HW 10

As we did not have a chance to get to the worksheet on Monday, we will instead tackle the Puerto Rico Mortality data as part of the homework! The dataset consists of all-cause mortality in Puerto Rico on each date from 2015 to 2018, and can be loaded by running the following commands:

```
library(tidyverse)
library(lubridate)
library(dslabs)
library(knitr)
library(fANCOVA)
library(caret)

puerto_path <- str_c("https://github.com/theodds/SDS-348/raw/master/",
                     "puerto_rico.csv")

puerto_rico <- read_csv(puerto_path) %>% drop_na() %>% arrange(date)

puerto_rico %>% head() %>% kable()
```

day	month	year	deaths	date
1	1	2015	107	2015-01-01
2	1	2015	101	2015-01-02
3	1	2015	78	2015-01-03
4	1	2015	121	2015-01-04
5	1	2015	99	2015-01-05
6	1	2015	104	2015-01-06

The dataset contains the number of deaths for each day of the year, with the column `date` formatted as a date.

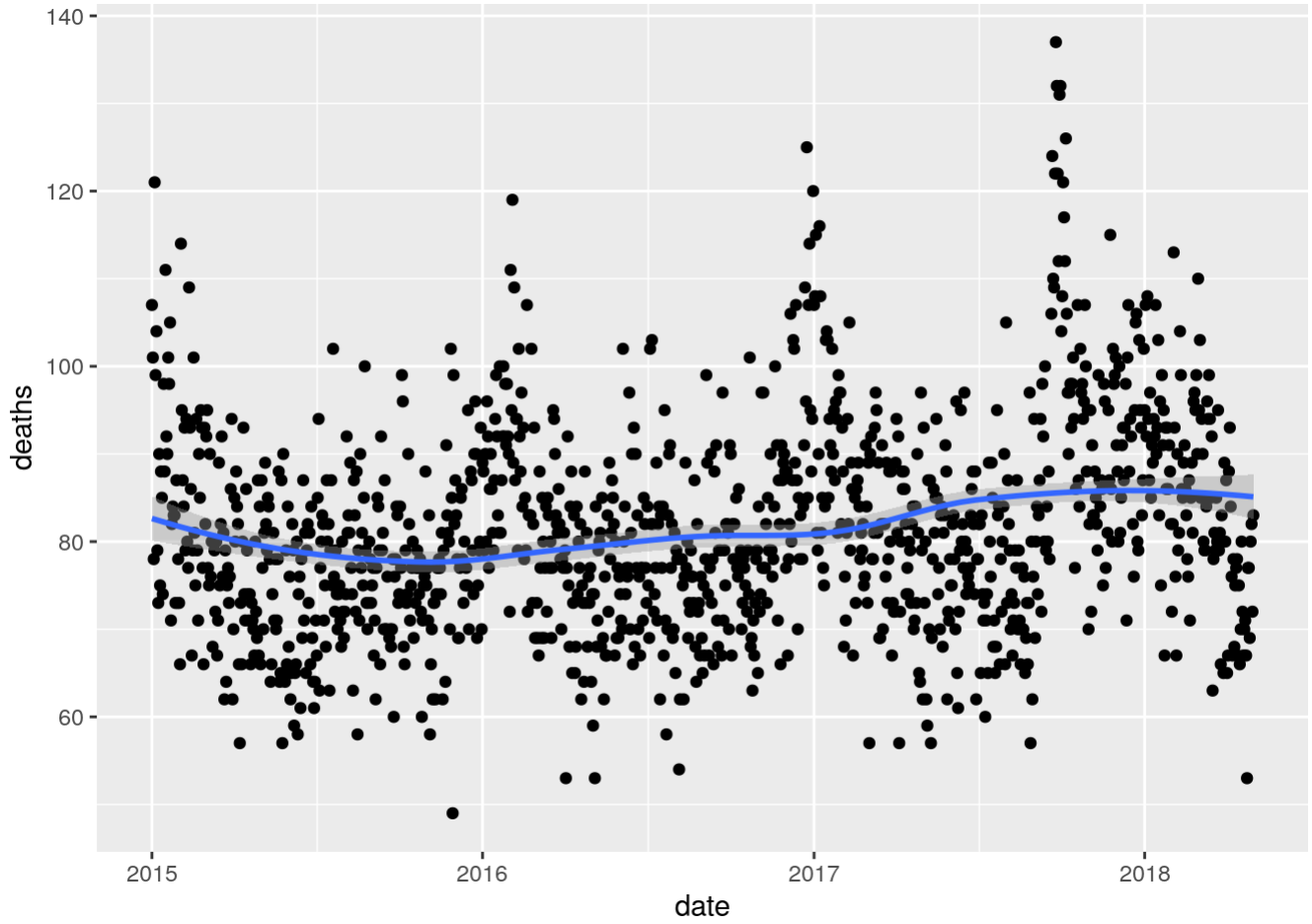
Question 1 (2 pts)

Use `ggplot` to plot the deaths over time and add a `loess` smooth to the data with the default `span`. Comment on any trends you see in deaths over time.

