COVID19_Analysis

Qizheng Wang

4/10/2020

```r
library(devtools)
```

```
## Loading required package: usethis
```

```r
library(mgcv)
```

```
## Loading required package: nlme
```

```
## This is mgcv 1.8-31. For overview type 'help("mgcv-package")'.
```

```r
library(gamm4)
```

```
## Loading required package: Matrix
```

```
## Loading required package: lme4
```

```
## Warning: package 'lme4' was built under R version 3.6.3
```

```
##
## Attaching package: 'lme4'
```

```
## The following object is masked from 'package:nlme':
##
##     lmList
```

```
## This is gamm4 0.2-6
```

```r
library(tidyverse)
```

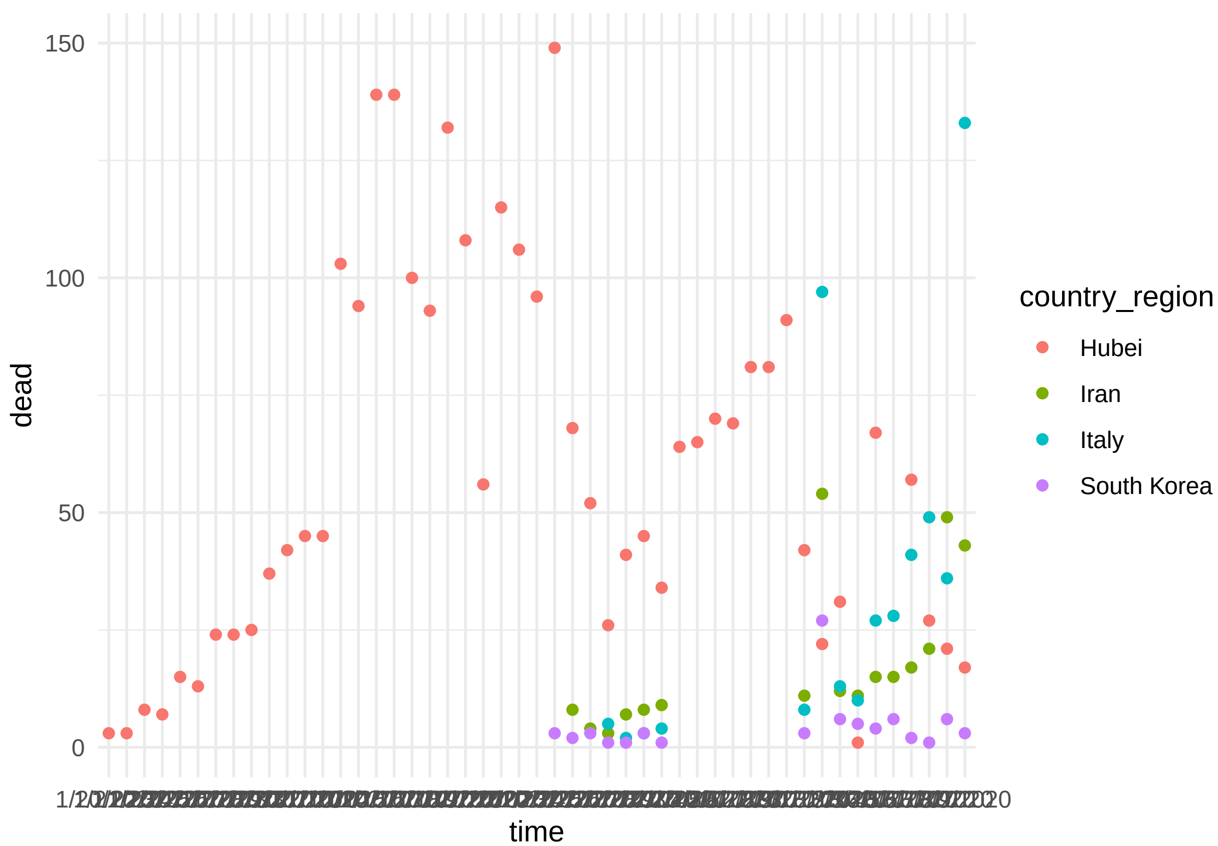
```
## -- Attaching packages ------------------------------------------------------------------------------
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.4
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

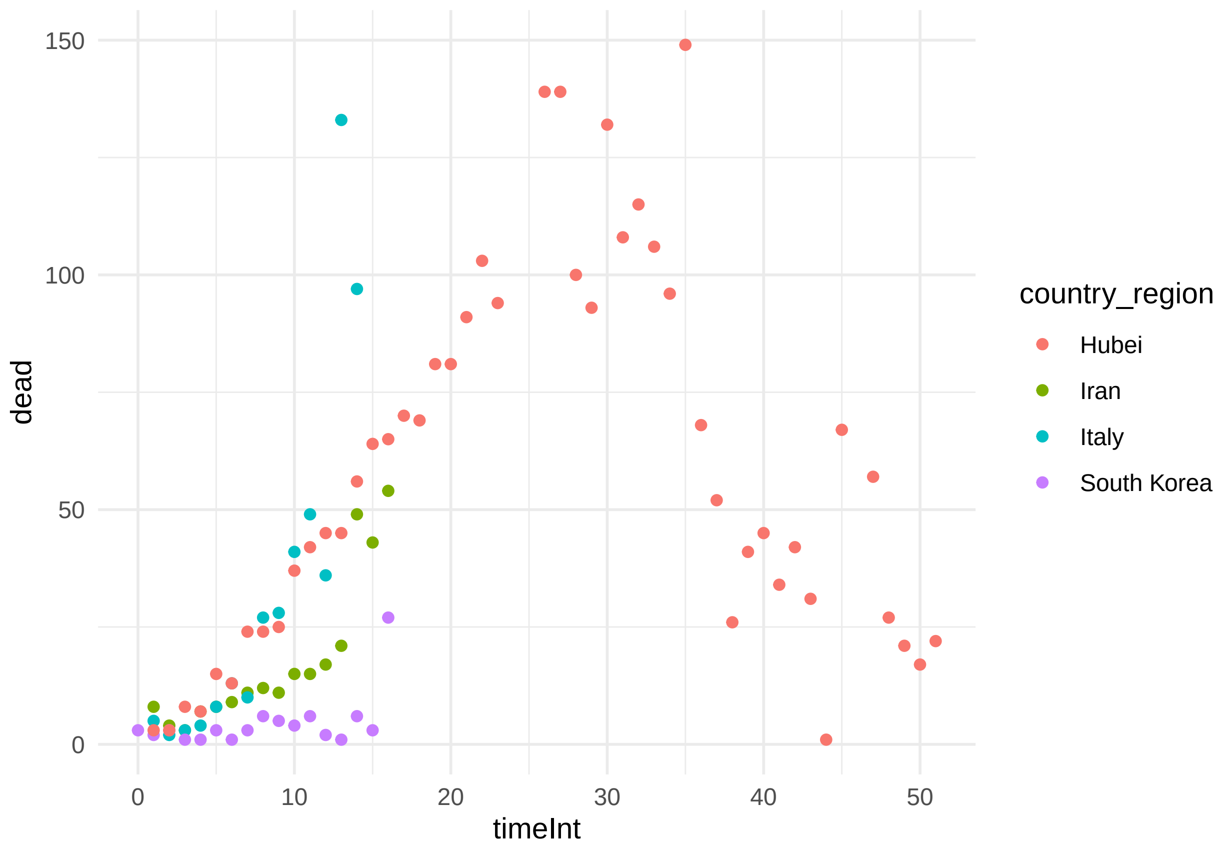
```
## -- Conflicts ------------------------------------------------------------------------------- tidyve
## x dplyr::collapse() masks nlme::collapse()
## x tidyr::expand()   masks Matrix::expand()
## x dplyr::filter()   masks stats::filter()
## x dplyr::lag()      masks stats::lag()
## x tidyr::pack()     masks Matrix::pack()
## x tidyr::unpack()   masks Matrix::unpack()
```

