Generalized Additive Model (GAM)

Quick Introduction to GAM using mgcv

Kamarul Imran Musa Jason Ng

2024-11-04

Table of contents

0.1 Prepare environment

- open new R project
- find and select a folder (that contains the data)
- load the packages
  - **tidyverse** for data management
  - **mgcv** for generalized additive model
  - **gamair** to get access to gam codes and dataset

```
library(tidyverse)
library(mgcv)
library(gamair)
```

0.2 Data 1

- Data `chicago`

```
data(chicago)
```

0.2.1 Explore data

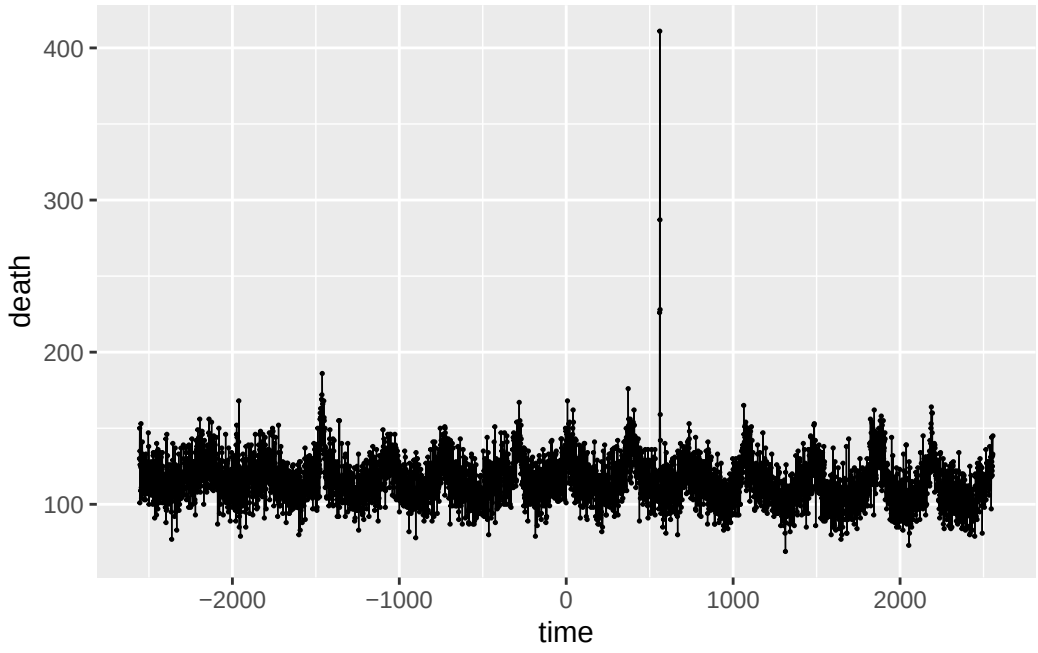
- outcome is death (count)
- median of pm10
- median of pm2.5
- median of ozone
- time
- temperature

```
summary(chicago)
```

```
     death          pm10median         pm25median         o3median      
 Min.   : 69.0   Min.   :-37.3761   Min.   :-16.426   Min.   :-24.779  
 1st Qu.:105.0   1st Qu.:-13.1082   1st Qu.: -6.588   1st Qu.:-10.232  
 Median :114.0   Median : -3.5391   Median : -1.326   Median : -3.326  
 Mean   :115.4   Mean   : -0.1464   Mean   :  0.243   Mean   : -2.179  
 3rd Qu.:124.0   3rd Qu.:  8.3029   3rd Qu.:  5.344   3rd Qu.:  4.468  
 Max.   :411.0   Max.   :320.7248   Max.   : 38.150   Max.   : 43.688  
                 NA's   :251        NA's   :4387                       
   so2median            time             tmpd      
```

```
 Min.   :-8.2061   Min.   :-2556   Min.   :-16.00
 1st Qu.:-2.6894   1st Qu.:-1278   1st Qu.: 35.00
 Median :-1.2183   Median :    0   Median : 51.00
 Mean   :-0.6361   Mean   :    0   Mean   : 50.19
 3rd Qu.: 0.8316   3rd Qu.: 1278   3rd Qu.: 67.00
 Max.   :28.9034   Max.   : 2556   Max.   : 92.00
 NA's   :27
```

```r
chicago |>
  ggplot(aes(x = time, y = death)) +
  geom_line(linewidth = 0.25) +
  geom_point(size = 0.25)
```



0.2.2 Analysis with GAM

- only one variable with smooth function

```r
ap0 <- gam(death ~ s(time, bs="cr", k = 200) +
             pm10median + so2median+  o3median + tmpd,
           data = chicago, family = poisson)
```