Note: for large numbers of observations, `geom_smooth` switches from `loess` to a different smoothing technique. We can force it to use `loess` by setting `method = "loess"`.

```
## Your code here
puerto_rico %>% ggplot(aes(x = date, y = deaths)) + geom_point() + geom_smooth(method = "loess")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

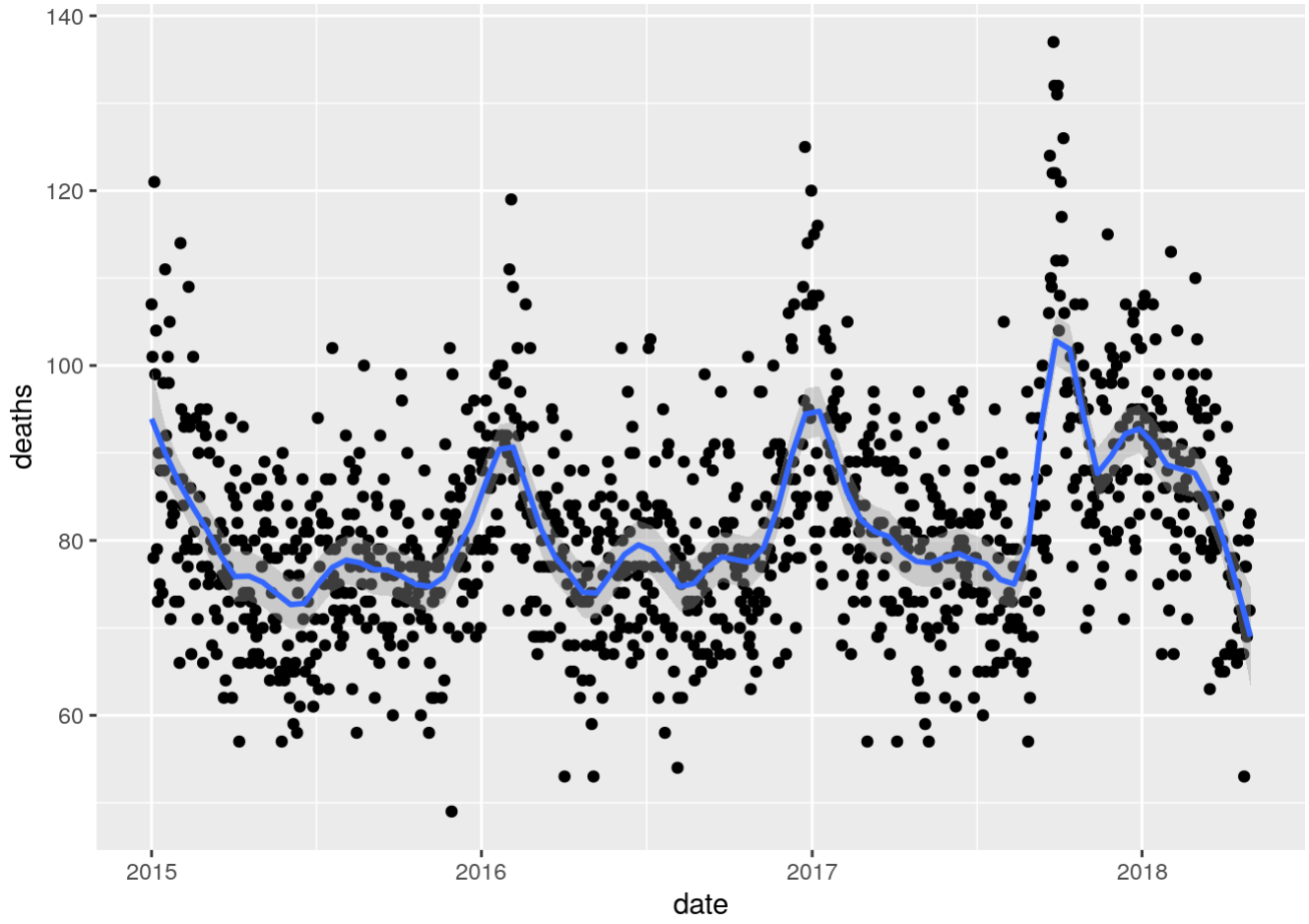


Answer: Based on the plot above, it can be seen that the number of deaths over time seems to fluctuate a lot, spiking more at intervals of about a year. The trend shown by the line though, indicates that the number of deaths over time seems to remain