COVID-19 Deaths Analysis

Zheyuan Fan

The purpose of this analysis is to analyze the relationship between COVID-19 death and time, in different coutries or regions.

```
install.packages("gamm4")
```

```
library(devtools)
library(mgcv)
library(gamm4)
library(tidyverse)
```

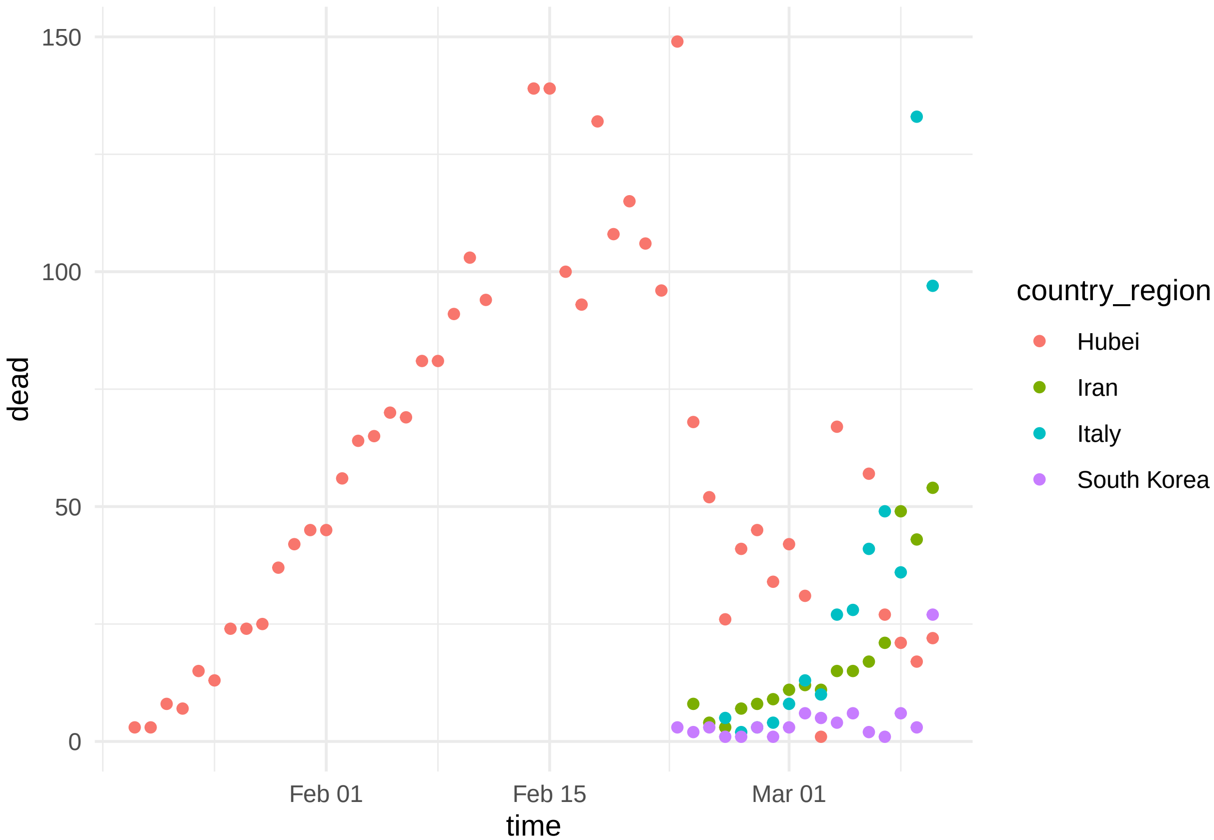
COVID-19 data

First, plot deaths from COVID-19, so we can visualize the deaths in five different regions.

