In [40]:
```r
library(ggplot2)
library(mgcv)
```

C3M4 Peer Reviewed Assignment

Outline:

The objectives for this assignment:

1. Observe the difference between GAMs and other regression models on simulated data.
2. Review how to plot and interpret the coefficient linearity for GAM models.

General tips:

1. Read the questions carefully to understand what is being asked.
2. This work will be reviewed by another human, so make sure that you are clear and concise in what your explanations and answers.

In [41]:
```r
# Load packages
library(ggplot2)
library(mgcv)
```

Problem 1: GAMs with Simulated Data

In this example, we show how to check the validity of a generalized additive model (GAM) (using the `gam()` function) using simulated data. This allows us to try and understand the intricacies of `gam()` without having to worry about the context of the data.

1. (a) Simulate the Data

Let $n = 200$. First, construct three predictor variables. The goal here is to construct a GAM with different types of predictor terms (e.g., factors, continuous variables, some that will enter linearly/parametrically, some that enter transformed).

1. x1: A continuous predictor that, we will suppose has a nonlinear relationship with the response.
2. x2: A categorical variable with three levels: `s` , `m` , and `t` .
3. x3: A categorical variable with two levels: `TRUE` and `FALSE` .

Then, make the response some nonlinear/nonparametric function of $\mathbf{x}$. For our case, use:

$$\log(\mu_i) = \beta_1 + \sin(0.5x_{i,1}^2) - x_{i,2} + x_{i,3}$$

This model is a Poisson GAM. In a realworld situation, we wouldn't know this functional relationship and would estimate it. Other terms are modeled parametrically. The response has normal noise.

Note that:

1. The construction of $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)^T$ has the linear predictor exponentiated, because of the nature of the link function.
2. We use $\boldsymbol{\mu}$ to construct $\mathbf{y} = (y_1, \ldots, y_n)^T$. The assumption for Poisson regression is that the random variable Y_i that generates y_i is Poisson with mean μ_i .
3. `as.integer(as.factor(VARIABLE))` converts the labels of VARIABLE to 1, 2, 3,.. so that we can construct the relationship for these factors.

Plot the relationship of $\mathbf{y}$ to each of the predictors. **Then, split the data into a training (`train_sim`) and test (`test_sim`) set.**

In [42]:
```
set.seed(0112358)
# n = number of data points
n = 200

#build dataframe
df = data.frame(x1= rnorm(200,12,12)^4,
                x2 = as.factor(sample(c('s','m','t'),size=n,replace=TR
                x3 = as.factor(sample(c('T','F'),size=n,replace=TRUE))
df$y = as.integer(as.factor(with(df, exp(sin(0.5 * x1^2) - as.integer(

#inspect
head(df)
summary(df)

#train test split
m = floor(.8 * length(df$x1))
idx = sample(seq(1,length(df$x1),1),m,replace=F)

tr = df[idx,]
```

```
te = df[-idx,]
print("Train Dimensions:")
dim(tr)
print("Test Dimensions:")
dim(te)

ggplot(df,aes(x1,y))+geom_point()+geom_smooth()+theme_bw()

ggplot(df,aes(x1,y, col=x2))+geom_point()+geom_smooth()+
        theme_bw()+scale_color_manual(values=c('black','#CFB87C', 'dar

ggplot(df,aes(x2,y, col=x2))+geom_point()+
        theme_bw()+scale_color_manual(values=c('black','#CFB87C', 'dar

ggplot(df,aes(x3,y, col=x2))+geom_point()+
        theme_bw()+scale_color_manual(values=c('black','#CFB87C', 'dar
```

A data.frame: 6 × 4

	x1	x2	x3	y
	<dbl>	<fct>	<fct>	<int>
1	1621.613	t	F	59
2	439788.318	m	F	151
3	113643.971	s	F	16
4	2899.356	s	F	118
5	537930.885	m	T	141
6	10286.277	t	F	42

```
       x1              x2       x3             y
 Min.    :       0   m:64   F:105   Min.   :  1.00
 1st Qu.:    1237   s:76   T: 95   1st Qu.: 50.75
 Median :   21906   t:60           Median :100.50
```

1. (b) Other Regression Models

Before jumping straight into GAMs, let's test if other regression models work. What about a regular linear regression model with ordinary least squares, and a generalized linear model for Poisson regression?

First fit a linear regression model to your `train_sim` data. We know that all of the predictors were used to make the response, but are they all significant in the linear regression model? Explain why this may be.

Then fit a Generalize Linear Model (GLM) to the `train_sim` data. Plot three diagnostic plots for your GLM:

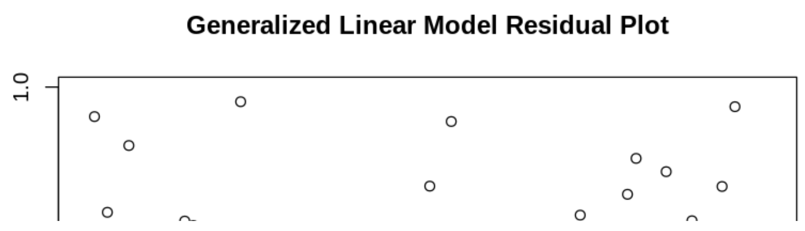
1. Residual vs. log(Fitted Values)
2. QQPlot of the Residuals
3. Actual Values vs. Fitted Values

