MTHM506 - Statistical Data Modelling

Topic 3 - Generalized Additive Models

Preliminaries

This practical session relates to Topic 3 – GAMs. The exercises refer to Topics 3.1-3.8 from the lecture notes as well as all the lecture sessions. The practical sessions are opportunities to ask questions on this material. If you do not finish these in the practical sessions, please complete these in your own time to further cement the course material and help prepare you for the coursework.

The file `datasets.RData` on the course ELE page contains the data that you will need for these exercises and can be loaded into R using the `load()` function. This practical will also use the `ggplot2` and `mgcv` packages which can be loaded using the `library()` function.

```
# Load required packages
library(mgcv)
library(ggplot2)

# Load data
load('datasets.RData')
```

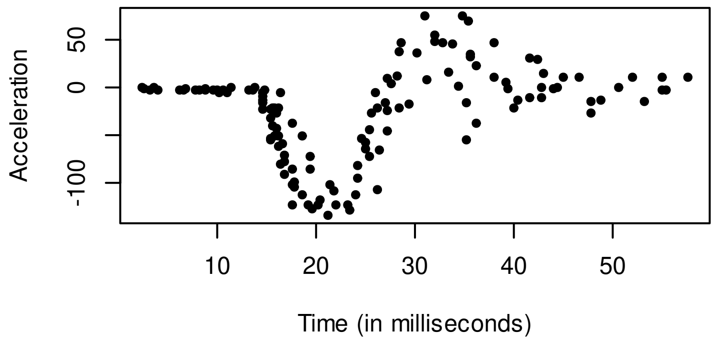
Exercise 1

The data set `mcycle` contains data on a series of measurements of head acceleration in a simulated motorcycle accident, used to test crash helmets.

(a) Plot the data and explain why a GAM make sense for this data set.

Solution: *The data take both positive and negative continuous values and is highly non-linear in time so a GAM makes sense here. The Normal distribution is sensible here, with an identity link with a function in time.*

```
# Plot the data
plot(mcycle$times,
  mcycle$accel,
  pch = 20,
  xlab = 'Time (in milliseconds)',
  ylab = 'Acceleration')
```



(b) Fit an appropriate GAM, ensuring it has adequate flexibility and that it fits the data well.

Solution: *The model is:*

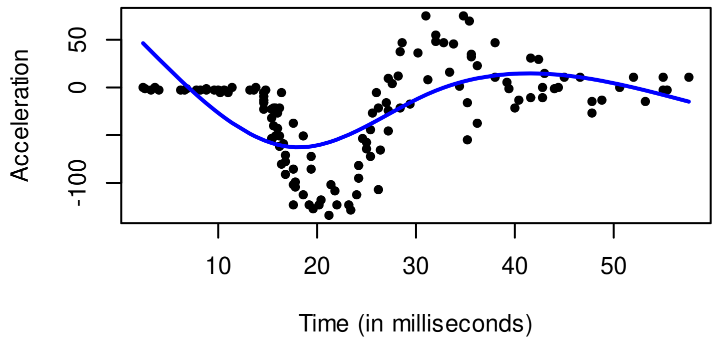
$$\begin{aligned} Y_t &\sim N(\mu_i, \sigma^2) \qquad Y_i \textit{ indep.} \\ \mu_i &= \beta_0 + f(t) \end{aligned}$$

where Y_t is the acceleration, t is the time step and $f(t)$ is a smooth function (defined by cubic splines). We fit this in R:

```
# GAM with 4 dof on the time step
Amodel <- gam(accel ~ s(times, k = 4, bs = "cs"), data = mcycle,
            family = gaussian)
# check the rank
k.check(Amodel)
         k'      edf   k-index p-value
s(times)  3 2.978009 0.4001372       0
```

The estimated degrees of freedom are close to 3, and k-index is less than 1 with a very p-value less than 0.05, so we should think about increasing the rank of our spline. This is also evident in the estimated line from this model, shown in blue below.

```
# Create a plot of the data
plot(mcycle$times,
  mcycle$accel,
  pch = 20,
  xlab = 'Time (in milliseconds)',
  ylab = 'Acceleration')
lines(mcycle$times, Amodel$fitted.values, col = 'blue', lwd = 2)
```



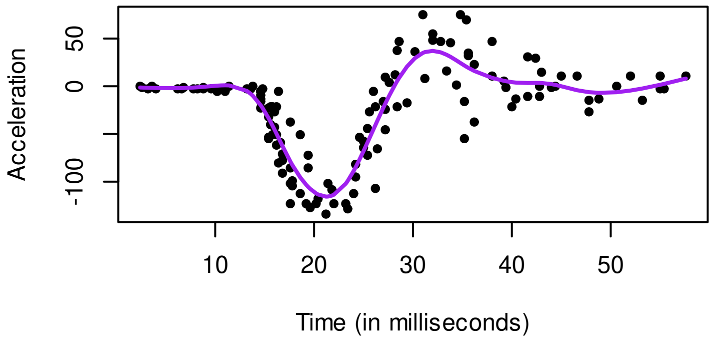
So let's increase the rank to 10 and see:

~~~
``` r
Amodel2 <- gam(accel ~ s(times, k = 10, bs = "cs"),
               data = mcycle, family = gaussian)
k.check(Amodel2)
         k'      edf  k-index p-value
s(times)  9 8.389483 1.148442    0.96
# Plot
plot(mcycle$times,
     mcycle$accel,
     pch = 20,
     xlab = 'Time (in milliseconds)',
     ylab = 'Acceleration')
lines(mcycle$times, Amodel2$fitted.values, col = 'red', lwd = 2)
```
~~~

\begin{center}\includegraphics{practicalsol_files/figure-latex/unnamed-chunk-5-1} \end{c

Model outputs indicate we should probably increase it much further to ensure it has enough flexibility. Let's try a much bigger rank:

```r
Amodel3 <- gam(accel ~ s(times, k = 25, bs = "cs"),
               data = mcycle, family = gaussian)
k.check(Amodel3)
         k'      edf  k-index p-value
s(times) 24 10.95962 1.155549    0.96

# Plot
plot(mcycle$times,
   mcycle$accel,
   pch = 20,
   xlab = 'Time (in milliseconds)',
   ylab = 'Acceleration')
lines(mcycle$times, Amodel3$fitted.values, col = 'purple', lwd = 2)
```