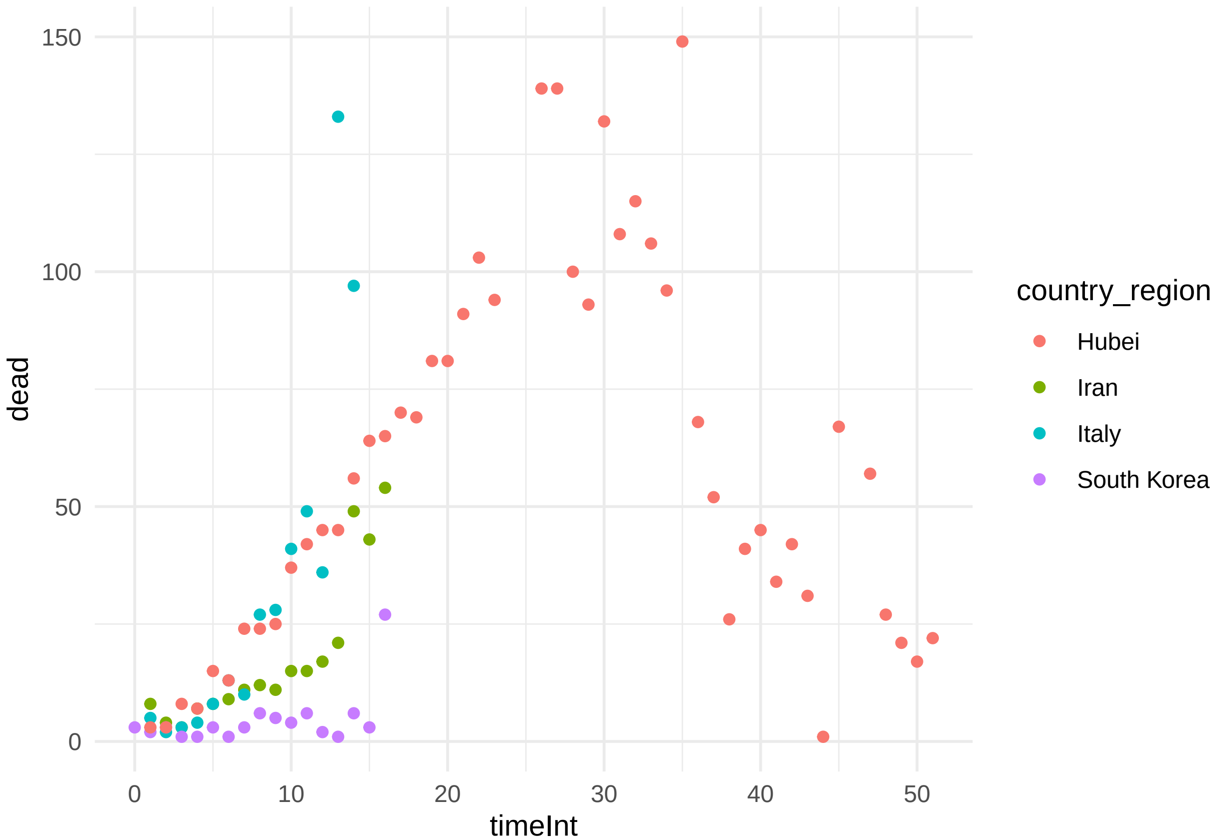
```
# Load nCOVID-19 data
covid_data <- read_csv("covid_data.csv")
```

```
## Parsed with column specification:
## cols(
##   time = col_date(format = ""),
##   timeInt = col_double(),
##   cum_confirm = col_double(),
##   cum_dead = col_double(),
##   incidence = col_double(),
##   dead = col_double(),
##   country_region = col_character()
## )
```

```
# Plot over time
covid_data %>%
  filter(country_region %in% c('Hubei','Italy','Iran','South Korea','USA')) %>%
  na.omit() %>%
  ggplot(aes(time, dead, color=country_region)) +
  geom_point() +
  theme_minimal()
```




```r
# Plot from initial death in region
covid_data %>%
  filter(country_region %in% c('Hubei','Italy','Iran','South Korea','USA')) %>%
  na.omit() %>%
  ggplot(aes(timeInt, dead, color=country_region)) +
  geom_point() +
  theme_minimal()
```



Now fit it a GAM resGam with `dead` as the response a smooth on `timeInt` and `country_region` as covariates.

```
resGam= mgcv::gam(
  dead ~ s(timeInt, pc=0) + country_region,
  data=covid_data,
  family=poisson(link='log'))
```

Now we summarize and get the conclusion of the model, then plot it.

```
summary(resGam)
```

```
##
## Family: poisson
## Link function: log
##
## Formula:
## dead ~ s(timeInt, pc = 0) + country_region
##
## Parametric coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                   -0.160352   0.583136  -0.275 0.783329
## country_regionAustralia        0.078106   1.155196   0.068 0.946094
## country_regionBeijing         -1.940556   0.739512  -2.624 0.008688 **
## country_regionChongqing       -0.535153   0.819679  -0.653 0.513833
## country_regionFrance           1.127419   0.610845   1.846 0.064940 .
## country_regionGuangdong       -1.608135   0.771882  -2.083 0.037215 *
```

