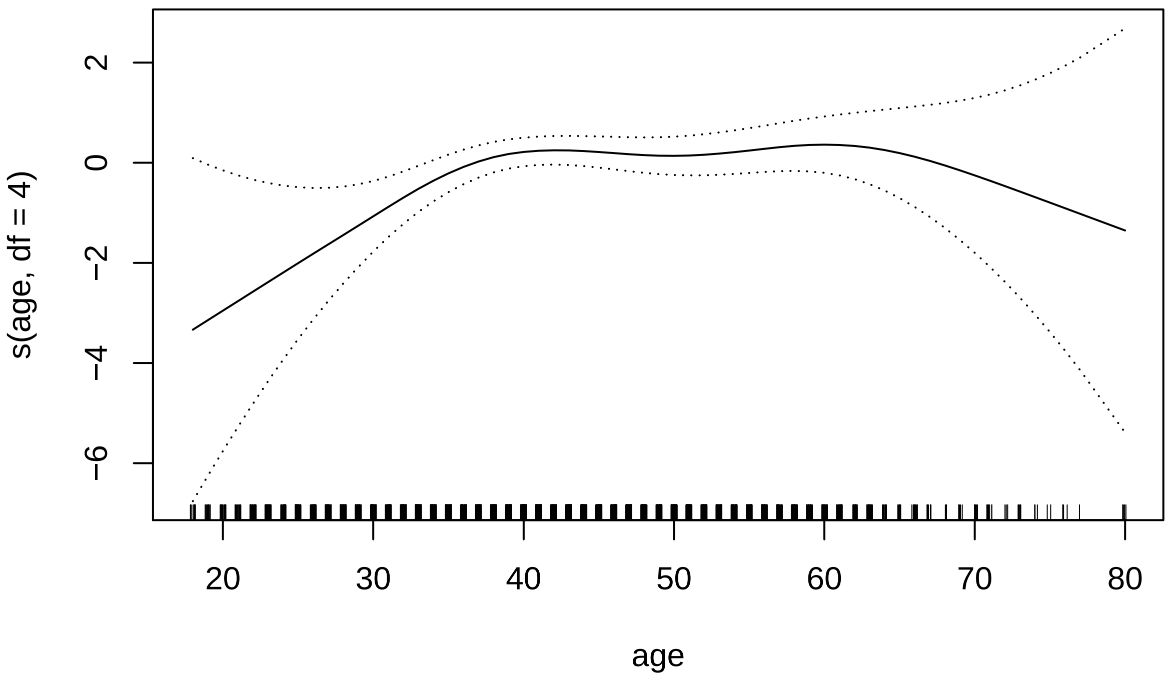
Nonparametric Effects" p-values for year and age correspond to a null hypothesis of a linear relationship versus the alternative of a non-linear relationship. The large p-value for year reinforces the conclusion from the ANOVA test that a linear function is adequate for this term.

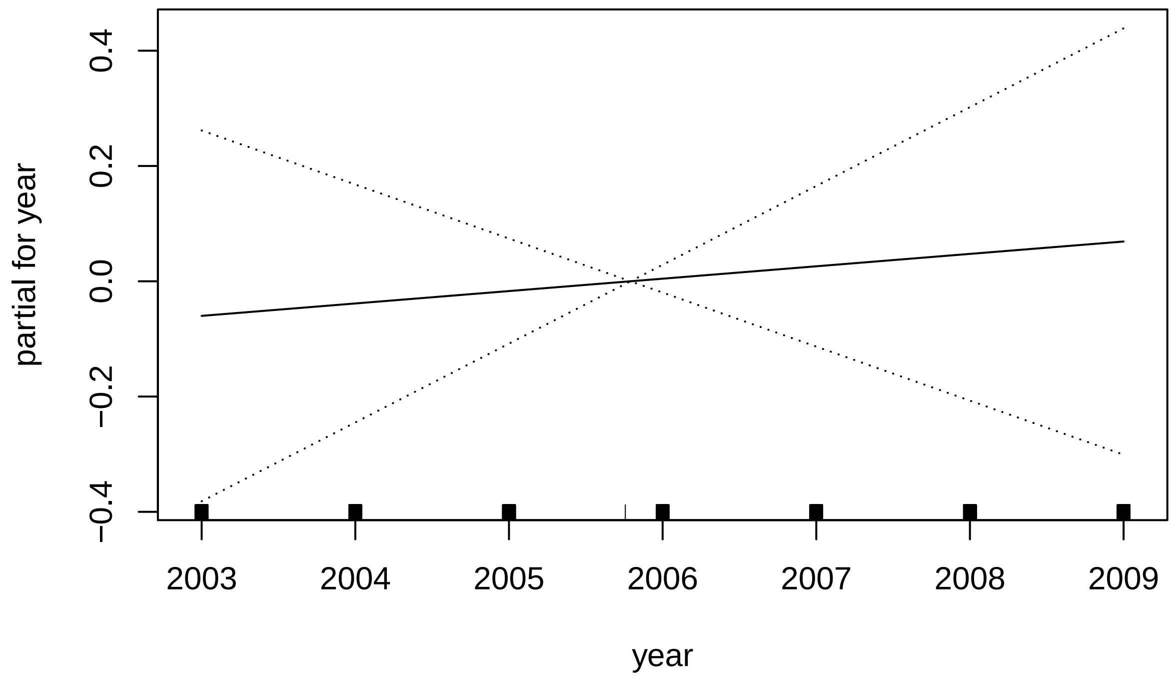
```
summary(gam.m3)
```

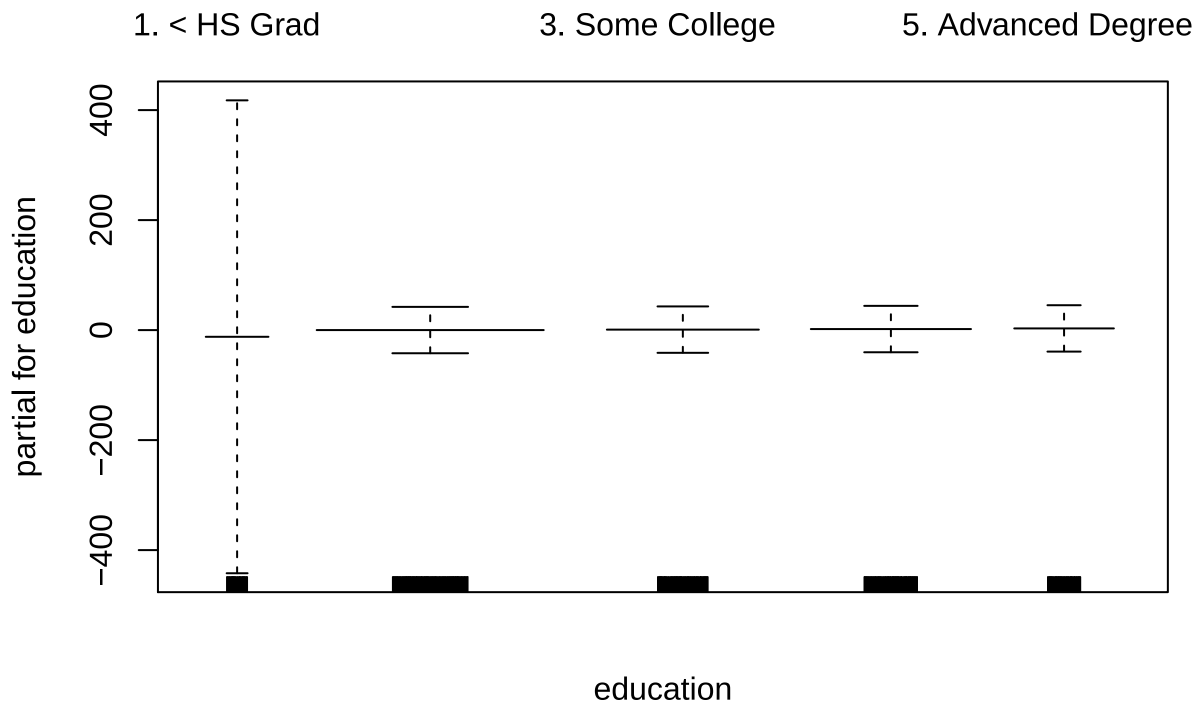
```
##
## Call: gam(formula = wage ~ s(age, df = 5) + s(year, df = 4) + education,
##     data = Wage)
## Deviance Residuals:
##     Min      1Q  Median      3Q     Max
## -119.43  -19.70   -3.33   14.17  213.48
##
## (Dispersion Parameter for gaussian family taken to be 1235.69)
##
##     Null Deviance: 5222086 on 2999 degrees of freedom
## Residual Deviance: 3689770 on 2986 degrees of freedom
## AIC: 29887.75
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##                   Df  Sum Sq Mean Sq F value    Pr(>F)
## s(age, df = 5)     1  200684  200684 162.406 < 2.2e-16 ***
## s(year, df = 4)    1   21817   21817  17.655 2.725e-05 ***
## education          4 1069726  267432 216.423 < 2.2e-16 ***
## Residuals       2986 3689770    1236
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                 Npar Df Npar F  Pr(F)
## (Intercept)
## s(age, df = 5)        4 32.380 <2e-16 ***
## s(year, df = 4)       3  1.086 0.3537
## education
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

GAM: LOGIT with smoothing spline

```
gam.l1 = gam(I(wage>250)~s(age,df=4)+year+education, data=Wage, family=binomial)
plot(gam.l1, se=T)
```


2
0
−2
−4
−6
s(age, df = 4)
20
30
40
50
60
70
80
age

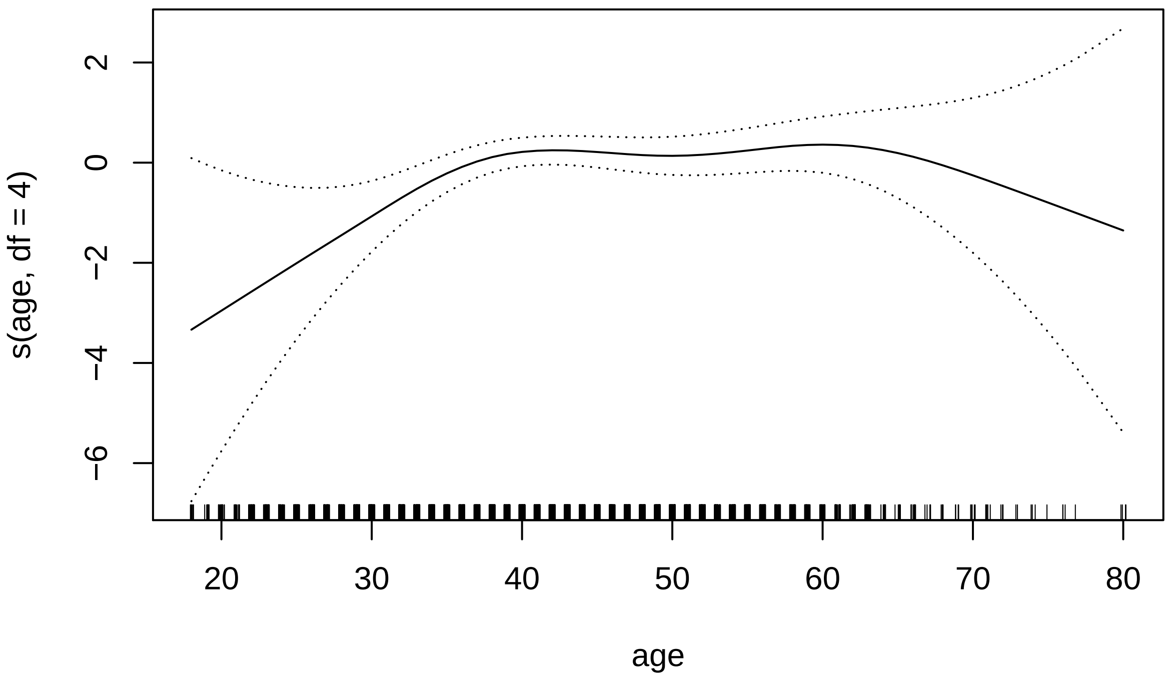

partial for year
0.4
0.2
0.0
−0.2
−0.4
2003
2004
2005
2006
2007
2008
2009
year



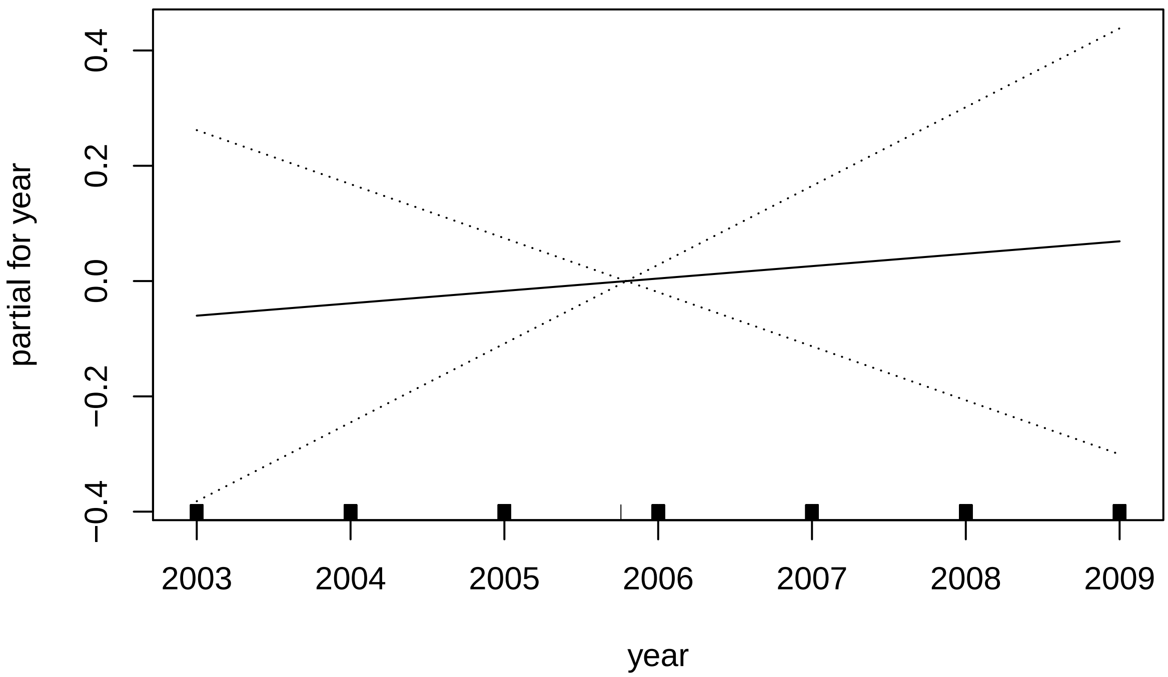

```r
#High SE for "<HS Grad" category?
