Homework 7 STA578

Jiaming

11/13/2017

Question 1

1.a

For $x \leq \xi$, $\beta_4(x-\xi)^3_+ = 0$. So $f_1(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$, then $a_1 = \beta_0; b_1 = \beta_1; c_1 = \beta_2; d_1 = \beta_3$

1.b

For $x > \xi$, $\beta_4(x-\xi)^3_+ = \beta_4(x-\xi)^3$. So

$f_2(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4(x-\xi)^3 =$
$(\beta_0 - \beta_4\xi^3) + (\beta_1 + 3\beta_4\xi^2)x + (\beta_2 - 3\beta_4\xi)x^2 + (\beta_3 + \beta_4)x^3$, then

$a_2 = (\beta_0 - \beta_4\xi^3)$;
$b_2 = (\beta_1 + 3\beta_4\xi^2)$;
$c_2 = (\beta_2 - 3\beta_4\xi)$;
$d_2 = (\beta_3 + \beta_4)$

1.c

$f_1(\xi) = \beta_0 + \beta_1\xi + \beta_2\xi^2 + \beta_3\xi^3$

$f_2(\xi) = \beta_0 + \beta_1\xi + \beta_2\xi^2 + \beta_3\xi^3 + \beta_4(\xi-\xi)^3 = f_1(\xi)$

1.d

$f_1'(\xi) = \beta_1 + 2\beta_2\xi + 3\beta_2\xi^2$

$f_2'(\xi) = \beta_1 + 2\beta_2\xi + 3\beta_3\xi^2 + 3\beta_4(\xi-\xi)^2$.

Therefore we have $f_1'(x) = f_2'(x)$

1.f

$f_1''(\xi) = 2\beta_2 + 6\beta_2\xi$
$f_2''(\xi) = 2\beta_2 + 2 * 3\beta_3\xi + 2 * 3\beta_4(\xi-\xi)^2$.

Therefore $f_1''(\xi) = f_2''(\xi) = 2\beta_2 + 6\beta_2\xi$. $f(x)$ is a cubic spine

Question 2

2.a

The difference between $\hat{g}_1$ and $\hat{g}_2$ is that $\hat{g}_2$ penalize the fourth order derivatives and $\hat{g}_1$ penalize thrid order wigglyness. So when $\lambda \to \infty$, $\hat{g}_2$ will be a higher order polynomial than $\hat{g}_1$. With a higher degree of freedom, $\hat{g}_2$ will have smaller training RSS.

2.b

$\hat{g}_2$ tend to overfit the noise in training data because it's more flexible and thus $\hat{g}_1$ will have a lower testing RSS.