- the observed numbers of deaths are Poisson random variables

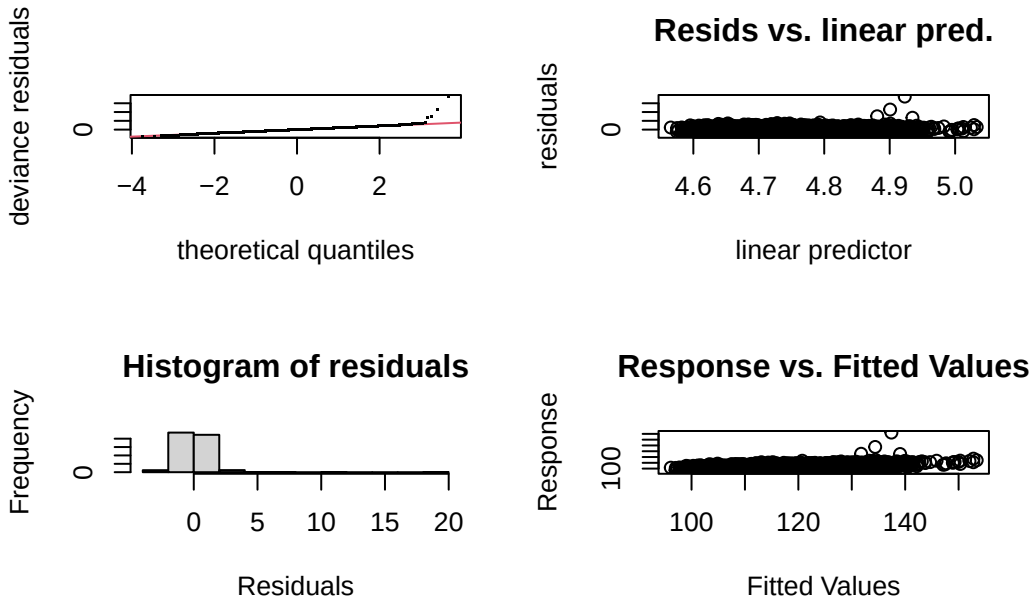
- an underlying mean that is the product of a basic, time varying, death rate
- modified through multiplication by pollution dependent effects.
- A cubic regression spline has been used for `f` to speed up computation a little

0.2.3 Results

```
summary(ap0)
```

```
Family: poisson
Link function: log

Formula:
death ~ s(time, bs = "cr", k = 200) + pm10median + so2median +
    o3median + tmpd

Parametric coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) 4.7006394  0.0091690 512.666  < 2e-16 ***
pm10median  0.0004321  0.0000889   4.861 1.17e-06 ***
so2median   0.0008184  0.0005523   1.482    0.138
o3median    0.0009306  0.0002032   4.581 4.63e-06 ***
tmpd        0.0009318  0.0001784   5.223 1.76e-07 ***
---
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(time) 168.6    188   2090  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.351   Deviance explained = 38.7%
UBRE = 0.25467  Scale est. = 1         n = 4841
```

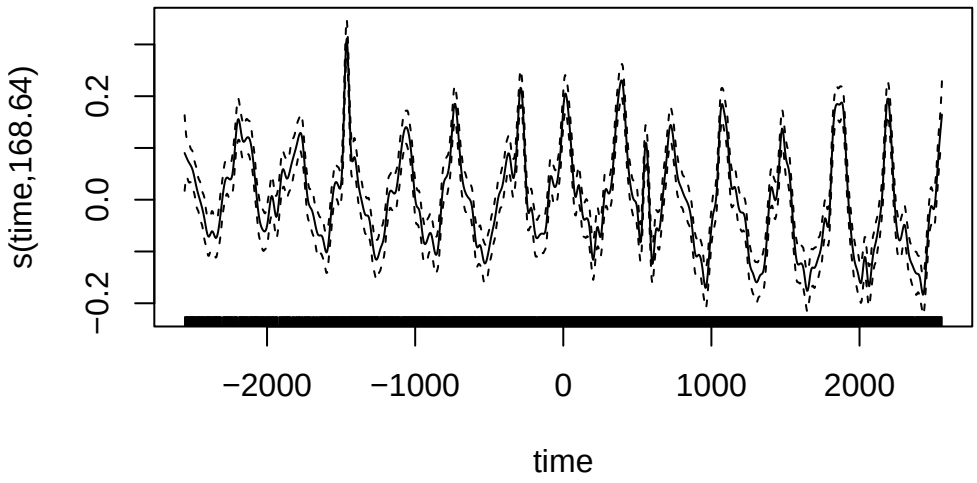
```
gam.check(ap0)
```



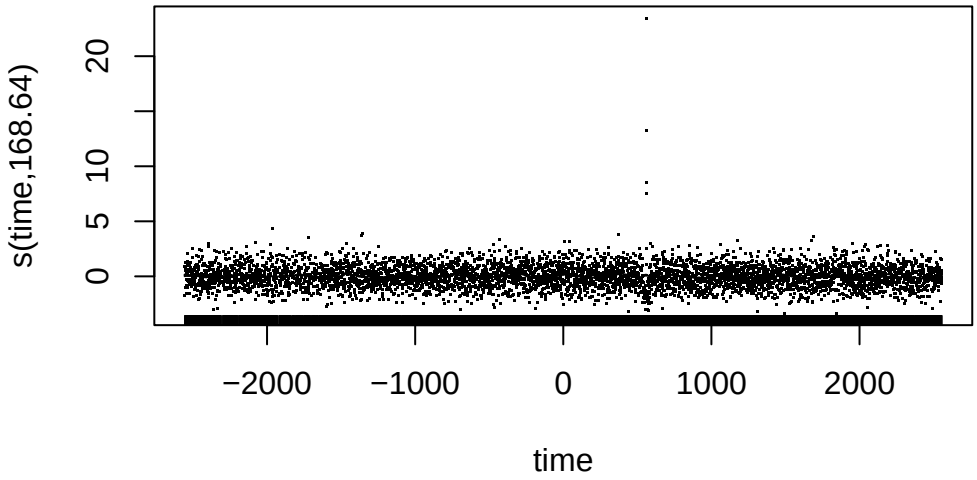

```
Method: UBRE   Optimizer: outer newton
full convergence after 3 iterations.
Gradient range [3.514596e-08,3.514596e-08]
(score 0.2546689 & scale 1).
Hessian positive definite, eigenvalue range [0.004247567,0.004247567].
Model rank =  204 / 204

Basis dimension (k) checking results. Low p-value (k-index<1) may
indicate that k is too low, especially if edf is close to k'.

         k' edf k-index p-value
s(time) 199 169    0.92  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

0.2.4 Plots

