Exploratory Time Series Analysis of COVID-19 in Rhode Island and the United States

Lin Zou

Question

How strict was the COVID-19 lockdown in Rhode Island compared to the overall COVID-19 lockdown in the United States? How did these similarities or differences in the strictness of lockdown affect the number of incidence cases of COVID-19 at the state level (Rhode Island) versus the national level (United States)? To answer these questions, we will perform an exploratory time series analysis of the incidence and stringency index.

Introduction

Over the past year, the novel coronavirus has proven to be difficult to control. We have seen many attempts to reduce the spread of COVID-19, including mandatory quarantine, social distancing, and remote learning. However, not all states had the same level of lockdown, which resulted in the expected consequences of dramatic increases in the incidence of the disease. Then, the hospitals experience a sudden shortage in beds and intensive care units to treat their patients. Could we have predicted these consequences if every state had similar lockdown requirements?

Methods

Library

```r
library(docstring)
library(dplyr)
library(kableExtra)
library(ggplot2)
library(ggpmisc)
library(zoo)
library(tidyr)
library(gridExtra)
```

Data Overview

```r
ri_inc <- read.csv("~/Desktop/PHP 2560/Midterm Project/RI Incidence.csv")
# Reformat the date for merging purposes
ri_inc$Date <- strptime(as.character(ri_inc$Date), "%m/%d/%Y")
ri_inc$Date <- format(ri_inc$Date, "%Y-%m-%d")
ri_inc_ts <- ri_inc %>%
  rename(RI_Incidence = Daily.number.of.positive.tests..may.count.people.more.than.once.) %>%
  select(Date, RI_Incidence)
str(ri_inc_ts)
```

```
## 'data.frame':    242 obs. of  2 variables:
##  $ Date        : chr  "2020-02-27" "2020-02-28" "2020-02-29" "2020-03-01" ...
##  $ RI_Incidence: int  0 0 0 2 0 0 0 0 1 0 ...
```

The code above reads in the COVID-19 data from the Rhode Island Department of Health (RIDOH). RIDOH has collected data since 2/27/2020 and has observations up to 10/25/2020. We are particularly interested in the incidence data from this database.

```
ri_stri <- read.csv("https://raw.githubusercontent.com/OxCGRT/covid-policy-tracker/master/data/OxCGRT_la
ri_stri <- ri_stri %>%
  filter(RegionName == "Rhode Island")
# Reformat the date for merging purposes
ri_stri$Date <- strptime(as.character(ri_stri$Date), "%Y%m%d")
ri_stri_ts <- ri_stri %>%
  rename(RI_Stringency_Index = StringencyIndex) %>%
  select(Date, RI_Stringency_Index)
str(ri_stri_ts)
```

```
## 'data.frame':    327 obs. of  2 variables:
##  $ Date               : POSIXlt, format: "2020-01-01" "2020-01-02" ...
##  $ RI_Stringency_Index: num  0 0 0 0 0 0 0 0 0 0 ...
```

The code above reads in the COVID-19 data from the University of Oxford's Oxford COVID-19 Government Response Tracker. This database has collected data since 1/1/2020 and has observations up to 11/4/2020. We are particularly interested in the stringency index data for Rhode Island from this database.

```
ri_ts <- merge(ri_inc_ts, ri_stri_ts, by = "Date")
str(ri_ts)
```

```
## 'data.frame':    242 obs. of  3 variables:
##  $ Date               : chr  "2020-02-27" "2020-02-28" "2020-02-29" "2020-03-01" ...
##  $ RI_Incidence       : int  0 0 0 2 0 0 0 0 1 0 ...
##  $ RI_Stringency_Index: num  16.7 16.7 16.7 16.7 19.4 ...
```

The code above merges the Rhode Island incidence and stringency index data by date.

```
us_data <- read.csv("~/Desktop/PHP 2560/Midterm Project/US Incidence and Stringency.csv")
us_data <- us_data %>%
  filter(location == "United States")
# Reformat the date for merging purposes
us_data$date <- strptime(as.character(us_data$date), "%Y-%m-%d")
us_data_ts <- us_data %>%
  rename(Date = date, US_Incidence = new_cases, US_Stringency_Index = stringency_index) %>%
  select(Date, US_Incidence, US_Stringency_Index)
str(us_data_ts)
```

```
## 'data.frame':    301 obs. of  3 variables:
##  $ Date               : POSIXlt, format: "2019-12-31" "2020-01-01" ...
##  $ US_Incidence       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ US_Stringency_Index: num  NA 0 0 0 0 0 0 0 0 0 ...
```

The code above reads in COVID-19 data from *Our World in Data*. This database has collected data since 12/31/2019 and has observations up to 10/26/2020. We are particularly interested in the incidence and stringency index data for the United States from this database.

```
ts_data <- merge(ri_ts, us_data_ts, by = "Date")
str(ts_data)
```

```
## 'data.frame':    242 obs. of  5 variables:
##  $ Date                : chr  "2020-02-27" "2020-02-28" "2020-02-29" "2020-03-01" ...
##  $ RI_Incidence        : int  0 0 0 2 0 0 0 0 1 0 ...
##  $ RI_Stringency_Index: num  16.7 16.7 16.7 16.7 19.4 ...
##  $ US_Incidence        : num  6 1 6 3 20 14 22 34 74 105 ...
##  $ US_Stringency_Index: num  5.56 5.56 5.56 8.33 11.11 ...
```

```
# Convert the Date columns to POSIXct
ri_ts$Date <- as.POSIXct(ri_ts$Date)
ts_data$Date <- as.POSIXct(ts_data$Date)
```

The code above merges the Rhode Island and United States incidence and stringency index data by date.

Data Exploration

```
summary_stat <- function(data){
  #' @title Data Exploration for COVID-19 Time Series Data
  #'
  #' @description Returns a table with the stringency index, number of days the stringency
  #' index was implemented, and the average number, standard deviation, and 95% confidence
  #' interval of new cases that was observed over these days
  #'
  #' @param data data frame containing daily incidence and stringency index observations
  #' @return a table with the stringency index, number of days the stringency index was
  #' implemented, and the average number of new cases that was observed over these days

  ##### Stop Functions #####
  if(!(is.data.frame(data))){
    stop("data must be a data frame")
  }

  ##### Summary Table Construction #####
  data %>%
    na.omit %>%
    # Rename the columns by looking for keywords to ignore difference in location
    rename(inc = sym(colnames(data)[grepl("Incidence", colnames(data))]),
           str_ind = sym(colnames(data)[grepl("Stringency", colnames(data))])) %>%
    mutate(`Stringency Index` = case_when((str_ind >= 0 & str_ind < 25) ~ "0-25",
                                          (str_ind >= 25 & str_ind < 50) ~ "25-50",
                                          (str_ind >= 50 & str_ind < 75) ~ "50-75",
                                          (str_ind >= 75 & str_ind <= 100) ~ "75-100")) %>%
    group_by(`Stringency Index`) %>%
    summarise(`Number of Days` = n(),
              `Average Number of New Cases` = round(mean(inc)),
              `SD of New Cases` = round(sd(inc)),
              `95% CI` = paste("(",
                              round(mean(inc)-qt(0.975, n()-1)*sd(inc)/sqrt(n()), digits=2),
                              round(mean(inc)+qt(0.975, n()-1)*sd(inc)/sqrt(n()), digits=2),
                                                  ")")) %>%
    kable(booktabs = TRUE)
}
```

The function above returns a table with the stringency index, number of days the stringency index was implemented, and the average number, standard deviation, and 95% confidence interval of new cases that was observed over these days.

```
# Summary statistics for Rhode Island
summary_stat(ri_ts)
```

Stringency Index	Number of Days	Average Number of New Cases	SD of New Cases	95% CI
0-25	14	0	1	(-0.01 0.87)
25-50	4	4	4	(-2.75 10.75)
50-75	146	141	92	(125.52 155.7)
75-100	69	291	134	(258.53 322.92)

From the table above, we can observe that although Rhode Island maintained a moderate to high stringency index over the majority of the days, the average number of new cases was still high. The stringency index was low and was not maintained when the average number of new cases was low.

```
# Summary statistics for United States
summary_stat(us_data_ts)
```

Stringency Index	Number of Days	Average Number of New Cases	SD of New Cases	95% CI
0-25	71	14	45	(3.78 25.09)
25-50	4	482	218	(134.03 828.97)
50-75	224	38276	17341	(35992.5 40559.05)

From the table above, we can observe that the United States had a low stringency index when the average number of new the cases was low, a moderate stringency index for only four days when there was a sudden spike in new cases, and has maintained a moderate to high stringency index, despite the large number of new cases.

From these two tables, it appears that in both the state level and national level, lockdown procedures are implemented in almost a "knee-jerk" fashion. That is, the stringency index only increases when the incidence is high; however, as the stringency index was not high before the increase in incidence, we see a rolling effect in the incidence as the test results come in even when a high stringency index is maintained.

Exploratory Time Series Analysis

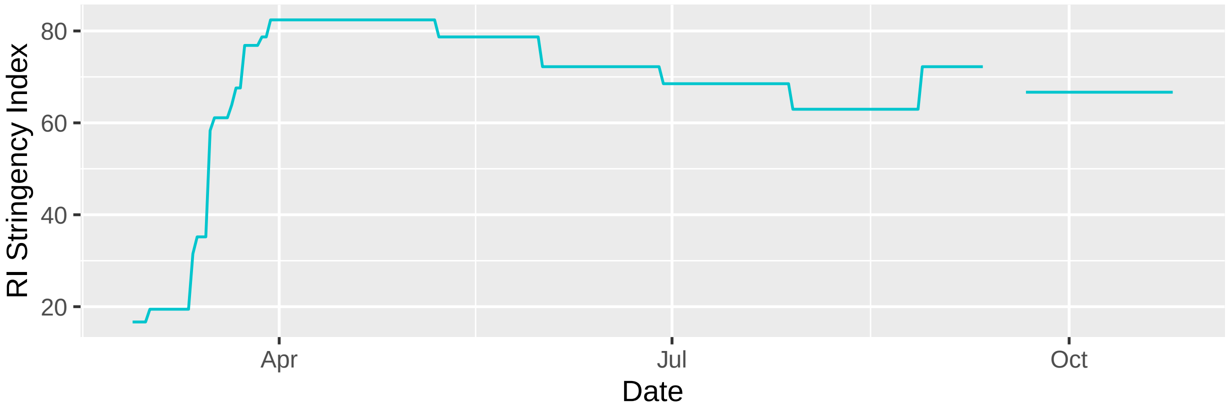
Perhaps comparing the peak incidence in Rhode Island with the peak incidence in the United States will reveal how the strictness of lockdown affect the number of incidence cases of COVID-19 at the state level (Rhode Island) versus the national level (United States). Since the data is noisy, let us first explore some smoothing methods for removing the noise. By doing this, we remove the high frequency behavior and analyze the low frequency behavior of the time series trends.