```
setwd("c:/Users/roder/OneDrive/Desktop/COVID")
covid_data <- read.csv("covid_data.csv")

# 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()
```




```
# 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()
```




```r
## Setting up GAM Model
## Because timeInt indicates date since the first death, so we should constrain the line to pass
## through the origin when timeInt = 0, indicating there are no death before any deaths occured

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

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 *
```

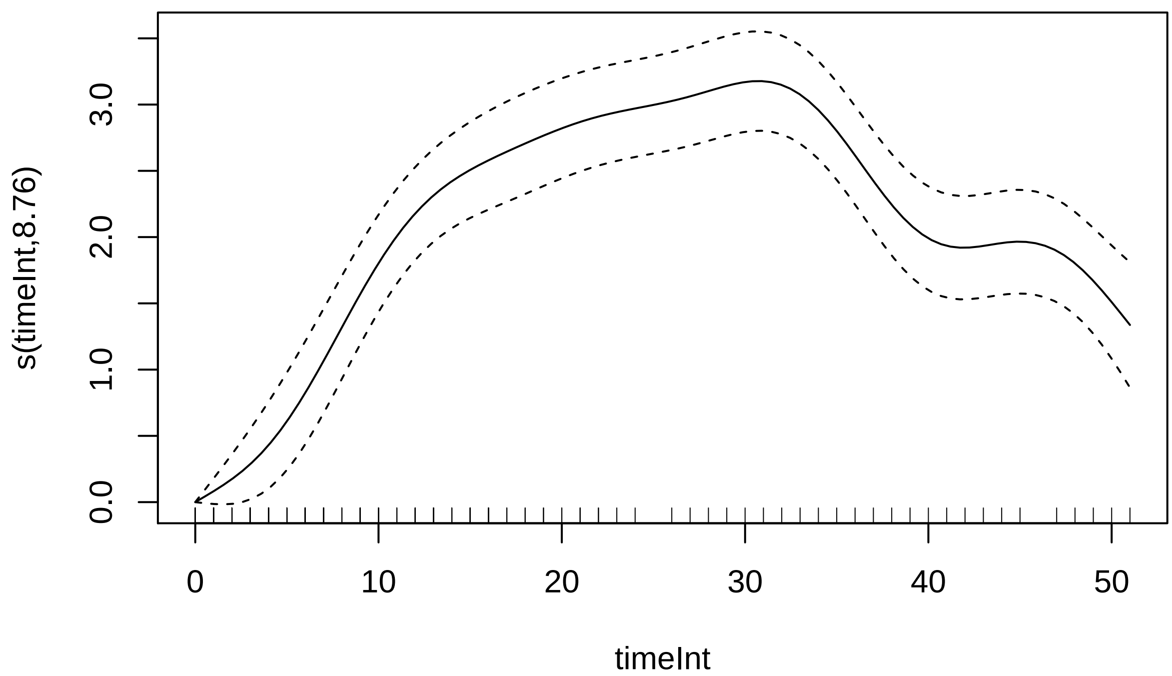
```
## country_regionHainan         -2.168937   0.824279  -2.631 0.008506 **
## country_regionHebei          -0.763389   0.823787  -0.927 0.354092
## country_regionHeilongjiang   -1.118993   0.666038  -1.680 0.092943 .
## country_regionHenan          -1.208796   0.631050  -1.916 0.055425 .
## country_regionHubei           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
```

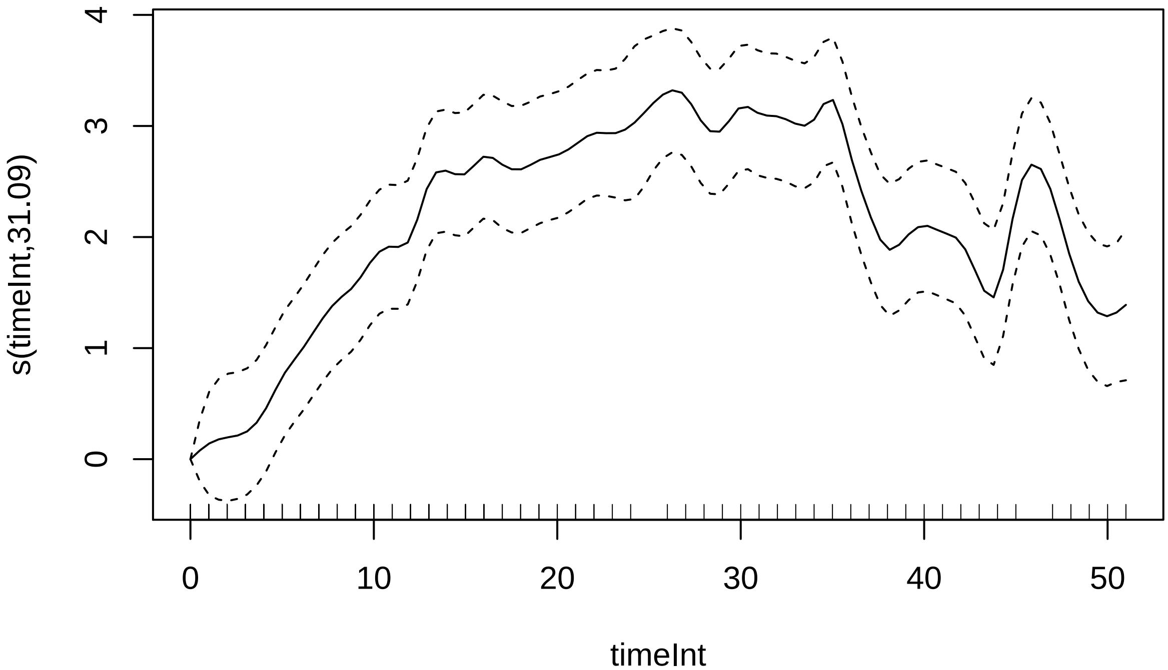
```
##                  -0.257405570                  0.133254518
##                  s(timeInt).6                 s(timeInt).7
##                   1.140898783                 -0.022139449
##                  s(timeInt).8                 s(timeInt).9
##                   4.992873514                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

```r
plot(resGam)
```