```
plot(ap0, n = 1000) # n increased to make plot smooth
```




```
plot(ap0, residuals = TRUE, n = 1000)
```



- and four gross outliers, in close proximity to each other, are clearly visible.
- examination of the data indicates that the outliers are the four highest daily death rates occurring in the data
- they occurred on consecutive days,

0.2.5 Adding more smooth functions

- some non-linear response of death rate to temperature and ozone is required
- replacing the linear dependencies on the air quality covariates with smooth functions
- so that the model structure becomes

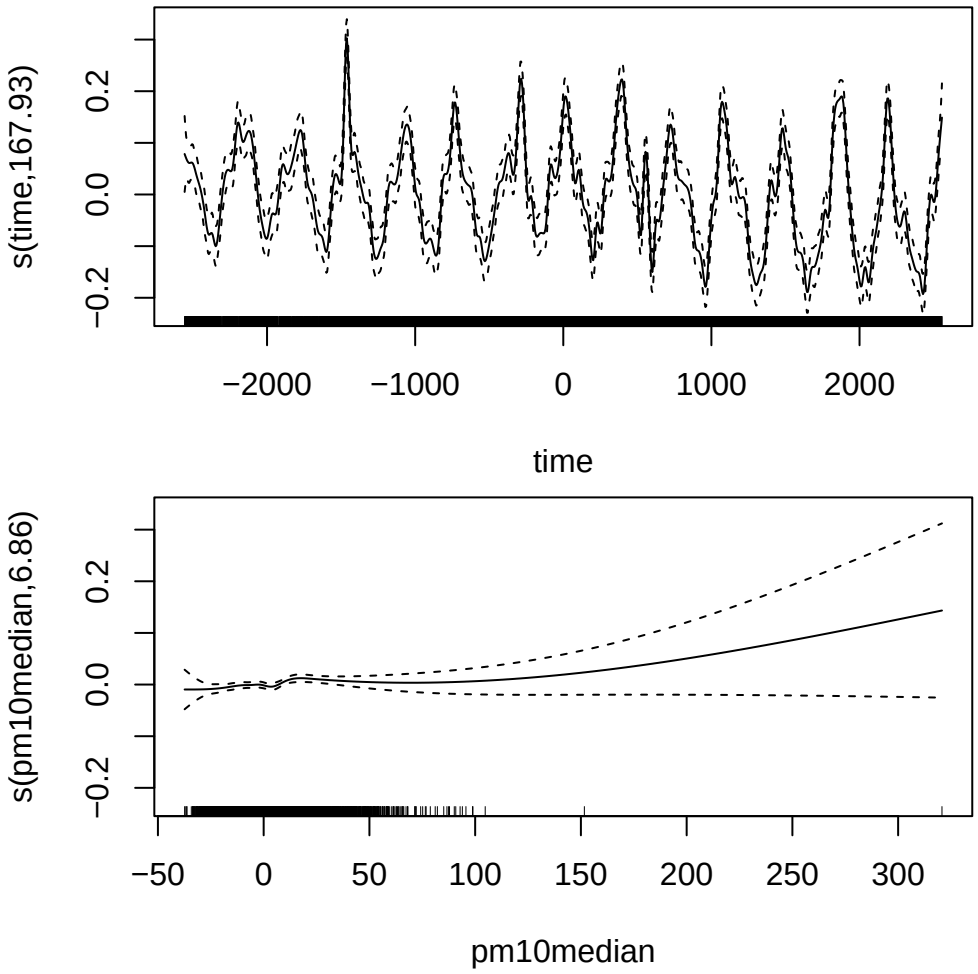
```r
ap1 <- gam(death ~ s(time, bs = "cr", k = 200) +
             s(pm10median, bs = "cr") +
             s(so2median, bs = "cr") +
             s(o3median, bs = "cr") +
             s(tmpd, bs = "cr"),
           data = chicago, family = poisson)
```

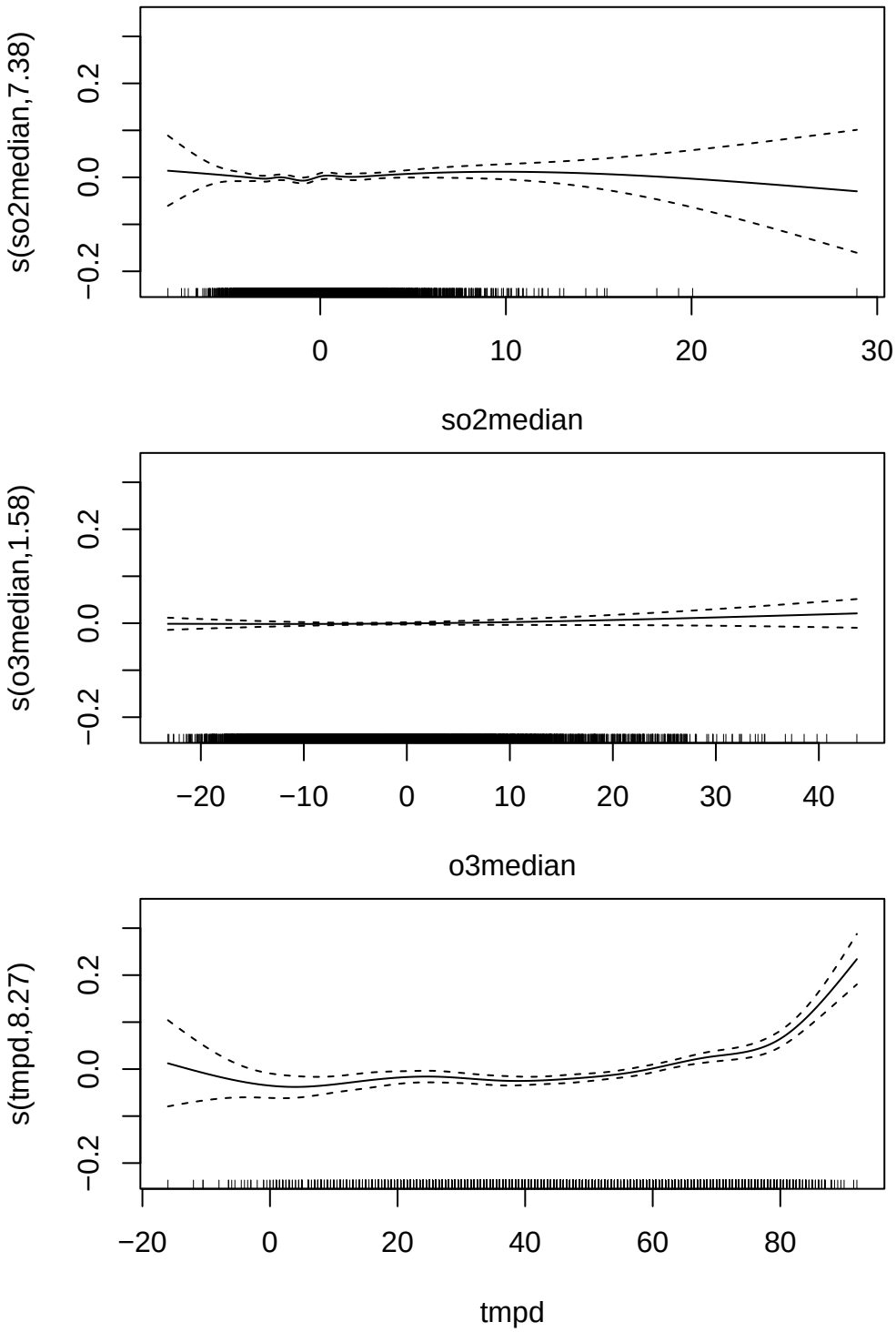
```r
summary(ap1)
```