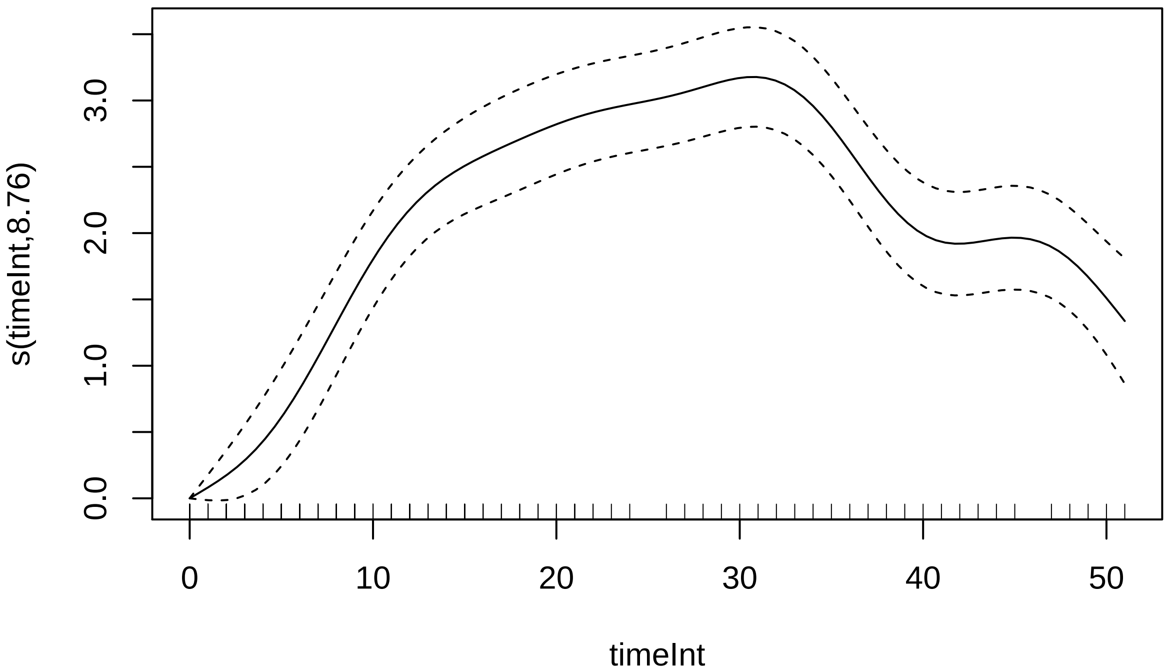
```
## country_regionHainan         -2.168937   0.824279  -2.631 0.008506 **
## country_regionHebei          -0.763389   0.823787  -0.927 0.354092
## country_regionHeilongjiang   -1.118993   0.666038  -1.680 0.092943 .
## country_regionHenan          -1.208796   0.631050  -1.916 0.055425 .
## country_regionHubei           1.815819   0.589066   3.083 0.002052 **
## country_regionHunan           0.078106   1.155196   0.068 0.946094
## country_regionIran            1.321243   0.590201   2.239 0.025180 *
## country_regionIraq            0.171690   0.764797   0.224 0.822375
## country_regionItaly           2.117238   0.588802   3.596 0.000323 ***
## country_regionJapan          -1.361864   0.654921  -2.079 0.037578 *
## country_regionShandong        0.215099   0.817422   0.263 0.792440
## country_regionSouth Korea    -0.005497   0.597876  -0.009 0.992664
## country_regionSpain           2.033865   0.605583   3.359 0.000784 ***
## country_regionUnited Kingdom  1.258965   0.820598   1.534 0.124979
## country_regionUnited States   0.827315   0.621365   1.331 0.183042
## ---
## 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(timeInt) 8.758  8.982   1309  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.894   Deviance explained = 93.5%
## UBRE = 2.0019  Scale est. = 1         n = 170
```

```
coef(resGam)
```

```
##                  (Intercept)      country_regionAustralia
##                 -0.160352456                  0.078105515
##       country_regionBeijing      country_regionChongqing
##                 -1.940556292                 -0.535153159
##        country_regionFrance      country_regionGuangdong
##                  1.127419488                 -1.608135374
##        country_regionHainan          country_regionHebei
##                 -2.168937066                 -0.763389041
##   country_regionHeilongjiang          country_regionHenan
##                 -1.118993096                 -1.208796089
##         country_regionHubei          country_regionHunan
##                  1.815818734                  0.078105515
##          country_regionIran           country_regionIraq
##                  1.321243223                  0.171690309
##         country_regionItaly          country_regionJapan
##                  2.117237701                 -1.361864231
##      country_regionShandong    country_regionSouth Korea
##                  0.215099168                 -0.005496802
##         country_regionSpain country_regionUnited Kingdom
##                  2.033864959                  1.258964745
##  country_regionUnited States                 s(timeInt).1
##                  0.827314895                  0.436070190
##                 s(timeInt).2                 s(timeInt).3
##                  0.162668721                  0.695995274
##                 s(timeInt).4                 s(timeInt).5
```

```
##                 -0.257405570                 0.133254518
##                 s(timeInt).6                s(timeInt).7
##                  1.140898783                -0.022139449
##                 s(timeInt).8                s(timeInt).9
##                  4.992873514                -1.020359041
```