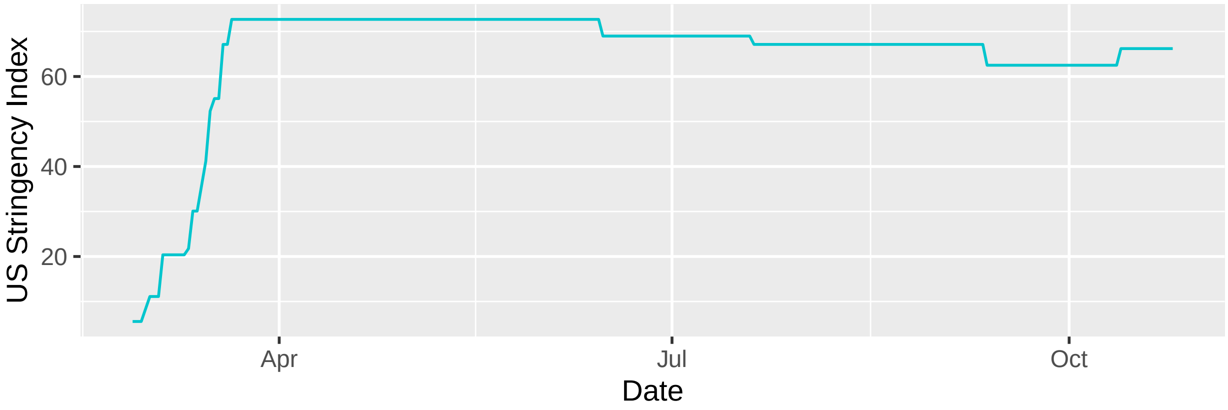
```
ts_covid <- function(data, date.name, data.name,
                     smoother = c("None", "ma", "tri_ma", "k_smooth"),
                     ma.window = 7, kernel = "normal", bandwidth = 14){
  #' @title Exploratory Time Series Analysis Using Smoothed Time Series Plots
  #'
  #' @description Returns a plot of the raw time series data with the corresponding
  #' smoothed time series. If smoother = "None" then only the plot of the raw time
  #' series data is returned.
  #'
  #' @param data data frame containing daily incidence and stringency index observations
  #' @param date.name name of the column containing dates
  #' @param data.name name of the column containing the data
  #' @param smoother the smoothing method to be used
  #' @param ma.window integer width of the rolling window. Corresponds to smoother = c("ma", "tri_ma").
  #' @param kernel the kernel to be used. Corresponds to smoother = "k_smooth"
  #' @param bandwidth the bandwidth to be used. The kernels are scaled so that their
  #' quartiles are at +/- 0.25*bandwidth
  #' @return a plot of the raw time series data with the corresponding smoothed time
  #' series. If smoother = "None" then only the plot of the raw time series data is returned.

  ##### Stop Functions #####
  if(!(is.data.frame(data))){
    stop("data must be a data frame")
  } else if(!(is.character(date.name) & is.character(data.name))){
    stop("date.name and data.name must be type character")
  } else if(!(is.character(smoother))){
    stop("smoother must be type character")
  } else if(!(is.numeric(ma.window))){
    stop("ma.window must be an integer")
  } else if(!(is.character(kernel))){
    stop("kernel must be type character")
  } else if(!(is.numeric(bandwidth))){
    stop("bandwidth must be type numeric")
  }

  ##### Control flow for smoother #####
  if(smoother == "None"){
    ts_plot <- ggplot(data) +
      geom_line(aes_string(x=date.name,y=data.name,group=1), color = "turquoise3") +
      xlab("Date") + ylab(gsub("_", " ", data.name))
  } else if(smoother == "ma"){
    data <- data %>%
      select(sym(date.name), sym(data.name)) %>%
      rename(date.name = sym(date.name), data.name = sym(data.name)) %>%
      mutate(ma = rollmean(data.name, k = ma.window, fill = NA))

    ts_plot <- data %>%
      gather(Metric, Value, data.name:ma) %>%
      ggplot(aes(date.name, Value, color = Metric)) +
      geom_line() +
      xlab("Date") + ylab(gsub("_", " ", data.name)) +
      scale_color_manual(name = "Data",
                         labels=c("Raw", paste(ma.window, "Day MA")),
                         values = c("turquoise3", "tomato"))
  } else if(smoother == "tri_ma"){
    data <- data %>%
      select(sym(date.name), sym(data.name)) %>%
      rename(date.name = sym(date.name), data.name = sym(data.name)) %>%
      mutate(ma = rollmean(rollmean(data.name, k = ma.window, fill = NA), k = ma.window, fill = NA))

    ts_plot <- data %>%
      gather(Metric, Value, data.name:ma) %>%
      ggplot(aes(date.name, Value, color = Metric)) +
      geom_line() +
      xlab("Date") + ylab(gsub("_", " ", data.name)) +
      scale_color_manual(name = "Data",
                         labels=c("Raw", paste(ma.window, "Day Triangular MA")),
                         values = c("turquoise3", "tomato"))
  } else if(smoother == "k_smooth"){
    data <- data %>%
      select(sym(date.name), sym(data.name)) %>%
      rename(date.name = sym(date.name), data.name = sym(data.name)) %>%
      mutate(ks = ksmooth(time(data.name), data.name, kernel, bandwidth)$y)


    ts_plot <- data %>%
      gather(Metric, Value, data.name:ks) %>%
      ggplot(aes(date.name, Value, color = Metric)) +
      geom_line() +
      xlab("Date") + ylab(gsub("_", " ", data.name)) +
      scale_color_manual(name = "Data",
                         labels=c("Raw", paste(kernel, "K-Smooth,", "B =", bandwidth)),
                         values = c("turquoise3", "tomato"))
  }

  return (ts_plot)
}
```

The function above returns a plot of the raw time series data with the corresponding smoothed time series.

Stringency Index

```
ri <- ts_covid(ts_data, "Date", "RI_Stringency_Index")
us <- ts_covid(ts_data, "Date", "US_Stringency_Index")
grid.arrange(ri, us)
```

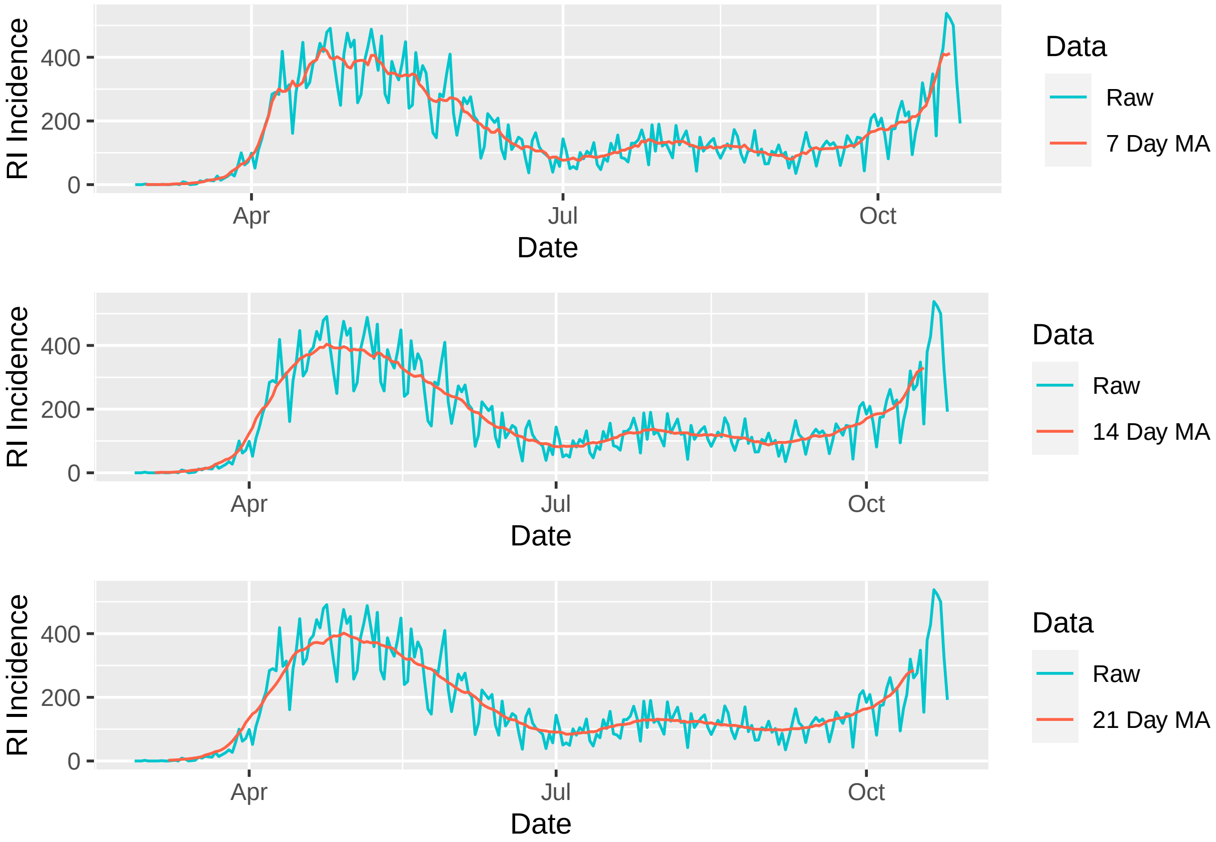




From the plot above, we can observe that the stringency index of Rhode Island and United States is similar for all time points.

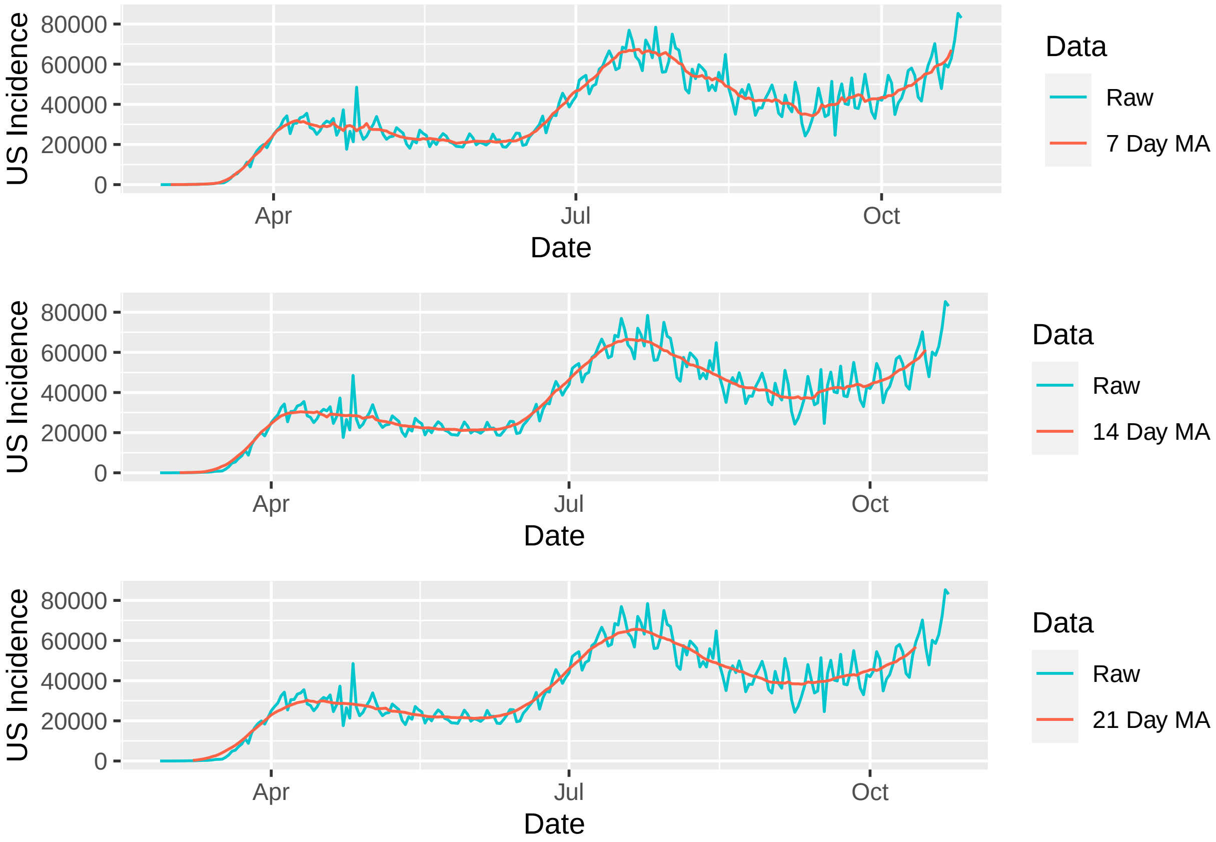
Incidence

```
ma7 <- ts_covid(ts_data, "Date", "RI_Incidence", "ma", ma.window = 7)
ma14 <- ts_covid(ts_data, "Date", "RI_Incidence", "ma", ma.window = 14)
ma21 <- ts_covid(ts_data, "Date", "RI_Incidence", "ma", ma.window = 21)
grid.arrange(ma7, ma14, ma21)
```