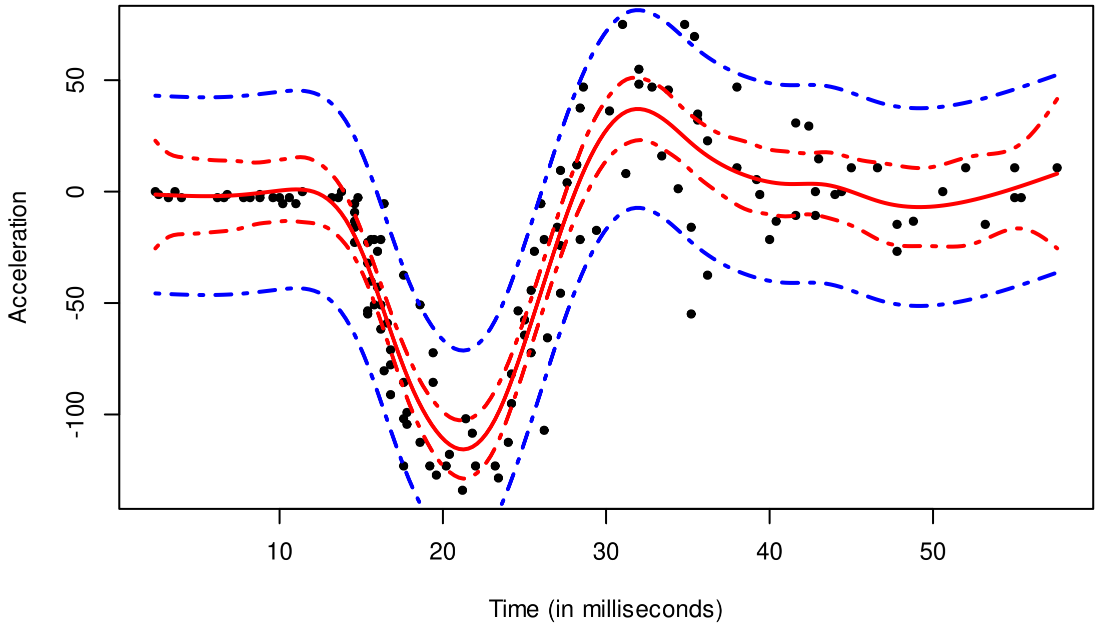
Now the edf is not too close to 24. Looking at the associated purple line however, indicates that we are probably starting to pick up the "noise" in the trend, so perhaps best to stop here.

(c) Produce a plot of the data and estimated mean (trend) together with 95% confidence and prediction intervals.

Solution: *We create a plot of (1) the predicted mean, (2) confidence intervals by using the predict function in R with the usual formula and (3) prediction intervals, by using the quantiles of the Normal distribution of our response, by using the estimate of σ^2 from the model:*

```
# Creating a plot of the data
plot(mcycle$times,
   mcycle$accel,
   pch = 20,
   xlab = 'Time (in milliseconds)',
   ylab = 'Acceleration')

# grid of time values
xx <- seq(min(mcycle$times), max(mcycle$times), length.out = 200)
preds <- predict(Amodel3, newdata = data.frame(times = xx),
                 type = 'response', se.fit = T)

# Predicted mean
lines(xx, preds$fit, col = "red", lwd = 2)

# 95% confidence intervals
lines(xx, preds$fit + 1.96 * preds$se.fit, col = "red", lwd = 2, lty = 4)
lines(xx, preds$fit - 1.96 * preds$se.fit, col = "red", lwd = 2, lty = 4)

# Get sigma
sig2 <- Amodel3$sig2
sig <- sqrt(sig2)

# Estimate prediction intervals
```

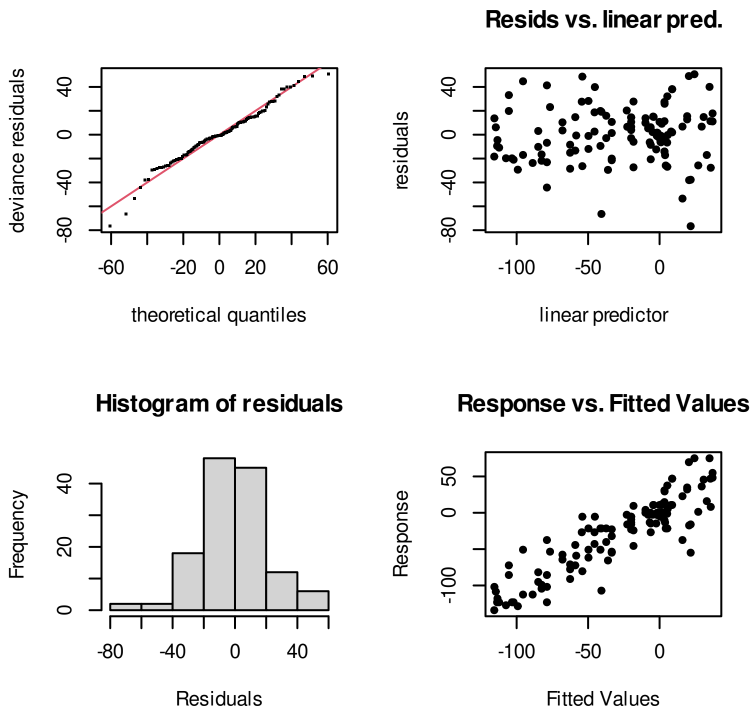
```r
upper <- qnorm(0.975,preds$fit,sig)
lower <- qnorm(0.025,preds$fit,sig)

# Add prediction intervals to the plot
lines(xx,lower,col="blue",lwd=2,lty=4)
lines(xx,upper,col="blue",lwd=2,lty=4)
```



(d) Produce the residual plots and comment on the overall fit of the model.