2.c

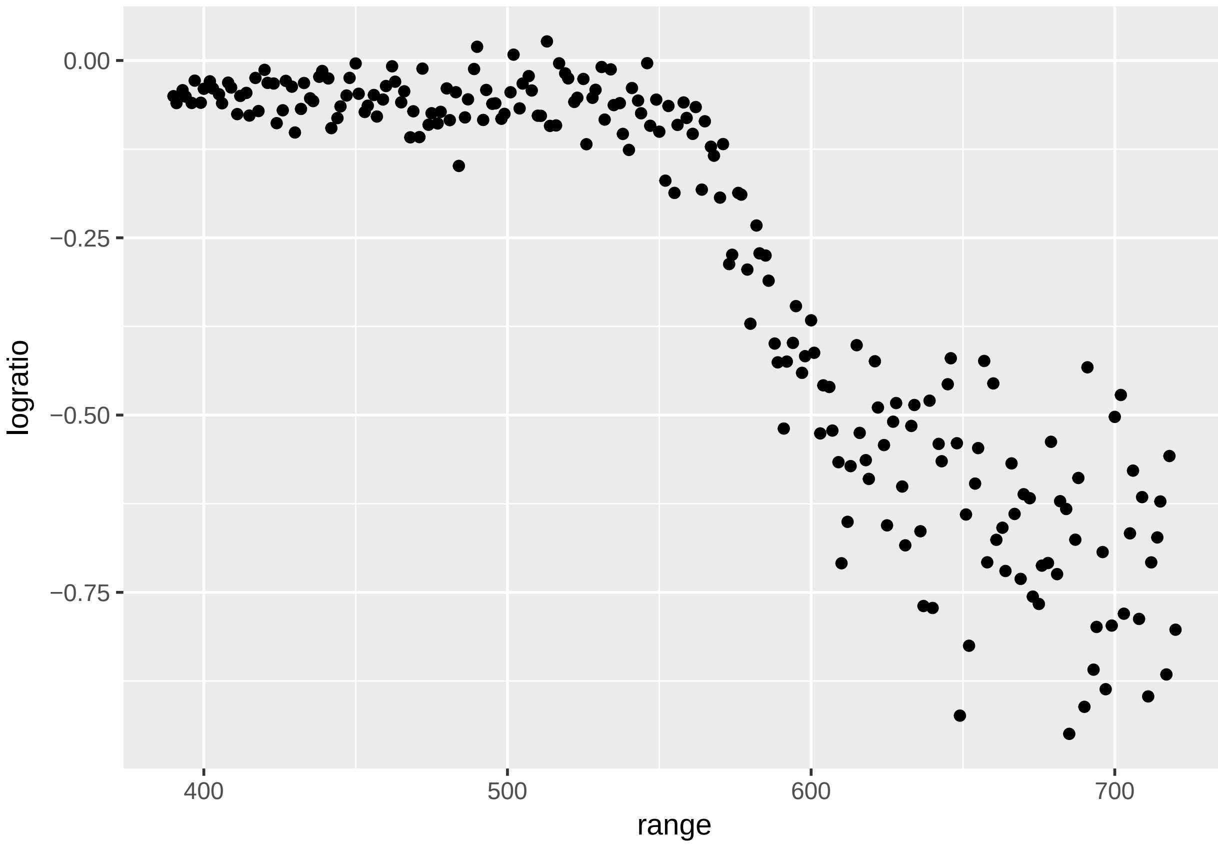
When $\lambda \to 0$, we can find $\hat{g}_1 = \hat{g}_2$. So we're expecting the same training and test RSS for them.

```r