relatively constant and does not really change much over time. As such, it's not a good approximation of the data.

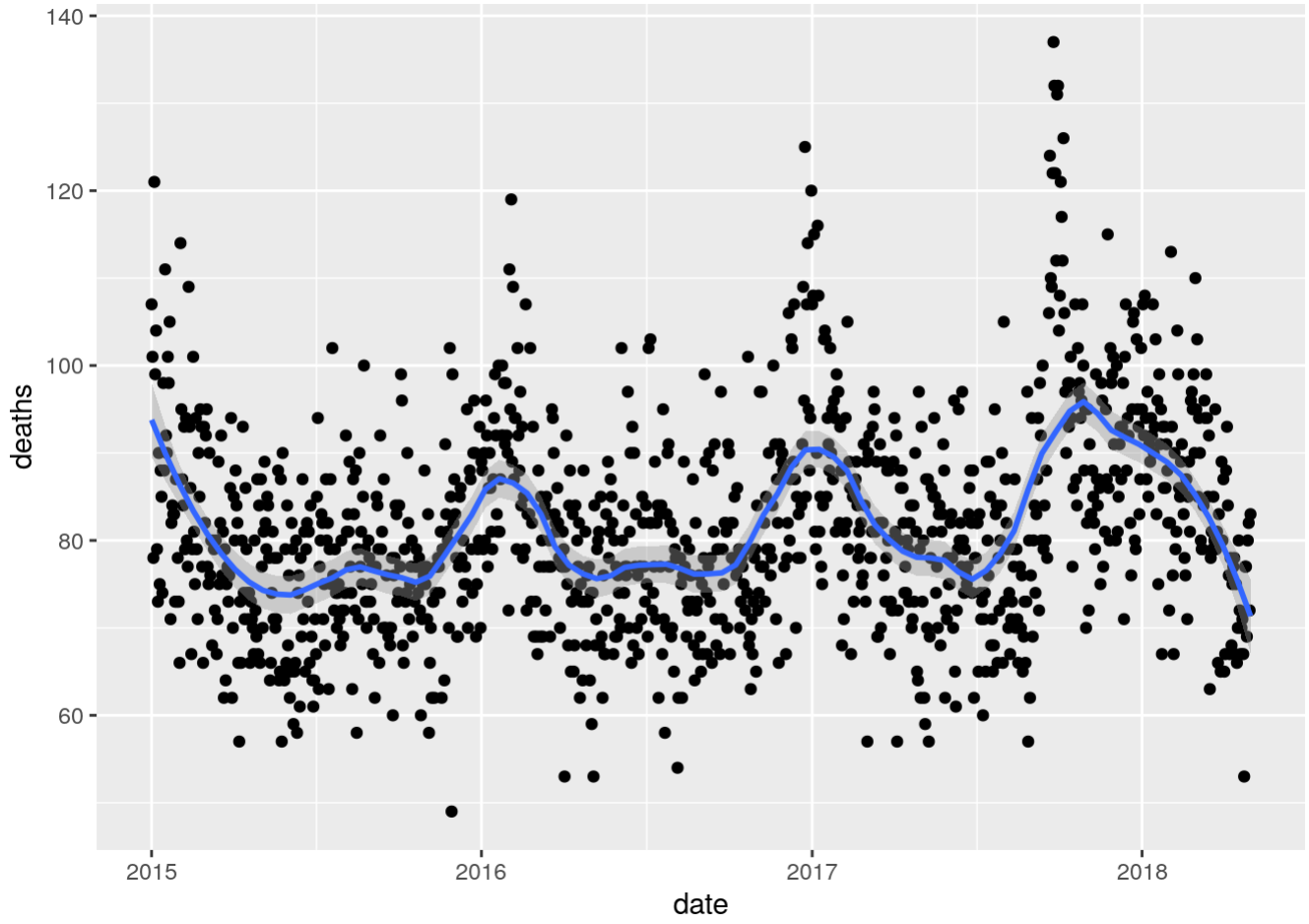
Question 2 (2 pts)

Play around with the `span` argument in `geom_smooth` and try to find a span that (to your eyes) does a better job than the default `span` of `0.75`. You may, or may not, conclude that the default `span` is good enough. **For this better span, comment on any trends you see now.**

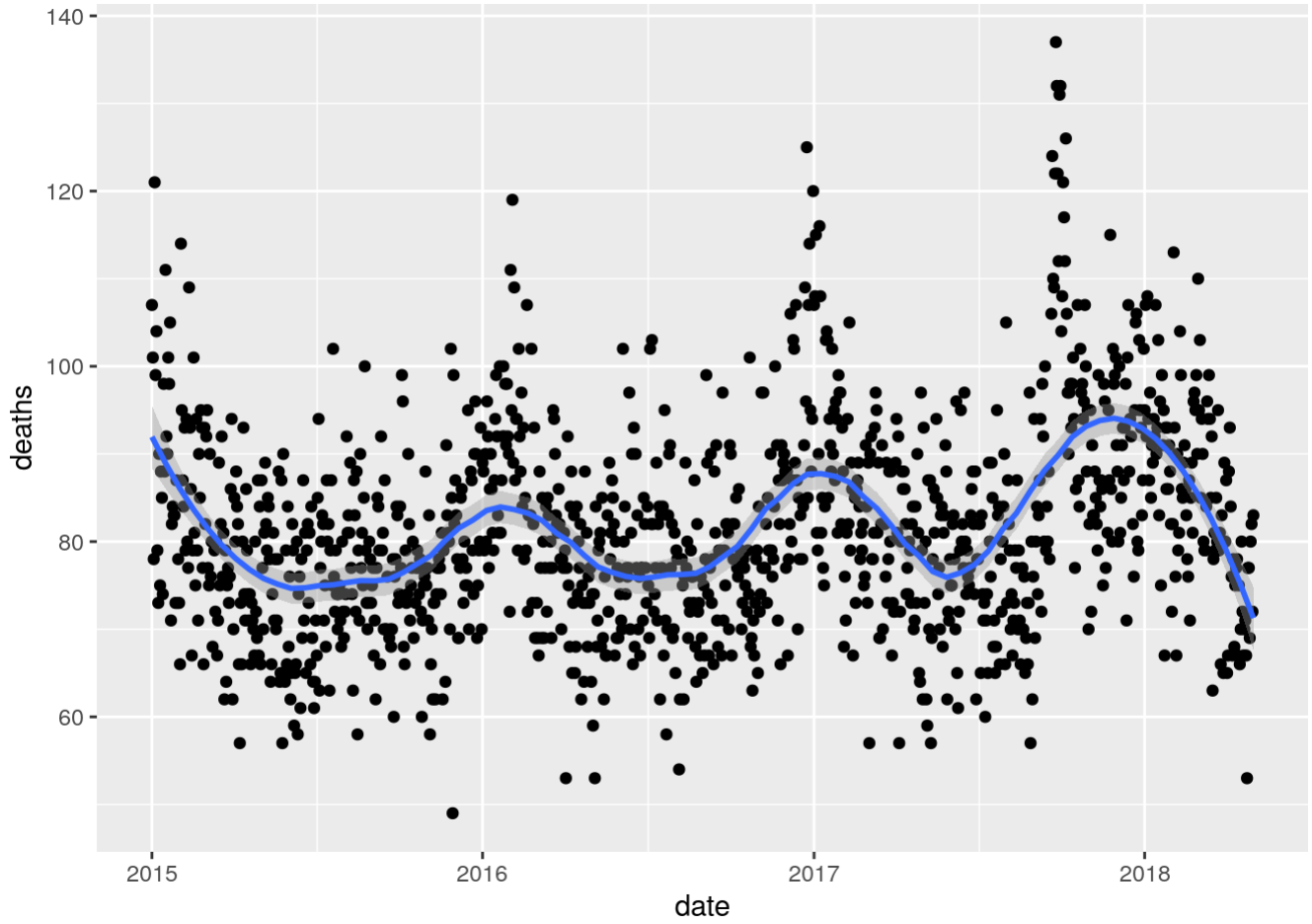
```
## Your code here
puerto_rico %>% ggplot(aes(x = date, y = deaths)) + geom_point() + geom_smooth(method = "loess",
span = 0.1)
```




