```r
# Load required libraries
library(fredr)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
## Registered S3 methods overwritten by 'forecast':
##   method       from
##   fitted.Arima TSA
##   plot.Arima   TSA
```

```r
library(readr)
```

```
## 
## Attaching package: 'readr'
```

```
## The following objects are masked from 'package:TSA':
## 
##     spec
```

```r
library(lubridate)
```

```
## 
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
## 
##     date, intersect, setdiff, union
```

```r
library(zoo)
```

```
## 
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
## 
##     as.Date, as.Date.numeric
```

```r
library(dplyr)
```

```
## 
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
## 
##     filter, lag
```

```
## The following objects are masked from 'package:base':
## 
##     intersect, setdiff, setequal, union
```

```r
library(ggplot2)
library(imputeTS)
```

```
## Warning: package 'imputeTS' was built under R version 4.3.3
```

```
## 
## Attaching package: 'imputeTS'
```

```
## The following object is masked from 'package:zoo':
## 
##     na.locf
```

```r
library(KFAS)
```

```
## Warning: package 'KFAS' was built under R version 4.3.3
```

```
## Please cite KFAS in publications by using: 
## 
##   Jouni Helske (2017). KFAS: Exponential Family State Space Models in R. Journal of Statistical Software, 78(10), 1-39. doi:10.18637/jss.v078.i10.
```

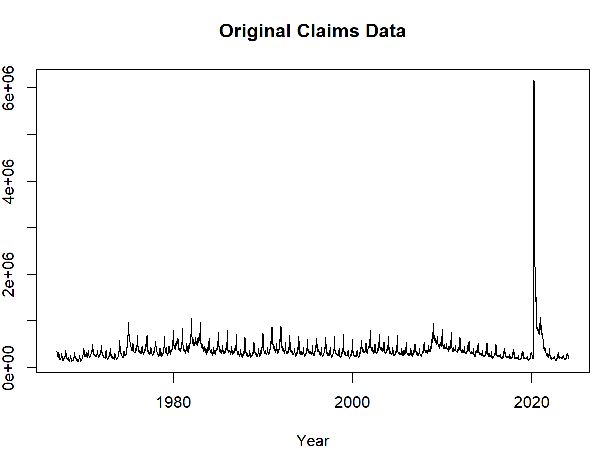
```r
library(MARSS)
```

```
## Warning: package 'MARSS' was built under R version 4.3.3
```

```r
# Set FRED API key
fredr_set_key("[illegible]")
# Load ICNSA data
icnsa <- fredr(series_id = "ICNSA")
icnsa$date <- as.Date(icnsa$date)
```

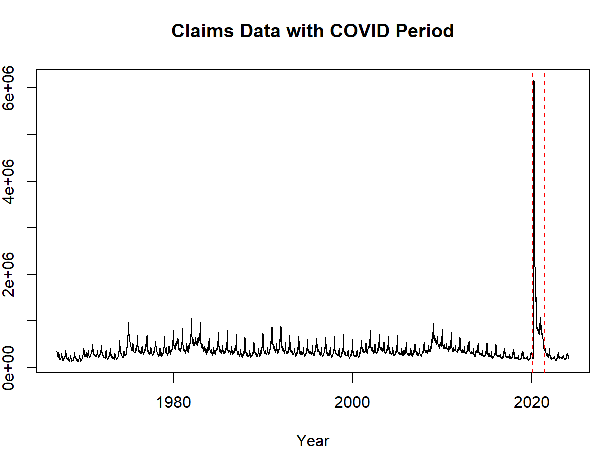
```r
# Plot original data
plot(icnsa$date, icnsa$value, type = "l",
     main = "Original Claims Data",
     xlab = "Year",
     ylab = "Number")
```

Original Claims Data




```r
# Define COVID start and end date
covid_start <- as.Date("2020-03-01")
covid_end <- as.Date("2021-06-30")
# Plot COVID period data
plot(icnsa$date, icnsa$value, type = "l",
     main = "Claims Data with COVID Period",
     xlab = "Year",
     ylab = "Number")
abline(v = as.numeric(covid_start), col = "red", lty = 2)
abline(v = as.numeric(covid_end), col = "red", lty = 2)
```