```
Family: poisson
Link function: log

Formula:
death ~ s(time, bs = "cr", k = 200) + s(pm10median, bs = "cr") +
    s(so2median, bs = "cr") + s(o3median, bs = "cr") + s(tmpd,
    bs = "cr")

Parametric coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 4.744645   0.001342    3534   <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(time)       167.933 187.535 1827.171  <2e-16 ***
s(pm10median)   6.863   7.695   14.480  0.0526 .
s(so2median)    7.382   8.141    9.221  0.3415
s(o3median)     1.579   1.985    1.916  0.3493
s(tmpd)         8.270   8.850  105.360  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.361   Deviance explained = 39.8%
UBRE = 0.24107  Scale est. = 1         n = 4841
```

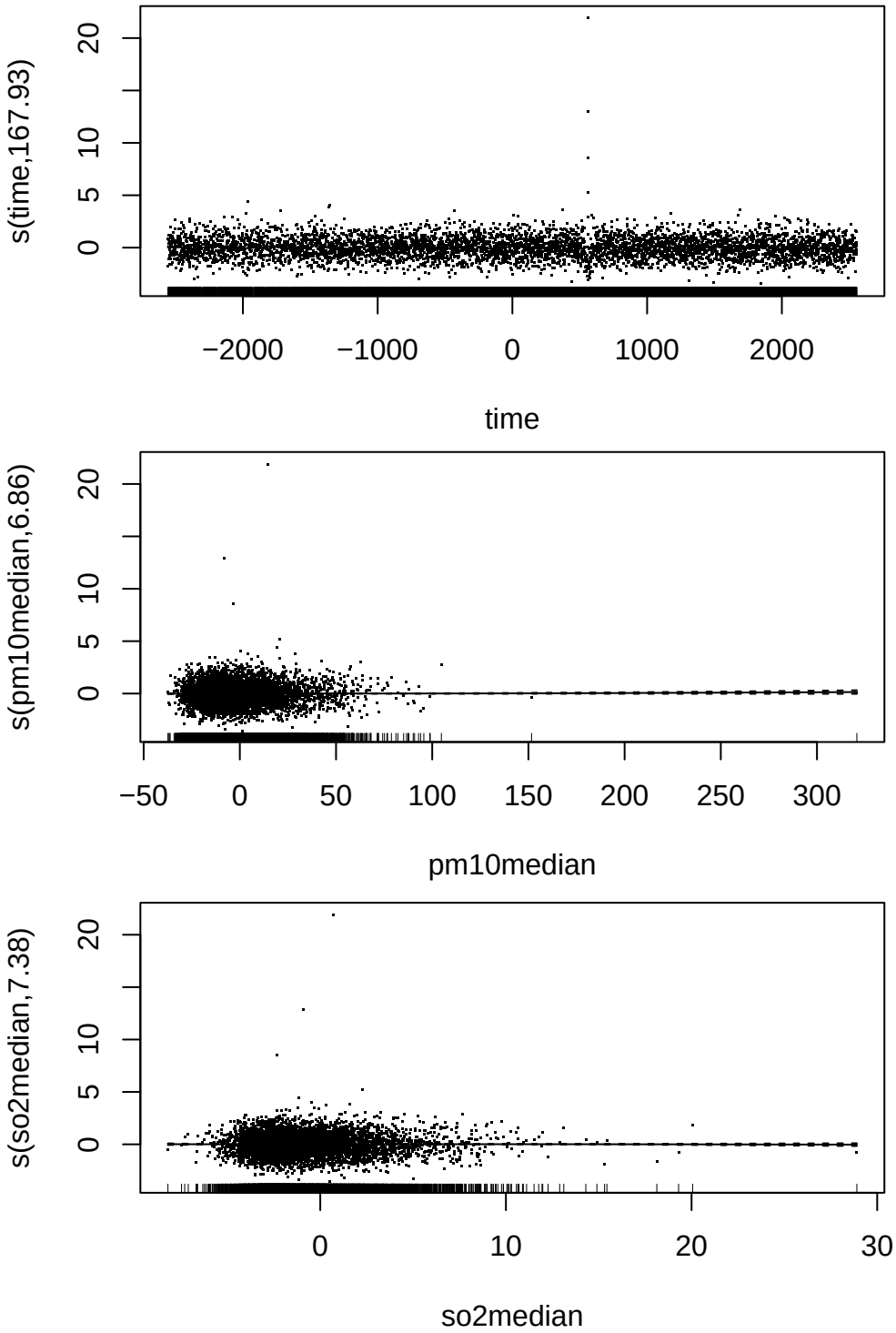
0.2.6 Plots

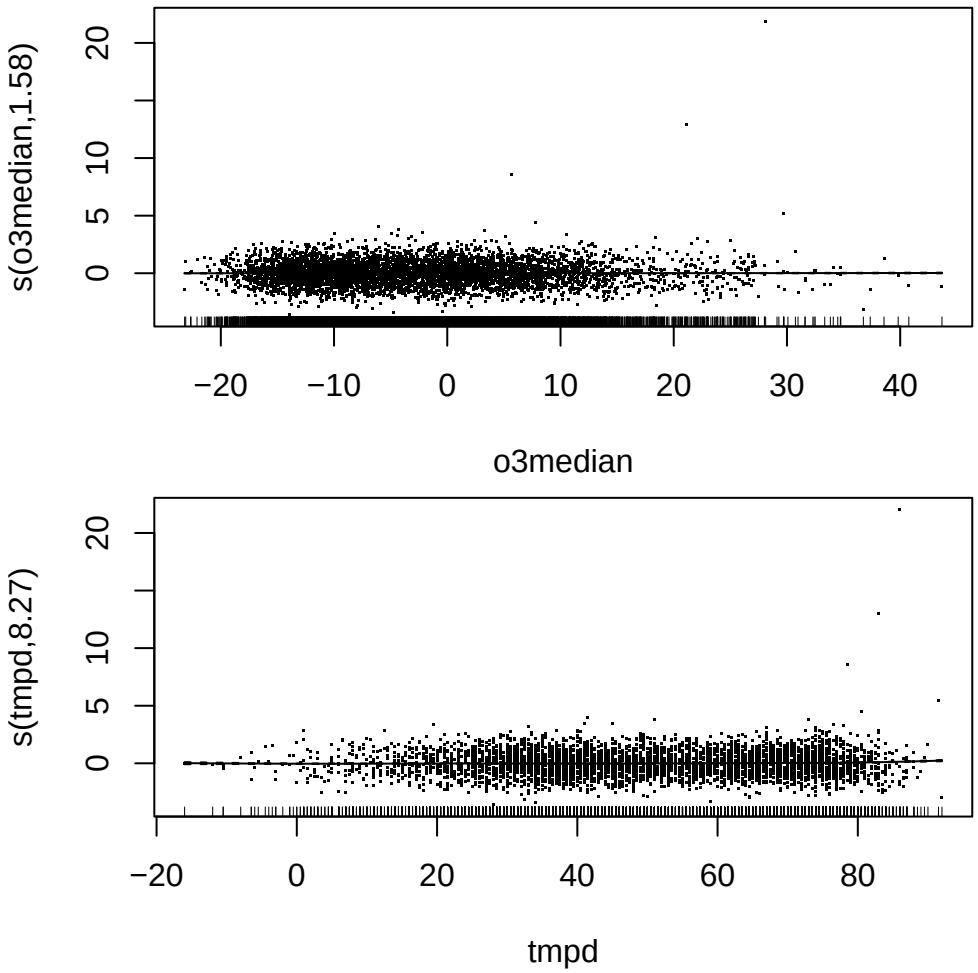
```r
plot(ap1, n = 1000)  # n increased to make plot smooth
```




s(so2median,7.38)
0.2
0.0
−0.2
0
10
20
30
so2median
s(o3median,1.58)
0.2
0.0
−0.2
−20
−10
0
10
20
30
40
o3median
s(tmpd,8.27)
0.2
0.0
−0.2
−20
0
20
40
60
80
tmpd