Solution: *Looking at the plot in part (c) it generally looks like a good fit, the mean trend follows the data and the prediction intervals capture the data well. However, the width of the prediction intervals might be a cause for concern (data has non-constant variance) so let's look at the residuals:*

```r
# Checking inputs from GAM
par(mfrow = c(2,2),pch=20)
gam.check(Amodel3)
```



```
Method: GCV   Optimizer: magic
Smoothing parameter selection converged after 5 iterations.
The RMS GCV score gradient at convergence was 0.003088512 .
The Hessian was positive definite.
Model rank =  25 / 25

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(times) 24  11    1.16    0.97
```

We can see that the QQ plot is generally good with only a few points straying away from the line, and that there is a nice random spread of the residuals.

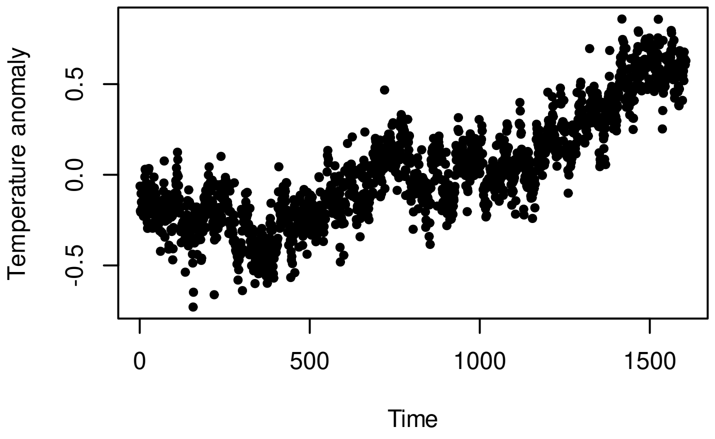
Exercise 2

The data set `globalMeanTemp` contains global mean temperature monthly “anomalies” for 1880–2013. The term “anomaly” means an overall mean value was substracted from the original data points. Interest lies in quantifying the temporal trend (change in the mean over time) in the data.

(a) Plot the data and explain why a GAM make sense for this data set.

Solution:*The data are anomalies so by definition take both positive and negative continuous values so the Normal distribution assumed by the additive model is sensible. In addition the* *trend is increasing but it a non-linear way so a smooth function in the mean is a sensible way of capturing this. In addition, the data are monthly so the additive model would enable us to add seasonal structure in the mean, either by treating month as a factor, or as another smooth function.*

```r
# Plot the data
plot(globalMeanTemp$timeStep, globalMeanTemp$temp, pch = 20,
  xlab = 'Time', ylab = 'Temperature anomaly')
```



(b) Fit an appropriate GAM ensuring this has adequate flexibility and that it fits the data well.

Solution: *The model we are fitting is:*

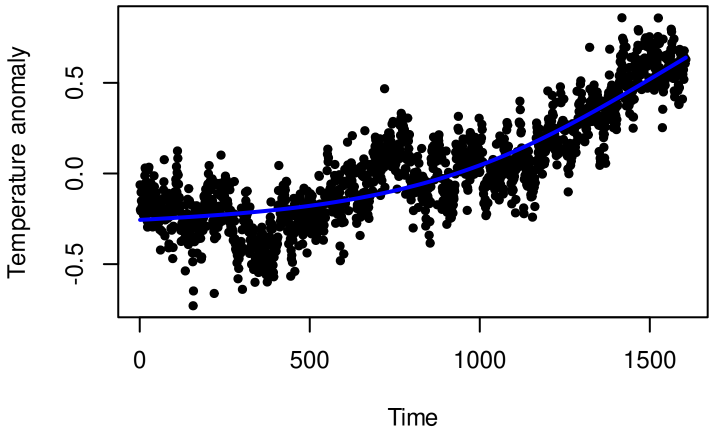
$$\begin{aligned} Y_t &\sim N(\mu_i, \sigma^2) \qquad Y_t \textit{ indep.} \\ \mu_t &= \beta_0 + f(x_t) \end{aligned}$$

where Y_t is the global mean temperature anomaly, t is the monthly time step and $f(t)$ is a smooth function (defined by cubic splines). We fit this in R:

```r
# GAM with 4 dof on the time step
Amodel <- gam(temp ~ s(timeStep, k = 4, bs = "cs"),
  data = globalMeanTemp, family = gaussian)

# Check rank
k.check(Amodel)
            k'      edf   k-index p-value
s(timeStep)  3 2.923338 0.2588401       0
```

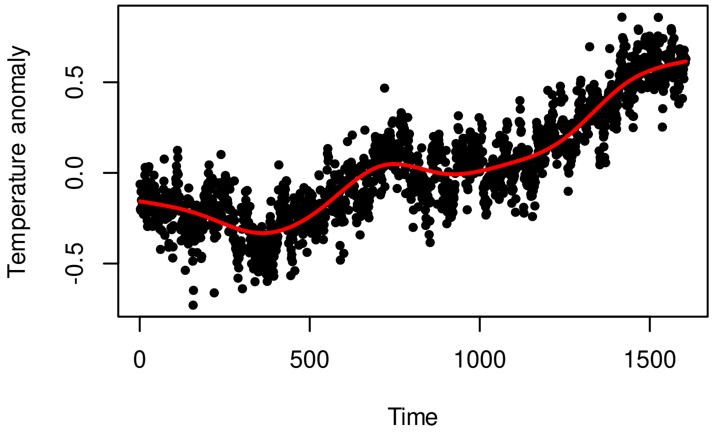
The EDF is close to 3, and `k-index` is less than 1 with a very low p-value so we should think about increasing the rank of our spline. This is also evident in the estimated line from this model, shown in blue below.

```r
# Plot
plot(globalMeanTemp$timeStep, globalMeanTemp$temp, pch = 20,
 xlab = 'Time', ylab = 'Temperature anomaly')
lines(globalMeanTemp$timeStep, Amodel$fitted.values, col = 'blue', lwd = 2)
```



So let's increase the rank to 9 and see:

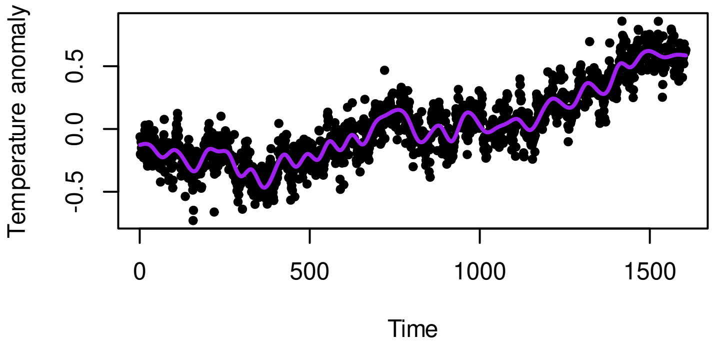
```
Amodel2 <- gam(temp ~ s(timeStep, k = 10, bs = "cs"),
  data = globalMeanTemp, family = gaussian)
# Checking inputs from GAM
k.check(Amodel2)
            k'      edf   k-index p-value
s(timeStep)  9 8.780688 0.3526832       0
plot(globalMeanTemp$timeStep, globalMeanTemp$temp, pch = 20,
 xlab = 'Time', ylab = 'Temperature anomaly')
lines(globalMeanTemp$timeStep, Amodel2$fitted.values, col = 'red', lwd = 2)
```



which indicates we need to increase more (red line on plot confirms this). Let's try a much bigger rank:

```
Amodel3 <- gam(temp ~ s(timeStep, k = 50, bs = "cs"),
  data = globalMeanTemp, family = gaussian)
k.check(Amodel2)
            k'      edf   k-index p-value
```

```
s(timeStep)  9 8.780688 0.3526832       0

# Creating a plot of the data
plot(globalMeanTemp$timeStep, globalMeanTemp$temp, pch = 20,
 xlab = 'Time', ylab = 'Temperature anomaly')
lines(globalMeanTemp$timeStep, Amodel3$fitted.values, col = 'purple', lwd = 2)
```



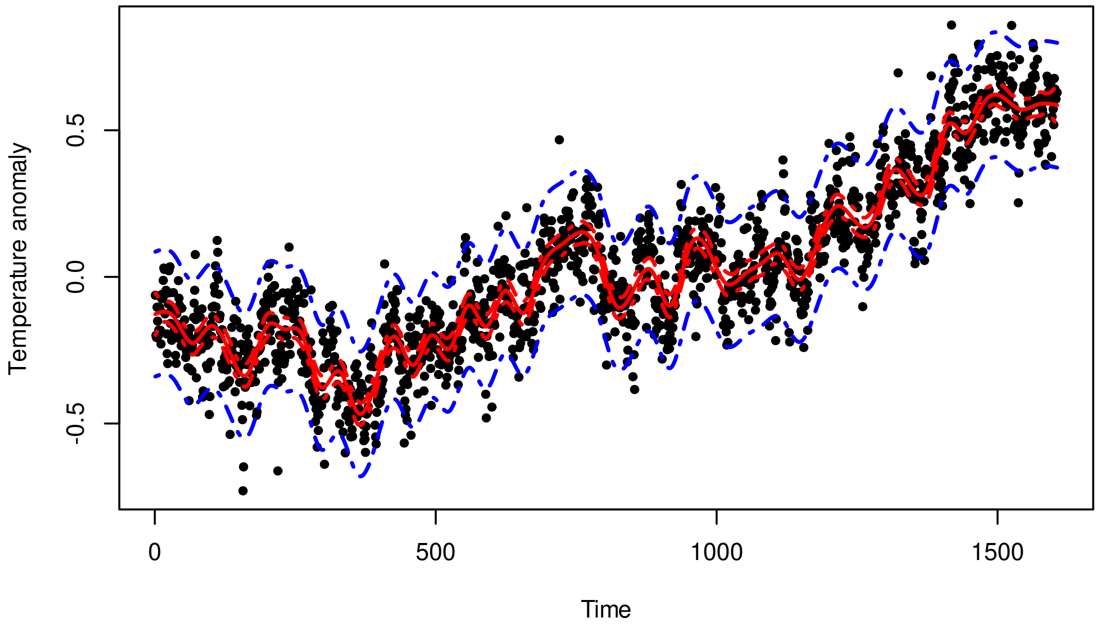
Now the edf is not too close to 49, although k-index and the p-value may indicate we need even bigger rank. Looking at the associated purple line however, indicates that we are probably starting to pick up the "noise" in the trend, so perhaps best to stop here.

(c) Produce a plot of the data and the estimated mean (trend) along with 95% confidence as well as prediction intervals.

Solution: *We create a plot of (1) the predicted mean, (2) confidence intervals by using the predict function in* R *with the usual formula and (3) prediction intervals, by using the quantiles of the Normal distribution of our response, by using the estimate of σ^2 from the model:*

```
# Plot the data
plot(globalMeanTemp$timeStep, globalMeanTemp$temp, pch = 20,
 xlab = 'Time', ylab = 'Temperature anomaly')

# grid of monthly time step
xx <- globalMeanTemp$timeStep
preds <- predict(Amodel3,newdata=data.frame(timeStep=xx),se.fit=T)

# Predicted mean
lines(xx,preds$fit,col="red",lwd=2)

# 95% confidence intervals
lines(xx,preds$fit+1.96*preds$se.fit,col="red",lwd=2,lty=4)
lines(xx,preds$fit-1.96*preds$se.fit,col="red",lwd=2,lty=4)

# Get sigma
sig2 <- Amodel3$sig2
```

```
sig <- sqrt(sig2)

# Estimating prediction intervals
upper <- qnorm(0.975,preds$fit,sig)
lower <- qnorm(0.025,preds$fit,sig)

# Adding prediction intervals to the plot
lines(xx,lower,col="blue",lwd=2,lty=4)
lines(xx,upper,col="blue",lwd=2,lty=4)
```



(d) Investigate (using GAMs) whether there is a significant seasonal cycle in the data.