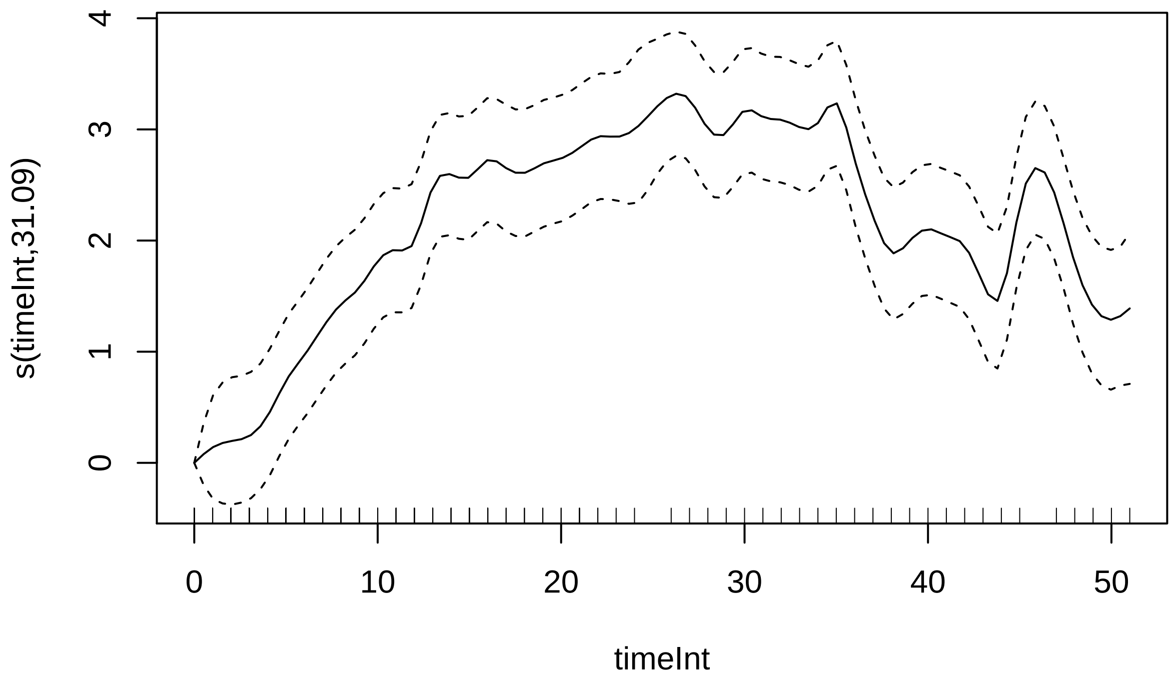
```r
plot(resGam)
```



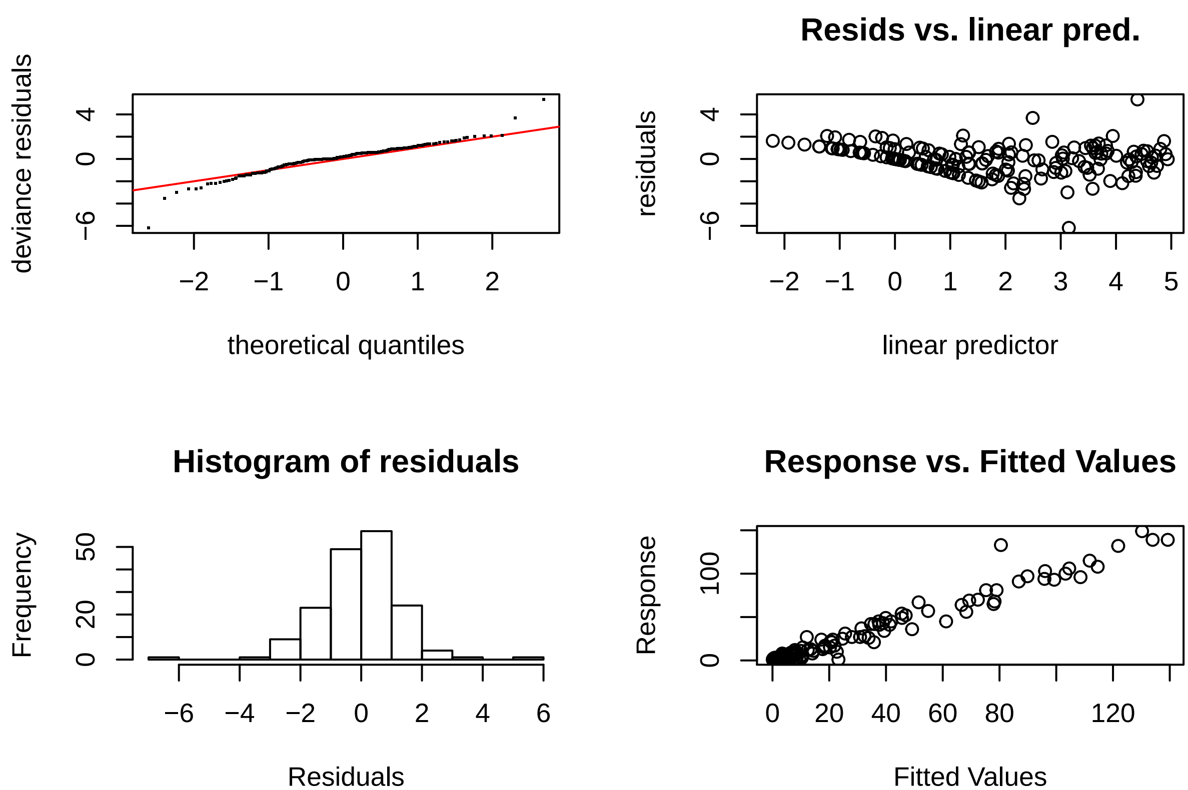
##
Conclusion from above The estimated degrees of freedom for the smooth of `timeInt` is 8.758, we see an edf is much higher than 1(which is 8.758), which means the relationship between deaths and time is not close to linear. We can interpret the coefficients for `country_region`. For example, country_regionAustralia has a coefficient of 0.078, means time has a positive relationship with deaths due to COVID-19 in Australia, one unit of time will cause 0.078 more deaths in Australia.

Next, we fit and plot two more GAMs with the same model but with `k = 50` and `k = 20`.

```r
resGam3= mgcv::gam(
  dead ~ s(timeInt, k=50, pc=0) + country_region, data=covid_data,
  family=poisson(link='log'), method='ML')
plot(resGam3)
```