```
puerto_rico %>% ggplot(aes(x = date, y = deaths)) + geom_point() + geom_smooth(method = "loess",
span = 0.2)
```




```
puerto_rico %>% ggplot(aes(x = date, y = deaths)) + geom_point() + geom_smooth(method = "loess",
span = 0.3)
```



Answer: This better span (to my eyes) of 0.3 shows a more predicted trend. It shows the number of deaths over time fluctuating and spiking at around the beginning of the year. It also takes on a rather sinusoidal shape which matches the data a lot better too, demonstrating how it continuously increases and decreases over time.

Question 3 (2 pts)

Use the `loess.as` function in the package `fANCOVA` to find a good `span` by cross-validation. How does this choice of `span` perform visually?

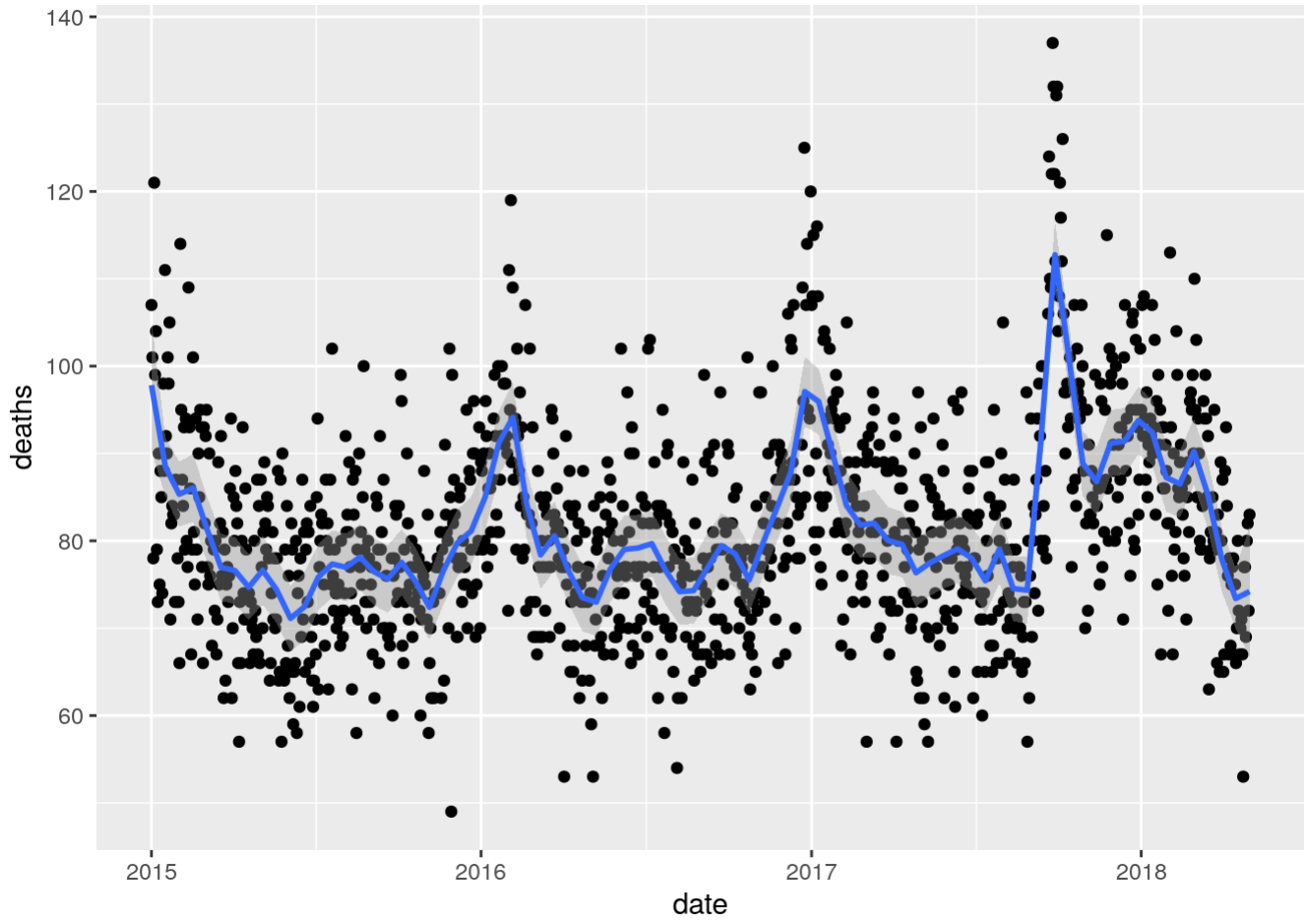
Hint: After fitting a `loess` using `loess.as`, we can see the best `span` by running `print(my_loess_fit$par)`.