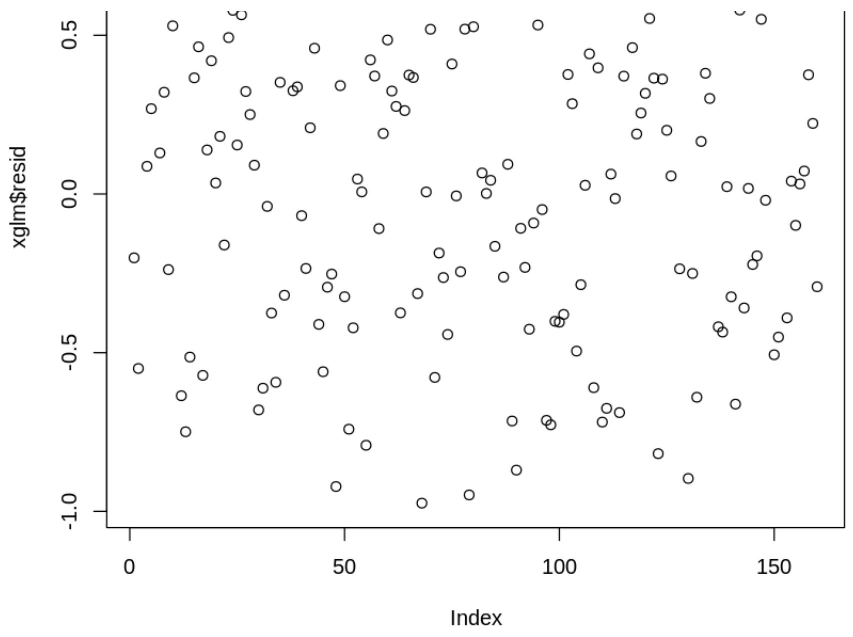
Using these plots, determine whether this model is a good fit for the data. Make sure to explain your conclusions and reasoning.

In [43]:
```
# Fit a LM model to the data
xlm = lm(y ~ x1 + x2 + x3, data=tr)
summary(xlm)

print("-----------------------------------------------------")
# Fit a GLM model to the data
xglm = glm(y~ x1 + x2 + x3, data = tr, family = poisson)
summary(xglm)
cat("R2: ", 1-(2717.1/6108.7))

#residual plot
plot(xglm$resid,main="Generalized Linear Model Residual Plot")

# Create the three specified plots
#Residual vs Log(Fitted)
plot(xglm$resid,log(xglm$fitted),main="Residuals vs Log(Fitted) for GL

# qq Plot
qqnorm(xglm$resid,main="Generalized Linear Model QQ Plot")
qqline(xglm$resid,col="red",lwd=2)

#fitted vs actual
plot(xglm$fitted,tr$y,main="Fitted vs Actual Values",xlab="Fitted Valu
```

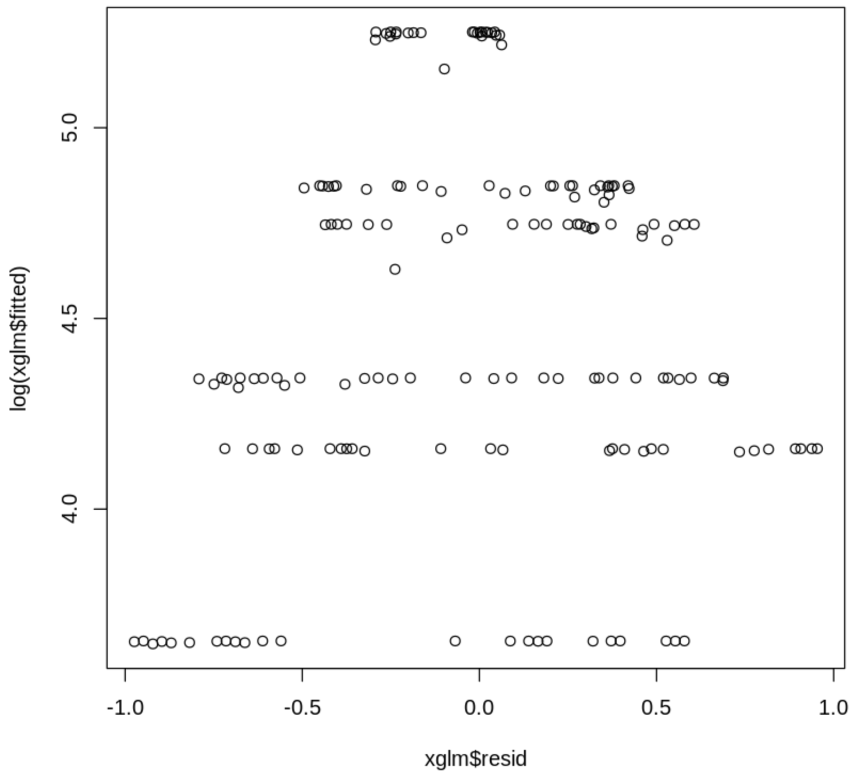
```
Call:
lm(formula = y ~ x1 + x2 + x3, data = tr)
```

```
Residuals:
    Min      1Q  Median      3Q     Max 
-62.717 -31.854   9.529  28.534  57.580 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.275e+02  6.114e+00  20.850  < 2e-16 ***
x1          -3.425e-06  4.434e-06  -0.773    0.441    
x2s         -5.015e+01  6.940e+00  -7.227 2.11e-11 ***
x2t         -1.020e+02  7.447e+00 -13.691  < 2e-16 ***
x3T          5.038e+01  5.694e+00   8.848 1.90e-15 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 35.95 on 155 degrees of freedom
Multiple R-squared:  0.6289,    Adjusted R-squared:  0.6193 
F-statistic: 65.66 on 4 and 155 DF,  p-value: < 2.2e-16

[1] "----------------------------------------------------"


Call:
glm(formula = y ~ x1 + x2 + x3, family = poisson, data = tr)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-8.4521  -3.8363   0.2803   3.1373   6.7385  

Coefficients:
              Estimate Std. Error z value Pr(>|z|)    
(Intercept)  4.747e+00  1.581e-02 300.151   <2e-16 ***
x1          -2.135e-08  1.090e-08  -1.958   0.0502 .  
x2s         -4.029e-01  1.734e-02 -23.235   <2e-16 ***
x2t         -1.093e+00  2.328e-02 -46.932   <2e-16 ***
x3T          5.042e-01  1.613e-02  31.254   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 6176.6  on 159  degrees of freedom
Residual deviance: 2717.2  on 155  degrees of freedom
AIC: 3713.2

Number of Fisher Scoring iterations: 5

R2:  0.5552081
```