table(education, I(wage>250))
```

```
##
## education            FALSE TRUE
##   1. < HS Grad         268    0
##   2. HS Grad           966    5
##   3. Some College      643    7
##   4. College Grad      663   22
##   5. Advanced Degree   381   45
```

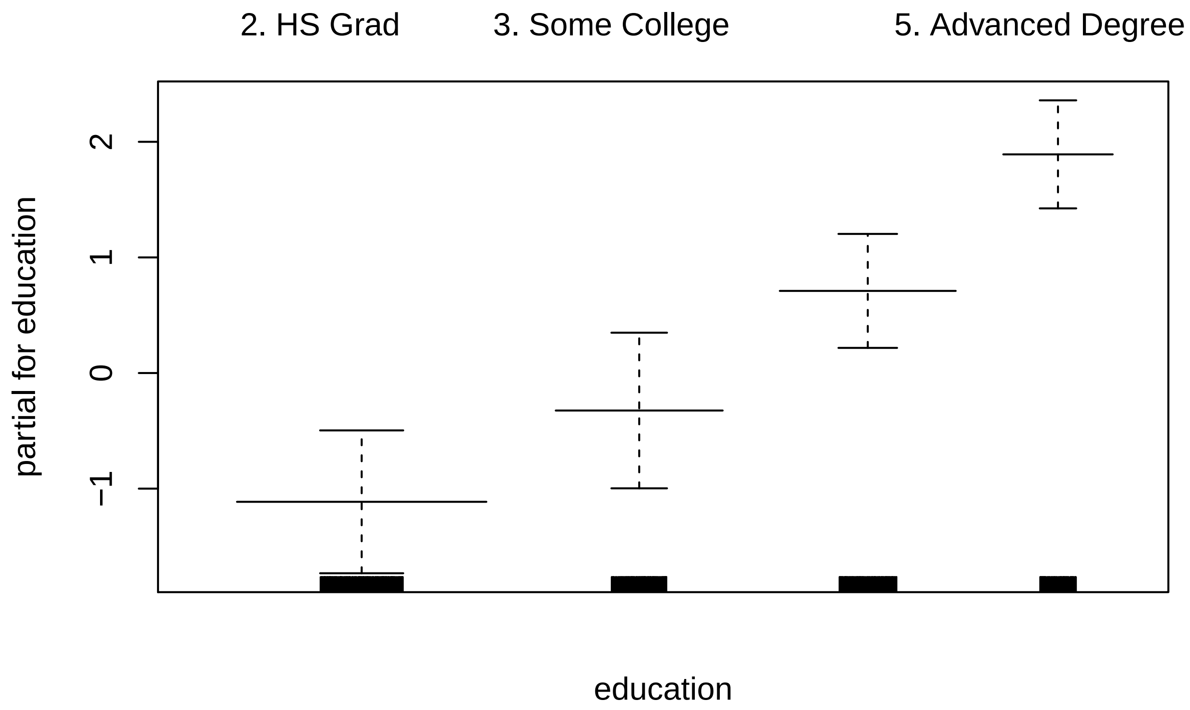
```r
gam.l1.sub = gam(I(wage>250)~s(age,df=4)+year+education, data=Wage,
                 family=binomial,
                 subset=(education != "1. < HS Grad"))
plot(gam.l1.sub, se=T)
```



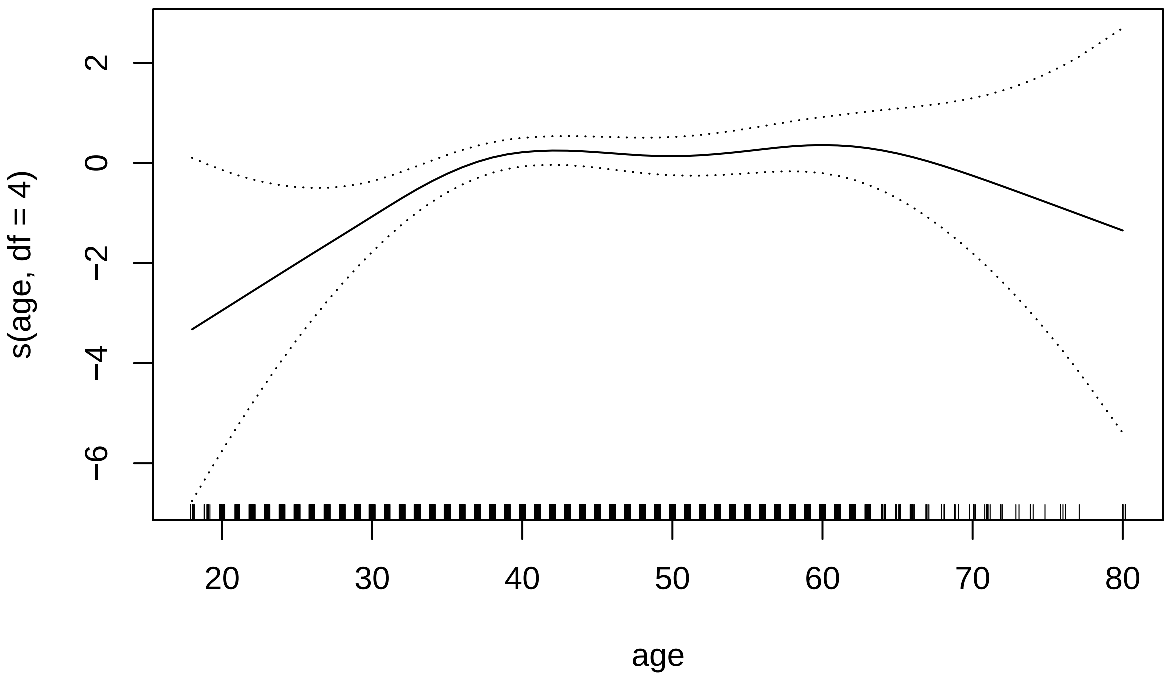
s(age, df = 4)
2
0
−2
−4
−6
20
30
40
50
60
70
80