Solution:*We can add a seasonal cycle in the model as a smooth function defined as a cubic spline, with the added restriction that the end-points join (i.e. that we smoothly move from December to January). The model is:*

$$\begin{aligned} Y_t &\sim N(\mu_t, \sigma^2) \qquad Y_t \textit{ indep.} \\ \mu_t &= \beta_0 + f_1(t) + f_2(DoY(t)) \end{aligned}$$

where $DoY(t)$ is the day-of-year that month t corresponds to. $f_2(\cdot)$ is a cyclic smooth function defined using cubic splines. In R:

```
# Model with a seasonal trend
Amodel4 <- gam(temp ~ s(timeStep, k = 50, bs = "cs") +
  s(month, bs = "cc", k = 5),
  data = globalMeanTemp, family = gaussian,
  knots=list(month=c(0,366)))
```

```
# Summarize model
summary(Amodel4)

Family: gaussian
Link function: identity

Formula:
temp ~ s(timeStep, k = 50, bs = "cs") + s(month, bs = "cc", k = 5)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.025102   0.002709   9.266   <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(timeStep) 47.006331     49 218.2  <2e-16 ***
s(month)     0.000424      3   0.0   0.342
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.869   Deviance explained = 87.3%
GCV = 0.012149  Scale est. = 0.011786  n = 1606

# Check ranks
k.check(Amodel4)
            k'          edf   k-index p-value
s(timeStep) 49 4.700633e+01 0.4755283       0
s(month)     3 4.240067e-04 1.1013811       1
```

The high p-value of the F-test for the cyclic spline indicates that the seasonal cycle is not significant – as expected. A plot of the estimated line confirms this (not shown).

Exercise 3

The dataframe `munichrent` contains data on net rent per month in Euros (`rent`) for more 3,000 randomly sampled similar sized apartments in Munich, Germany. Other variables in the data set are the year of construction (`yearc`), the quality of the location according to an expert assessment (a three level factor `location`: average location (1), good location (2), top location (3)) and the variable `area`, quantifying the size of each property in m^2. Interest lies in the relationship (if any) between net rent, year of construction, location and area.

(a) Fit a Normal GLM *with a log-link* to model the relationship between net rent and year of construction, location assessment and the interaction between them. Produce a density plot of the distribution of the residuals and a Q-Q plot of the residuals and also plot the residuals against the linear predictor. On the basis of these plots explain with reasons why the model may not be suitable for this data set.

Solution: *The model we are fitting is:*

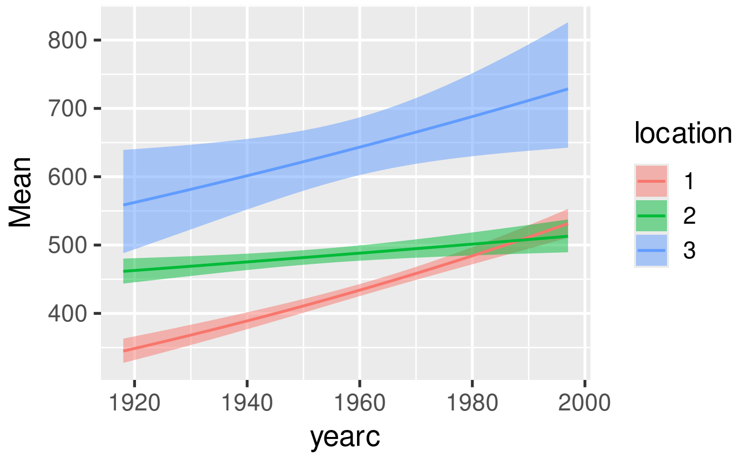
$$\begin{aligned} Y_i &\sim N(\mu_i, \sigma^2) \qquad Y_i \text{ indep.} \\ \log(\mu_i) &= \beta_0 + \beta_1 x_{1_i} + \beta_2 x2_i + \beta_3 x_{3_i} + \beta_4 x_{1_i} x_{2_i} + \beta_5 x_{1_i} x_{3,i} \end{aligned}$$

where Y_i is the rent, x_1 is year, $x_2 = 1$ if location is 2 and 0 otherwise and $x_3 = 1$ if location is 3 and 0 otherwise.

```
# Fit model
model1 <- glm(rent ~ yearc + location + yearc:location,
              data = munichrent, family = gaussian(link="log"))

# Summary
summary(model1)

Call:
glm(formula = rent ~ yearc + location + yearc:location, family = gaussian(link = "l
    data = munichrent)

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)     -4.6599343  1.0347314  -4.504 6.93e-06 ***
yearc            0.0054760  0.0005268  10.394  < 2e-16 ***
location2        8.2340360  1.3957775   5.899 4.05e-09 ***
location3        4.5359025  3.0300550   1.497    0.135
yearc:location2 -0.0041410  0.0007125  -5.812 6.79e-09 ***
yearc:location3 -0.0021134  0.0015459  -1.367    0.172
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 35657.36)

    Null deviance: 117945363  on 3081  degrees of freedom
Residual deviance: 109680448  on 3076  degrees of freedom
AIC: 41059

Number of Fisher Scoring iterations: 6
```

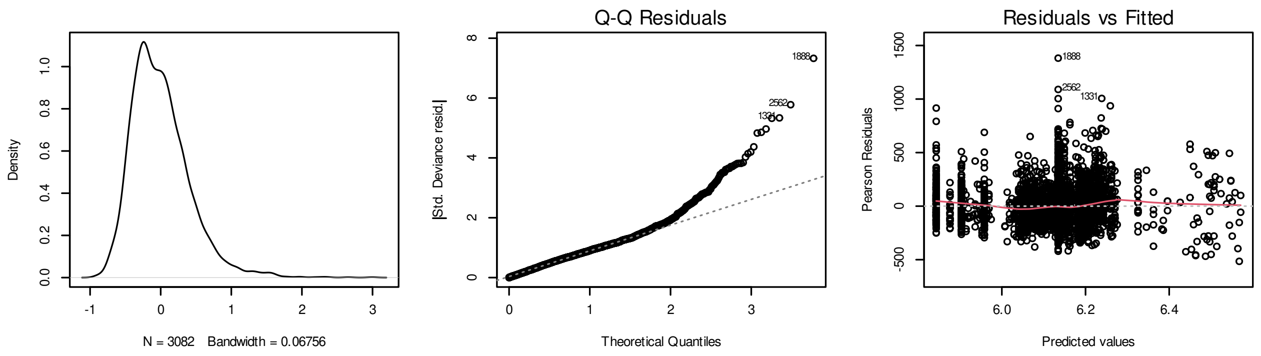
Hard to interpret the output so let's produce estimates of the mean for different combinations of `yearc` and `location` to understand the inference.

```r
# Grid of covariate values
years <- 1918:1997
plotDat <- expand.grid(years,1:3)
plotDat <- data.frame(plotDat)
names(plotDat) <- c("yearc","location")
plotDat$location <- factor(plotDat$location)

# predict mean and standard errors
preds <- predict(model1,newdata=plotDat,se.fit=T,type="link")
# put in the dataframe
plotDat$Mean <- exp( preds$fit )
plotDat$Upper <- exp( preds$fit + 1.96*preds$se.fit )
plotDat$Lower <- exp( preds$fit - 1.96*preds$se.fit )

# and plot
library(ggplot2)
myPlot <- ggplot(data=plotDat) +
geom_ribbon(aes(x=yearc,ymin=Lower,ymax=Upper,fill=location),alpha=0.5) +
geom_line(aes(x=yearc,y=Mean,col=location))
myPlot
```