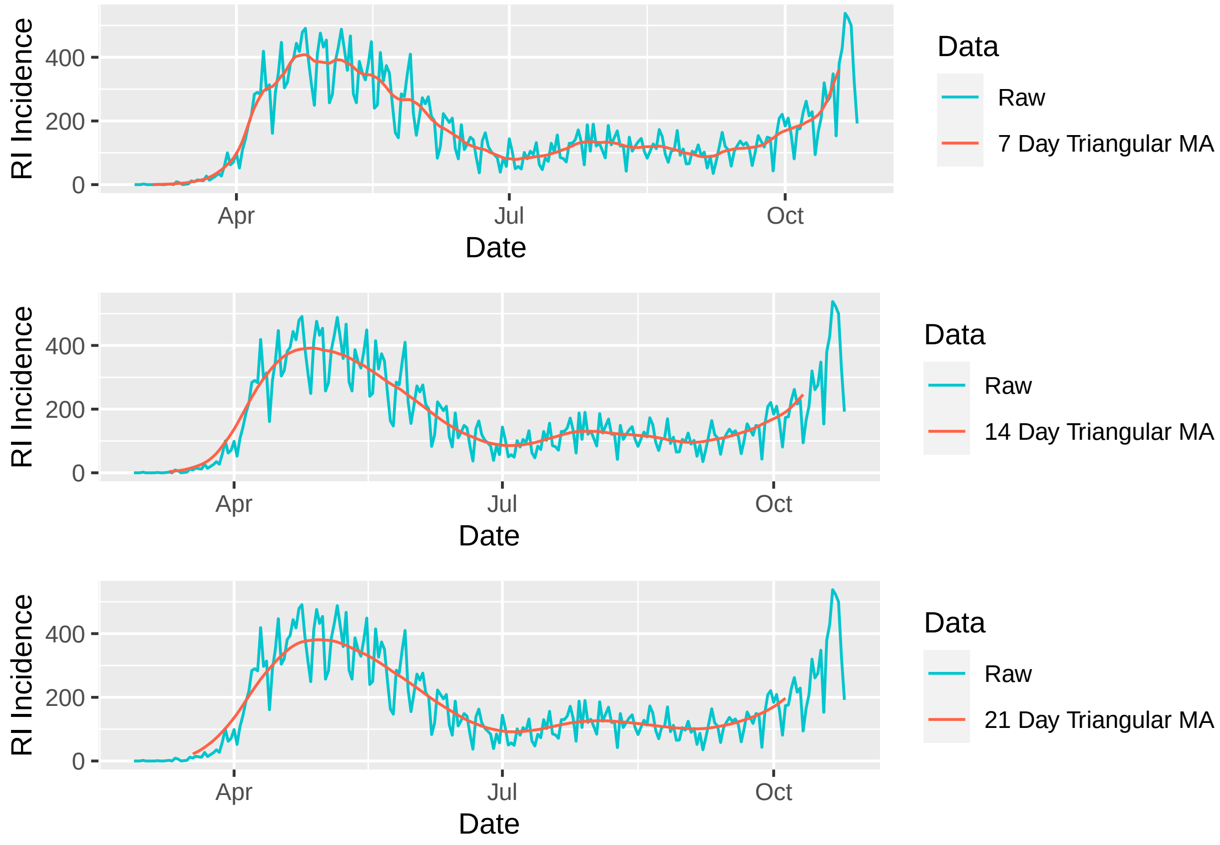
From the plot above, the 21 day moving average provides the smoothest time series, as expected.

```r
ma7 <- ts_covid(ts_data, "Date", "US_Incidence", "ma", ma.window = 7)
ma14 <- ts_covid(ts_data, "Date", "US_Incidence", "ma", ma.window = 14)
ma21 <- ts_covid(ts_data, "Date", "US_Incidence", "ma", ma.window = 21)
grid.arrange(ma7, ma14, ma21)
```



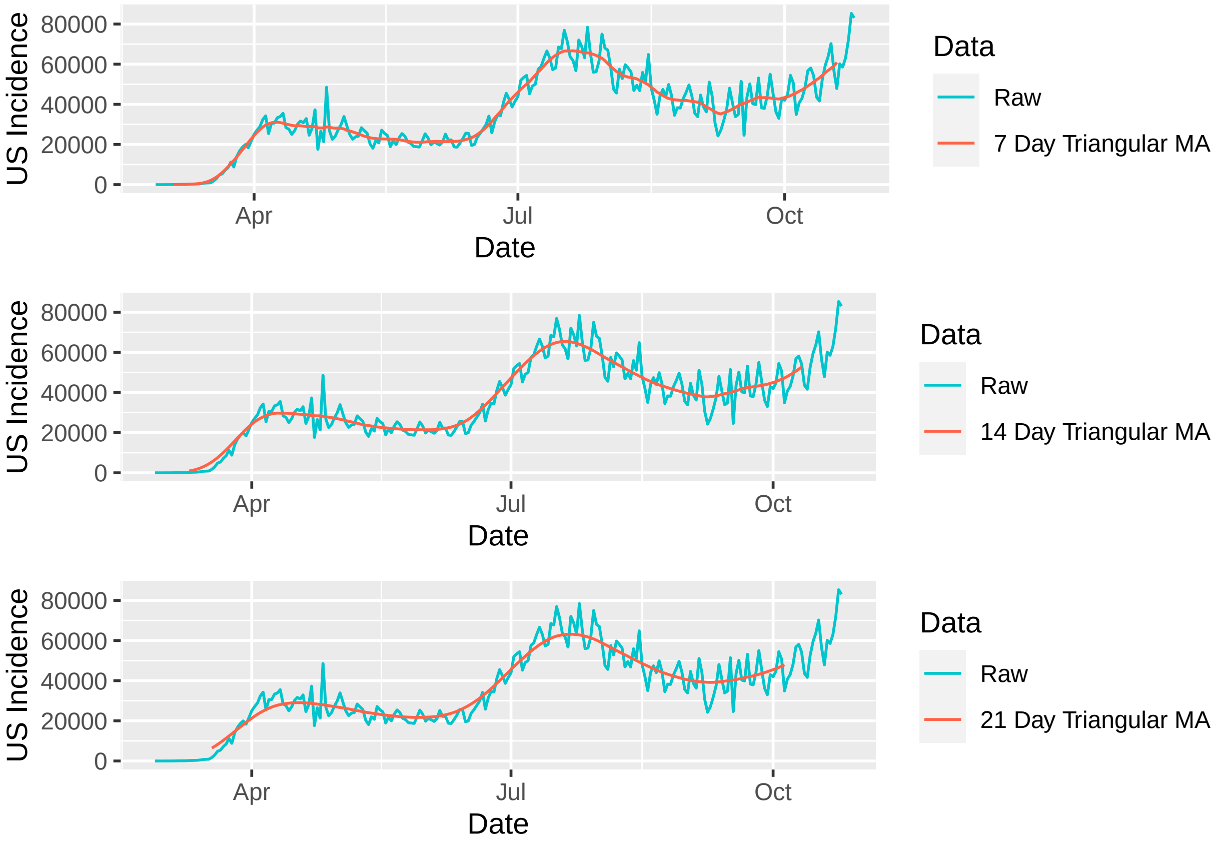
From the plot above, the 21 day moving average provides the smoothest time series, as expected.

```
ma7 <- ts_covid(ts_data, "Date", "RI_Incidence", "tri_ma", ma.window = 7)
ma14 <- ts_covid(ts_data, "Date", "RI_Incidence", "tri_ma", ma.window = 14)
ma21 <- ts_covid(ts_data, "Date", "RI_Incidence", "tri_ma", ma.window = 21)
grid.arrange(ma7, ma14, ma21)
```



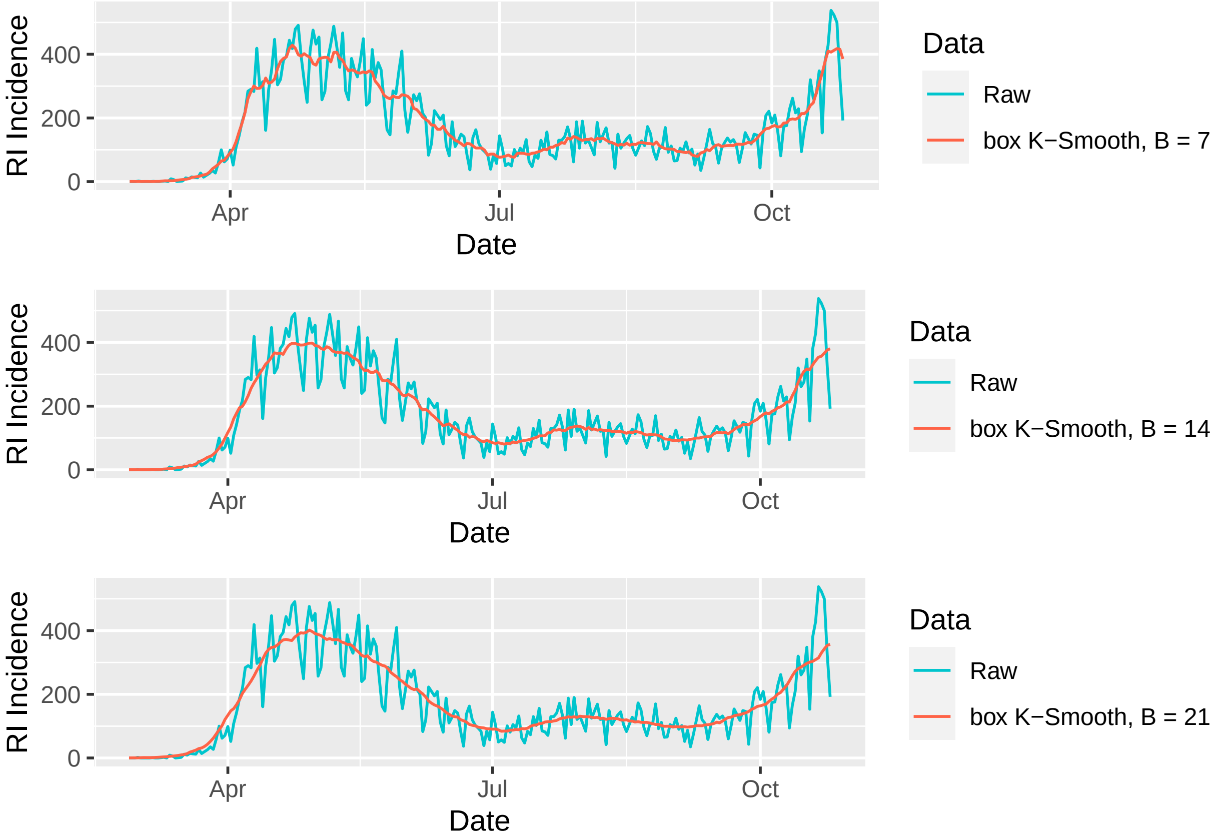
From the plot above, the 14 day and 21 day triangular moving averages provide similar smoothness.

```r
ma7 <- ts_covid(ts_data, "Date", "US_Incidence", "tri_ma", ma.window = 7)
ma14 <- ts_covid(ts_data, "Date", "US_Incidence", "tri_ma", ma.window = 14)
ma21 <- ts_covid(ts_data, "Date", "US_Incidence", "tri_ma", ma.window = 21)
grid.arrange(ma7, ma14, ma21)
```