age



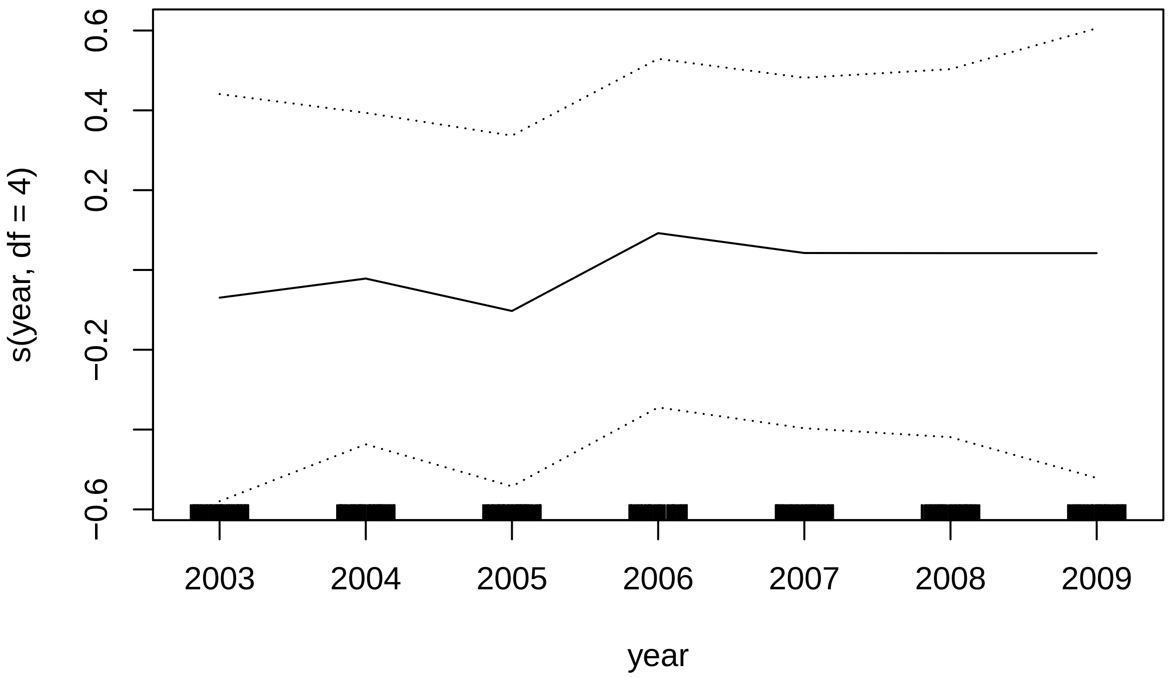
0.4
0.2
0.0
−0.2
−0.4
partial for year
2003
2004
2005
2006
2007
2008
2009
year



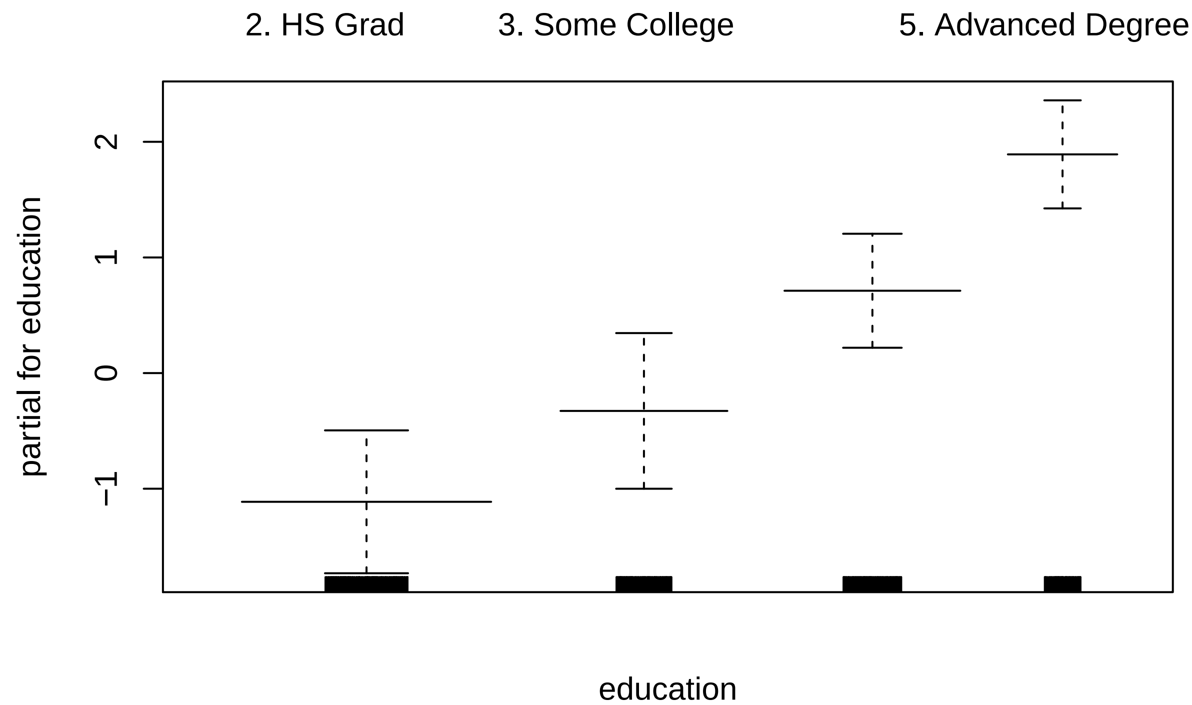

```
gam.l2.sub = gam(I(wage>250)~s(age,df=4)+s(year,df=4)+education, data=Wage,
                 family=binomial,
                 subset=(education != "1. < HS Grad"))
plot(gam.l2.sub, se=T)
```



2
0
−2
−4
−6
s(age, df = 4)
20
30
40
50
60
70
80
age



s(year, df = 4)
0.6
0.4
0.2
−0.2
−0.6
2003
2004
2005
2006
2007
2008
2009
year

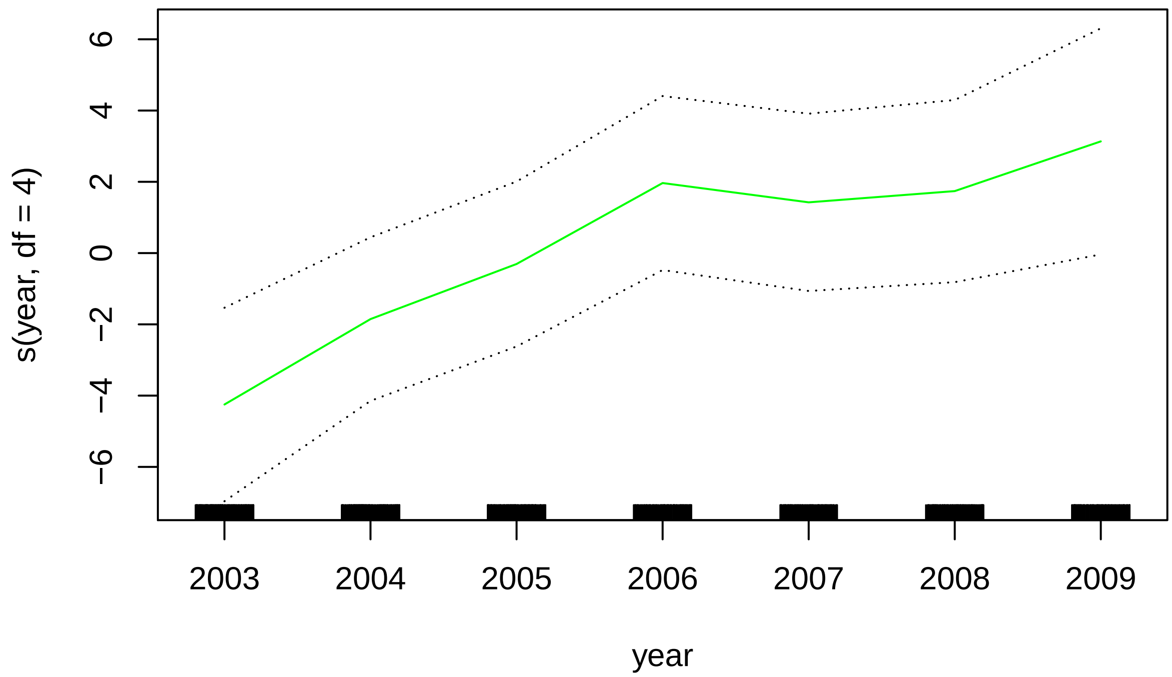


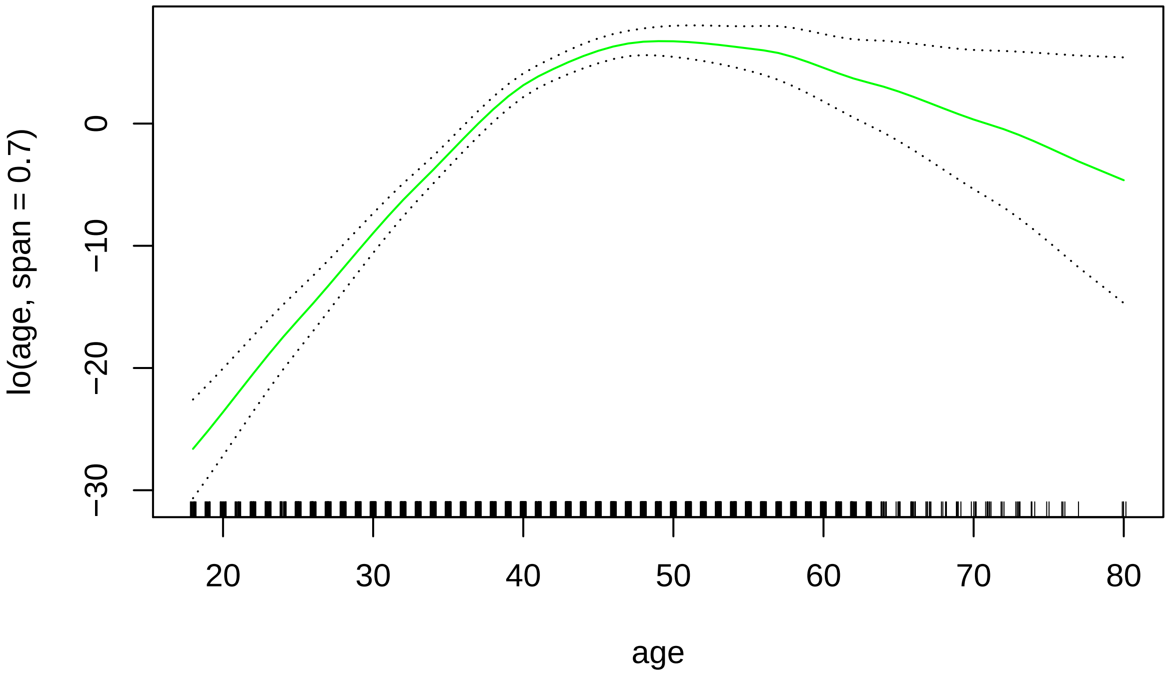