```
library(fANCOVA)
y <- puerto_rico$deaths
x <- puerto_rico$date %>% as.numeric()

## Your code here
my_loess_fit <- loess.as(x,y)
print(my_loess_fit$par$span)
```

```
## [1] 0.0534068
```

```
puerto_rico %>% ggplot(aes(x = date, y = deaths)) + geom_point() + geom_smooth(method = "loess",
span = my_loess_fit$par$span)
```

```
## `geom_smooth()` using formula 'y ~ x'
```

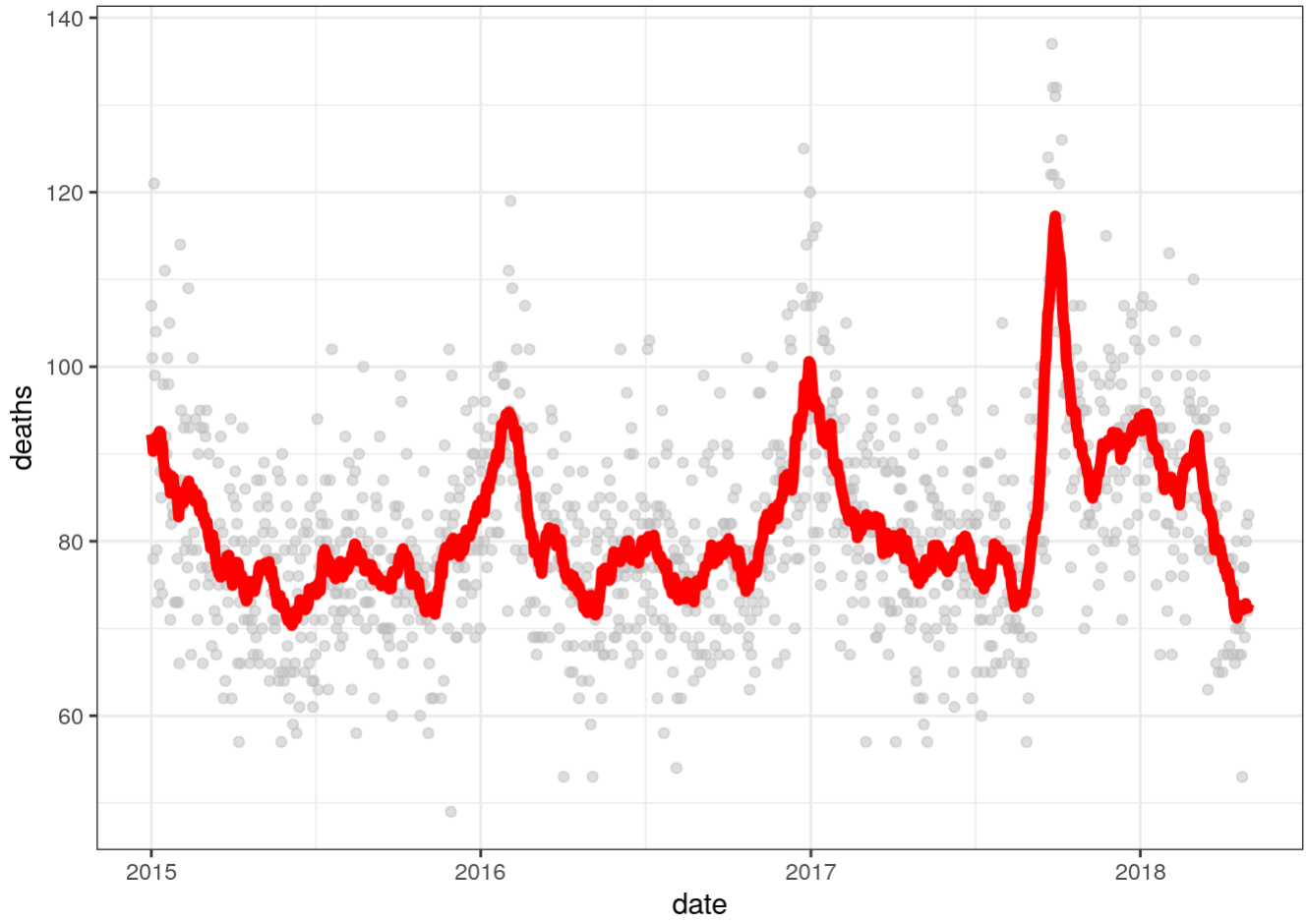


Answer: When using the `loess.as` function, the good `span` by cross-validation was 0.0534068. While this may be a good span for approximating