Rent prices increase with location and the relationship with `yearc` is slightly non-linear. The interaction is also evident as the lines are not parallel.

```r
par(mfrow = c(1,3))
plot(density(model1$residuals), main = '')
plot(model1, 2)
plot(model1, 1)
```



The residual plots below indicate that the residuals are non-Normally distributed, in particular they are positively skewed. As such we may want to consider a model based on the Gamma distribution

---

(b) Suggest a GAM for this data set using the same predictors and choosing a suitable distribution from the exponential family with an appropriate link function. Justify your choices. Fit your proposed model, report and interpret the results and carry out appropriate model checking.

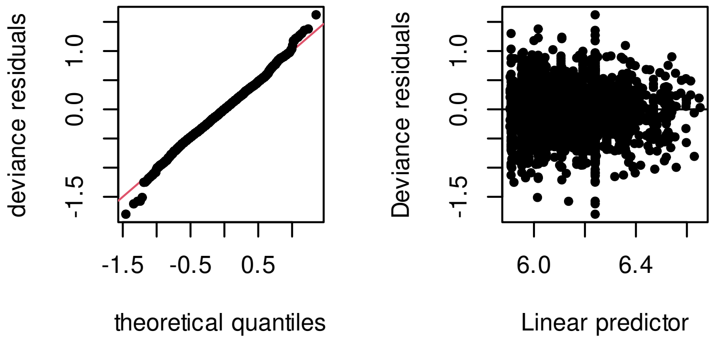
Solution: *We note that rent takes non-negative values and that the residuals from the linear model in (a) are positively skewed, so a sensible option is to go for a Gamma GAM, which can also better capture non-linear relationships with `yearc`. We will use a log-link to ensure the mean is in the correct range:*

$$\begin{aligned} Y_i &\sim \text{Gamma}(\mu_i, \phi) \qquad Y_i \text{ indep.} \\ \log(\mu_i) &= \beta_0 + \beta_2 z_{2,i} + \beta_3 z_{3,i} + z_{1,i} f_1(x_i) + z_{2,i} f_2(x_i) + z_{3,i} f_3(x_i) \end{aligned}$$

where z_1, z_2 and z_3 are dummy variables for `location`. We fit the model via:

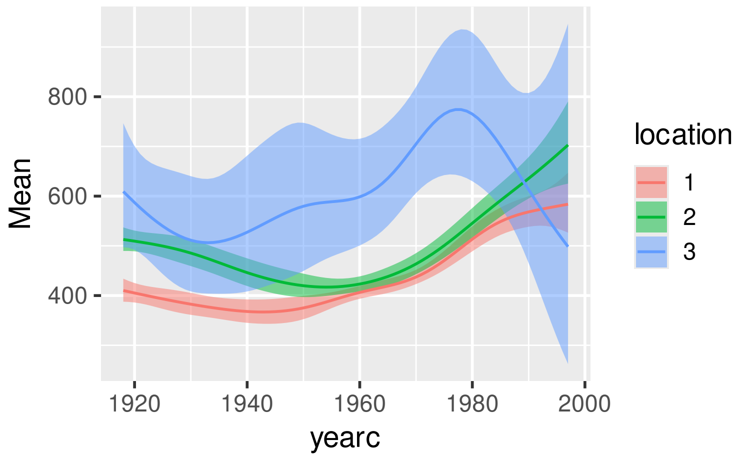
```
model2 <- gam(rent ~ location + s(yearc, by = location, k = 10),
    data = munichrent, family = Gamma(link="log"))

# check rank
k.check(model2)
                        k'      edf   k-index p-value
s(yearc):location1  9 5.452146 0.9507918  0.0375
s(yearc):location2  9 4.003695 0.9507918  0.0200
s(yearc):location3  9 4.354029 0.9507918  0.0300

## Check residuals
par(mfrow=c(1,2))
# QQ plot
qq.gam(model2,pch=20)
# deviance residuals vs linear predictor
xx <- model2$linear.predictors
yy <- residuals(model2,type="deviance")
```

```
plot(xx,yy,pch=20,xlab="Linear predictor",ylab="Deviance residuals")
abline(h=0)
```



The 3 splines have adequate ranks and the residual plots look fairly OK. Slight funnelling on the right plot, and a bit of deviation for the very high and low extremes on the QQ plot, but not very much. Let's look the estimated curves:

```
# predict mean and standard errors
preds <- predict(model2,newdata=plotDat,se.fit=T,type="link")
# put in the dataframe
plotDat$Mean <- exp( preds$fit )
plotDat$Upper <- exp( preds$fit + 1.96*preds$se.fit )
plotDat$Lower <- exp( preds$fit - 1.96*preds$se.fit )

# and plot
myPlot <- ggplot(data=plotDat) +
geom_ribbon(aes(x=yearc,ymin=Lower,ymax=Upper,fill=location),alpha=0.5) +
geom_line(aes(x=yearc,y=Mean,col=location))
myPlot
```



The relationship with `yearc` is definitely non-linear, with a dip and then a rise for all 3 locations. Location 3 carries a lot of uncertainty though, so the "plunge" in rent prices at the very end might be an artifact (judging from the width of the CIs). Location 3 has consistently higher rents than location 2 which in turn has higher rents compared to location 1.