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00  
```

Question 3

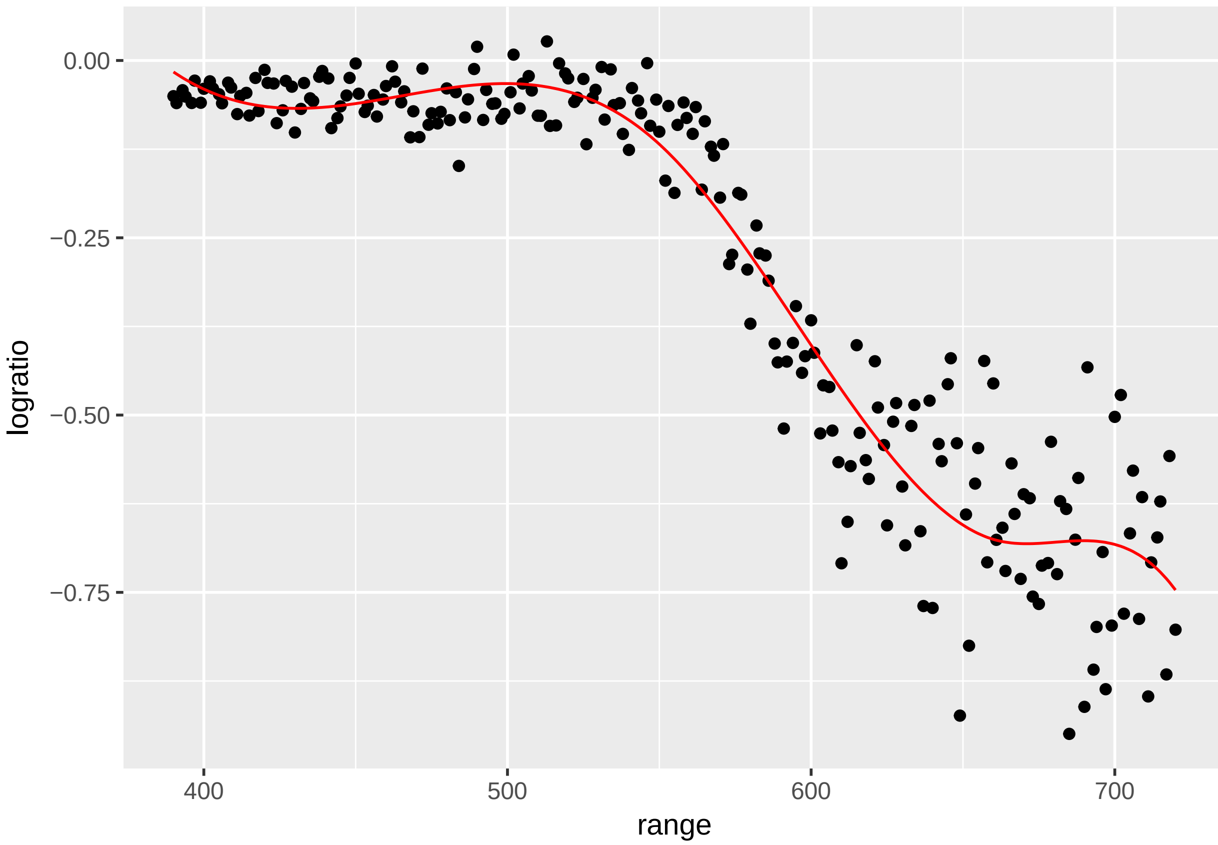
```r
data('lidar', package='SemiPar')
ggplot(lidar, aes(x=range, y=logratio)) +
  geom_point()
```