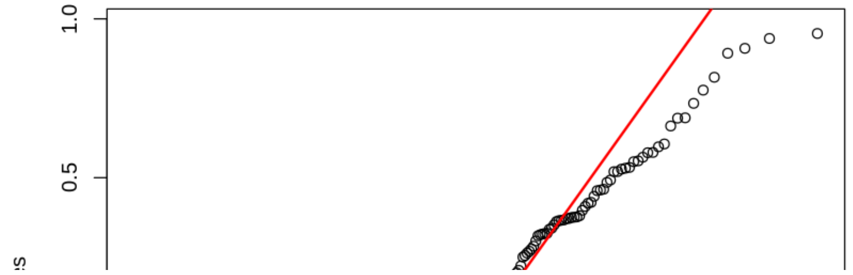


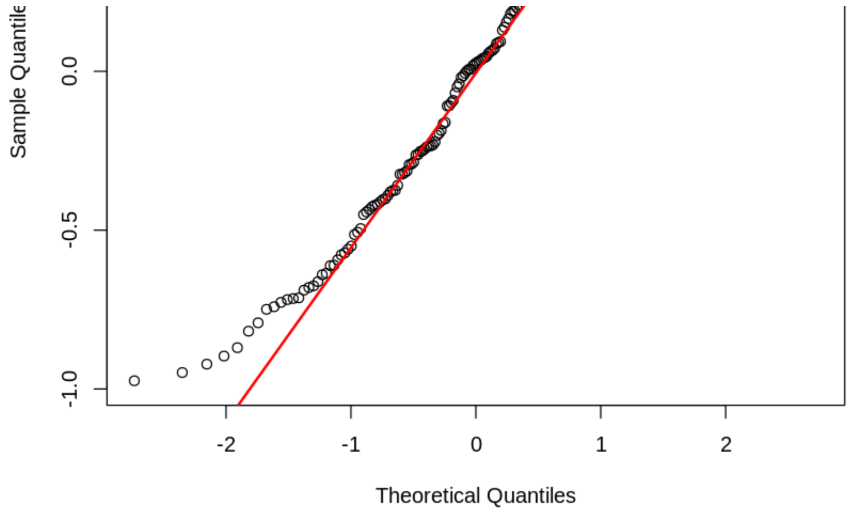


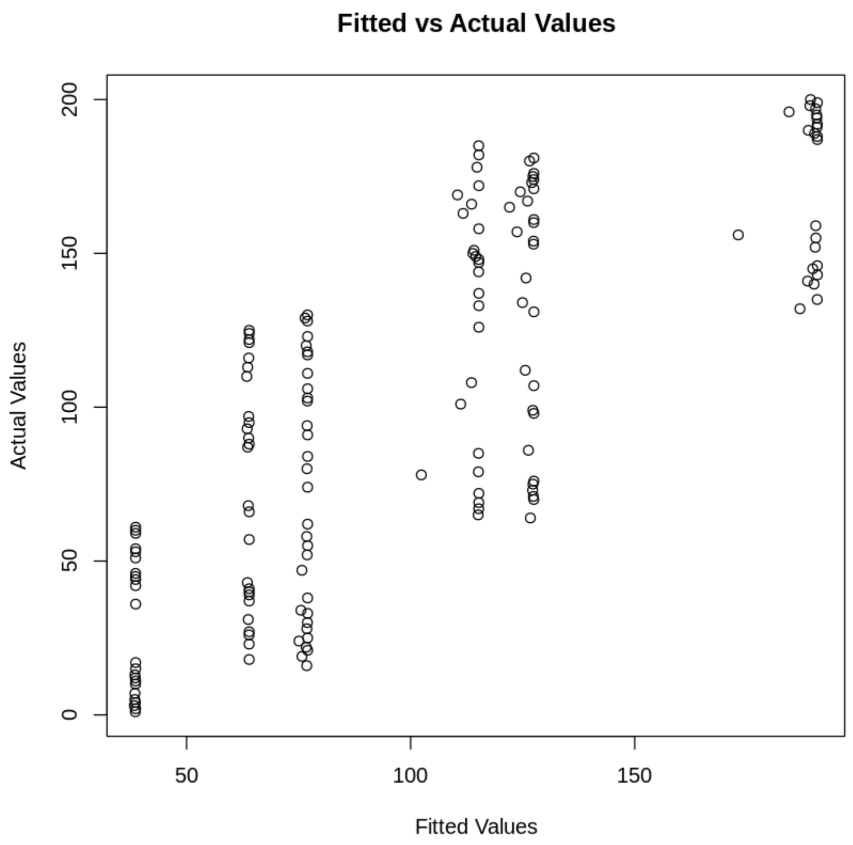
Residuals vs Log(Fitted) for GLM



Generalized Linear Model QQ Plot







The linear model is able to generalize about this data. However the accuracy is not adequate for implementation with an R^2 of roughly 62%. Several of the factors involved (x2 and x3) are linear, so it's not surprising that they have p-values that are statistically significant. The shortcoming of the model is apparent when scrutinizing the non-linear predictor (x1) which is not statistically insignificant and has a pvalue above 0.44.

The poisson GLM is able to explain roughly 55% of the variance in the data. This is interesting, the GLM isn't as accurate as the LM. The x1 predictor is statistically significant at the 0.1 alpha, but fails tests for significance at alphas of 0.05 or 0.01. The other two variables (in all their factors) are statistically significant in the GLM.