```r
plot(ap1, residuals = TRUE, n = 1000)
```





0.3 Data 2

- let's say we have a simulated data with only
  - outcome is admission
  - a predictor which is `pm2.5`

0.3.1 Data with one predictor

```
glimpse(data_simple)
```

```
Rows: 1,000
```

```
Columns: 2
$ pm25       <dbl> 28.757752, 78.830514, 40.897692, 88.301740, 94.046728, 4.55~
$ admissions <dbl> 20, 60, 36, 79, 75, 11, 35, 66, 42, 34, 84, 35, 58, 45, 11,~
```

0.3.2 Data with three predictors

- outcome is admission (count)
- predictors are
  - Particulate Matter 2.5 `PM2.5`
  - Temperature `temp`
  - Relative Humidity `rh`

```
data_complex |>
  View()
```

```
glimpse(data_complex)
```

```
Rows: 1,000
Columns: 4
$ pm25       <dbl> 28.757752, 78.830514, 40.897692, 88.301740, 94.046728, 4.55~
$ temp       <dbl> 15.89507, 18.46371, 15.48951, 23.13534, 25.60178, 30.63607,~
$ rh         <dbl> 44.91779, 89.36779, 73.02731, 69.10370, 44.43337, 35.20757,~
$ admissions <dbl> 34, 90, 40, 104, 129, 22, 65, 99, 71, 55, 118, 48, 72, 77, ~
```

0.3.3 GAM with one predictor

- Estimate effect of PM2.5

```
# Fit simple GAM model
model_simple <- gam(admissions ~ s(pm25),
                    family = poisson(),  # for count data
                    data = data_simple)
summary(model_simple)
```

```
Family: poisson
Link function: log

Formula:
```

```
admissions ~ s(pm25)

Parametric coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 3.547039   0.005907   600.4   <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(pm25) 3.295  4.093  11246  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.955   Deviance explained = 93.1%
UBRE = -0.023647  Scale est. = 1         n = 1000
```

- Used `s()` smooth function for PM2.5
- Poisson family as we're modeling count data

Arguments explained:

`s()` creates smooth term `family = poisson()` specifies distribution family `data` is input dataset

0.3.4 Predict mortality

- values for predictions

```r
# Generate predictions
pm25_seq <- seq(min(pm25), max(pm25), length.out = 100)
```

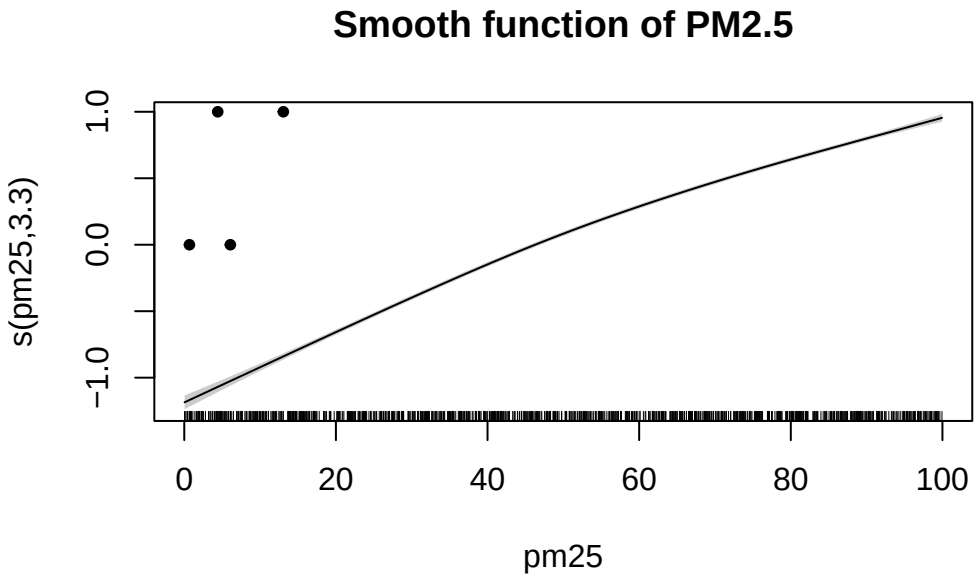
- make predictions