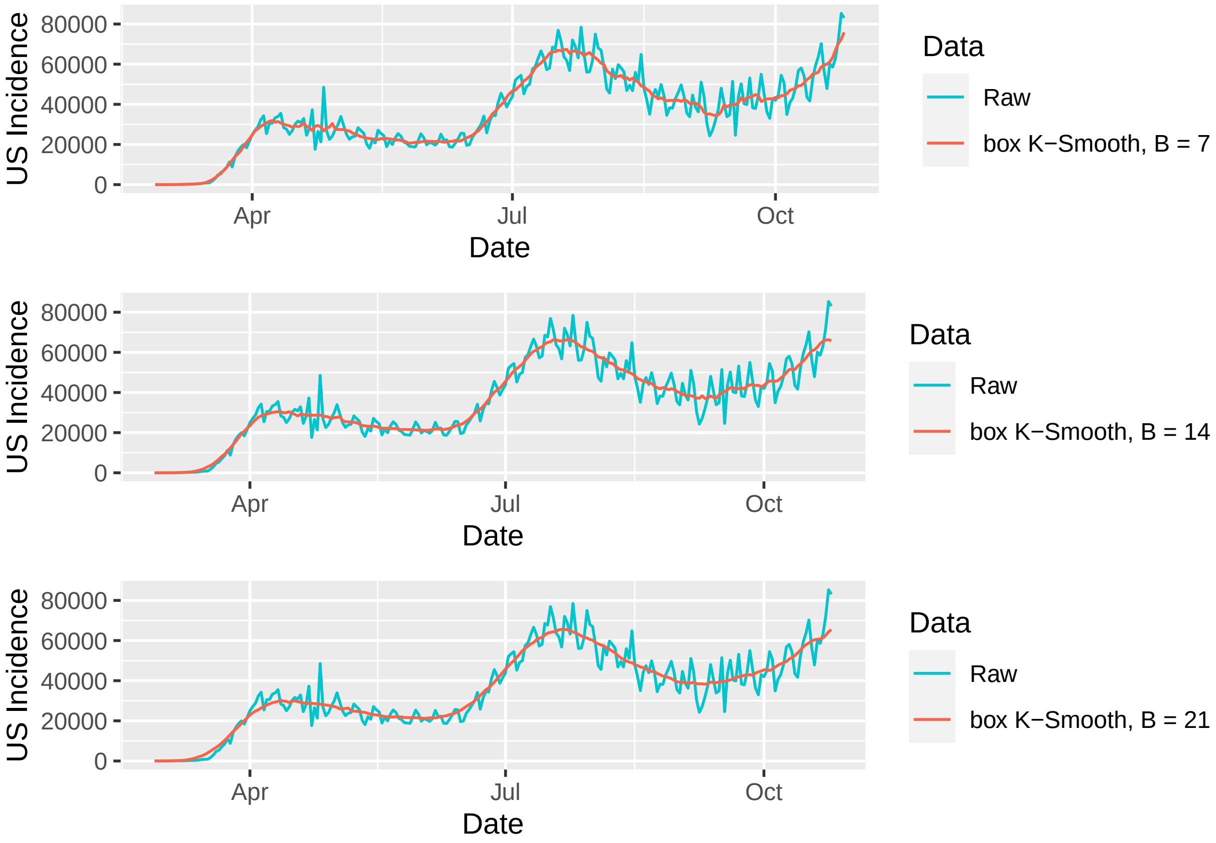
From the plot above, the 14 day and 21 day triangular moving averages provide similar smoothness.

```r
bw7 <- ts_covid(ts_data, "Date", "RI_Incidence", "k_smooth",
                kernel = "box", bandwidth = 7)
bw14 <- ts_covid(ts_data, "Date", "RI_Incidence", "k_smooth",
                 kernel = "box", bandwidth = 14)
bw21 <- ts_covid(ts_data, "Date", "RI_Incidence", "k_smooth",
                 kernel = "box", bandwidth = 21)
grid.arrange(bw7, bw14, bw21)
```



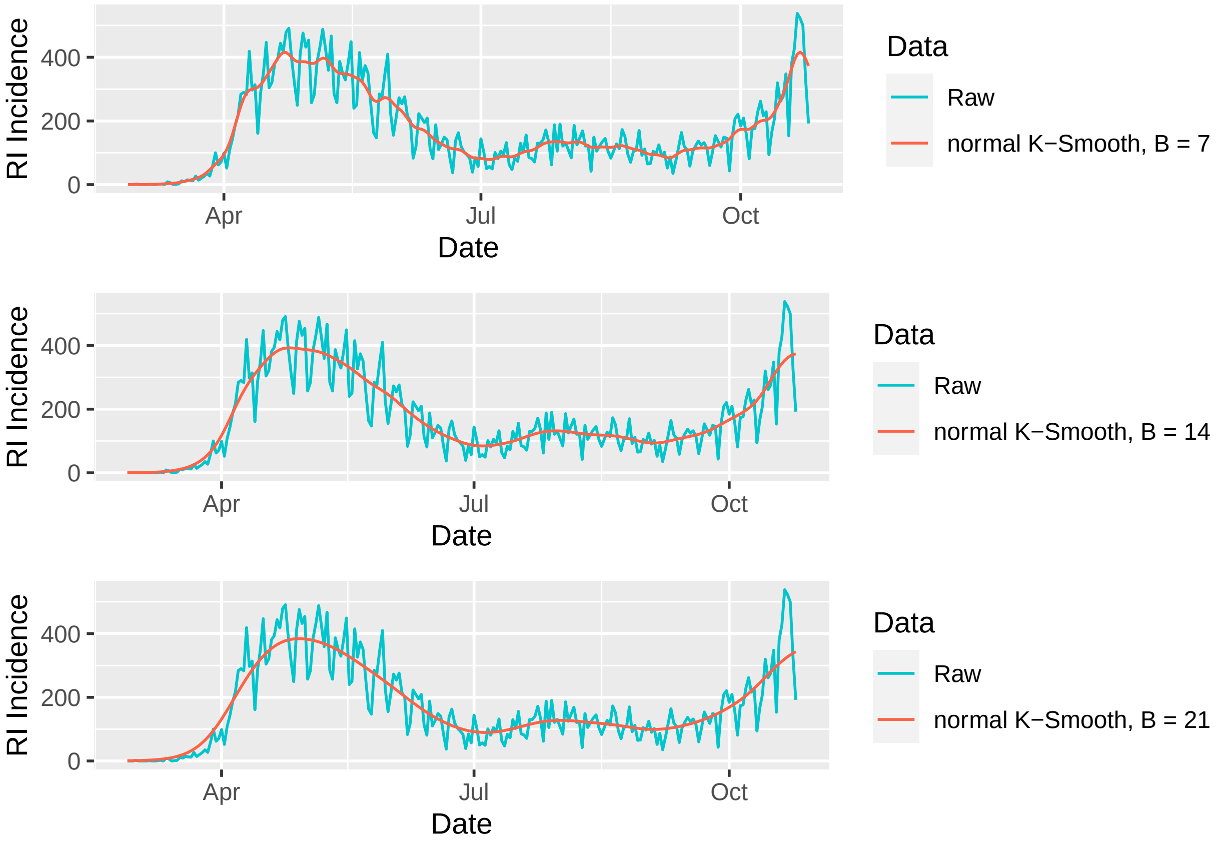
From the plot above, the box kernel smoothing with bandwidth = 21 provides the smoothest function.

```
bw7 <- ts_covid(ts_data, "Date", "US_Incidence", "k_smooth",
                kernel = "box", bandwidth = 7)
bw14 <- ts_covid(ts_data, "Date", "US_Incidence", "k_smooth",
                 kernel = "box", bandwidth = 14)
bw21 <- ts_covid(ts_data, "Date", "US_Incidence", "k_smooth",
                 kernel = "box", bandwidth = 21)
grid.arrange(bw7, bw14, bw21)
```



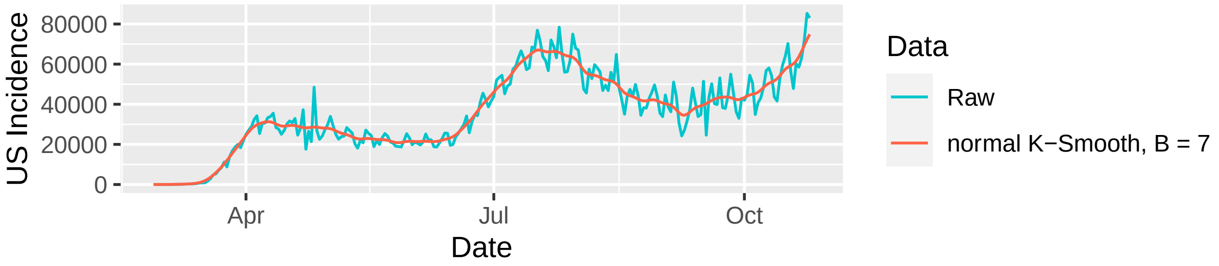
From the plot above, the box kernel smoothing with bandwidth = 21 provides the smoothest function.

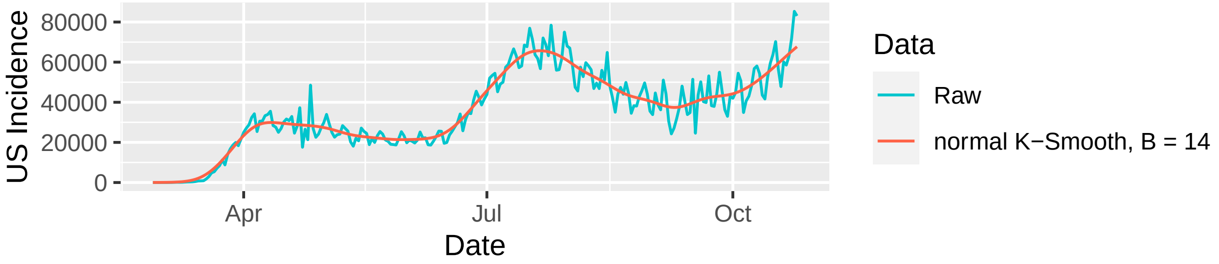
```r
bw7 <- ts_covid(ts_data, "Date", "RI_Incidence", "k_smooth",
                kernel = "normal", bandwidth = 7)
bw14 <- ts_covid(ts_data, "Date", "RI_Incidence", "k_smooth",
                 kernel = "normal", bandwidth = 14)
bw21 <- ts_covid(ts_data, "Date", "RI_Incidence", "k_smooth",
                 kernel = "normal", bandwidth = 21)
grid.arrange(bw7, bw14, bw21)
```

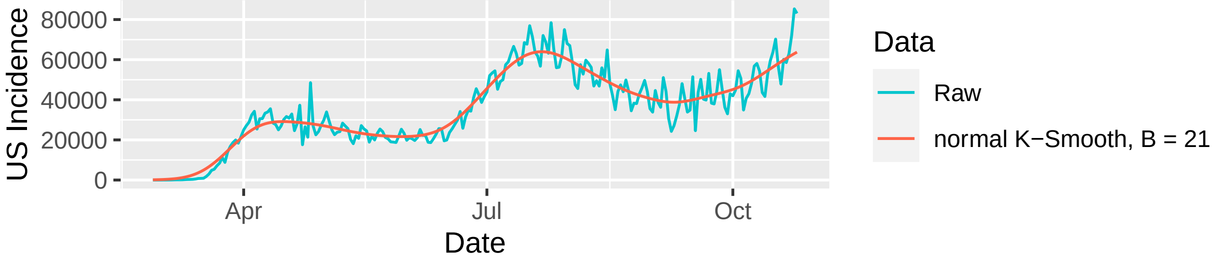


From the plot above, the gaussian kernal smoothing with bandwidth = 14 and bandwidth = 21 provide similar smoothness.

```
bw7 <- ts_covid(ts_data, "Date", "US_Incidence", "k_smooth",
                kernel = "normal", bandwidth = 7)
bw14 <- ts_covid(ts_data, "Date", "US_Incidence", "k_smooth",
                 kernel = "normal", bandwidth = 14)
bw21 <- ts_covid(ts_data, "Date", "US_Incidence", "k_smooth",
                 kernel = "normal", bandwidth = 21)
grid.arrange(bw7, bw14, bw21)
```







From the plot above, the gaussian kernal smoothing with bandwidth = 14 and bandwidth = 21 provide similar smoothness.