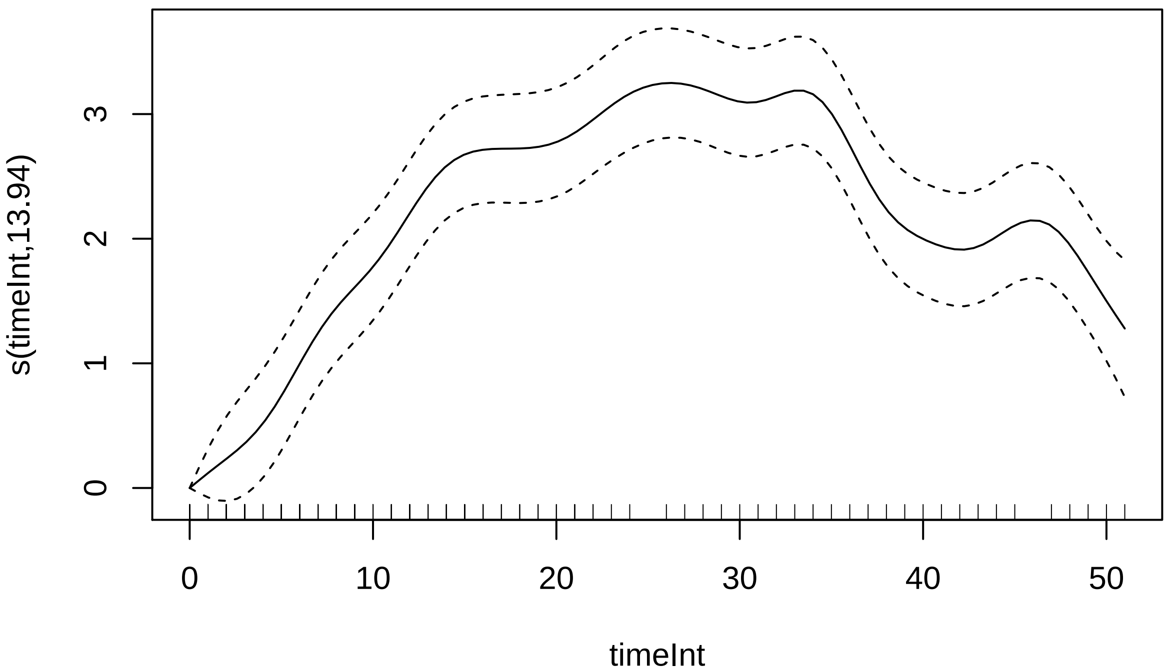

```r
gam.check(resGam3)
```



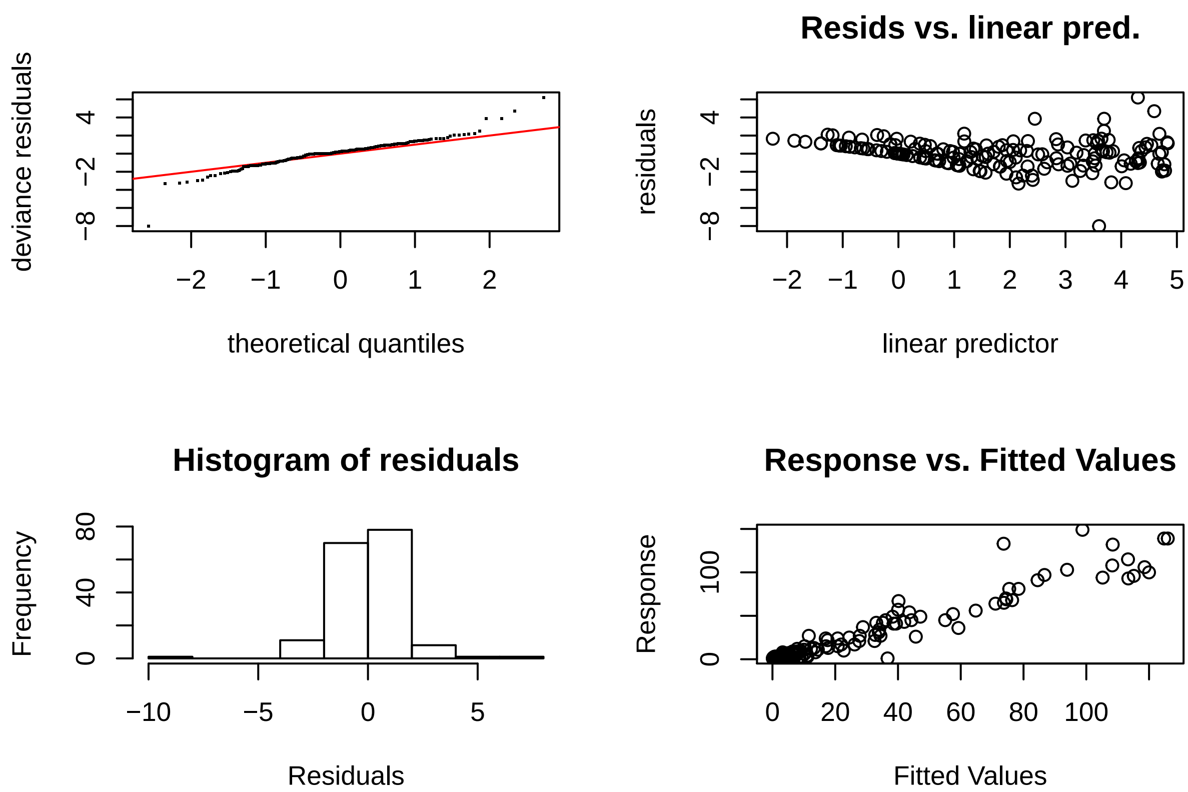

```
##
## Method: ML   Optimizer: outer newton
## full convergence after 6 iterations.
## Gradient range [-1.704072e-05,-1.704072e-05]
```

```
## (score 540.3471 & scale 1).
## Hessian positive definite, eigenvalue range [4.080029,4.080029].
## Model rank =  70 / 70
##
## 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(timeInt) 49.0 31.1    1.25       1
```

```
resGam4 = mgcv::gam(
  dead ~ s(timeInt, k=20, pc=0) + country_region, data=covid_data,
  family=poisson(link='log'), method='ML')
plot(resGam4)
```




```
gam.check(resGam4)
```