(c) Fit another GAM to assess the interaction between `area` and `yearc`. Write your model down mathematically, justify the choice of distribution and link, check whether it fits and discuss the results.

Solution: *We keep the Gamma distribution with a log-link and formulate the model:*

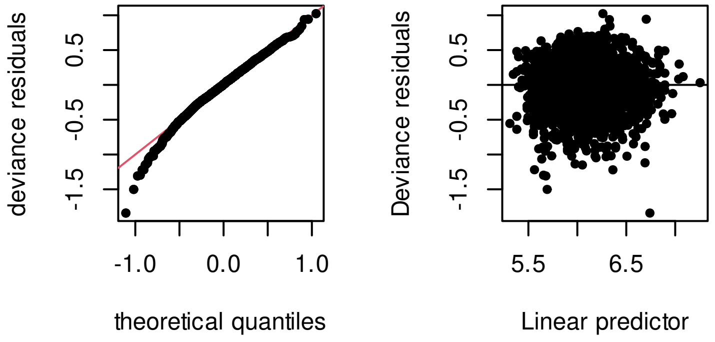
$$\log(\mu_i) = \beta_0 + f(x_{1,i}, x_{2,i})$$

where $f(x_1, x_2)$ is a tensor product, of two cubic splines. We fit this model via:

```
model3 <- gam(rent ~ te(yearc,area,k=c(10,10),bs=c("cs","cs")),
              data = munichrent,
              family = Gamma(link="log"))

# Model summary
summary(model3)

Family: Gamma
Link function: log

Formula:
rent ~ te(yearc, area, k = c(10, 10), bs = c("cs", "cs"))

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 6.090745   0.005385    1131   <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
te(yearc,area) 22.54     98 26.94  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.461   Deviance explained = 44.8%
GCV = 0.098349  Scale est. = 0.089385   n = 3082

# check rank
k.check(model3)
               k'      edf   k-index p-value
te(yearc,area) 99 22.54458 0.9756435     0.2
```

There are enough degrees of freedom, so let's look at the residual plots:

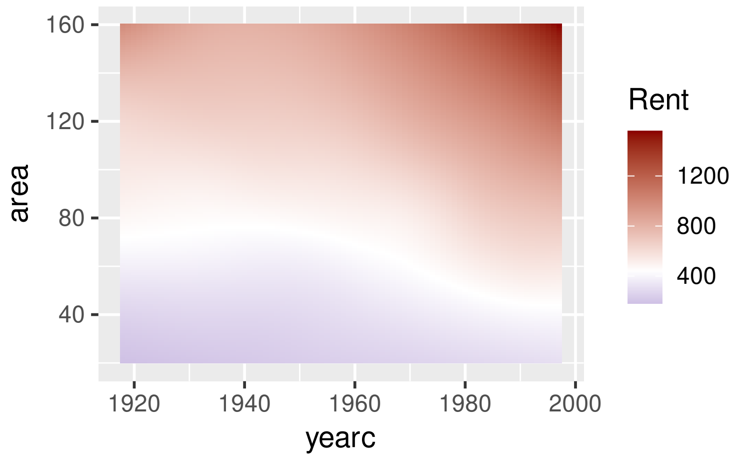
```r
## Check residuals
par(mfrow=c(1,2))
# QQ plot
qq.gam(model3,pch=20)
# deviance residuals vs linear predictor
xx <- model3$linear.predictors
yy <- residuals(model3,type="deviance")
plot(xx,yy,pch=20,xlab="Linear predictor",ylab="Deviance residuals")
abline(h=0)
```



The QQ plot indicates issues at the lower extremes while the residuals vs linear predictor shows slight evidence of funnelling. We could try adding more covariates to fix some of these problems. Alternatively we with `mgcv` we can fit a Gamma GAM where the dispersion parameters is also a function of the covariates using the `gammals` family. This is however beyond the scope of the module. Let's try to understand the estimates using plots.

```r
# create a grid area values
range(munichrent$area)
[1]  20 160
areas <- 20:160
# grid of years
range(munichrent$yearc)
[1] 1918 1997
years <- 1918:1997
# grid of area and years
plotDat <- expand.grid(years,areas)
plotDat <- data.frame(plotDat)
names(plotDat) <- c("yearc","area")
head(plotDat)
  yearc area
1  1918   20
2  1919   20
3  1920   20
4  1921   20
5  1922   20
6  1923   20
```

```r
# predict mean and standard errors
preds <- predict(model3,newdata=plotDat,se.fit=T,type="link")

# put in the dataframe
plotDat$Mean <- exp(preds$fit)
plotDat$Upper <- exp( preds$fit + 1.96*preds$se.fit )
plotDat$Lower <- exp( preds$fit - 1.96*preds$se.fit )

# Plot as a 2D raster plot
myPlot <- ggplot(data=plotDat) +
geom_raster(aes(x=yearc,y=area,fill=Mean)) +
scale_fill_gradient2(midpoint=exp(model3$coefficients[1]),high="darkred",low="darkblue"
myPlot
```



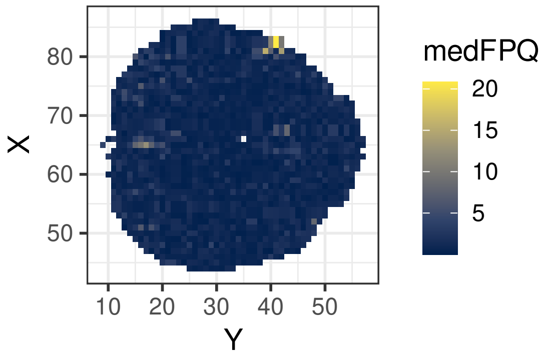
Rent is highest for bigger properties as expected, and its peak is for the newest build. There is also a smaller peak for very old build properties.