these values given the data, it does not look very good visually despite matching the trend of the data/capturing the peaks of the data well. It looks really messy and is likely just overfitting the data.

Question 4 (2 pts)

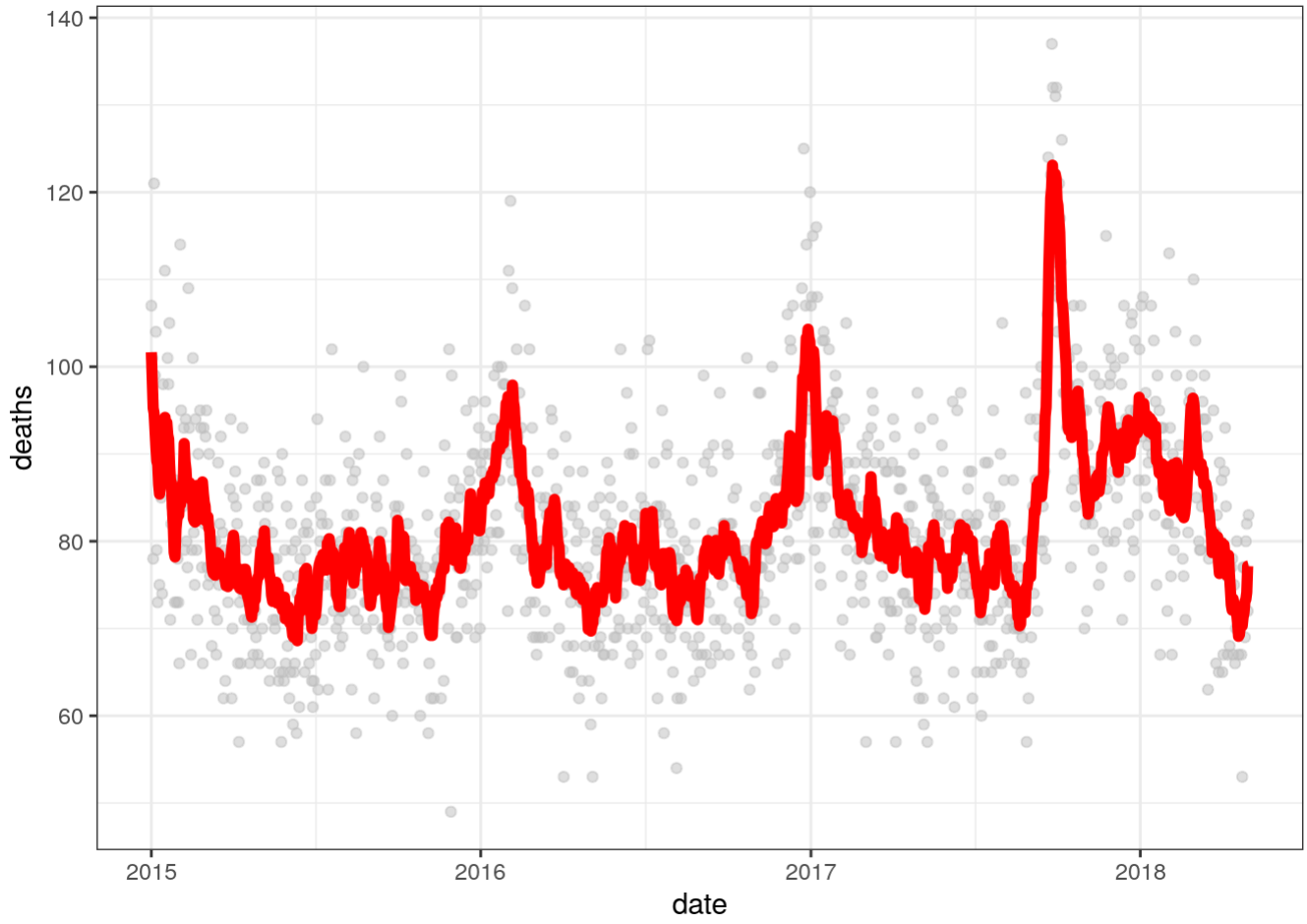
Code for getting a moving average with a bandwidth of 20 days is given below:

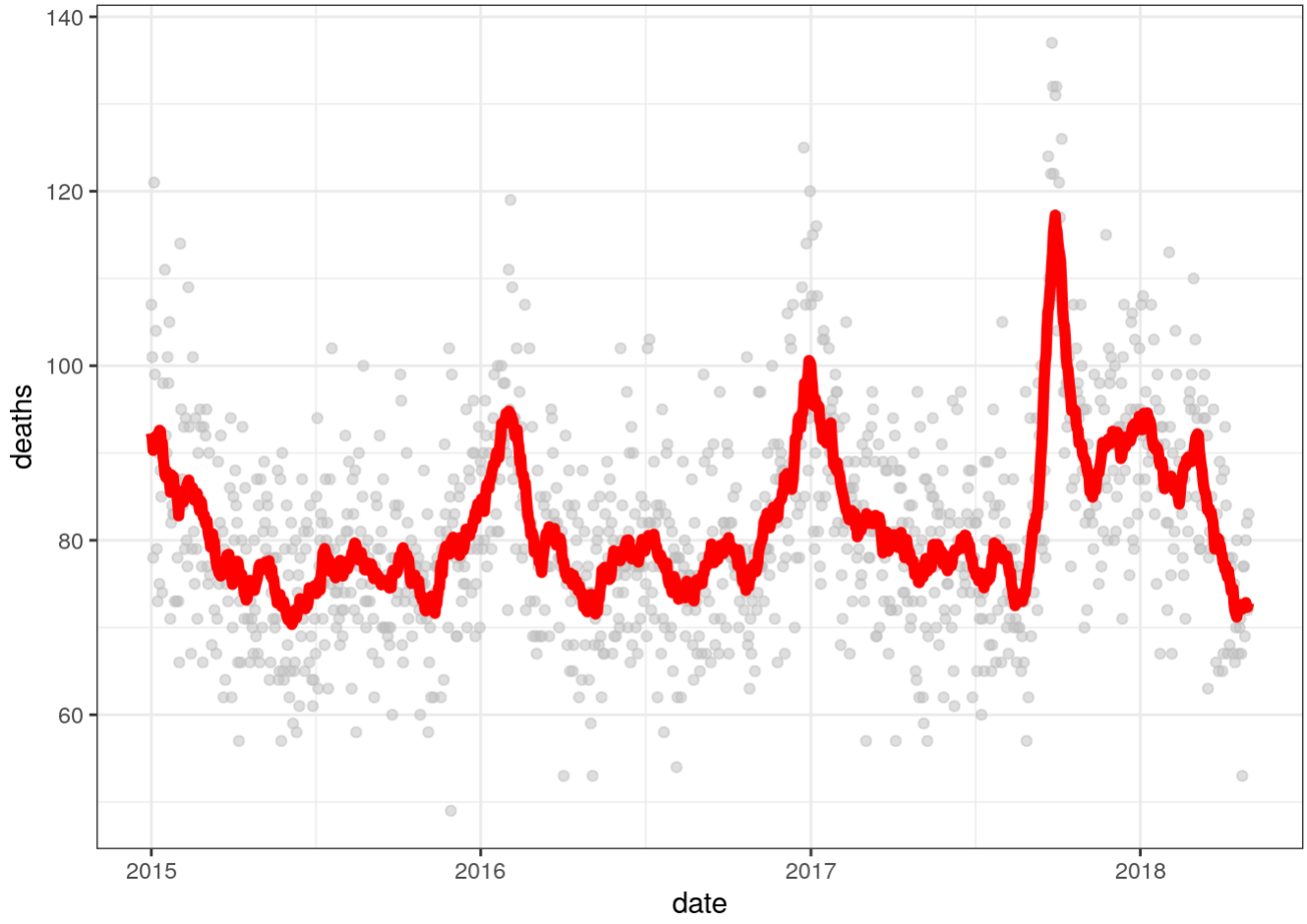
```
puerto_moving <- ksmooth(x = puerto_rico$date,
                         y = puerto_rico$deaths,
                         kernel = "box",
                         bandwidth = 20)