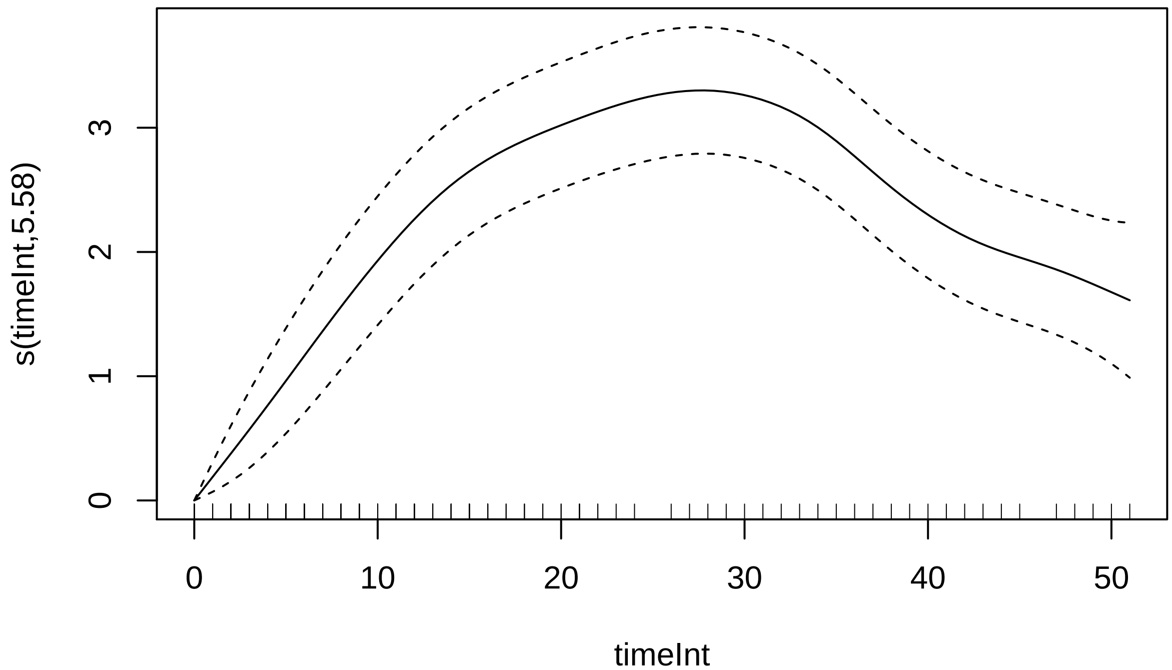



```
##
## Method: ML   Optimizer: outer newton
## full convergence after 6 iterations.
## Gradient range [3.691928e-06,3.691928e-06]
## (score 554.3095 & scale 1).
## Hessian positive definite, eigenvalue range [3.724135,3.724135].
## Model rank =  40 / 40
##
## Basis dimension (k) checking results. 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(timeInt) 19.0 13.9    1.15    0.97
```

```
covid_data$timeIntInd = covid_data$timeInt

resGammInd = gamm4::gamm4(
  dead ~ country_region +
      s(timeInt, k=20, pc=0),
    random = ~ (1|timeIntInd),
    data=covid_data, family=poisson(link='log'))

plot(resGammInd$gam)
```




```r
summary(resGammInd$mer)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: poisson  ( log )
##
##      AIC      BIC   logLik deviance df.resid
##   1082.2   1157.4   -517.1   1034.2      146
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.2542 -0.5002  0.0522  0.8694  5.2817
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  timeIntInd (Intercept) 0.08203  0.2864
##  Xr         s(timeInt)  5.18868  2.2779
## Number of obs: 170, groups:  timeIntInd, 50; Xr, 18
##
## Fixed effects:
##                                Estimate Std. Error z value Pr(>|z|)
## X(Intercept)                  -0.306160   0.605059  -0.506 0.612857
## Xcountry_regionAustralia       0.005782   1.163560   0.005 0.996035
## Xcountry_regionBeijing        -2.011839   0.741288  -2.714 0.006648 **
## Xcountry_regionChongqing      -0.657144   0.823363  -0.798 0.424800
## Xcountry_regionFrance          1.045114   0.612790   1.706 0.088101 .
## Xcountry_regionGuangdong      -1.642045   0.775352  -2.118 0.034192 *
## Xcountry_regionHainan         -2.299688   0.843723  -2.726 0.006418 **
## Xcountry_regionHebei          -0.882840   0.825789  -1.069 0.285031
## Xcountry_regionHeilongjiang   -1.055389   0.668758  -1.578 0.114535
## Xcountry_regionHenan          -1.242027   0.632998  -1.962 0.049747 *
## Xcountry_regionHubei           1.771843   0.590886   2.999 0.002712 **
```

```
## Xcountry_regionHunan           0.005607   1.163584   0.005 0.996155
## Xcountry_regionIran            1.236077   0.592175   2.087 0.036857 *
## Xcountry_regionIraq            0.150635   0.768603   0.196 0.844621
## Xcountry_regionItaly           2.044601   0.590686   3.461 0.000537 ***
## Xcountry_regionJapan          -1.418249   0.656942  -2.159 0.030861 *
## Xcountry_regionShandong        0.083484   0.822716   0.101 0.919175
## Xcountry_regionSouth Korea    -0.088985   0.599738  -0.148 0.882049
## Xcountry_regionSpain           2.017840   0.604858   3.336 0.000850 ***
## Xcountry_regionUnited Kingdom  1.337925   0.832874   1.606 0.108187
## Xcountry_regionUnited States   0.744861   0.623195   1.195 0.231998
## Xs(timeInt)Fx1                 2.801306   0.765064   3.662 0.000251 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

##
## Correlation matrix not shown by default, as p = 22 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it
```

```r
summary(resGammInd$gam)
```

```
##
## Family: poisson
## Link function: log
##
## Formula:
## dead ~ country_region + s(timeInt, k = 20, pc = 0)
##
## Parametric coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -0.306160   0.608512  -0.503 0.614874
## country_regionAustralia      0.005782   1.169958   0.005 0.996057
## country_regionBeijing       -2.011839   0.744806  -2.701 0.006910 **
## country_regionChongqing     -0.657144   0.827554  -0.794 0.427149
## country_regionFrance         1.045114   0.616594   1.695 0.090080 .
## country_regionGuangdong     -1.642045   0.779111  -2.108 0.035067 *
## country_regionHainan        -2.299688   0.850166  -2.705 0.006831 **
## country_regionHebei         -0.882840   0.829945  -1.064 0.287450
## country_regionHeilongjiang  -1.055389   0.672452  -1.569 0.116540
## country_regionHenan         -1.242027   0.636659  -1.951 0.051075 .
## country_regionHubei          1.771843   0.594531   2.980 0.002880 **
## country_regionHunan          0.005607   1.170033   0.005 0.996176
## country_regionIran           1.236077   0.595806   2.075 0.038021 *
## country_regionIraq           0.150635   0.773106   0.195 0.845515
## country_regionItaly          2.044601   0.594323   3.440 0.000581 ***
## country_regionJapan         -1.418249   0.660555  -2.147 0.031789 *
## country_regionShandong       0.083484   0.827629   0.101 0.919653
## country_regionSouth Korea   -0.088985   0.603363  -0.147 0.882753
## country_regionSpain          2.017840   0.608684   3.315 0.000916 ***
## country_regionUnited Kingdom  1.337925   0.839165   1.594 0.110857
## country_regionUnited States   0.744861   0.627042   1.188 0.234874
## ---
## 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(timeInt) 5.579  5.579  289.7  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.884
## glmer.ML = 250.06  Scale est. = 1         n = 170
```