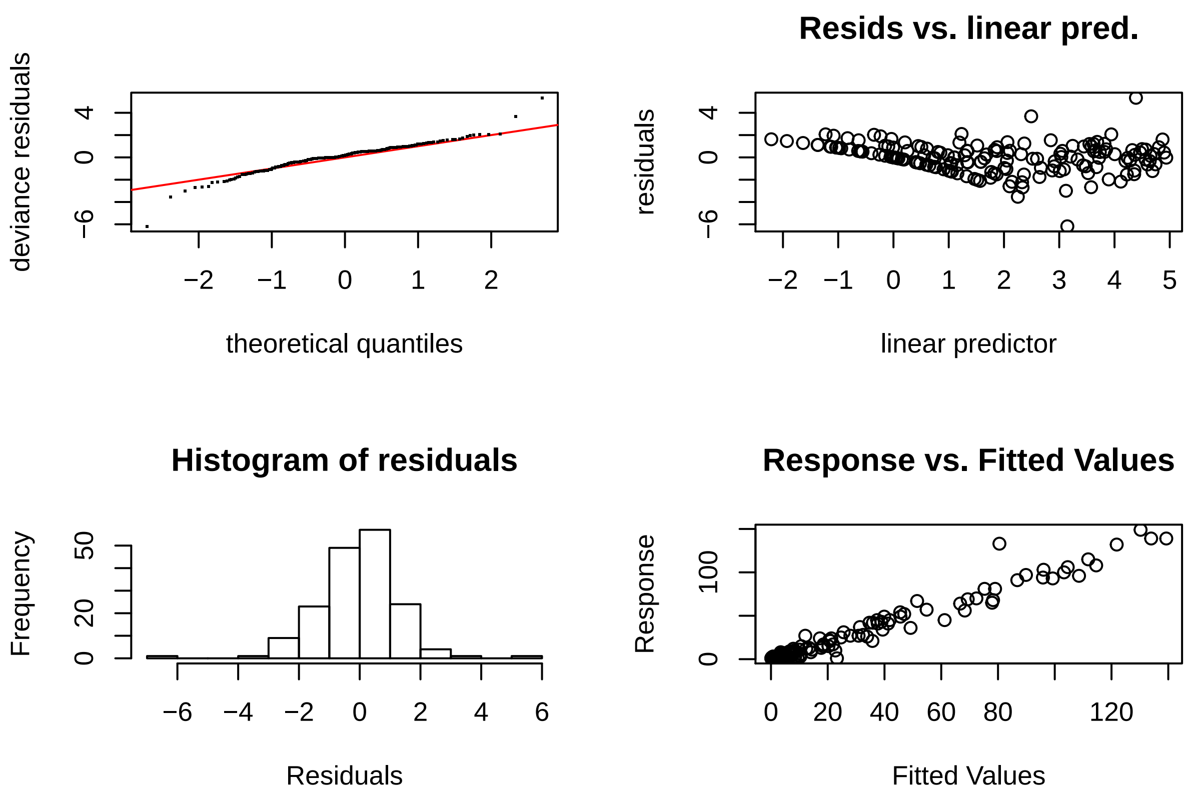


Analysis: the estimated degrees of freedom for the smooth of timeInt is 8.758, Since edf is much higher than 1, the relationship is not linear. And we can't interpret the coefficients for the smooth of timeInt because they are coefficients for the different splines taht make up our curve, but don't have a scientific interpretation, thus we cannot interpret the coefficients for country_region as usual.