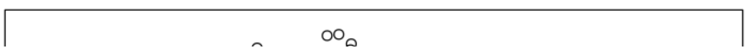
1. (c) Looking for those GAMs

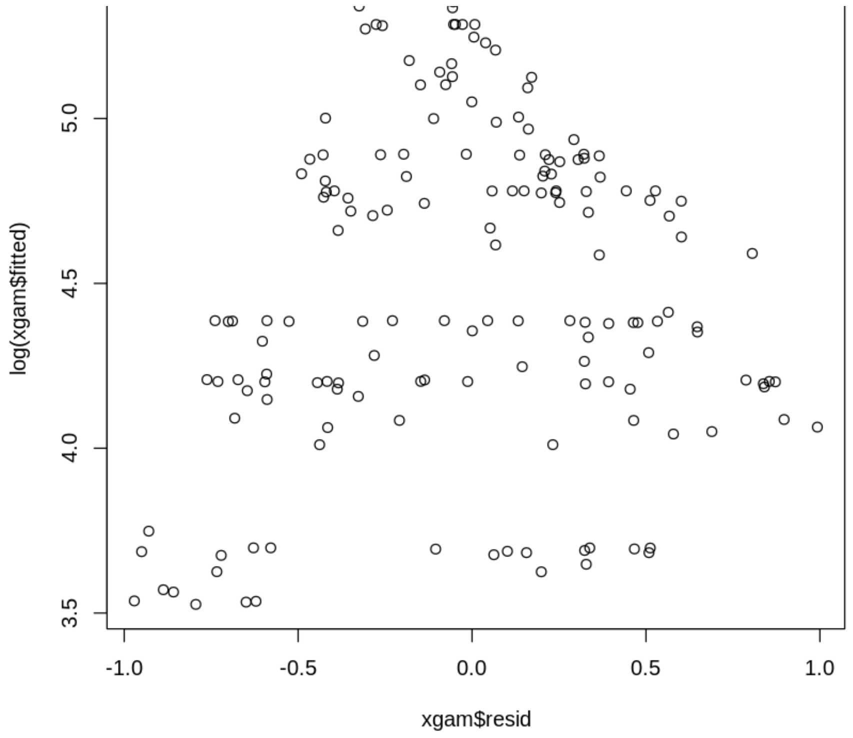
Now, it's time to see how a generalized additive model (GAM) performs! Fit a GAM to the data. Construct the same three plots for your GAM model. Do these plots look better than those of the GLM?

In [44]:
```
# Fit a GAM model to the data
xgam = gam(y~ s(x1) + x2 + x3, data = tr, family = poisson)
summary(xgam)

# Construct the three specified plots

#residual vs fitted
plot(xgam$resid,log(xgam$fitted),main="Residuals vs Fitted")

## qqplot
qqnorm(xgam$resid,main="qq Plot")
qqline(xgam$resid,col="red",lwd=2)

#fitted vs actual
plot(xgam$fitted,tr$y,main="Fitted vs Actual",xlab="Fitted Values",yla
```

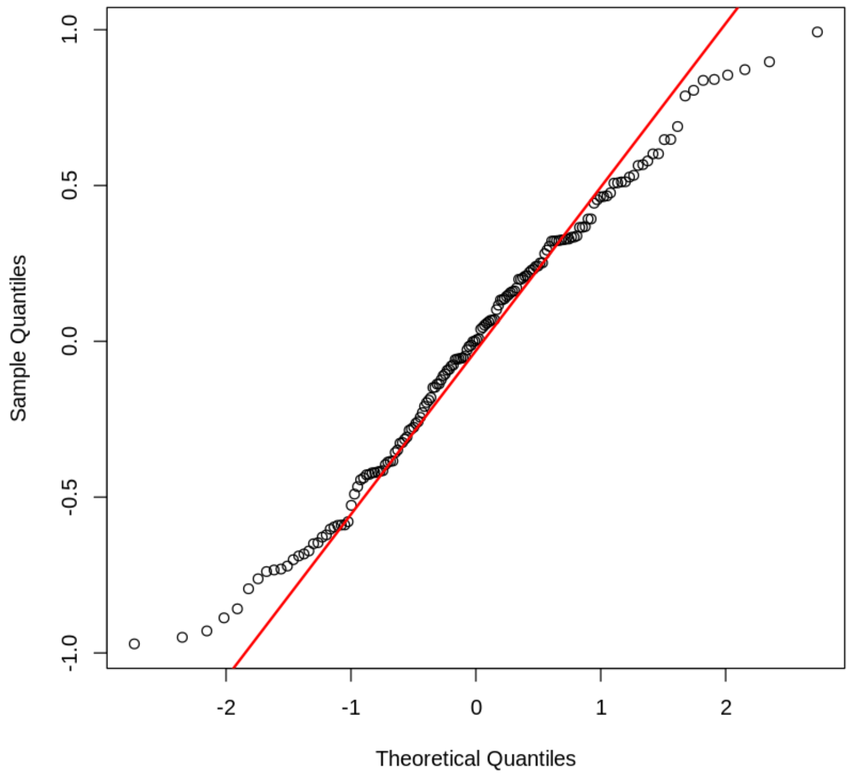
```
Family: poisson
Link function: log

Formula:
y ~ s(x1) + x2 + x3

Parametric coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.72907    0.01543  306.42   <2e-16 ***
x2s         -0.39366    0.01783  -22.08   <2e-16 ***
x2t         -1.08280    0.02358  -45.91   <2e-16 ***
x3T          0.50493    0.01654   30.53   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
        edf Ref.df Chi.sq p-value
s(x1) 8.819  8.987  172.2  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

R-sq.(adj) =  0.596   Deviance explained = 58.8%
UBRE = 15.056  Scale est. = 1         n = 160
```