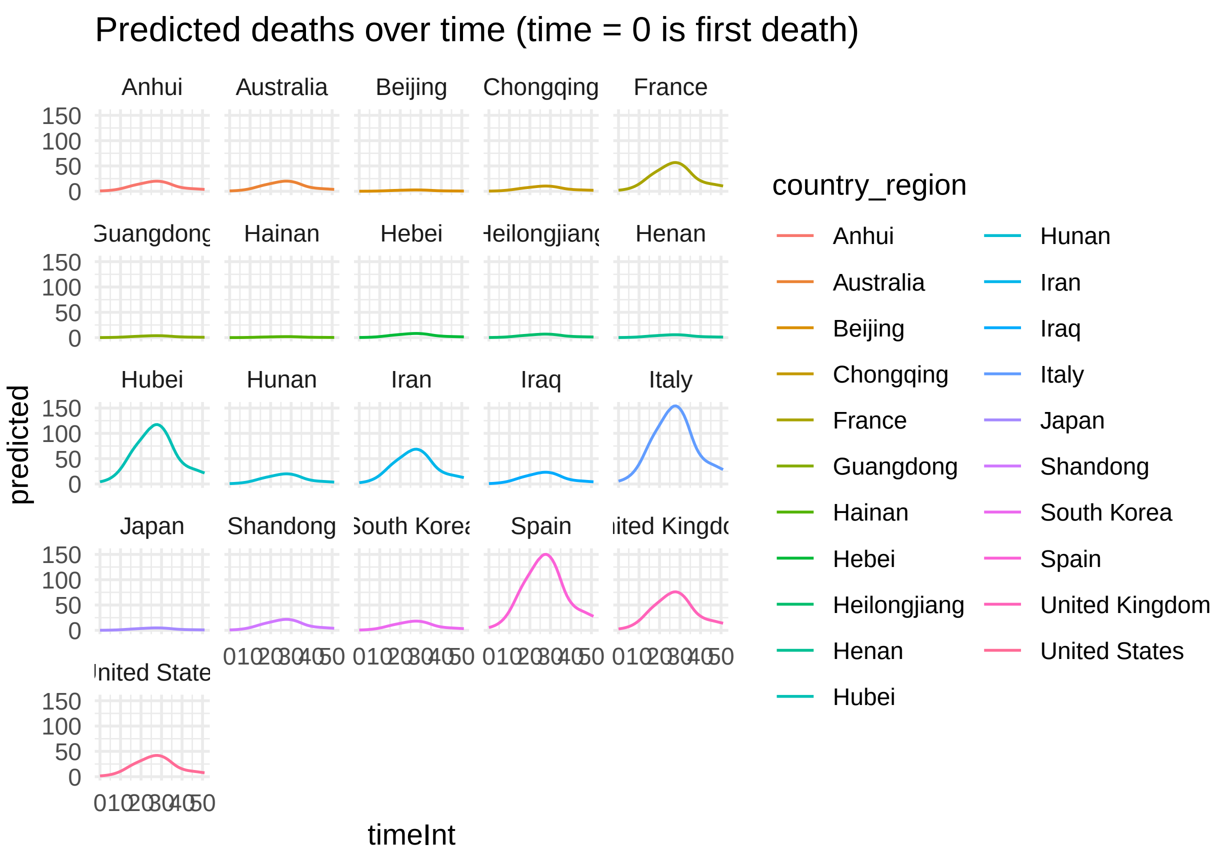
The plot suggests a trend where we estimate a sharper increase in deaths per day over the first 25 days to a month and then the number decreases from about day 30 onwards.

```
covid_data_2 <- expand_grid(covid_data$timeInt, covid_data$country_region) %>%
  as_tibble() %>%
  rename(timeInt = 1, country_region = 2) %>%
  distinct()

covid_data_2$predicted <- predict(resGammInd$gam, newdata=covid_data_2, type="response")

#covid_data_3 <- bind_cols(covid_data_2, predicted) %>%
  #mutate(lower = fit - 2*se.fit, upper = fit + 2*se.fit)

covid_data_2 %>%
  ggplot(aes(timeInt, predicted, colour=country_region)) +
  geom_line() +
  theme_minimal() +
  facet_wrap(~country_region) +
  ggtitle("Predicted deaths over time (time = 0 is first death)")
```