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

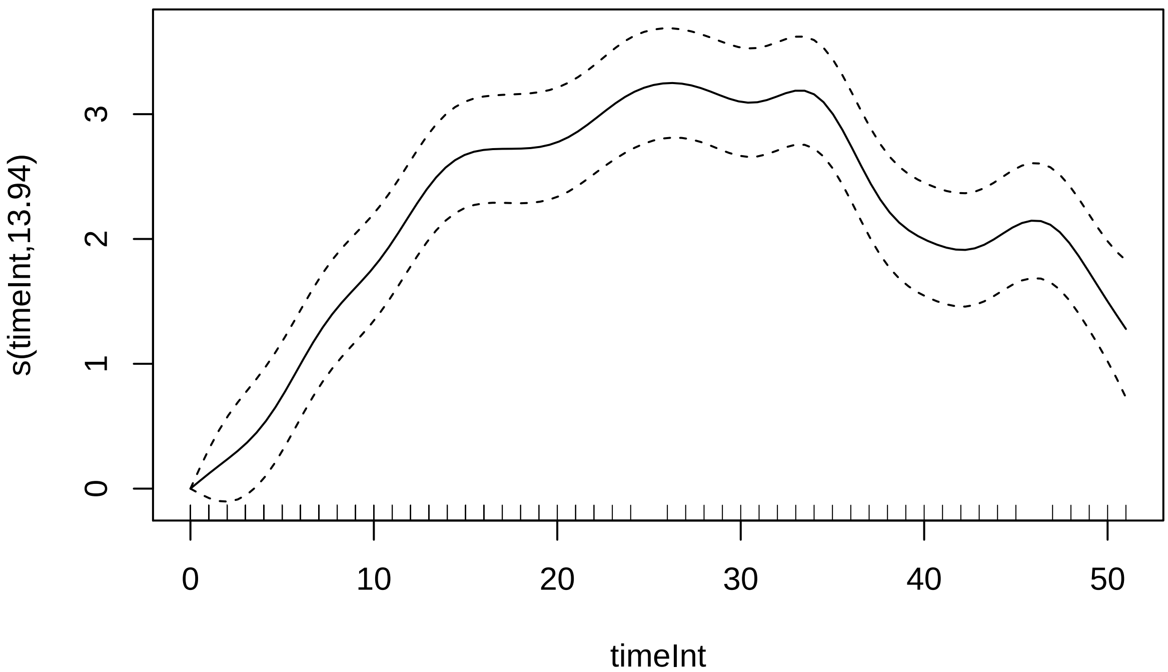



```
gam.check(resGam3)
```

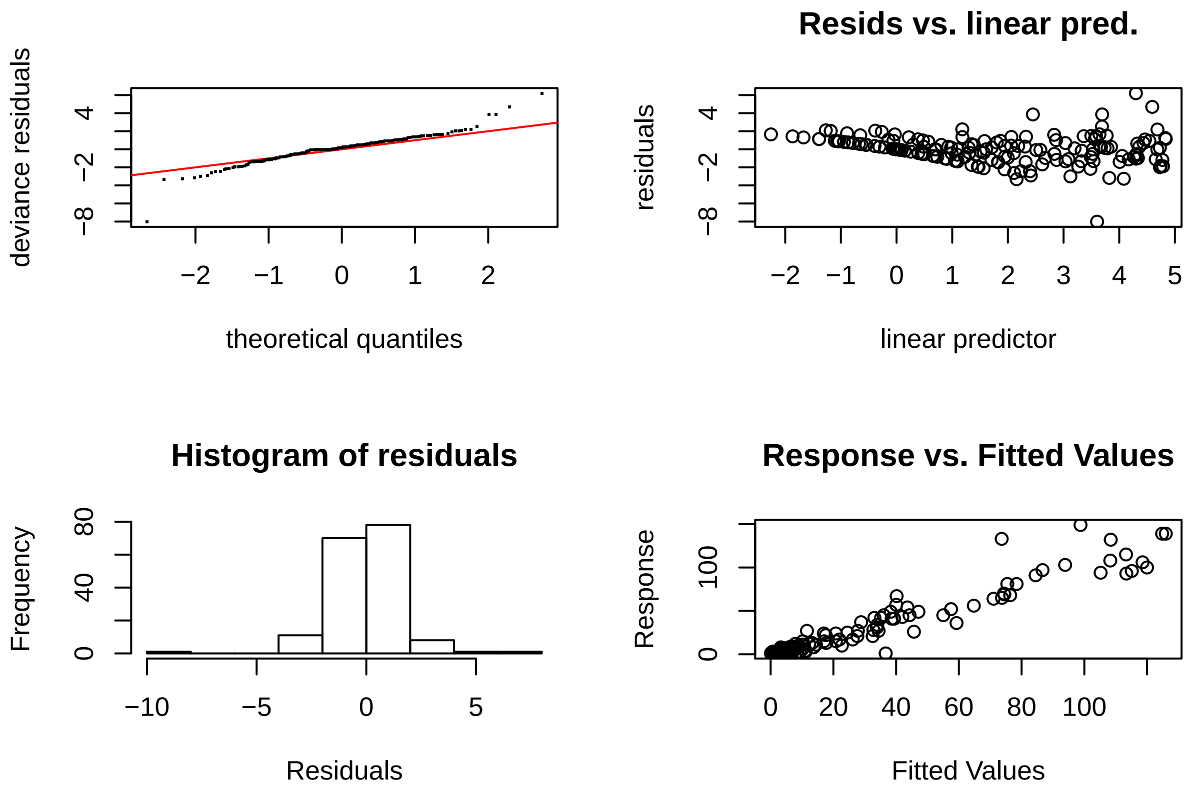



```
##
## Method: ML   Optimizer: outer newton
## full convergence after 6 iterations.
## Gradient range [-1.704072e-05,-1.704072e-05]
## (score 540.3471 & scale 1).