```
anova(gam.l1.sub, gam.l2.sub, test="Chisq") #no-need for adding non-linear terms for 'year'
```

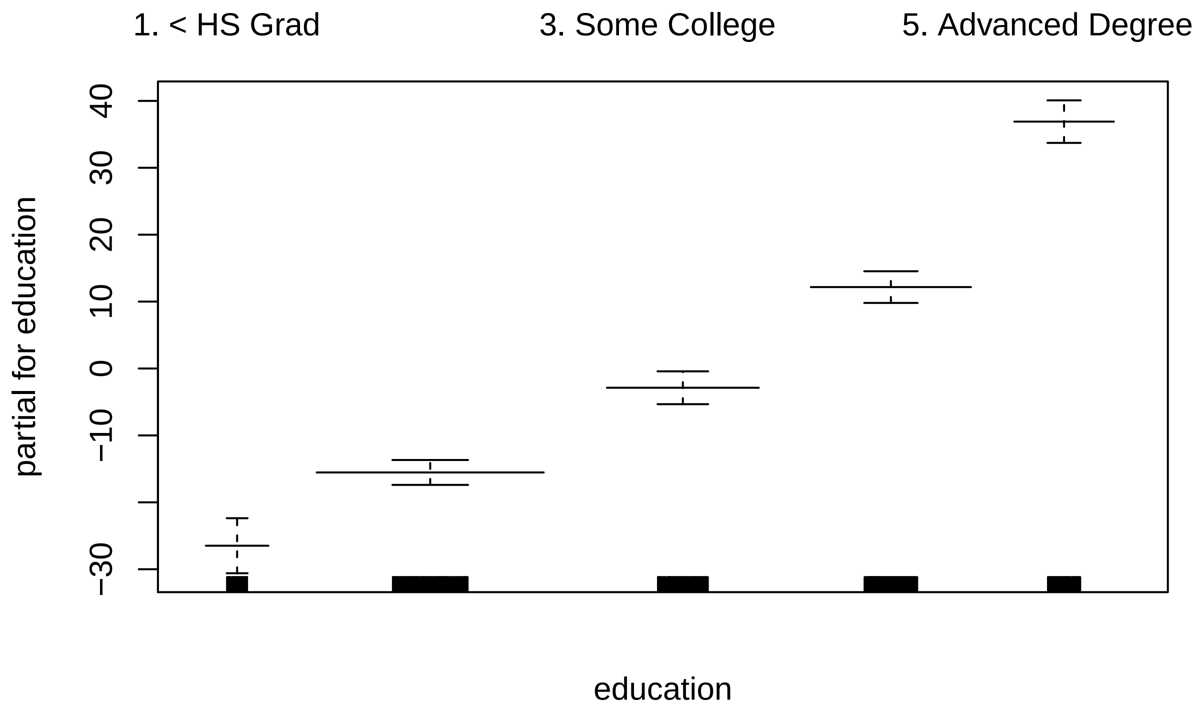
```
## Analysis of Deviance Table
##
## Model 1: I(wage > 250) ~ s(age, df = 4) + year + education
## Model 2: I(wage > 250) ~ s(age, df = 4) + s(year, df = 4) + education
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      2723     603.78
## 2      2720     602.87  3  0.90498   0.8242
```

GAM: Using LOESS & Interaction

```
gam.lo <- gam(wage~s(year, df=4) + lo(age, span=0.7) + education, data=Wage)