3.a

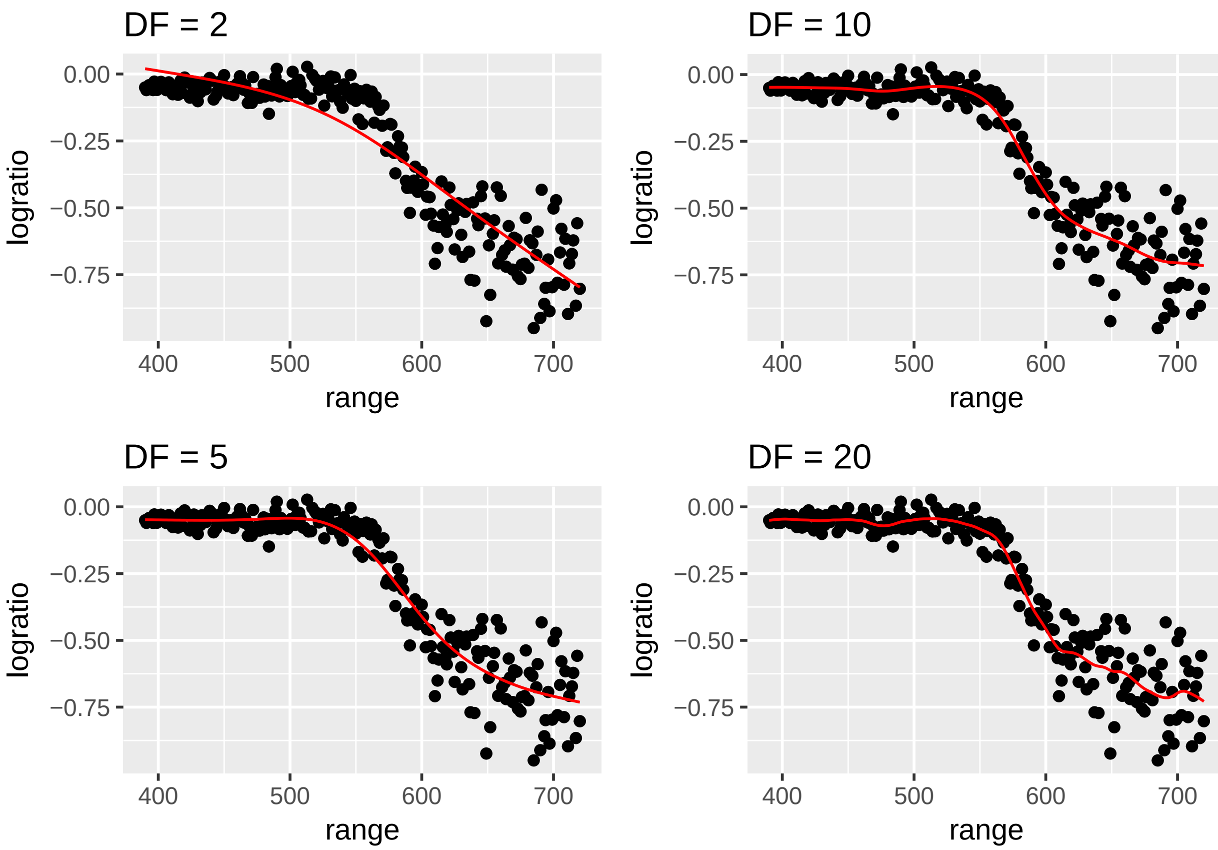
It seems like that we have a breakpoint at x=550 and at x = 660(or 670?)

```
model <- lm( logratio ~ bs(range, degree = 3, knots = c(550, 660)), data=lidar )

lidar$yhat = predict(model)
ggplot(lidar, aes(x=range, y=logratio)) +
  geom_point() +
  geom_line(aes(y=yhat), color='red')
```