## Hessian positive definite, eigenvalue range [4.080029,4.080029].
## Model rank =  70 / 70
##
## Basis dimension (k) checking results. 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
```

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



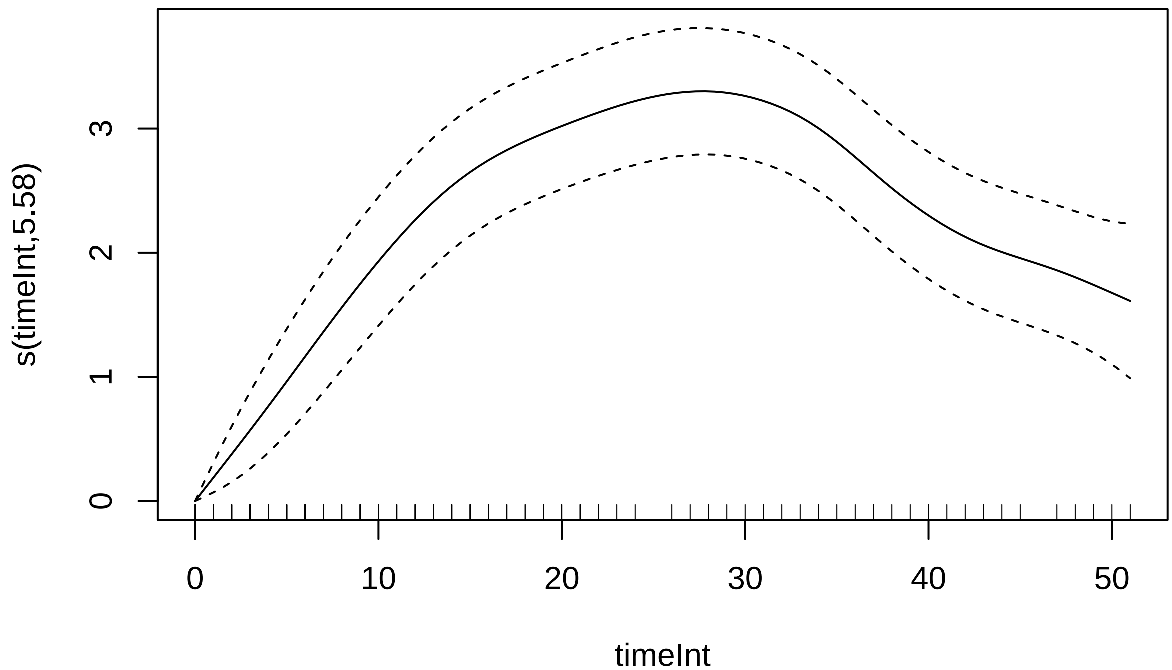

```
gam.check(resGam4)
```




```
##
## 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    0.95
```