```r
pred <- predict(model_simple,
                newdata = data.frame(pm25 = pm25_seq),
                se.fit = TRUE)
```

0.3.5 Plot results

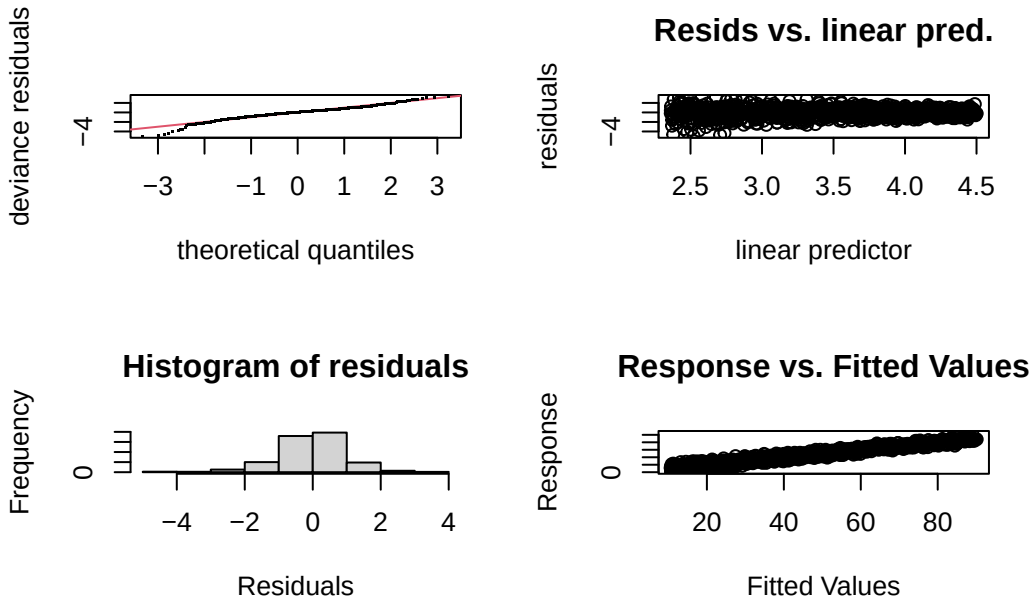
- smooth term

```
# Plot smooth term
plot(model_simple, shade = TRUE,
     main = "Smooth function of PM2.5")
# Add observed data points
points(data_simple$pm25,
       data_simple$admissions, col = "black", pch = 20)
```



0.3.6 Model checking

```
# Model checking
gam.check(model_simple)
```




```
Method: UBRE   Optimizer: outer newton
full convergence after 4 iterations.
Gradient range [7.028009e-08,7.028009e-08]
(score -0.02364664 & scale 1).
Hessian positive definite, eigenvalue range [0.002102548,0.002102548].
Model rank =  10 / 10

Basis dimension (k) checking results. Low p-value (k-index<1) may
indicate that k is too low, especially if edf is close to k'.

         k' edf k-index p-value
s(pm25) 9.0 3.3    1.01    0.66
```

0.3.7 Compare with glm

- glm model

```
glm_simple <- glm(admissions ~ pm25, family = poisson(), data = data_simple)
```

- compare

```r
anova(glm_simple, model_simple, test = "Chisq")
```

```
Analysis of Deviance Table

Model 1: admissions ~ pm25
Model 2: admissions ~ s(pm25)
  Resid. Df Resid. Dev    Df Deviance  Pr(>Chi)
1     998.0    1118.08
2     995.7     967.76 2.295   150.32 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

0.3.8 GAM with four predictors

```r
# Fit complex GAM model with interaction
model_complex <- gam(admissions ~ s(pm25) + s(temp) + s(rh) +
                       ti(pm25, temp),  # tensor product interaction
                     family = poisson(),
                     data = data_complex)
```

- Added multiple smooth terms `s()` for each predictor
- Included tensor product interaction `ti()` between PM2.5 and temperature

Key arguments:

- `ti()` is the tensor product interaction
- Multiple `s()` terms for each predictor