3.b

```
# initialize P
P <- NULL
i <- 1
for( df in c(2,5,10,20) ){
  # calling gam() function for smoothing spline
  model <- gam( logratio ~ s(range, df), data=lidar )
  lidar$yhat <- predict(model)
  P[[i]] <- ggplot(lidar, aes(x=range)) +
    geom_point( aes(y=logratio) ) +
    geom_line( aes(y=yhat), color='red') +
    labs(title=paste('DF =',df))
  i <- i + 1
}
Rmisc::multiplot(P[[1]], P[[2]], P[[3]], P[[4]], cols = 2)
```



Looks like the best *df* is between 3 to 18.

Cross Validation

```
ctrl <- trainControl( method='repeatedcv', repeats=10, number=4 )
grid <- data.frame(df=3:18)
model <- train( logratio ~ range, data=lidar, method='gamSpline',
                tuneGrid=grid, trControl=ctrl )

# Best tune?
model$bestTune
```

```
##   df
## 6  8
```

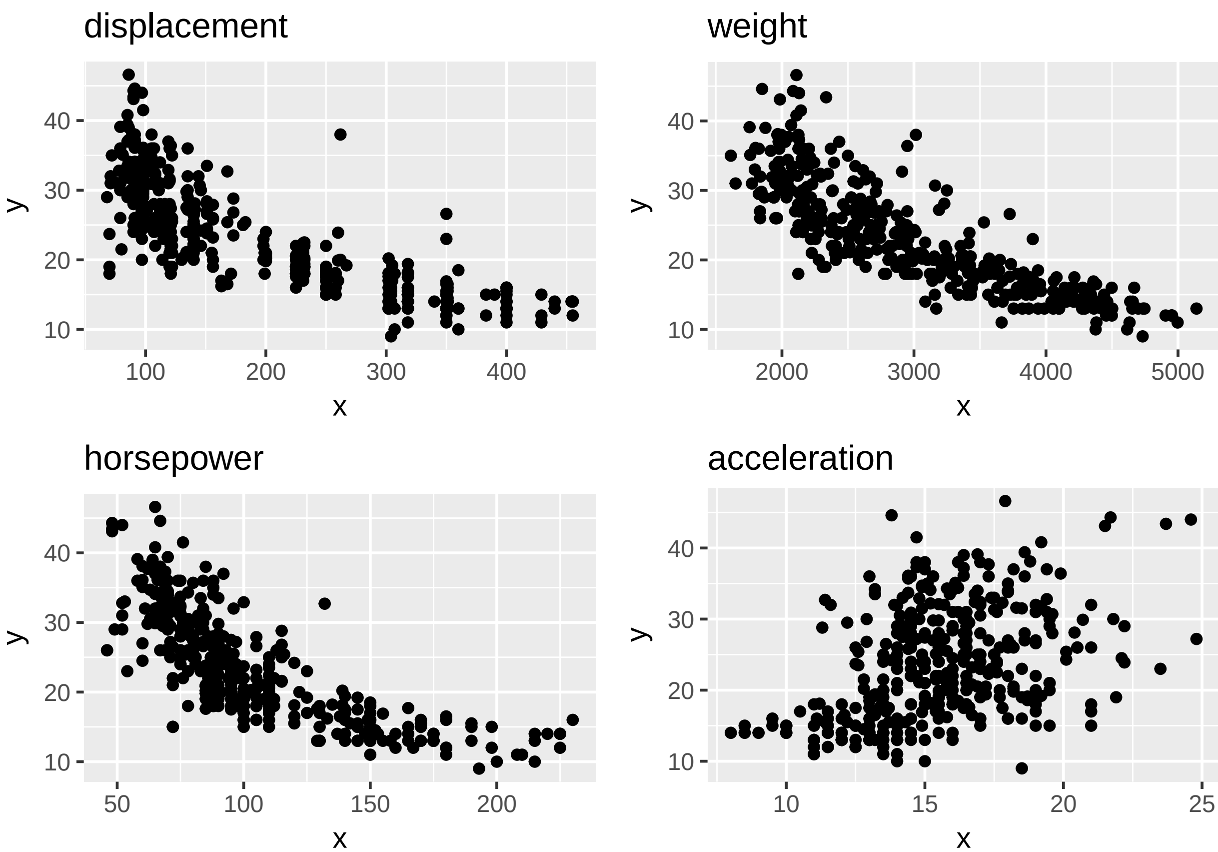
```
# OneSE best tune?
caret::oneSE(model$results,
             'RMSE',             # which metric are we optimizing with respect to
             num=nrow(model$resample), # how many hold outs did we make
             maximize=FALSE)     # Is bigger == better?
```

```
## [1] 4
```

Question 4

```
data('Auto', package='ISLR')
```

```r
varlist <- c('displacement','horsepower','weight','acceleration')
plots <- list()
i <- 1
for(col_ in  varlist){

  df <- data.frame(
    x = Auto[[col_]],
    y = Auto$mpg
  )
  plots[[col_]] <- ggplot(df,aes(x=x, y=y)) +
  geom_point() +
  ggtitle(col_)
  i <- i+1
}

multiplot(plotlist = plots,cols=2)
```



We can actually assume that all these four predictors,especially acceleration and horsepower. have nonlinear relationship with the response *mpg*.

Let's fit a smoother to horsepower, acceleration and weight but a standard linear relationship to displacement.

```r
library(mgcv)
```