In this part, choose different k, k =50 and k = 20. And run gam.check() for both of them.
After seeing the result, I would choose k =20 for capturing patterns in the data without overfittingthe data.
However data seems highly correlated. A random effect for the country should be fitted.

```
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  5.2818
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  timeIntInd (Intercept) 0.08203  0.2864
##  Xr         s(timeInt)  5.19007  2.2782
## Number of obs: 170, groups:  timeIntInd, 50; Xr, 18
##
## Fixed effects:
##                              Estimate Std. Error z value Pr(>|z|)
## X(Intercept)                -0.306480   0.605145  -0.506 0.612536
## Xcountry_regionAustralia     0.006297   1.163527   0.005 0.995682
## Xcountry_regionBeijing      -2.011547   0.741361  -2.713 0.006661 **
## Xcountry_regionChongqing    -0.656657   0.823396  -0.797 0.425162
```

```
## Xcountry_regionFrance          1.045405   0.612873   1.706 0.088055 .
## Xcountry_regionGuangdong      -1.641456   0.775393  -2.117 0.034265 *
## Xcountry_regionHainan         -2.299227   0.843758  -2.725 0.006430 **
## Xcountry_regionHebei          -0.882402   0.825837  -1.068 0.285298
## Xcountry_regionHeilongjiang   -1.054884   0.668823  -1.577 0.114744
## Xcountry_regionHenan          -1.241604   0.633073  -1.961 0.049852 *
## Xcountry_regionHubei           1.772212   0.590969   2.999 0.002710 **
## Xcountry_regionHunan           0.006266   1.163493   0.005 0.995703
## Xcountry_regionIran            1.236439   0.592258   2.088 0.036828 *
## Xcountry_regionIraq            0.151040   0.768669   0.196 0.844223
## Xcountry_regionItaly           2.044959   0.590769   3.462 0.000537 ***
## Xcountry_regionJapan          -1.417716   0.657006  -2.158 0.030940 *
## Xcountry_regionShandong        0.083982   0.822740   0.102 0.918697
## Xcountry_regionSouth Korea    -0.088619   0.599821  -0.148 0.882546
## Xcountry_regionSpain           2.018165   0.604940   3.336 0.000850 ***
## Xcountry_regionUnited Kingdom  1.338358   0.832920   1.607 0.108093
## Xcountry_regionUnited States   0.745296   0.623271   1.196 0.231782
## Xs(timeInt)Fx1                 2.801119   0.765139   3.661 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
```