puerto_smooth <- data.frame(x = puerto_moving$x, y = puerto_moving$y)

ggplot(puerto_rico, aes(x = date, y = deaths)) +
  geom_point(color = 'gray', alpha = 0.5) +
  geom_line(aes(x = x, y = y),
            data = puerto_smooth,
            color = "red",
            size = 2) +
  theme_bw()
```

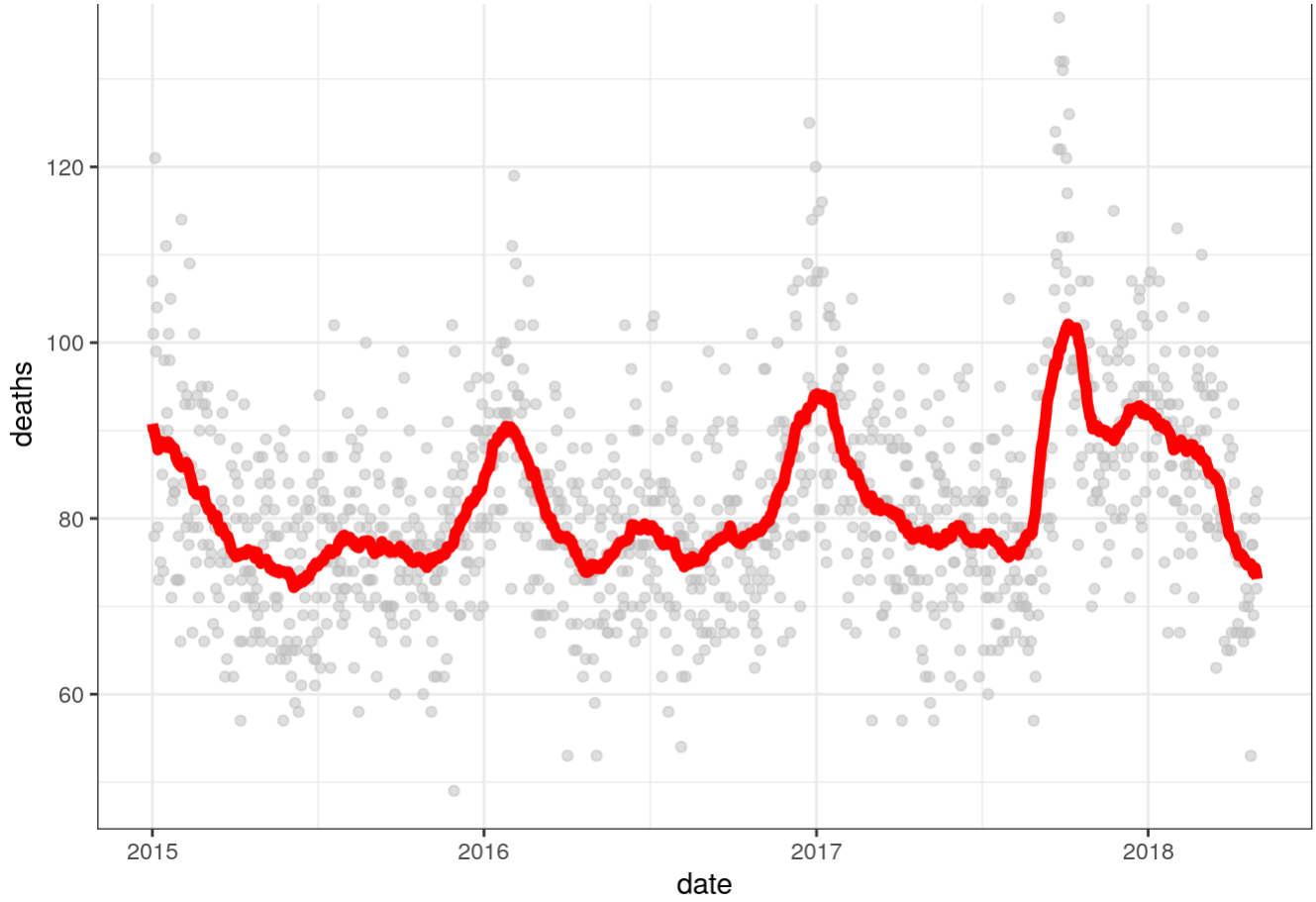


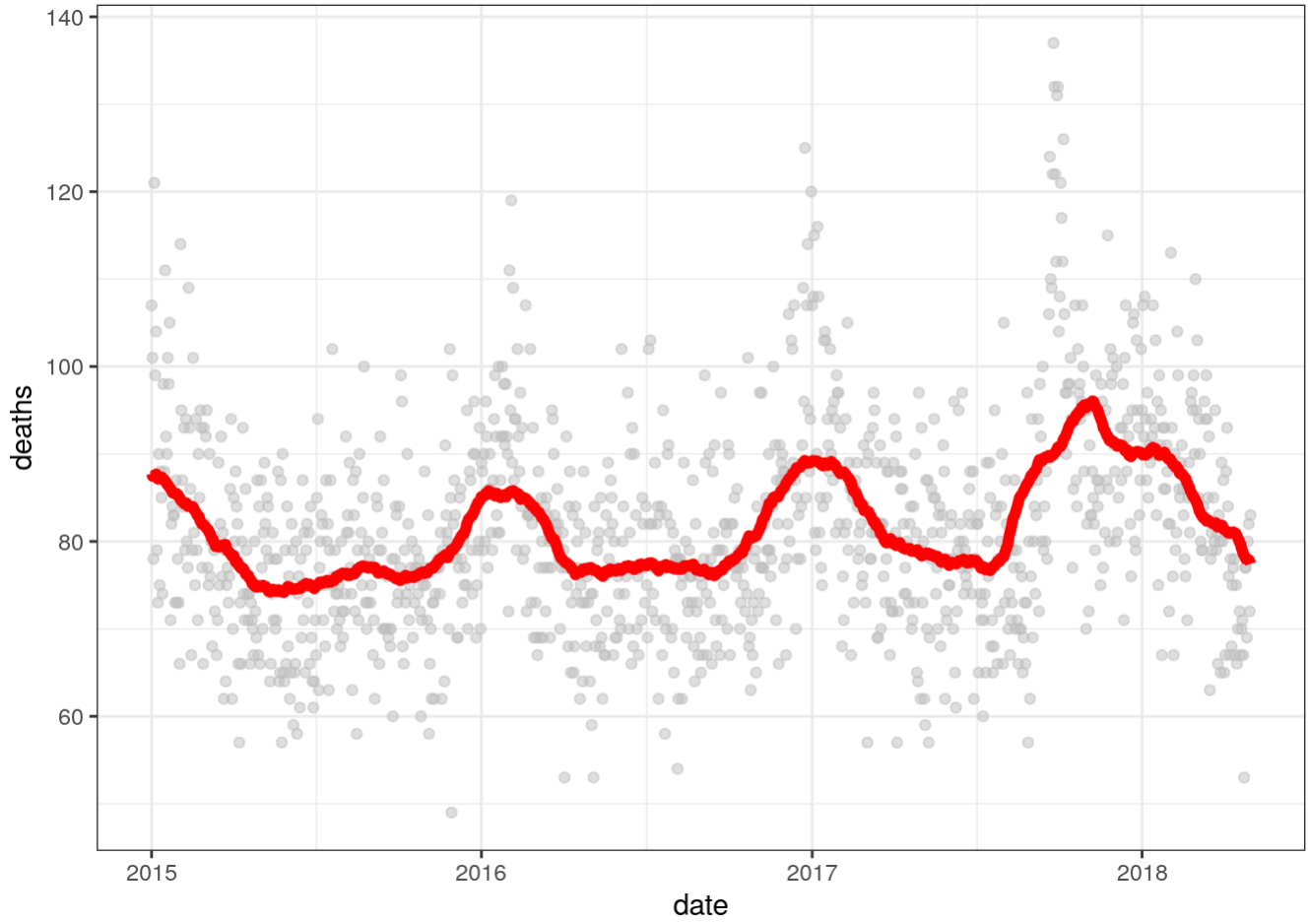
Try out the bandwidths 10, 20, 50, and 100. Which bandwidths seem to do the best job of capturing the peaks in the data? Which produces the smoothest estimates? Which do you prefer best visually?