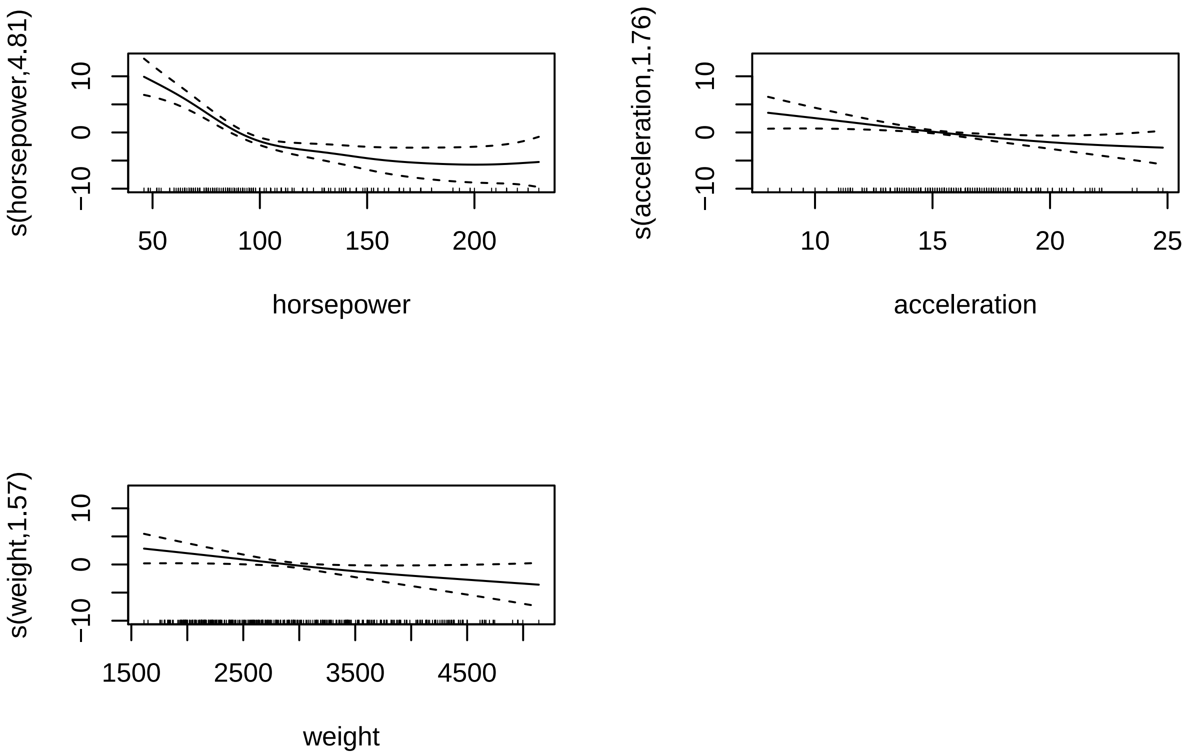
```
## Loading required package: nlme

##
## Attaching package: 'nlme'
```

```
## The following object is masked from 'package:dplyr':
##
##     collapse
## This is mgcv 1.8-15. For overview type 'help("mgcv-package")'.
##
## Attaching package: 'mgcv'
## The following objects are masked from 'package:gam':
##
##     gam, gam.control, gam.fit, plot.gam, predict.gam, s,
##     summary.gam
```

```
model <- mgcv::gam(mpg ~ displacement + s(horsepower) + s(acceleration) + s(weight), data=Auto)
mgcv::summary.gam(model)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## mpg ~ displacement + s(horsepower) + s(acceleration) + s(weight)
##
## Parametric coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  27.454784   1.293669  21.222  < 2e-16 ***
## displacement -0.020620   0.006579  -3.134  0.00186 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                   edf Ref.df      F  p-value
## s(horsepower)   4.806  5.901 11.335 1.21e-11 ***
## s(acceleration) 1.757  2.255  4.311   0.0112 *
## s(weight)       1.569  1.959  2.435   0.0775 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.759   Deviance explained = 76.5%
## GCV = 15.063  Scale est. = 14.673    n = 392
```

```
mgcv::plot.gam(model, pages=1 )
```



It looks like the relationship between *weight* , *acceleration* and *mpg* is not necessarily nonlinear. We can try to only fit nonlinear spline with horsepower.

```
model <- mgcv::gam(mpg ~ displacement + s(horsepower) + acceleration + weight, data=Auto)
mgcv::summary.gam(model)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## mpg ~ displacement + s(horsepower) + acceleration + weight
##
## Parametric coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  39.4567292  1.7101995  23.071  < 2e-16 ***
## displacement -0.0213751  0.0063519  -3.365 0.000842 ***
## acceleration -0.4046805  0.1278694  -3.165 0.001676 **
## weight       -0.0018693  0.0008574  -2.180 0.029858 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                 edf Ref.df    F p-value
## s(horsepower) 4.86  5.963 15.8  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.758   Deviance explained = 76.3%
## GCV = 15.075  Scale est. = 14.734    n = 392
```

```r
mgcv::plot.gam(model, pages=1 )
```