Peak detection

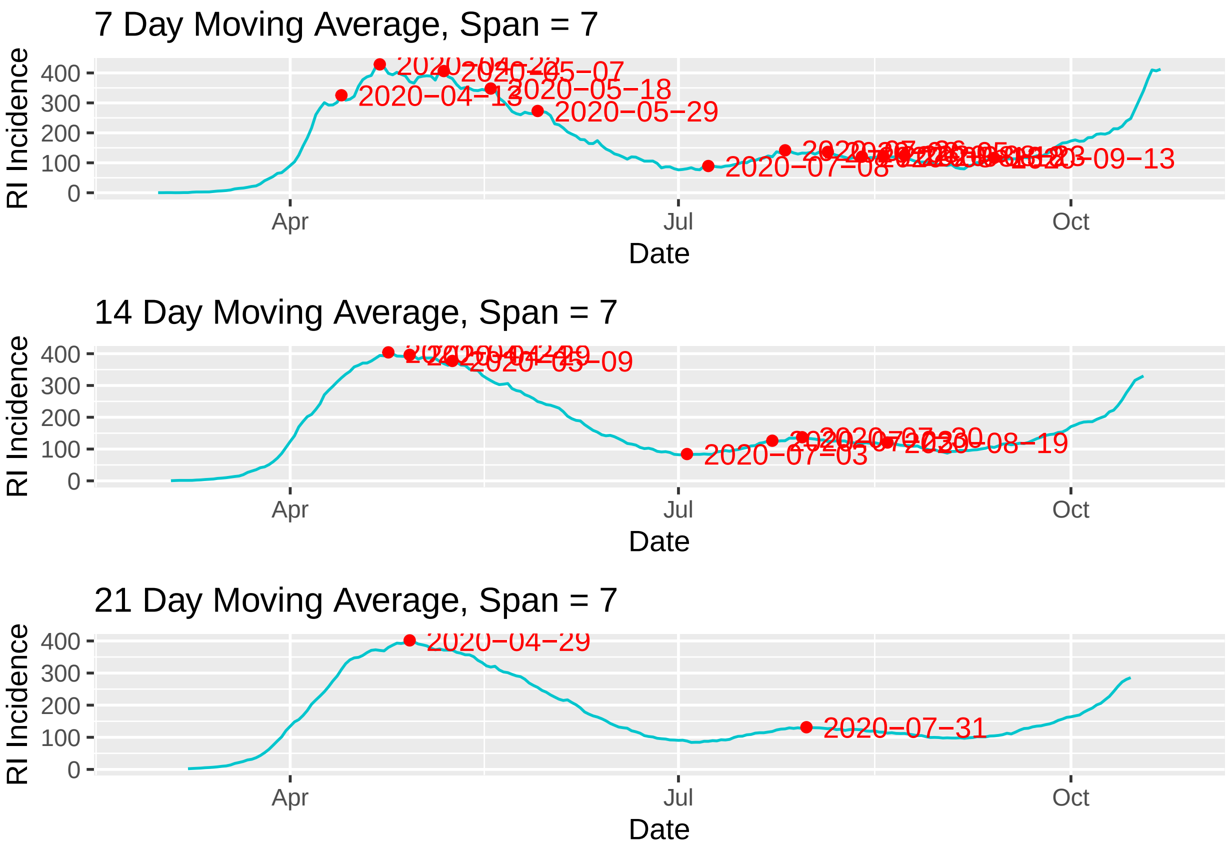
Now let us implement peak detection.

```
peak_detect <- function(data, date.name, data.name,
                        smoother = c("ma", "tri_ma", "k_smooth"),
                        ma.window = 7, kernel = "normal", bandwidth = 14,
                        span = 7){
  #' @title Peak Detection for Time Series Data
  #'
  #' @description Returns a smoothed time series with the detected peaks highlighted and labeled
  #'
  #' @param data data frame containing daily incidence and stringency index observations
  #' @param date.name name of the column containing dates
  #' @param data.name name of the column containing the data
  #' @param smoother the smoothing method to be used
  #' @param ma.window integer width of the rolling window. Corresponds to smoother = c("ma", "tri_ma").
  #' @param kernel the kernel to be used. Corresponds to smoother = "k_smooth"
  #' @param bandwidth the bandwidth to be used. The kernels are scaled so that their
  #'quartiles are at +/- 0.25*bandwidth
  #' @param span a peak is defined as an element in a sequence which is greater than all other
  #' elements within a window of width span centered at that element. The default value is 7
  #' meaning that a peak is greater than three consecutive neighbors on each side. A NULL value
  #' for span corresponds to a span covering the entire range of the data.
  #' @return a smoothed time series with the detected peaks highlighted and labeled

  ##### Stop Functions #####
  if(!(is.data.frame(data))){
    stop("data must be a data frame")
  } else if(!(is.character(date.name) & is.character(data.name))){
    stop("date.name and data.name must be type character")
  } else if(!(is.character(smoother))){
    stop("smoother must be type character")
  } else if(!(is.numeric(ma.window))){
    stop("ma.window must be an integer")
  } else if(!(is.character(kernel))){
    stop("kernel must be type character")
  } else if(!(is.numeric(bandwidth))){
    stop("bandwidth must be type numeric")
  } else if(!(is.numeric(span))){
    stop("span must be type numeric")
  }

  ##### Control flow for smoother #####
  if(smoother == "ma"){
    data <- data %>%
      select(sym(date.name), sym(data.name)) %>%
      rename(date.name = sym(date.name), data.name = sym(data.name)) %>%
      mutate(ma = rollmean(data.name, k = ma.window, fill = NA))

    ts_plot <- ggplot(data, aes(x=date.name, y=ma)) +
      geom_line(color = "turquoise3") +
      stat_peaks(color = "red", span = span) +
      stat_peaks(geom = "text", colour = "red", span = span,
                 hjust = -0.1, x.label.fmt = "%Y-%m-%d") +
      xlab("Date") + ylab(gsub("_", " ", data.name)) +
      ggtitle(paste(ma.window, "Day Moving Average, Span =", span))
  } else if(smoother == "tri_ma"){
    data <- data %>%
      select(sym(date.name), sym(data.name)) %>%
      rename(date.name = sym(date.name), data.name = sym(data.name)) %>%
      mutate(ma = rollmean(rollmean(data.name, k = ma.window, fill = NA), k = ma.window, fill = NA))

    ts_plot <- ggplot(data, aes(x=date.name, y=ma)) +
      geom_line(color = "turquoise3") +
      stat_peaks(color = "red", span = span) +
      stat_peaks(geom = "text", colour = "red", span = span,
                 hjust = -0.1, x.label.fmt = "%Y-%m-%d") +
      xlab("Date") + ylab(gsub("_", " ", data.name)) +
      ggtitle(paste(ma.window, "Day Triangular Moving Average, Span =", span))
  } else if(smoother == "k_smooth"){
    data <- data %>%
      select(sym(date.name), sym(data.name)) %>%
      rename(date.name = sym(date.name), data.name = sym(data.name)) %>%
      mutate(ks = ksmooth(time(data.name), data.name, kernel, bandwidth)$y)

    ts_plot <- ggplot(data, aes(x=date.name, y=ks)) +
      geom_line(color = "turquoise3") +
      stat_peaks(color = "red", span = span) +
      stat_peaks(geom = "text", colour = "red", span = span,
                 hjust = -0.1, x.label.fmt = "%Y-%m-%d") +
      xlab("Date") + ylab(gsub("_", " ", data.name)) +
      ggtitle(paste(kernel, "K-Smooth,", "B =", bandwidth, ", Span =", span))
  }

  return (ts_plot)
}
```

The function above returns a smoothed time series with the detected peaks highlighted and labeled.

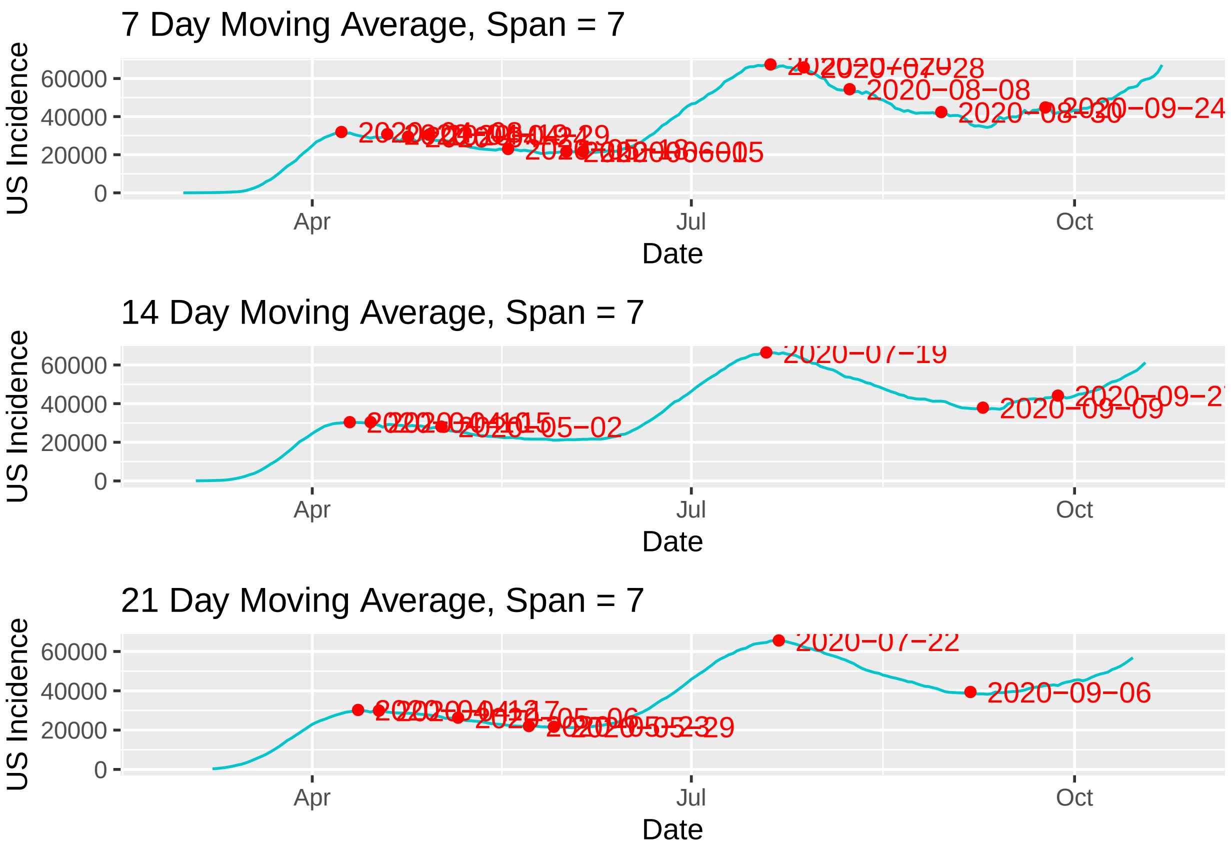
Incidence Peak Detection with Span = 7

```
ma7 <- peak_detect(ts_data, "Date", "RI_Incidence", "ma", ma.window = 7, span = 7)
ma14 <- peak_detect(ts_data, "Date", "RI_Incidence", "ma", ma.window = 14, span = 7)
ma21 <- peak_detect(ts_data, "Date", "RI_Incidence", "ma", ma.window = 21, span = 7)
grid.arrange(ma7, ma14, ma21)
```