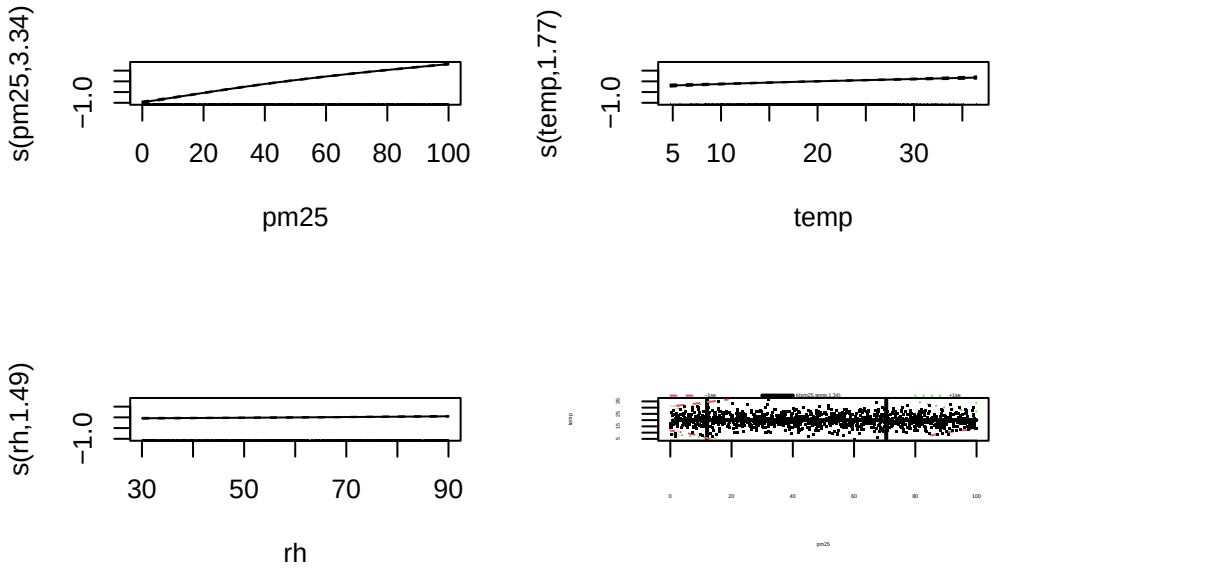
0.3.9 Results

```r
# Model summary
summary(model_complex)
```

```
Family: poisson
Link function: log

Formula:
```

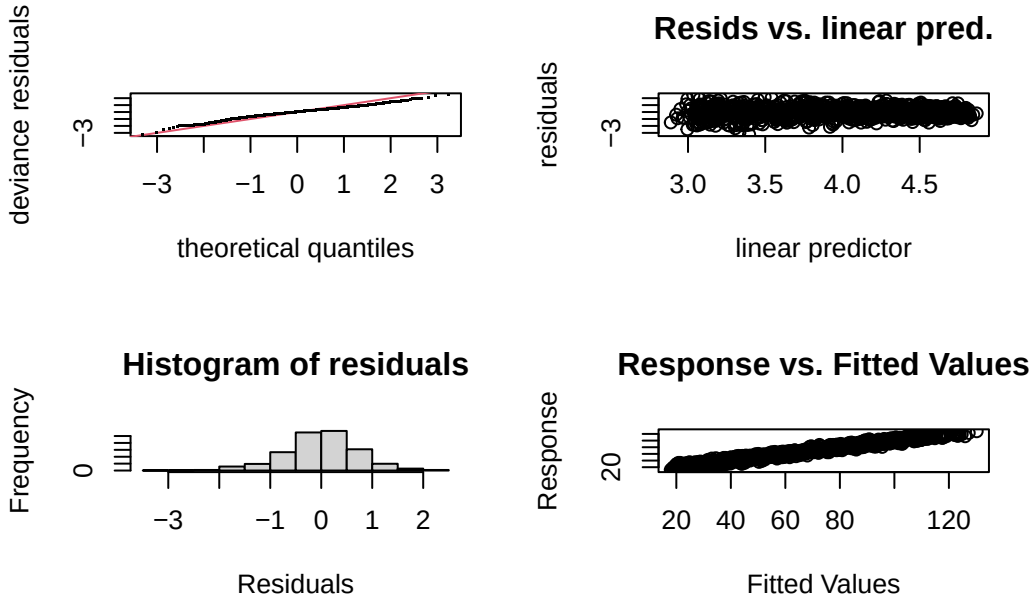
```
admissions ~ s(pm25) + s(temp) + s(rh) + ti(pm25, temp)

Parametric coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 3.990664   0.004603   866.9   <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(pm25)       3.343  4.149 12446.74  <2e-16 ***
s(temp)       1.768  2.249   154.46  <2e-16 ***
s(rh)         1.492  1.831    40.49  <2e-16 ***
ti(pm25,temp) 1.342  1.602     0.27   0.734
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.971   Deviance explained = 96.3%
UBRE = -0.44909  Scale est. = 1         n = 1000
```

```r
# Plot results for complex model
plot(model_complex, pages = 1)
```



0.3.10 Model checking

```
# For complex model
gam.check(model_complex)
```




```
Method: UBRE   Optimizer: outer newton
full convergence after 7 iterations.
Gradient range [-2.15315e-07,1.51255e-08]
(score -0.4490946 & scale 1).
Hessian positive definite, eigenvalue range [2.143967e-07,0.001694236].
Model rank =  44 / 44

Basis dimension (k) checking results. Low p-value (k-index<1) may
indicate that k is too low, especially if edf is close to k'.

                 k'   edf k-index p-value
s(pm25)        9.00  3.34    0.95    0.06 .
s(temp)        9.00  1.77    1.03    0.73
s(rh)          9.00  1.49    1.00    0.53
ti(pm25,temp) 16.00  1.34    1.02    0.82
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

`gam.check()` provides:

- Q-Q plot of residuals
- Residuals vs linear predictor
- Histogram of residuals
- Response vs fitted values