Residuals vs Fitted

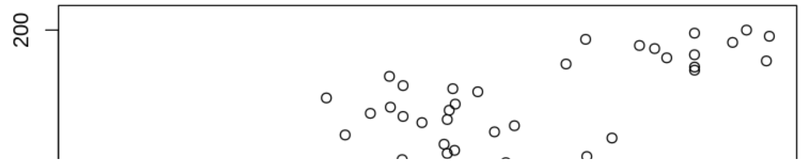


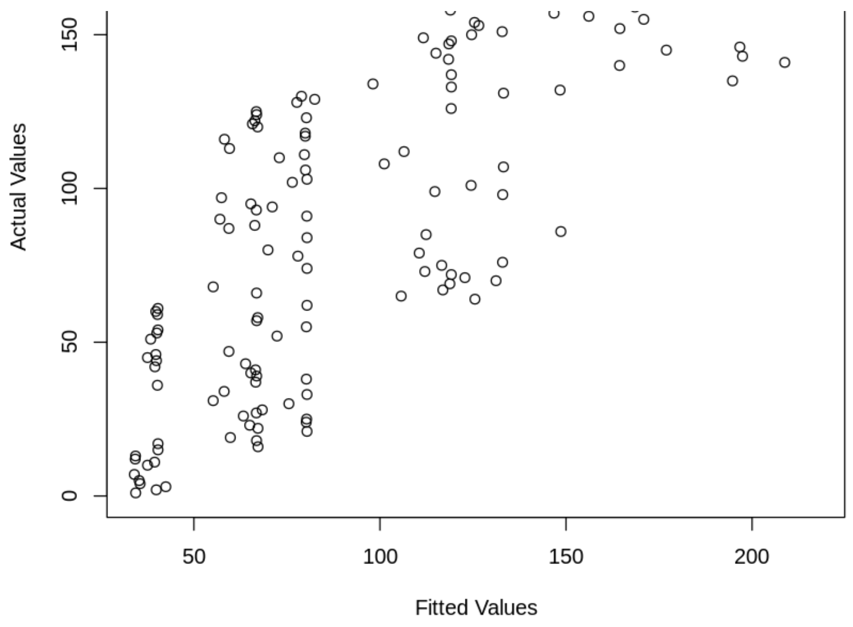


qq Plot



Fitted vs Actual





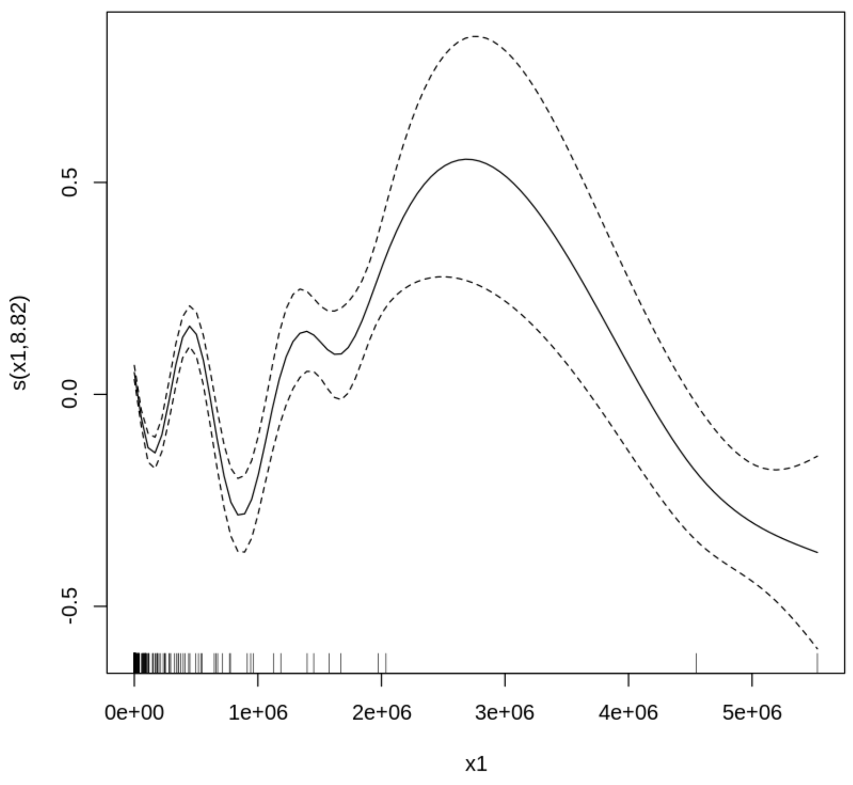
Yes, these graphs do look better. The residuals vs log(fitted) graph is not horizontally layered, the points are spread out in this graph. This points to the model's understanding of at least some of the non-linear pattern here. The qqplot illustrates more normal residuals, which is good. The Fitted Vs Actual graph is not as striated either, again showing that the model picked up on some of the non-linear relationship.

1. (d) Interpreting GAMs

We made a GAM model! However GAMs are harder to interpret than regular linear regression models. How do we determine if a GAM model was necesary? Or, in other words, how do we determine if our predictors have a linear relationship with the response?

Use the `plot.gam()` function in the mgcv library to plot the relationship between `y` and `x1`. Recall that x_1 entered our model as $\sin(0.5x_{i,1}^2)$, and we plotted that relationship in **1. (a)**. Does your plot confirm this relationship?

In [45]:
```
plot.gam(xgam)
```



Yes. The plot does confirm the general pattern shown above. There is a difference in the scales used in the two visualizations that, when taken into account, confirms the pattern.

1.(e) Model comparison

Compute the mean squared prediction error (MSPE) for each of the three models above (regression model, GLM, and GAM). State which model performs based according to this metric.

Remember, the MSPE is given by

$$MSPE = \frac{1}{k}\sum_{i=1}^{k}\left(y_i^\star - \widehat{y}_i^\star\right)^2$$

where $y_i^\star$ are the observed response values in the test set and $\widehat{y}_i^\star$ are the predicted values for the test set (using the model fit on the training set).

In [46]:
```
#mspe for lm
lmMSPE = mean((tr$y - xlm$fitted)^2)
cat("Linear Model MSPE:" , lmMSPE)

#mspe for glm
glmMSPE = mean((tr$y - xglm$fitted)^2)
cat("\nGeneralized Linear Model MSPE:" , glmMSPE)

# mspe for gam
gamMSPE = mean((tr$y - xgam$fitted)^2)
cat("\nGeneralized Additive Model MSPE:" , gamMSPE)
```

```
Linear Model MSPE: 1252.092
Generalized Linear Model MSPE: 1362.554
Generalized Additive Model MSPE: 1262.004
```

Surprisingly, the linear model wins with a MSPE of 1252. The GAM isn't far behind though with a MSPE of 1262. This is encouraging as the data has two linear predictors and a non-linear predictor. That the GAM is able to model the data fast and accurately justifies trying on unknown data.

Problem 2 Additive models with the advertising data

The following dataset containts measurements related to the impact of three advertising medias on sales of a product, P. The variables are:

- `youtube` : the advertising budget allocated to YouTube. Measured in thousands of dollars;
- `facebook` : the advertising budget allocated to Facebook. Measured in thousands of dollars; and
- `newspaper` : the advertising budget allocated to a local newspaper. Measured in thousands of dollars.
- `sales` : the value in the i^{th} row of the sales column is a measurement of the sales (in thousands of units) for product P for company i.