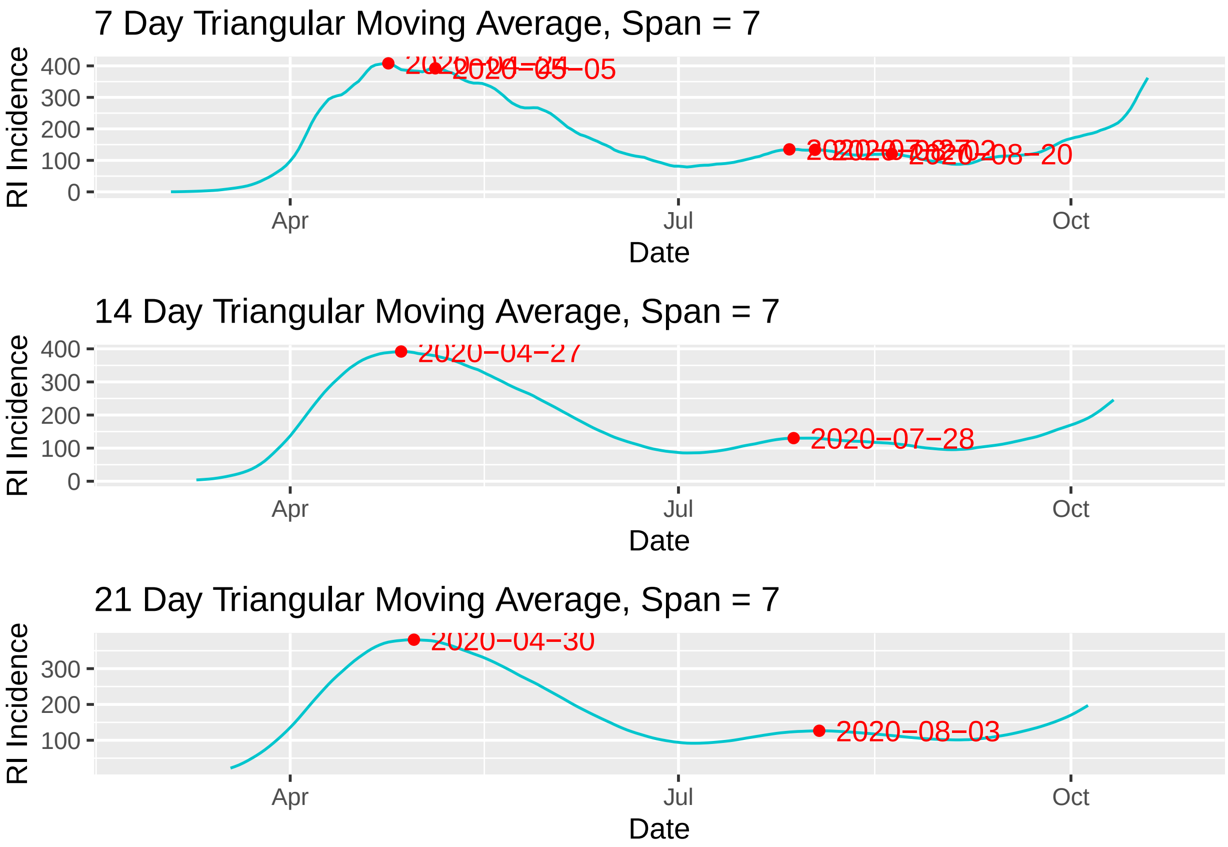
From the plot above, we can observe that two potential peaks in incidence that occurred in Rhode Island are 4/29/2020 and 7/31/2020.

```
ma7 <- peak_detect(ts_data, "Date", "US_Incidence", "ma", ma.window = 7, span = 7)
ma14 <- peak_detect(ts_data, "Date", "US_Incidence", "ma", ma.window = 14, span = 7)
ma21 <- peak_detect(ts_data, "Date", "US_Incidence", "ma", ma.window = 21, span = 7)
grid.arrange(ma7, ma14, ma21)
```



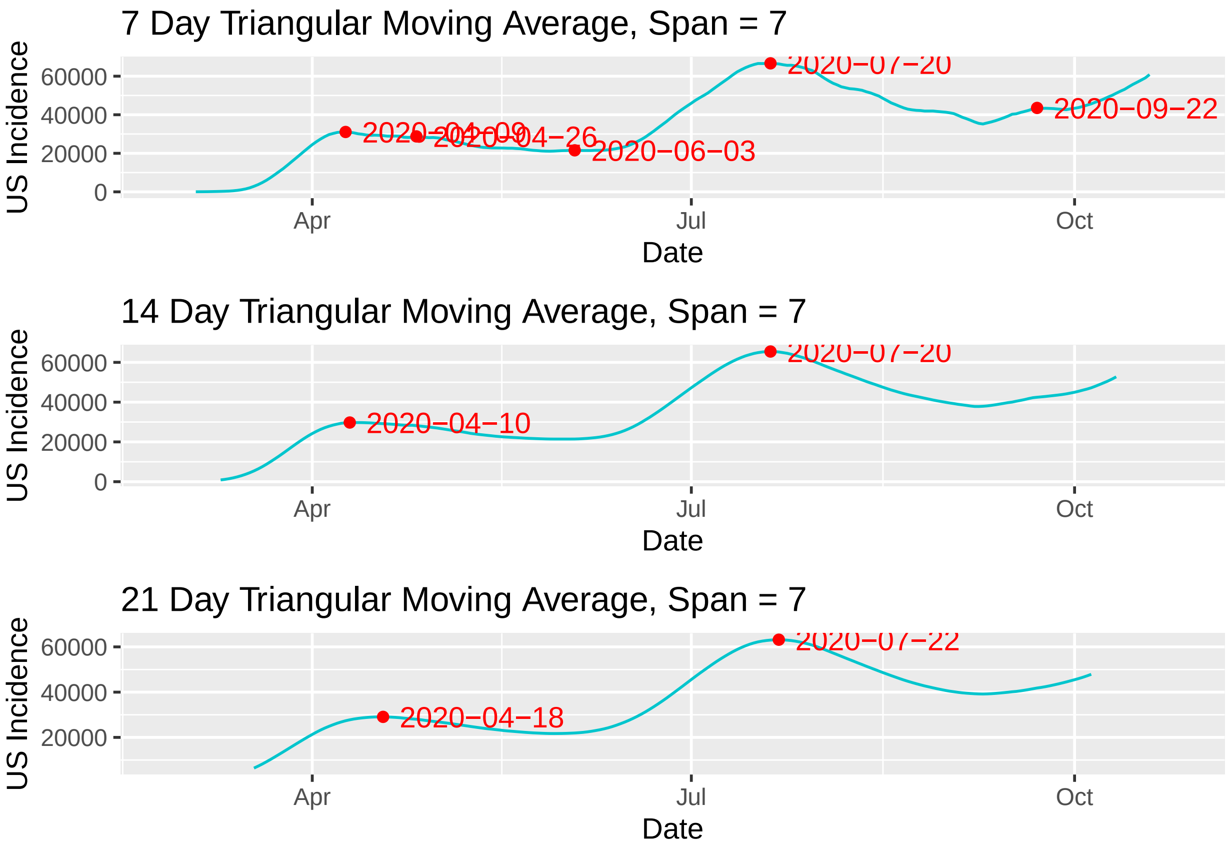
From the plot above, we can observe that two potential peaks in incidence that occurred in the United States are 4/12/2020 and 7/22/2020. However, it is unclear whether or not these are peaks.

```
ma7 <- peak_detect(ts_data, "Date", "RI_Incidence", "tri_ma", ma.window = 7, span = 7)
ma14 <- peak_detect(ts_data, "Date", "RI_Incidence", "tri_ma", ma.window = 14, span = 7)
ma21 <- peak_detect(ts_data, "Date", "RI_Incidence", "tri_ma", ma.window = 21, span = 7)
grid.arrange(ma7, ma14, ma21)
```



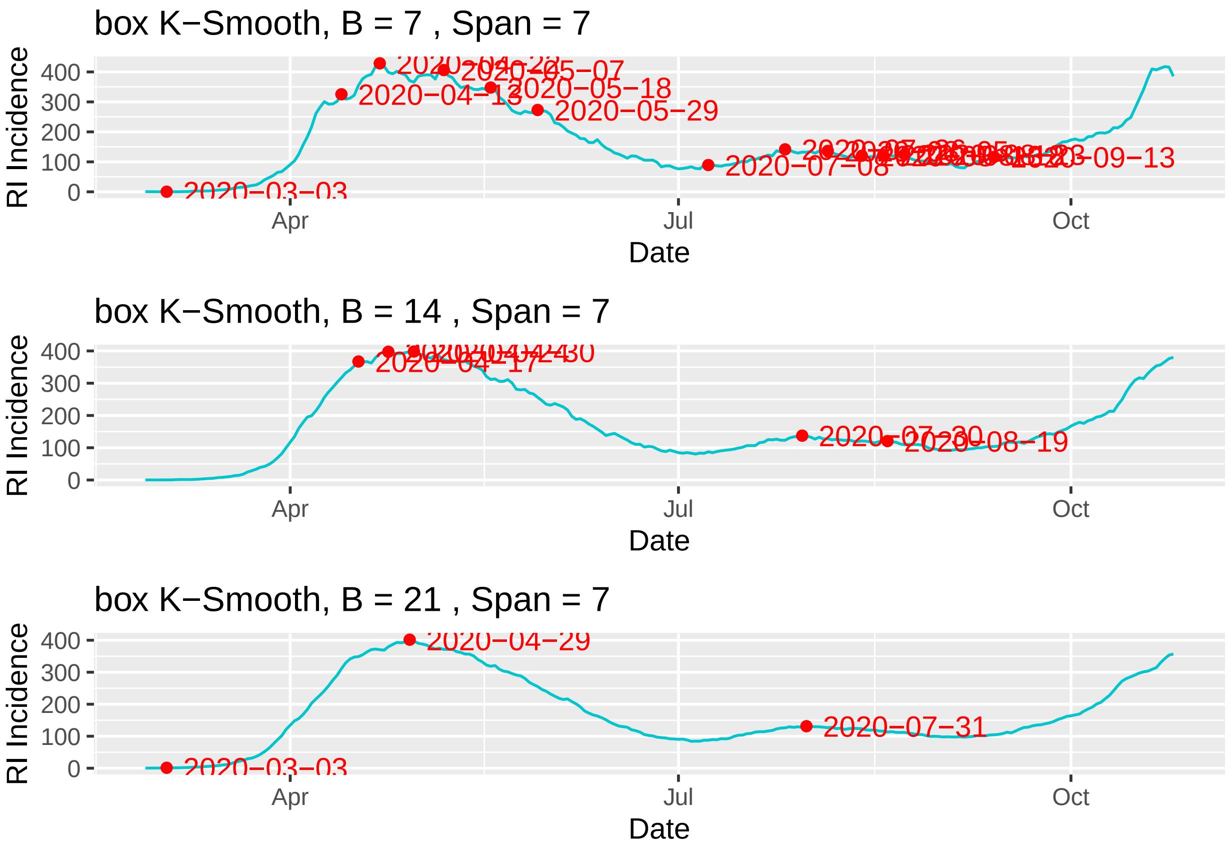
From the plot above, we can observe that two potential peaks in incidence that occurred in Rhode Island are 4/27/2020 and 7/28/2020, or 4/30/2020 and 8/3/2020.

```
ma7 <- peak_detect(ts_data, "Date", "US_Incidence", "tri_ma", ma.window = 7, span = 7)
ma14 <- peak_detect(ts_data, "Date", "US_Incidence", "tri_ma", ma.window = 14, span = 7)
ma21 <- peak_detect(ts_data, "Date", "US_Incidence", "tri_ma", ma.window = 21, span = 7)
grid.arrange(ma7, ma14, ma21)
```



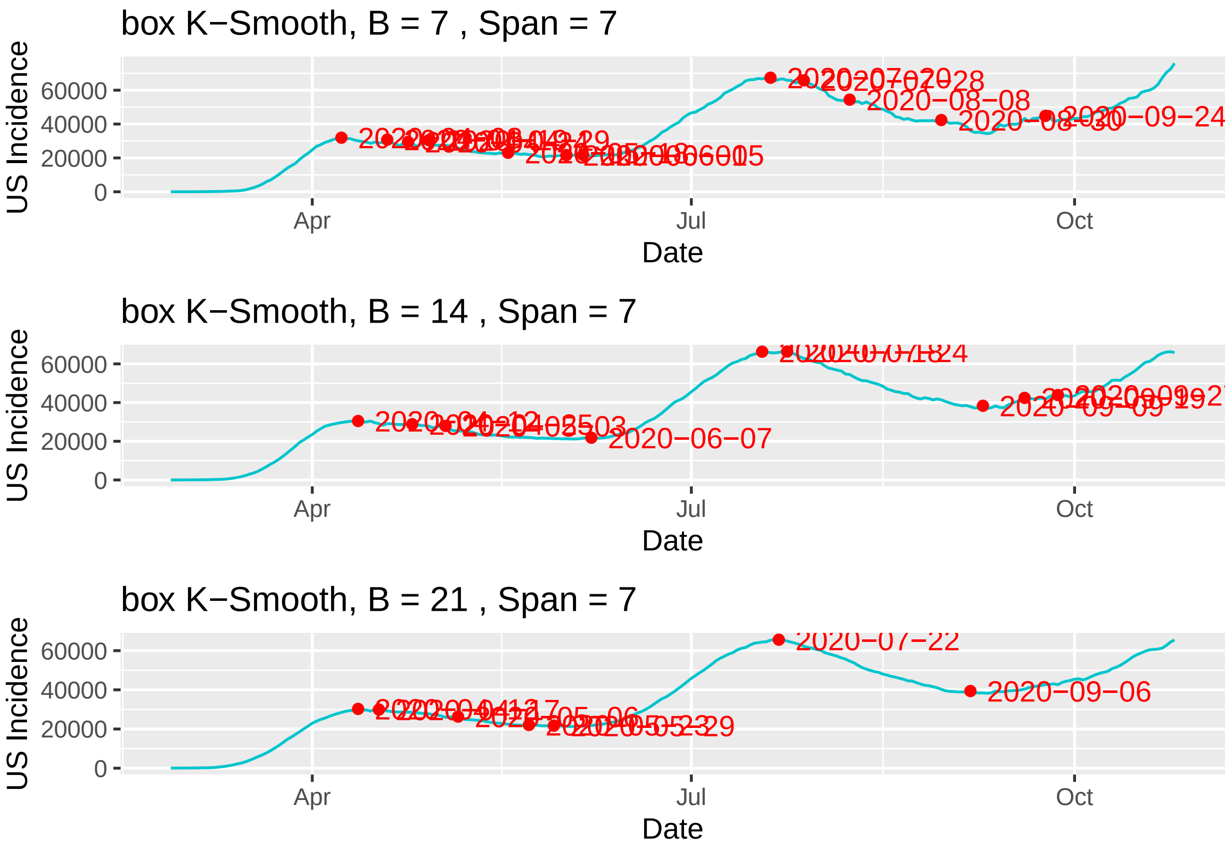
From the plot above, we can observe that two potential peaks in incidence that occurred in the United States are 4/10/2020 and 7/20/2020, or 4/18/2020 and 7/22/2020.

```
bw7 <- peak_detect(ts_data, "Date", "RI_Incidence", "k_smooth",
                   kernel = "box", bandwidth = 7, span = 7)
bw14 <- peak_detect(ts_data, "Date", "RI_Incidence", "k_smooth",
                    kernel = "box", bandwidth = 14, span = 7)
bw21 <- peak_detect(ts_data, "Date", "RI_Incidence", "k_smooth",
                    kernel = "box", bandwidth = 21, span = 7)
grid.arrange(bw7, bw14, bw21)
```