```
## Your code here
bandwidths <- c(10, 20, 50, 100)
for (band in bandwidths){
  puerto_moving <- ksmooth(x = puerto_rico$date,
                           y = puerto_rico$deaths,
                           kernel = "box",
                           bandwidth = band)

  puerto_smooth <- data.frame(x = puerto_moving$x, y = puerto_moving$y)

  print(ggplot(puerto_rico, aes(x = date, y = deaths)) +
    geom_point(color = 'gray', alpha = 0.5) +
    geom_line(aes(x = x, y = y),
              data = puerto_smooth,
              color = "red",
              size = 2) +
    theme_bw())
}
```


140
120
100
80
60
deaths
2015
2016
2017
2018
date


140
120
100
80
60
deaths
2015
2016
2017
2018
date


deaths
120
100
80
60
2015
2016
2017
2018
date


deaths
140
120
100
80
60
2015
2016
2017
2018
date

Answer: Based on the produced plots, a bandwidth of 10 is best at capturing the peaks in the data as it contains the most peaks in great detail and looks a lot more precise, a bandwidth of 100 produces the smoothest estimates as it contains smoother transitions between values in the function (it appears to have less steep/dynamic change in deaths per small changes in time), and a bandwidth of 20 looks the best to me visually as it combines the precise capture of peaks in the data with the smoother transitions.

Question 5 (2 pts)

The following code divides the dataset into a training set and a testing set.

```
set.seed(112112)
train_idx <- createDataPartition(puerto_rico$deaths, p = 0.75)[[1]]
puerto_train <- puerto_rico[train_idx, ]
puerto_test <- puerto_rico[-train_idx,]
```

Using the value of `span` from your favorite `loess` fit and the value of `bw` from your favorite `ksmooth` fit (with `kernel = "box"`), fit (i) a `loess` and (ii) moving average to `puerto_train` and predict the number of deaths on `puerto_test`. Using the function below

```
rmse <- function(x,y) sqrt(mean((x-y)^2))
```

find the RMSE on the test set for the two models. **Which model performs better: your favorite `loess` or your favorite `ksmooth`?**

Hint: there is no predict function for `ksmooth`, but you can choose the points to get predictions at by setting `x.points = as.numeric(puerto_test$date)`. Then the `x` and `y` of the output will be for the test data.

```
## Your code here
loess_fit <- loess(deaths ~ as.numeric(date), data = puerto_train, span = 0.3, degree = 1)
ksmooth_fit <- ksmooth(as.numeric(puerto_train$date), puerto_train$deaths, bandwidth = 20, kerne
l = "box", x.points = as.numeric(puerto_test$date))

loess_pred <- predict(loess_fit, puerto_test)
ksmooth_pred <- ksmooth_fit$y

rmse(puerto_test$deaths, loess_pred)
```

```
## [1] 10.41932
```

```
rmse(puerto_test$deaths, ksmooth_pred)
```

```
## [1] 9.098103
```

Answer: Based on the produced RMSE values, it can be seen that my favorite `ksmooth` model performs better as it contains a smaller RMSE value than my favorite `loess` model (9.098103 < 10.41932).