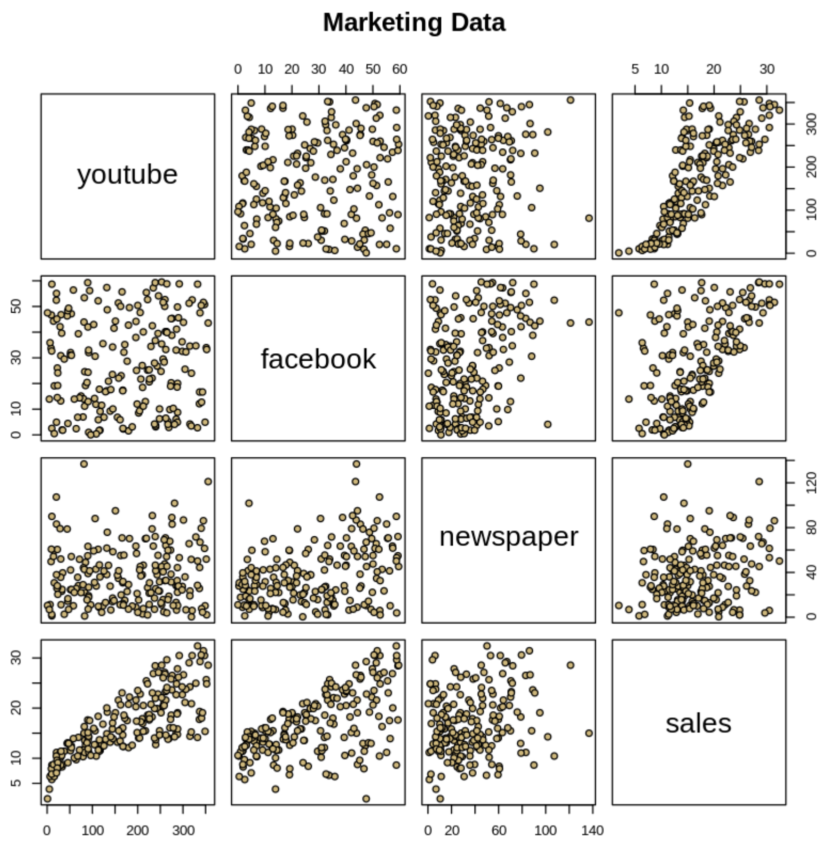
The advertising data treat "a company selling product P" as the statistical unit, and "all companies selling product P" as the population. We assume that the $n = 200$ companies in the dataset were chosen at random from the population (a strong assumption!).

First, we load the data, plot it, and split it into a training set (`train_marketing`) and a test set (`test_marketing`).

```
In [47]: # Load in the data
         marketing = read.csv("marketing.txt", sep = "")
         head(marketing)
         pairs(marketing, main = "Marketing Data", pch = 21,
               bg = c("#CFB87C"))
```

A data.frame: 6 × 4

	youtube	facebook	newspaper	sales
	<dbl>	<dbl>	<dbl>	<dbl>
1	276.12	45.36	83.04	26.52
2	53.40	47.16	54.12	12.48
3	20.64	55.08	83.16	11.16
4	181.80	49.56	70.20	22.20
5	216.96	12.96	70.08	15.48
6	10.44	58.68	90.00	8.64



In [48]:
```
set.seed(177) #set the random number generator seed.
n = floor(0.8 * nrow(marketing)) #find the number corresponding to 80%
index = sample(seq_len(nrow(marketing)), size = n) #randomly sample in

train_marketing = marketing[index, ] #set the training set to be the r
test_marketing = marketing[-index, ] #set the testing set to be the re
dim(test_marketing) #check the dimensions
dim(train_marketing) #check the dimensions
```

40 · 4

160 · 4

2.(a) Let's try a GAM on the marketing data!

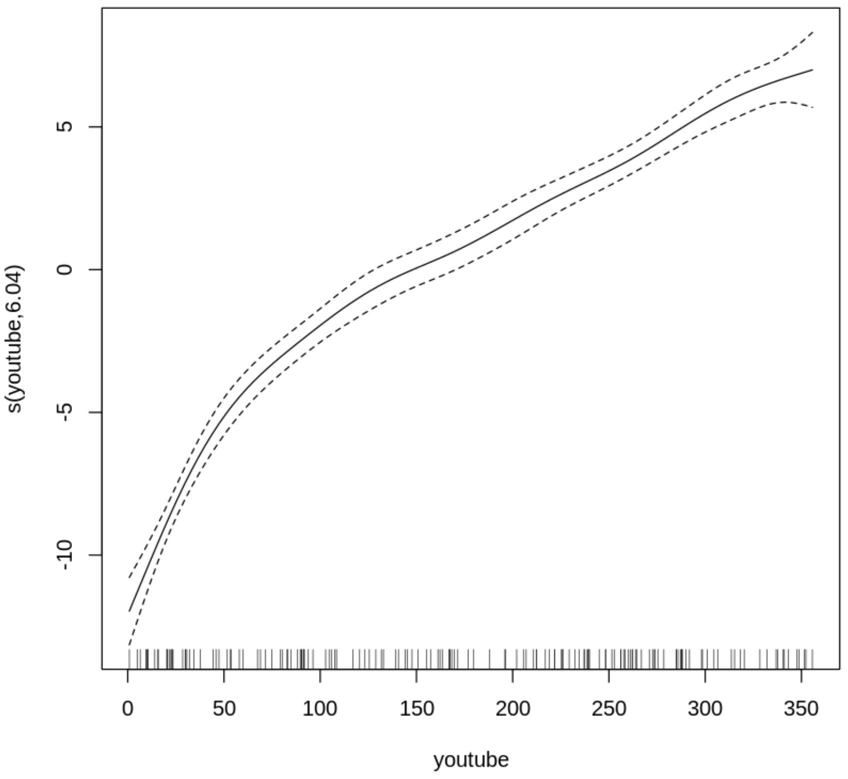
Note that the relationship between `sales` and `youtube` is nonlinear. This was a problem for us back in the first course in this specialization, when we modeled the data as if it were linear. In the last module, we focused on modeling the relationship between `sales` and `youtube`, omitting the other variables. Now it's time to include the additional predictors.