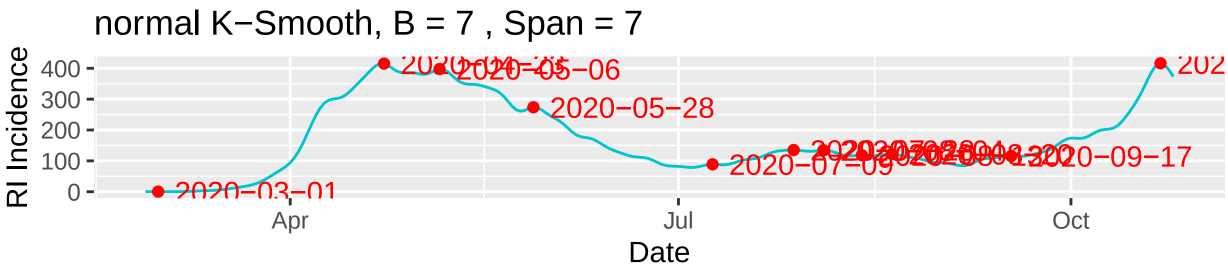
From the plot above, we can observe that two potential peaks in incidence that occurred in Rhode Island are 4/29/2020 and 7/31/2020.

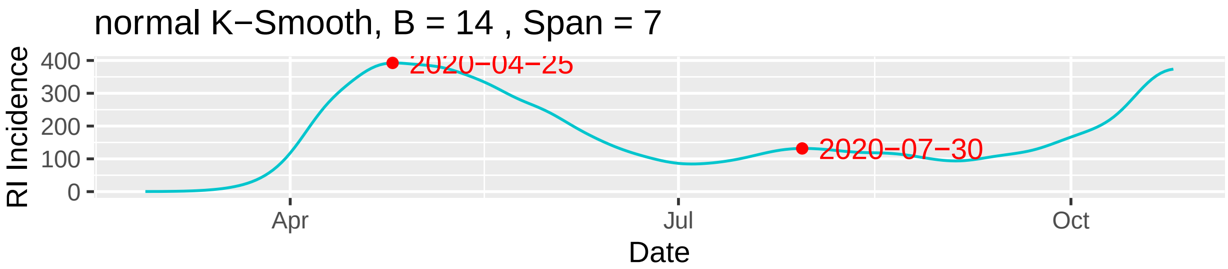
```r
bw7 <- peak_detect(ts_data, "Date", "US_Incidence", "k_smooth",
                   kernel = "box", bandwidth = 7, span = 7)
bw14 <- peak_detect(ts_data, "Date", "US_Incidence", "k_smooth",
                    kernel = "box", bandwidth = 14, span = 7)
bw21 <- peak_detect(ts_data, "Date", "US_Incidence", "k_smooth",
                    kernel = "box", bandwidth = 21, span = 7)
grid.arrange(bw7, bw14, bw21)
```

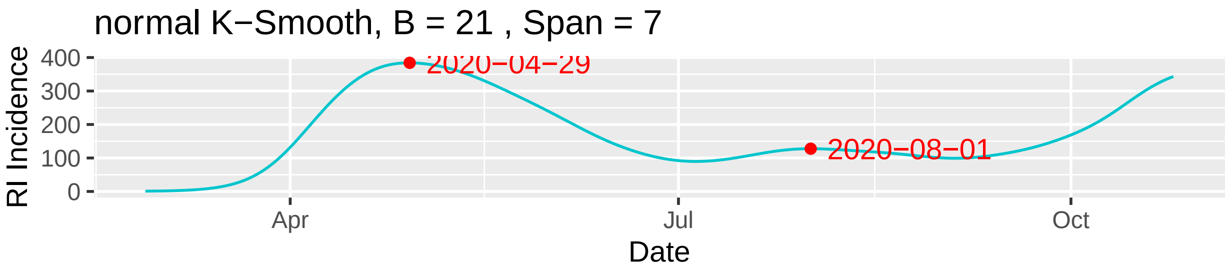


From the plot above, we can observe that two potential peaks in incidence that occurred in the United States are 4/12/2020 and 7/22/2020. However, it is unclear whether or not these are peaks.

```
bw7 <- peak_detect(ts_data, "Date", "RI_Incidence", "k_smooth",
                   kernel = "normal", bandwidth = 7, span = 7)
bw14 <- peak_detect(ts_data, "Date", "RI_Incidence", "k_smooth",
                    kernel = "normal", bandwidth = 14, span = 7)
bw21 <- peak_detect(ts_data, "Date", "RI_Incidence", "k_smooth",
                    kernel = "normal", bandwidth = 21, span = 7)
grid.arrange(bw7, bw14, bw21)
```

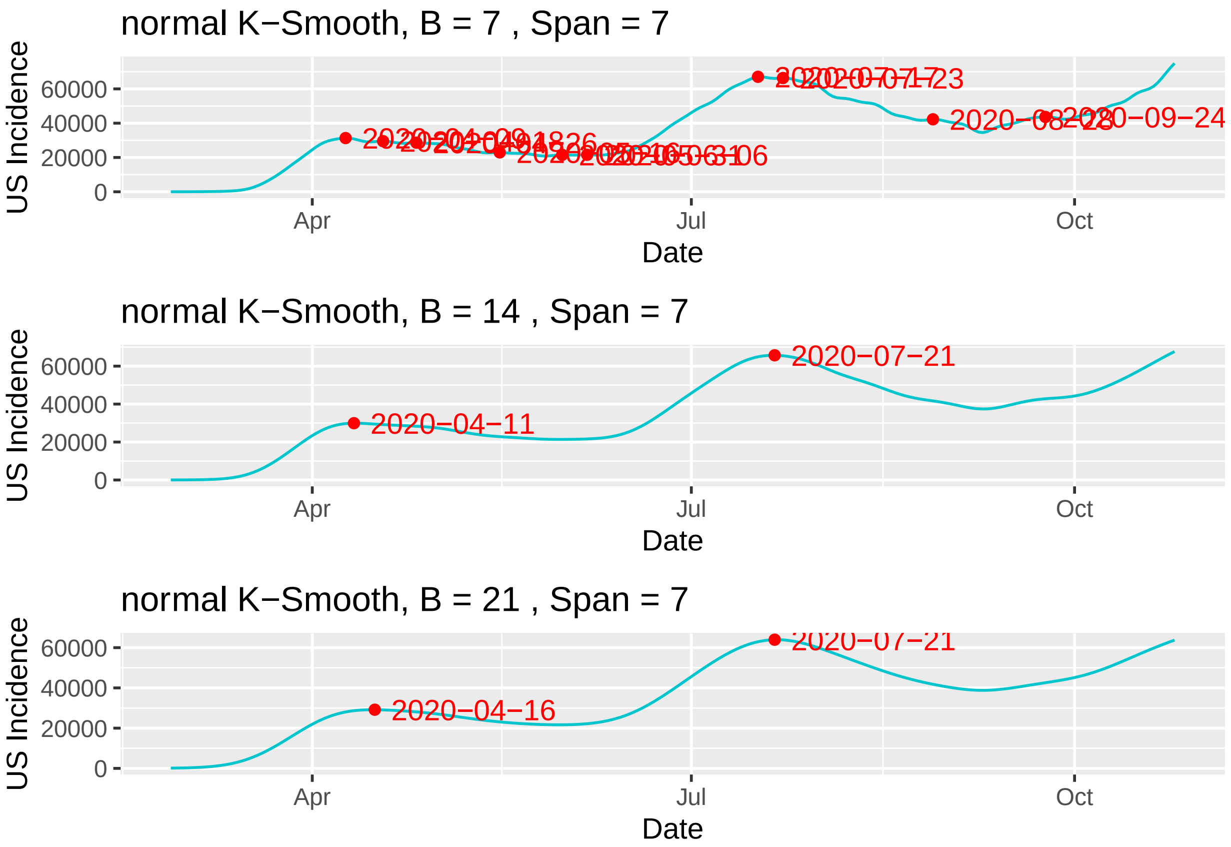






From the plot above, we can observe that two potential peaks in incidence that occurred in Rhode Island are 4/25/2020 and 7/30/2020, or 4/29/2020 and 8/1/2020.

```
bw7 <- peak_detect(ts_data, "Date", "US_Incidence", "k_smooth",
                   kernel = "normal", bandwidth = 7, span = 7)
bw14 <- peak_detect(ts_data, "Date", "US_Incidence", "k_smooth",
                    kernel = "normal", bandwidth = 14, span = 7)
bw21 <- peak_detect(ts_data, "Date", "US_Incidence", "k_smooth",
                    kernel = "normal", bandwidth = 21, span = 7)
grid.arrange(bw7, bw14, bw21)
```



From the plot above, we can observe that two potential peaks in incidence that occurred in the United States are 4/11/2020 and 7/21/2020, or 4/16/2020 and 7/21/2020.

Generalized peak detection function

```
general_peak_detect <- function(data, date.name, data.name, peak_func){
  #' @title Peak Detection of Time Series Data Using User-Defined Peak Detection Algorithm
  #'
  #' @description Returns a smoothed time series plot with the detected peaks highlighted and labeled.
  #' The peaks are detected using a user-defined function for peak detection.
  #'
  #' @param data data frame containing daily incidence and stringency index observations
  #' @param date.name name of the column containing dates
  #' @param data.name name of the column containing the data
  #' @param peak_func list output of a user-defined function for peak detection. Must be a list
  #' containing the x value and index corresponding to the peak and the y values of the
  #' smoothed time series.
  #' @return a smoothed time series plot with the detected peaks highlighted and labeled.
  #' The peaks are detected using a user-defined function for peak detection.

  ##### Stop Functions #####
  if(!(is.data.frame(data))){
    stop("data must be a data frame")
  } else if(!(is.character(date.name) & is.character(data.name))){
    stop("date.name and data.name must be type character")
  } else if(!(is.list(peak_func))){
    stop("peak_func must be a list")
  }

  ##### Time Series Plot Construction #####
  data <- data %>%
    select(sym(date.name), sym(data.name)) %>%
    rename(date.name = sym(date.name), data.name = sym(data.name))

  ts_plot <- ggplot(data, aes(x=date.name, y=data.name)) +
    geom_point() +
    geom_line(aes(x=date.name, y=peak_func$y.hat)) +
    geom_point(aes(x=date.name[peak_func$index],
                   y=peak_func$y.hat[peak_func$index]), color = "red") +
    geom_text(aes(x=date.name[peak_func$index],
                  y=peak_func$y.hat[peak_func$index],
                  label=date.name[peak_func$index]),
              hjust = -0.1, vjust = -0.1, color = "red") +
    xlab("Date") + ylab(gsub("_", " ", data.name)) +
    ggtitle(paste("Window =", peak_func$w, ", Span =", peak_func$span))

  return (ts_plot)
}
```

The function above returns a smoothed time series plot with the detected peaks highlighted and labeled. The peaks are detected using a user-defined function for peak detection.