Exercise 4

The dataframe `brain` contains data from a brain image obtained from functional magnetic resonance scanning (see Landau et. al (2004) for more details). The columns of interest are: `X` and `Y` refer to a voxel location and `medFPQ` is a brain activity level measurement (median Fundamental Power Quotient over three measurements); This exercise will consider models for `medFPQ` as a function of `X` and `Y`.

We can plot the brain scan image using the following:

```r
ggplot(brain,
  aes(y = X, x = Y, fill = medFPQ)) +
  geom_tile() + scale_fill_viridis_c(option = "cividis") +
  theme_bw()
```

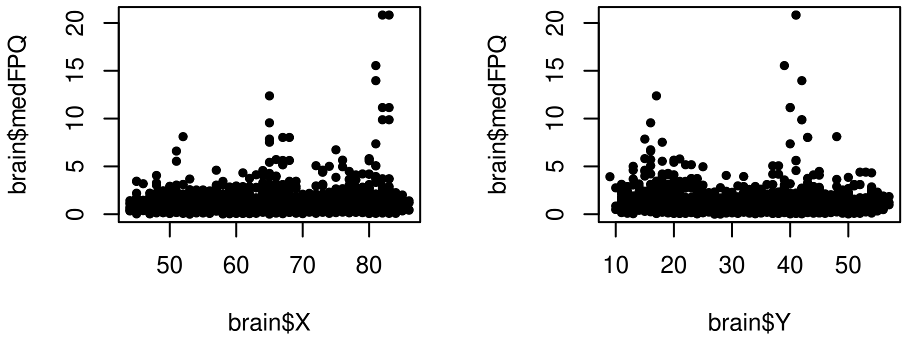


The data is quite noisy and interest lies in understanding the actual level of brain activity “under all that noise”. Such situations can be tackled by smoothing the data to estimate the **underlying trend**, which is something we can do with GAMs, by characterising the mean (μ_i) of the data as a smooth function of the covariates.

(a) Suggest and fit an appropriate GAM to this data, with `X` and `Y` as interacting covariates. You can think of those as coordinates. Make sure the model has enough flexibility.

Solution: *Let’s plot the response `medFPQ` against each coordinate:*

```
par(mfrow=c(1,2),pch=20)
plot(brain$X,brain$medFPQ)
plot(brain$Y,brain$medFPQ)
```



The response is non-negative, skewed and we have values close to zero, so best to use a model based on the Gamma distribution (rather than Normal). Let’s fit a Gamma GAM with a log-link and a tensor product 2D smooth function of `X` and `Y`:

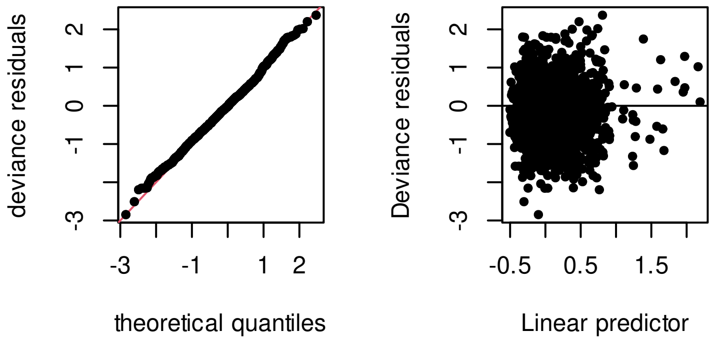
```
Gmodel <- gam(medFPQ ~ te(X,Y,k=c(10,10),bs=c("cs","cs")),data=brain,
 family=Gamma(link="log"))

# check rank
k.check(Gmodel)
        k'      edf   k-index p-value
te(X,Y) 99 58.31613 0.8560737       0
```

Looks like we have enough flexibility in the spline as the EDF is much lower than `k'`.

(b) Plot the residuals and comment on the overall fit.

Solution: *Let's look at the residuals:*

```r
## Check residuals
par(mfrow=c(1,2))
# QQ plot
qq.gam(Gmodel,pch=20)
# deviance residuals vs linear predictor
xx <- Gmodel$linear.predictors
yy <- residuals(Gmodel,type="deviance")
plot(xx,yy,pch=20,xlab="Linear predictor",ylab="Deviance residuals")
abline(h=0)
```

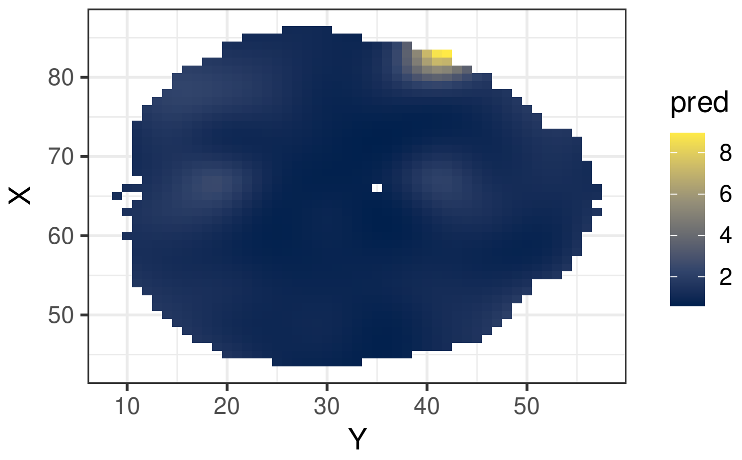


Both plots look OK so model fits well. Let's proceed to produce predictions.

(c) Use your model to predict the mean brain activity (this is the noise-free estimate of the brain activity) for all obsrved values of `X` and `Y`. Plot this and comment on what you see.

Solution: *Predict and plot:*

```r
# Predict mean brain activity
brain$pred <- predict(Gmodel, newdata = brain, type = 'response')

# Plot the predicted brain activity
ggplot(brain, aes(y = X, x = Y, fill = pred)) +
  geom_tile() + scale_fill_viridis_c(option = "cividis") + theme_bw()
```



We can see that there is high brain activity on the left hand side of the brain here with a couple of other areas in the centre spiking too. This is where we will stop, but for further details about this example, please see Generalized Additive Models: An Introduction in R by Simon Wood Chapter 7.2 for more details. Chapter 7 might also prove useful for coursework purposes.