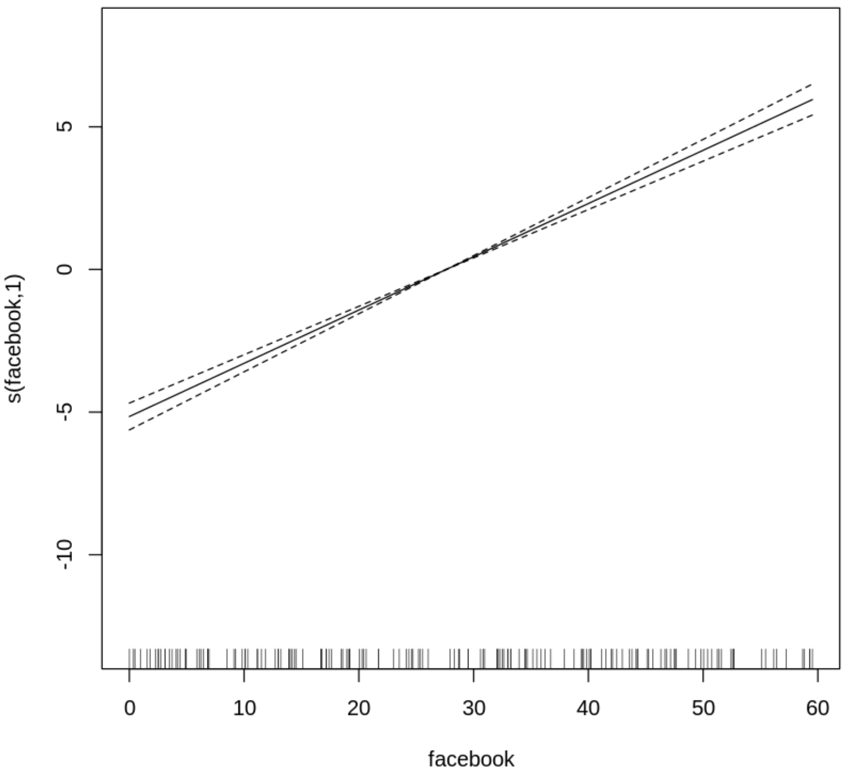
Using the `train_marketing` fit an additive model to the data and store it in `gam_marketing`. Produce the relevant added variable plots using `plot(gam_marketing)`. Comment on the fit of the model.

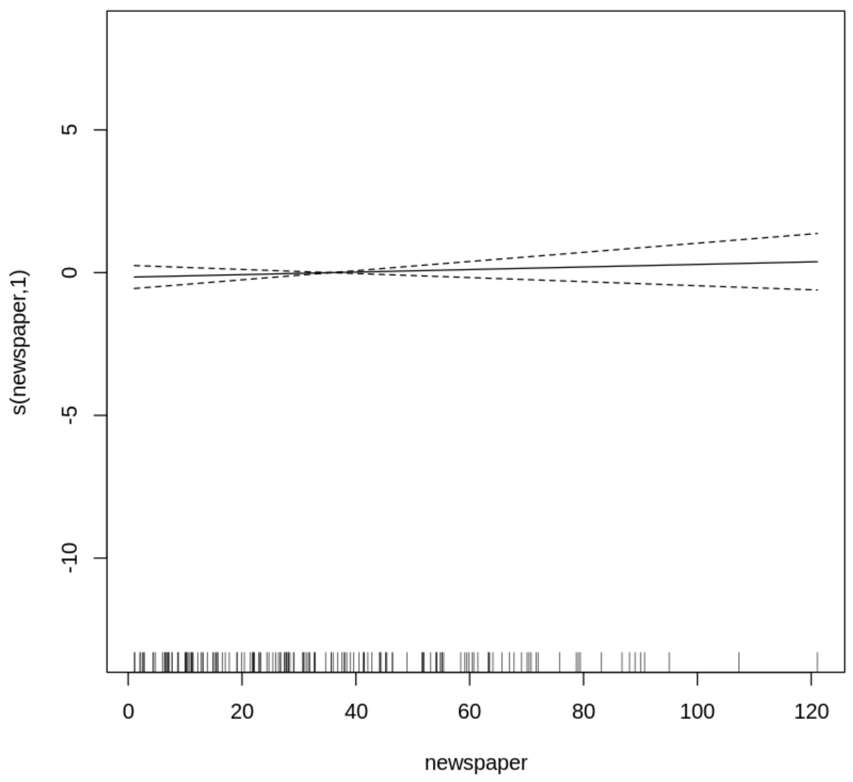
In [49]:
```
gam_marketing <- gam(sales~ s(youtube) + s(facebook) + s(newspaper), d
summary(gam_marketing)

plot.gam(gam_marketing)
```

```
Family: gaussian
Link function: identity

Formula:
sales ~ s(youtube) + s(facebook) + s(newspaper)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  16.5743     0.1321   125.5   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
               edf Ref.df       F p-value
s(youtube)   6.037  7.127 201.356  <2e-16 ***
s(facebook)  1.000  1.000 481.319  <2e-16 ***
s(newspaper) 1.000  1.000   0.593   0.442
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
R-sq.(adj) =  0.929   Deviance explained = 93.3%
GCV = 2.9572  Scale est. = 2.7902    n = 160
```