plot.Gam(gam.lo, se=TRUE, col="green")
```


6
4
2
0
−2
−4
−6
s(year, df = 4)
2003
2004
2005
2006
2007
2008
2009
year

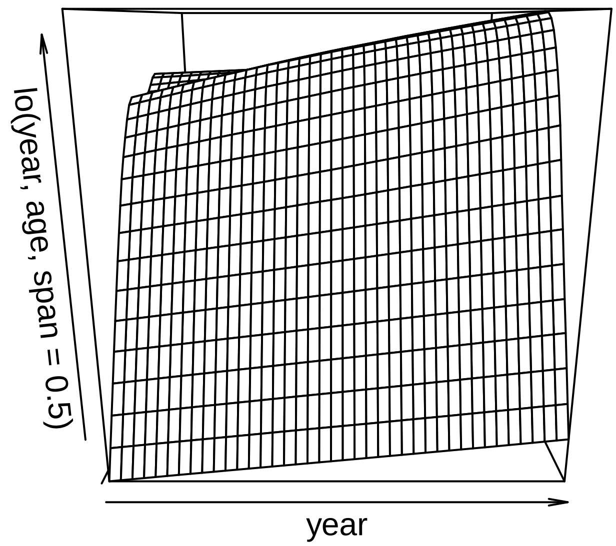

lo(age, span = 0.7)
0
−10
−20
−30
20
30
40
50
60
70
80
age

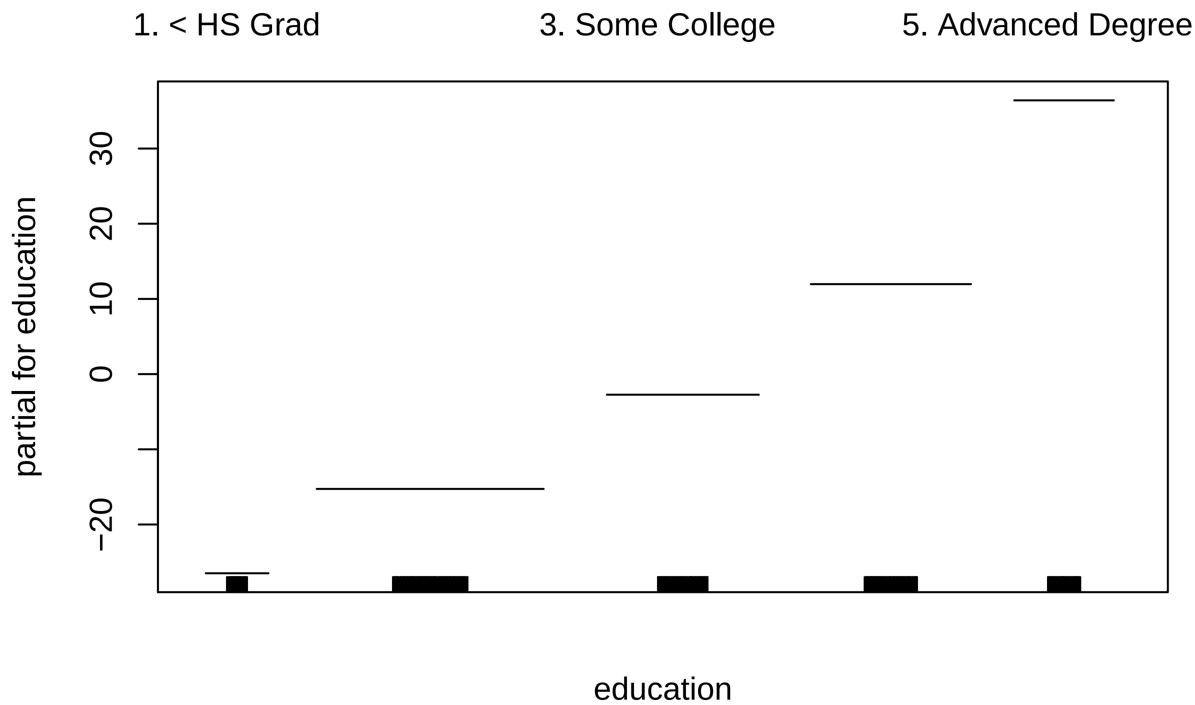



```
gam.lo.i <- gam(wage~lo(year, age, span=0.5)+ education, data=Wage)
```

```
## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame, bf.maxit, : liv
## too small. (Discovered by lowesd)

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame, bf.maxit, : lv
## too small. (Discovered by lowesd)

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame, bf.maxit, : liv