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

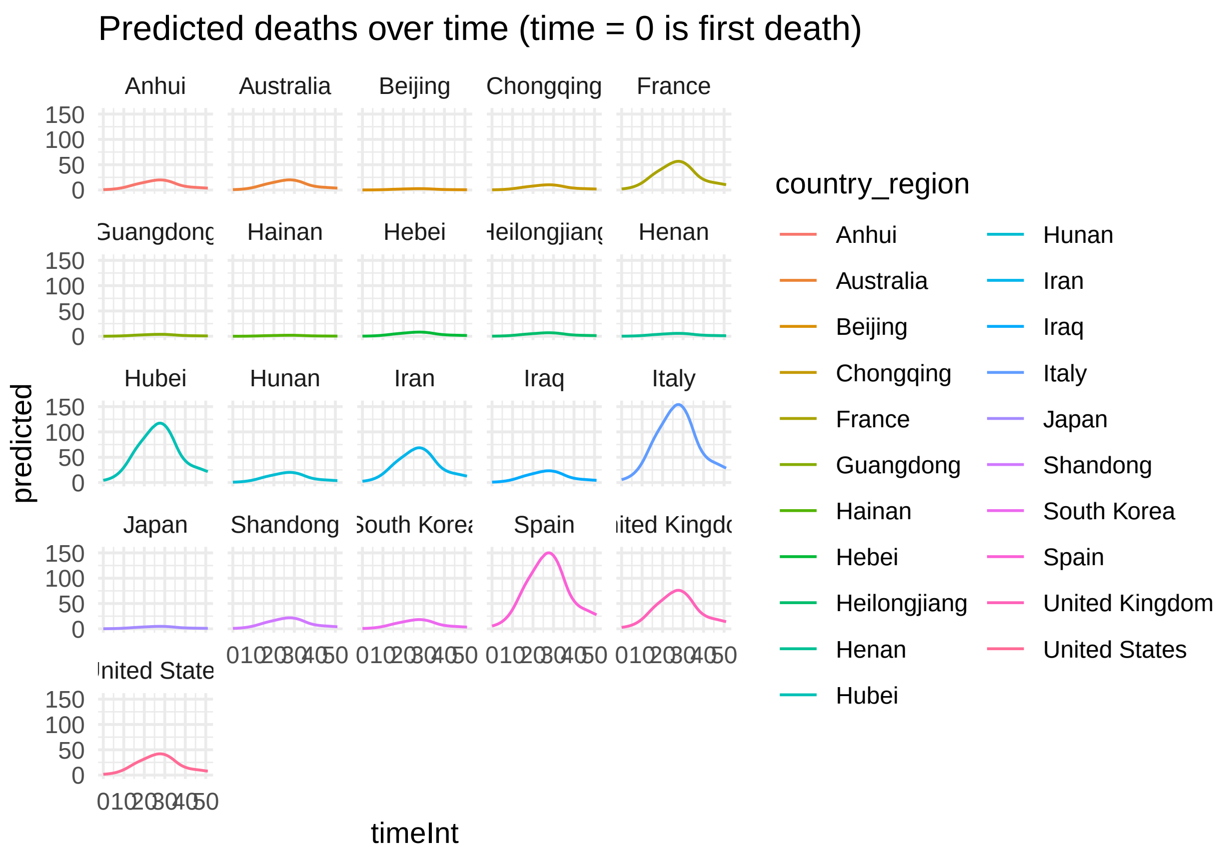
```
##
## Family: poisson
## Link function: log
##
## Formula:
## dead ~ country_region + s(timeInt, k = 20, pc = 0)
##
## Parametric coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -0.306480   0.608607  -0.504 0.614559
## country_regionAustralia      0.006297   1.169943   0.005 0.995705
## country_regionBeijing       -2.011547   0.744893  -2.700 0.006925 **
## country_regionChongqing     -0.656657   0.827599  -0.793 0.427517
## country_regionFrance         1.045405   0.616690   1.695 0.090040 .
## country_regionGuangdong     -1.641456   0.779151  -2.107 0.035141 *
## country_regionHainan        -2.299227   0.850216  -2.704 0.006845 **
## country_regionHebei         -0.882402   0.830000  -1.063 0.287721
## country_regionHeilongjiang  -1.054884   0.672528  -1.569 0.116756
## country_regionHenan         -1.241604   0.636748  -1.950 0.051186 .
## country_regionHubei          1.772212   0.594628   2.980 0.002879 **
## country_regionHunan          0.006266   1.169956   0.005 0.995727
## country_regionIran           1.236439   0.595903   2.075 0.037996 *
## country_regionIraq           0.151040   0.773174   0.195 0.845119
## country_regionItaly          2.044959   0.594420   3.440 0.000581 ***
## country_regionJapan         -1.417716   0.660632  -2.146 0.031873 *
```

```
## country_regionShandong        0.083982   0.827671   0.101 0.919180
## country_regionSouth Korea    -0.088619   0.603459  -0.147 0.883249
## country_regionSpain           2.018165   0.608779   3.315 0.000916 ***
## country_regionUnited Kingdom  1.338358   0.839214   1.595 0.110762
## country_regionUnited States   0.745296   0.627131   1.188 0.234667
## ---
## 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) 5.58   5.58  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
```

So use counry_region nessted within timIntInd to fit another same model.
This suggests a tren where there is a shaper increase in the deaths per day over the first 25 days to a month and the number decreases the following 30 days.