User Defined Smoothing Function and Peak Detection Algorithm

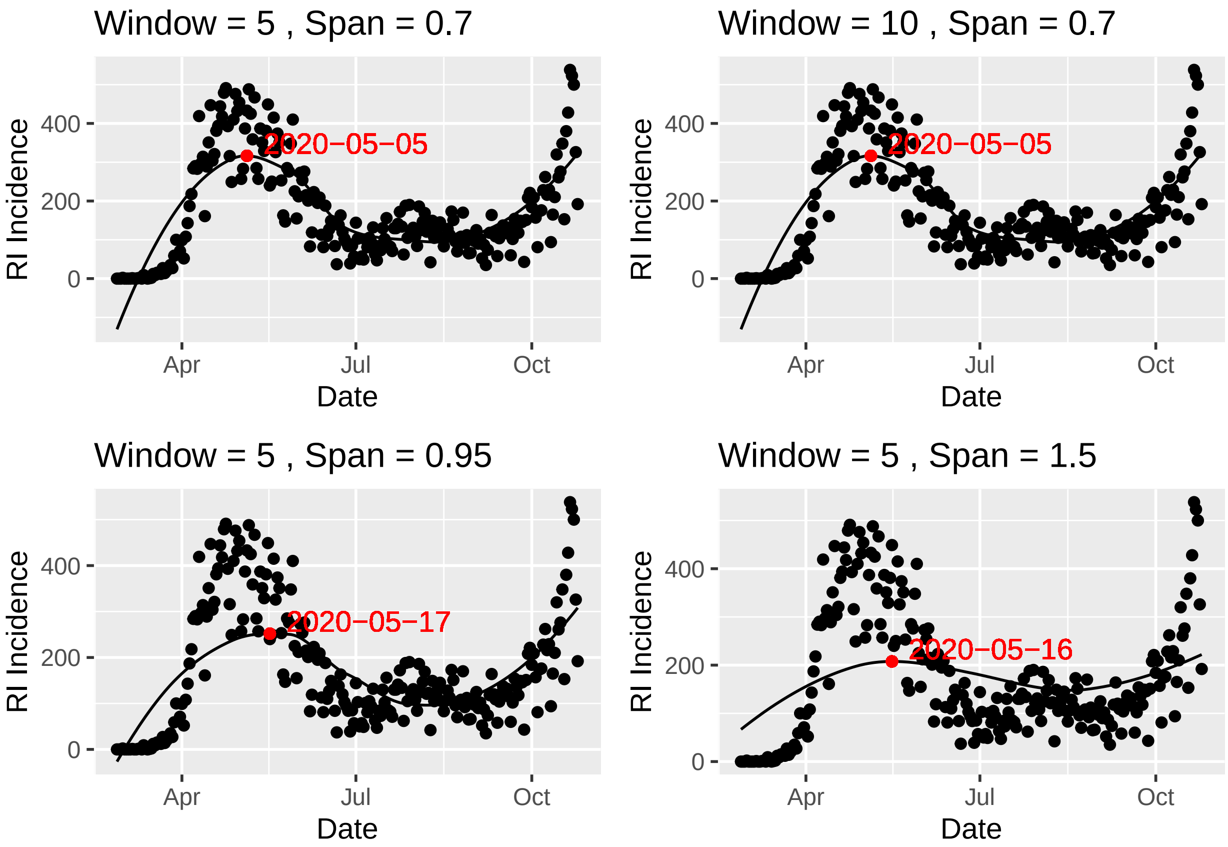
```
peak_func <- function(data, date.name, data.name, w, span){
  #' @title Peak Detection of Time Series Data Using Local Polynomial Regression Fitting
  #'
  #' @description Returns a list containing the x value and index corresponding to the peak,
  #' the y values of the smoothed time series, the window size used to compute the local
  #' maximum, and the span used to control the degree of smoothing
  #'
  #' @param data data frame containing daily incidence and stringency index observations
  #' @param date.name name of the column containing dates
  #' @param data.name name of the column containing the data
  #' @param w the window size used to compute the local maximum
  #' @param span the parameter that controls the degree of smoothing
  #' @return a list containing the x value and index corresponding to the peak, the y values
  #' of the smoothed time series, the window size used to compute the local maximum, and the
  #' span used to control the degree of smoothing

  ##### Stop Functions #####
  if(!(is.data.frame(data))){
    stop("data must be a data frame")
  } else if(!(is.character(date.name) & is.character(data.name))){
    stop("date.name and data.name must be type character")
  } else if(!(is.numeric(w))){
    stop("w must be numeric")
  } else if(!(is.numeric(span))){
    stop("span must be numeric")
  }

  ##### Peak Detection Algorithm #####
  data <- data %>%
    select(sym(date.name), sym(data.name)) %>%
    rename(date.name = sym(date.name), data.name = sym(data.name))

  n <- length(data$data.name)
  # Compute y.hat for the smoothed time series
  data.smooth <- loess(data.name ~ time(date.name), data = data, span = span)$fitted
  # Compute the local maximum
  data.max <- rollapply(zoo(data.smooth), 2*w+1, max, align="center")
  delta <- data.max - data.smooth[-c(1:w, n+1-1:w)]
  time.max <- which(delta <= 0) + w

  return (list(date=data$date.name[time.max], index=time.max, y.hat=data.smooth, w=w, span=span))
}

str(peak_func(ts_data, "Date", "RI_Incidence", w = 5, span = 0.7))
```

```
## List of 5
##  $ date : POSIXct[1:1], format: "2020-05-05"
##  $ index: num 69
##  $ y.hat: num [1:242] -130.7 -118.3 -106 -93.8 -81.8 ...
##  $ w    : num 5
##  $ span : num 0.7
```

The function above returns a list containing the x value and index corresponding to the peak, the y values of the smoothed time series, the window size used to compute the local maximum, and the span used to control the degree of smoothing.

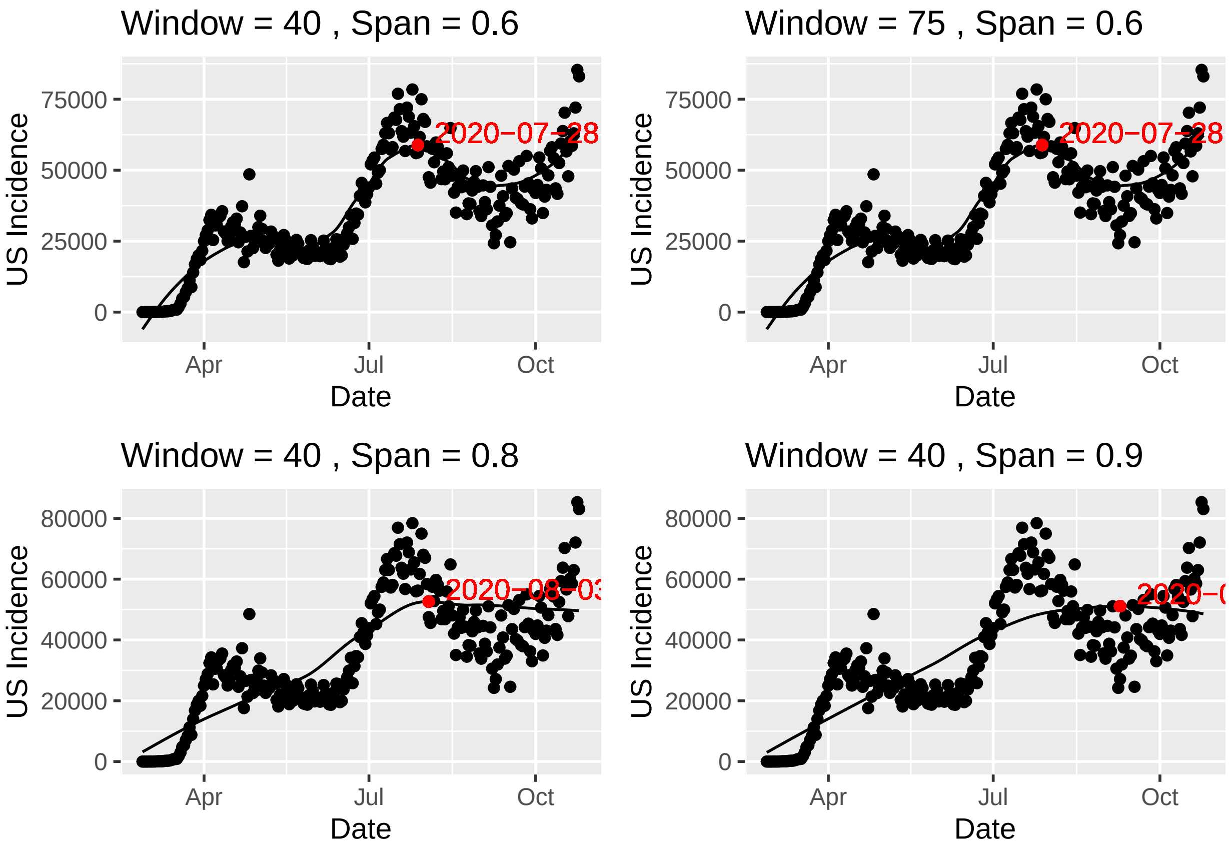
Incidence Peak Detection Using User-Defined Smoothing Function and Peak Detection Algorithm

```
w5_s0.7 <- general_peak_detect(ts_data, "Date", "RI_Incidence",
                               peak_func(ts_data, "Date", "RI_Incidence", w = 5, span = 0.7))
w10_s0.7 <- general_peak_detect(ts_data, "Date", "RI_Incidence",
                                peak_func(ts_data, "Date", "RI_Incidence", w = 10, span = 0.7))
w5_s0.95 <- general_peak_detect(ts_data, "Date", "RI_Incidence",
                                peak_func(ts_data, "Date", "RI_Incidence", w = 5, span = 0.95))
w5_s1.5 <- general_peak_detect(ts_data, "Date", "RI_Incidence",
                               peak_func(ts_data, "Date", "RI_Incidence", w = 5, span = 1.5))
grid.arrange(w5_s0.7, w10_s0.7, w5_s0.95, w5_s1.5)
```



It appears that a potential peak in incidence in Rhode Island occurred in the beginning of May.

```
w40_s0.6 <- general_peak_detect(ts_data, "Date", "US_Incidence",
                                peak_func(ts_data, "Date", "US_Incidence", w = 40, span = 0.6))
w75_s0.6 <- general_peak_detect(ts_data, "Date", "US_Incidence",
                                peak_func(ts_data, "Date", "US_Incidence", w = 75, span = 0.6))
w40_s0.8 <- general_peak_detect(ts_data, "Date", "US_Incidence",
                                peak_func(ts_data, "Date", "US_Incidence", w = 40, span = 0.8))
w40_s0.9 <- general_peak_detect(ts_data, "Date", "US_Incidence",
                                peak_func(ts_data, "Date", "US_Incidence", w = 40, span = 0.9))
grid.arrange(w40_s0.6, w75_s0.6, w40_s0.8, w40_s0.9)
```



It appears that a potential peak in incidence in the United States occurred in the beginning of July.

Conclusion

From our results, we can observe that the potential peaks in incidence for both Rhode Island and the United States are similar. We expected this to occur because the stringency index for Rhode Island and the United States were similar for all time points. We could not implement the best parameter selection methods for our smoothing methods, hence our smoothing results may not be trustworthy. We encourage further similar analyses to be done for other states. We expect the potential peaks in incidence to be different at the state level and the national level when their stringency index is significantly different.