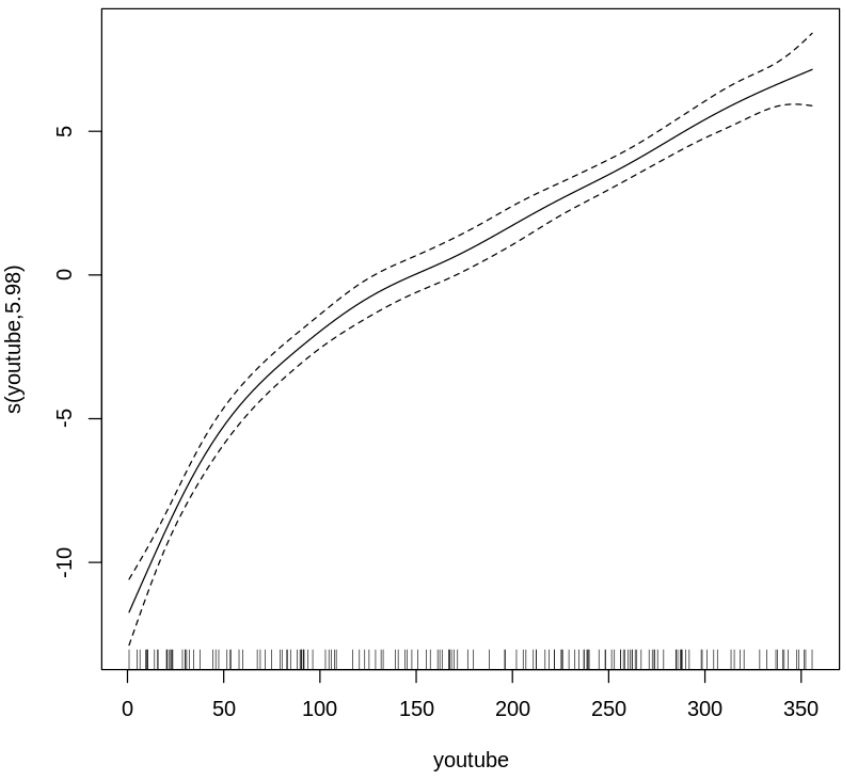
With a single line of code, this model fits very well. Exhibiting an adjusted R^2 of 0.929 is fairly impressive. However, there are some issues with the initial model. The GAM plots show us that two of the predictors are probably linear (a straight line inside the confidence intervals). These findings are corroborated by EDS at 1 for both newspaper and facebook predictors. Worse, the newspaper predictor has an associated p-value of 0.44, telling us that the fit fails an Ftest (where the null hypothesis states that the newspaper slope is equal to 0). We should change both facebook and newspaper predictors to linear inputs, and recheck the pvalues associated to ensure they are statistically significant. This model is pretty good, but not good enough.

2.(b) Semiparametric modeling of the marketing data

Refit the additive model based on your results from 2.(a). That is, if any predictors above should enter linearly, refit the model to reflect that. If any predictors are statistically insignificant, remove them from the model. Store your final model in `semiparametric_marketing`.

In [50]:
```
# semiparametric_marketing = gam(sales ~ s(youtube) + facebook + newsp
# newspaper predictor fails t-test at 0.10 alpha. Model Yields 0.929 a

semiparametric_marketing = gam(sales ~ s(youtube) + facebook , data=tr
summary(semiparametric_marketing)
plot.gam(semiparametric_marketing)
```

```
Family: gaussian
Link function: identity

Formula:
sales ~ s(youtube) + facebook

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 11.352099   0.253895   44.71   <2e-16 ***
facebook     0.189085   0.007848   24.09   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
             edf Ref.df     F p-value
s(youtube) 5.979  7.127 200.9  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

R-sq.(adj) =  0.929   Deviance explained = 93.2%
GCV = 2.9438  Scale est. = 2.797     n = 160
```



Great, in two lines of code we were able to establish that the newspaper predictor is in fact statistically insignificant. Not only did newspaper fail a t-test at the .1 alpha threshold for a linear predictor, removing the variable resulted in no change to the $_aR^2$ value of 0.929. Moving the facebook predictor to a linear predictor resulted in a pvalue of 2e-16 which is highly significant. The plot confirms that we have a non-linear predictor in the youtube predictor, and aren't smoothing linear predictors.

This model is probably better than the first because it is more simple, has an identical R^2 value, and will be easier to explain or use.

```
In [ ]:
```

2.(c) Model comparisons

Now, let's do some model comparisons on the test data. Compute the mean squared prediction error (MSPE) on the `test_marketing` data for the following three models:

- `gam_marketing` from 2.(a)
- `semiparametric_marketing` from 2.(b)
- `lm_marketing` , a linear regression model with `sales` is the response and `youtube` and `facebook` are predictors (fit on the `train_marketing` data).

State which model performs based according to this metric.

In [51]:
```
#mspe for gam
mod1MSPE <- mean((predict.gam(gam_marketing, newdata=test_marketing)-t
cat("GAM Marketing MSPE:", mod1MSPE )
cat('\n------------------------------------------------------------\n'

#mspe for semiparametric
mod2MSPE <- mean((predict.gam(semiparametric_marketing, newdata=test_m
cat("\nSemiparametric Marketing Model MSPE:", mod2MSPE)
cat('\n------------------------------------------------------------\n'

# mspe for lm
lm_marketing <- lm(sales ~ youtube + facebook , data = train_marketing
summary(lm_marketing)
cat('\n------------------------------------------------------------\n'

lmMSPE <- mean((predict(lm_marketing,newdata = test_marketing) - test_
cat("\nLinear Marketing Model MSPE:", lmMSPE)
```

```
GAM Marketing MSPE: 3.438202
------------------------------------------------------------

Semiparametric Marketing Model MSPE: 3.297448
------------------------------------------------------------



Call:
lm(formula = sales ~ youtube + facebook, data = train_marketing)

Residuals:
     Min       1Q   Median       3Q      Max 
-10.1914  -0.9730   0.3786   1.4414   3.2354 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 3.617054   0.390963   9.252   <2e-16 ***
youtube     0.046430   0.001522  30.500   <2e-16 ***
facebook    0.177932   0.009302  19.129   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.022 on 157 degrees of freedom
Multiple R-squared:  0.8973,	Adjusted R-squared:  0.896 
F-statistic:   686 on 2 and 157 DF,  p-value: < 2.2e-16



------------------------------------------------------------


Linear Marketing Model MSPE: 4.197701
```

The semiparametric marketing model performed better than the alternatives, having the lowest MSPE at 3.297. The scrutiny paid to insignificant predictors (removing newspaper) and altering the input method for linear predictors (removing the smooth on facebook) results in an easier to explain and more accurate model. The basic GAM marketing model does beat a vanilla linear model, even with the removal of the insignificant newspaper predictor.

```
In [ ]:
```

```
In [ ]:
```