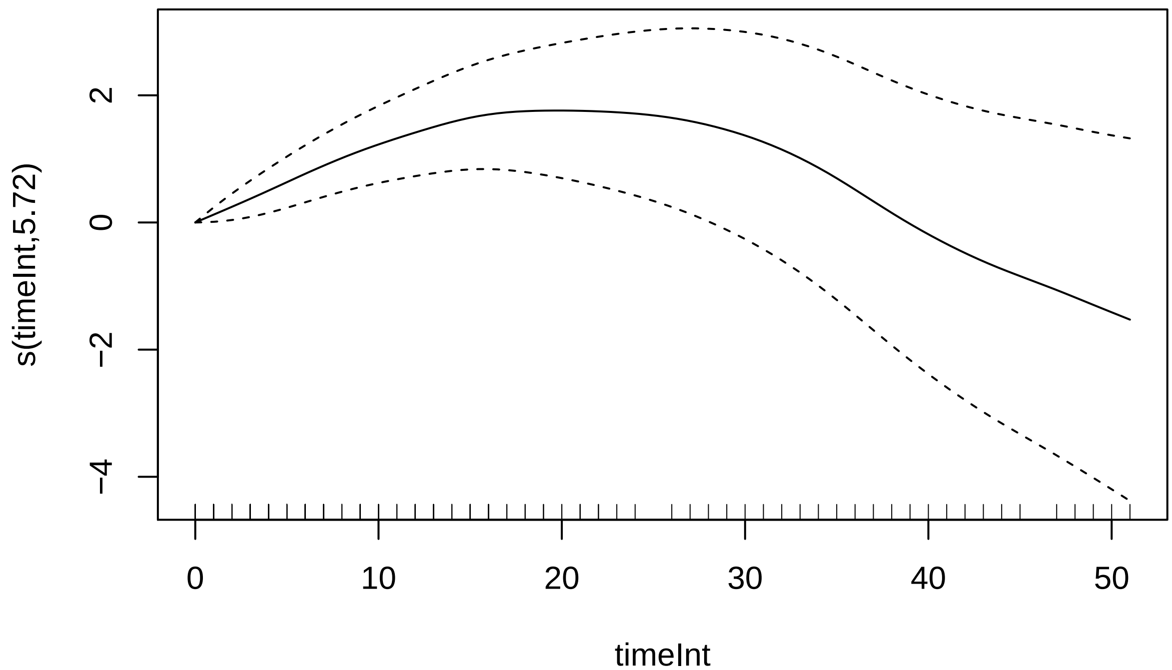
```r
covid_data_2 <- expand_grid(covid_data$timeInt, covid_data$country_region) %>%
as_tibble() %>%
rename(timeInt = 1, country_region = 2) %>%
distinct()
covid_data_2$predicted <- predict(resGammInd$gam, newdata=covid_data_2, type="response")

covid_data_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)")
```




```
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'))

plot(resGammSlope$gam)
```




```r
summary(resGammSlope$mer)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: poisson  ( log )
##
##      AIC      BIC   logLik deviance df.resid
##    991.2   1069.6   -470.6    941.2      145
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.2172 -0.3074 -0.0140  0.2211  2.0847
##
## Random effects:
##  Groups                    Name        Variance Std.Dev.
##  timeIntInd:country_region (Intercept)  0.08517 0.2918
##  Xr                        s(timeInt)   3.57360 1.8904
##  country_region            timeSlope   55.12944 7.4249
## Number of obs: 170, groups:
## timeIntInd:country_region, 170; Xr, 28; country_region, 21
##
## Fixed effects:
##                               Estimate Std. Error z value Pr(>|z|)
## X(Intercept)                  -0.24988    0.62034  -0.403  0.68709
## Xcountry_regionAustralia       0.09264    1.20949   0.077  0.93894
```

```
## Xcountry_regionBeijing        -0.63006    1.20219  -0.524  0.60021
## Xcountry_regionChongqing      -0.25543    0.92027  -0.278  0.78135
## Xcountry_regionFrance          0.95325    0.69546   1.371  0.17048
## Xcountry_regionGuangdong      -0.31269    0.95823  -0.326  0.74418
## Xcountry_regionHainan         -0.56894    1.18326  -0.481  0.63065
## Xcountry_regionHebei          -0.55685    0.98029  -0.568  0.57000
## Xcountry_regionHeilongjiang    0.14675    0.77880   0.188  0.85053
## Xcountry_regionHenan           0.43946    0.72421   0.607  0.54397
## Xcountry_regionHubei           1.80314    0.65407   2.757  0.00584 **
## Xcountry_regionHunan           0.09280    1.20941   0.077  0.93884
## Xcountry_regionIran            1.34649    0.66567   2.023  0.04310 *
## Xcountry_regionIraq            0.16359    0.83447   0.196  0.84458
## Xcountry_regionItaly           0.98696    0.68097   1.449  0.14724
## Xcountry_regionJapan           0.17602    0.82260   0.214  0.83056
## Xcountry_regionShandong        0.24189    0.89838   0.269  0.78774
## Xcountry_regionSouth Korea     0.46834    0.69393   0.675  0.49974
## Xcountry_regionSpain           1.97649    0.66869   2.956  0.00312 **
## Xcountry_regionUnited Kingdom  1.31487    0.89625   1.467  0.14235
## Xcountry_regionUnited States   0.93941    0.69632   1.349  0.17730
## Xs(timeInt)Fx1                 1.57790    0.81609   1.933  0.05318 .
## ---
## 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
```

```
names(lme4::ranef(resGammSlope$mer))
```

```
## [1] "timeIntInd:country_region" "Xr"
## [3] "country_region"
```

```
theRanef = lme4::ranef(resGammSlope$mer, condVar = TRUE)$country_region
(theRanefVec = sort(drop(t(theRanef))))
```

```
##          Japan          Henan   Heilongjiang      Guangdong         Hainan
##    -7.45640077    -7.39197752    -6.59756826    -4.01398844    -3.18040196
##        Beijing  United States      Chongqing          Anhui          Hebei
##    -2.65471706    -1.74396156    -1.45000907    -0.17970022    -0.15082900
##           Iraq United Kingdom          Hunan      Australia       Shandong
##    -0.02958292     0.00000000     0.01712546     0.01720012     0.25261765
##    South Korea          Spain         France           Iran          Hubei
##     1.40347325     3.16651164     5.55065837     5.63517909     6.01194601
##          Italy
##    16.14488005
```

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

```r
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')
```