The plot shows the Predicted deaths over time (time = 0 is first death).

Fit a different model.

Now we fit a different model with time being a random slope.

```
covid_data$timeSlope = covid_data$timeInt/100

resGammSlope = gamm4::gamm4(
  dead ~ country_region + s(timeInt, k=30, pc=0),
    random = ~(0+timeSlope|country_region) +
    (1|timeIntInd:country_region),
  data=covid_data, family=poisson(link='log'))
#save(resGammSlope, file='resGamSlope.RData')
plot(resGammSlope$gam)
summary(resGammSlope$mer)
names(lme4::ranef(resGammSlope$mer))
theRanef = lme4::ranef(resGammSlope$mer, condVar = TRUE)$country_region
(theRanefVec = sort(drop(t(theRanef))))

Dcountry = 'France'
toPredict = expand.grid(
  timeInt = 0:100,
  country_region = Dcountry)
toPredict$timeSlope = toPredict$timeIntInd =
  toPredict$timeInt
thePred = predict(resGammSlope$gam,
```

```
                  newdata=toPredict, se.fit=TRUE)

matplot(toPredict$timeInt,
        exp(do.call(cbind, thePred) %*% Pmisc::ciMat(0.75)),
        type='l',
        col=c('black','grey','grey'),
        ylim = c(0, 25))
points(covid_data[covid_data$country_region == Dcountry,c('timeInt','dead')],
       col='red')
```

Appendix

1. The COVID-19 data was retrieved from GitHub and the procedure is shown below.

```
install.packages("devtools")
devtools::install_github("GuangchuangYu/nCov2019")

x1 <- nCov2019::load_nCov2019(lang = 'en')

cutoff=3

x2 = by(x1$global, x1$global[,'country', drop=FALSE],
        function(xx) {
        xx$incidence = diff(c(0, xx$cum_confirm))
        xx$dead = diff(c(0, xx$cum_dead))
        if(any(xx$cum_dead >= cutoff)) {
            cutoffHere = min(xx[xx$cum_dead >= cutoff,'time'], na.rm=TRUE) +1
            xx$timeInt = as.numeric(difftime(xx$time, cutoffHere, units='days'))
            xx = xx[xx$timeInt >= 0, ]
            xx=                 xx[,c('time','timeInt','cum_confirm','cum_dead','incidence','dead','cou
        } else {
            xx = NULL
        }
        xx
    }, simplify=FALSE)

x3 = by(x1$province, x1$province[,'province', drop=FALSE],
        function(xx) {
        xx$incidence = diff(c(0, xx$cum_confirm))
        xx$dead = diff(c(0, xx$cum_dead))
        colnames(xx) = gsub("province","country", colnames(xx))
        if(any(xx$cum_dead >= cutoff)) {
            cutoffHere = min(xx[xx$cum_dead >= cutoff,'time'], na.rm=TRUE) +1
            xx$timeInt = as.numeric(difftime(xx$time, cutoffHere, units='days'))
            xx = xx[xx$timeInt >= 0, ]
            xx=                 xx[,c('time','timeInt','cum_confirm','cum_dead','incidence','dead','cou
        } else {
            xx = NULL
        }
        xx
    }, simplify=FALSE)
class(x2) = class(x3) = 'list'
```

```r
x2 = x2[grep('China', names(x2), invert=TRUE)]
x = c(x2, x3)
x$Hubei[x$Hubei$incidence > 4000,c('dead','incidence')] = NA

tidy_data <- compact(x) %>% bind_rows() %>%
  rename(country_region = country) %>%
  filter(dead>0)
write_csv(tidy_data, "covid_data.csv")
```

2. Some codes were modified from the assignment of the class ran by Prof. Brown and Prof. Bolton from the University of Toronto.