## too small. (Discovered by lowesd)

## Warning in lo.wam(x, z, wz, fit$smooth, which, fit$smooth.frame, bf.maxit, : lv
## too small. (Discovered by lowesd)
```

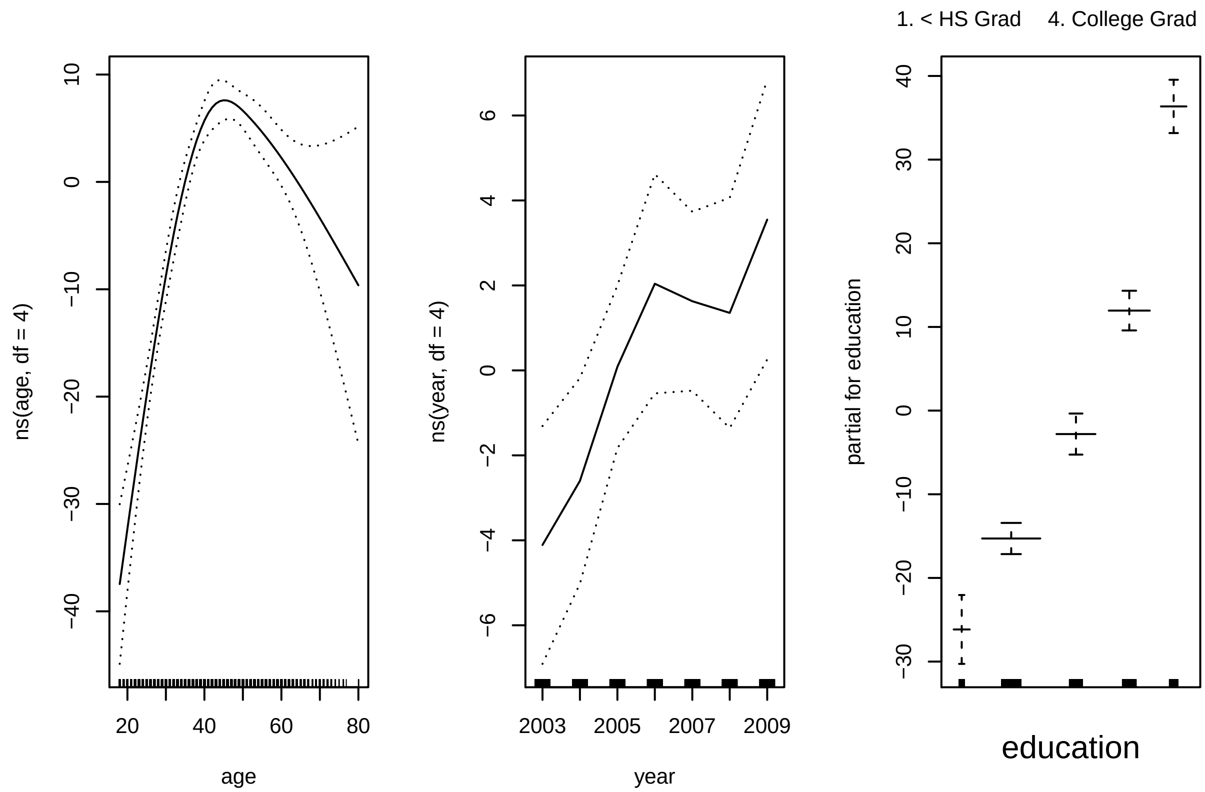
```
plot(gam.lo.i)
```


lo(year, age, span = 0.5)
year



Using GAM plotting functionality with lm() models.

```
par(mfrow=c(1,3))
lm1 = lm(wage~ns(age,df=4)+ns(year,df=4)+education,data=Wage)
plot.Gam(lm1, se=T)
```


1. < HS Grad
4. College Grad
ns(age, df = 4)
age
ns(year, df = 4)
year
partial for education
education