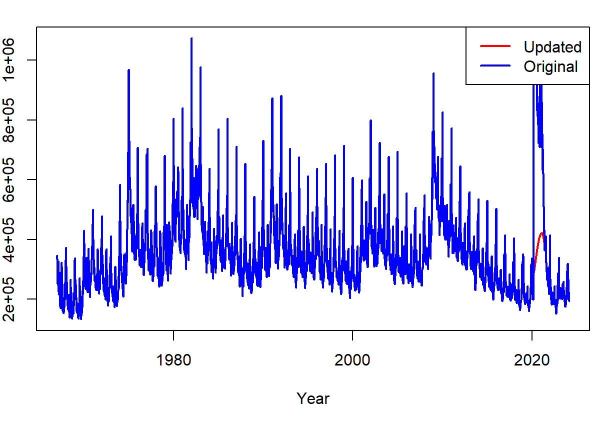
Claims Data with COVID Period



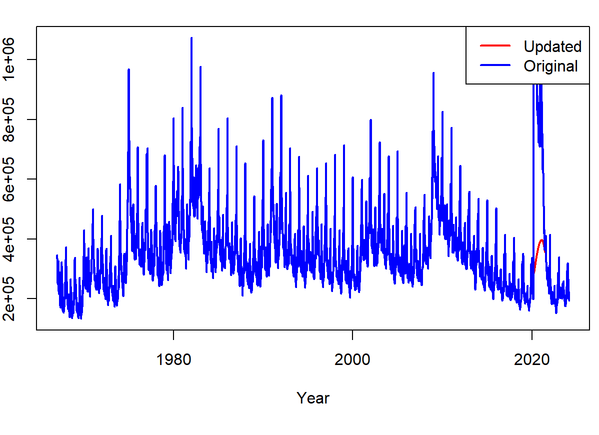

```r
# Filter non-COVID and COVID period data
non_covid <- icnsa[icnsa$date < covid_start | icnsa$date > covid_end, ]
covid <- icnsa[icnsa$date >= covid_start & icnsa$date <= covid_end, ]
```

```r
# Impute missing COVID data using cubic splines
lambda_values <- seq(0.1, 1.0, by = 0.1)
for (lam in lambda_values) {
  spline_fit <- smooth.spline(x = as.numeric(non_covid$date), y = non_covid$value, spar = lam)
  imputed_values <- predict(spline_fit, x = as.numeric(covid$date))$y
  covid$value <- imputed_values
  updated_icnsa <- rbind(non_covid, covid)
  updated_icnsa <- updated_icnsa %>% arrange(date)
  plot(updated_icnsa$date, updated_icnsa$value, type = "l", col = "red", lwd = 2,
       main = paste("Comparison of Time Series (spar =", lam, ")"), xlab = "Year", ylab = "Number")
  lines(icnsa$date, icnsa$value, col = "blue", lwd = 2)
  legend("topright", legend = c("Updated", "Original"), col = c("red", "blue"), lty = 1, lwd = 2)
}
```

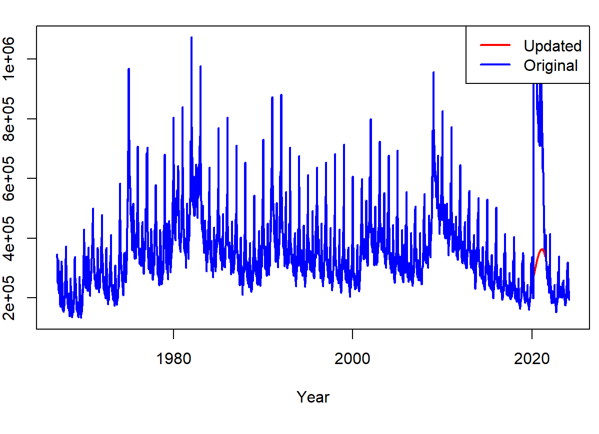
Comparison of Time Series (spar = 0.1)



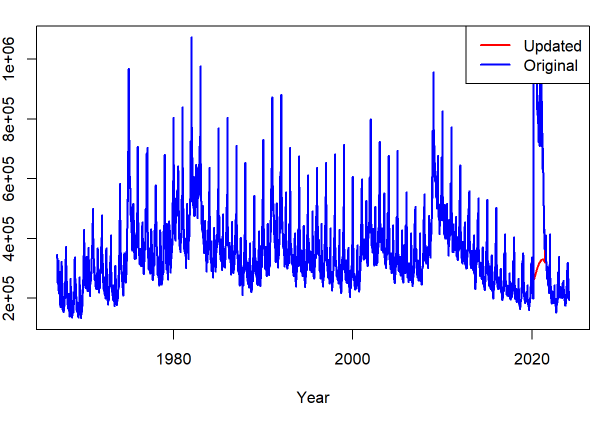
Comparison of Time Series (spar = 0.2)



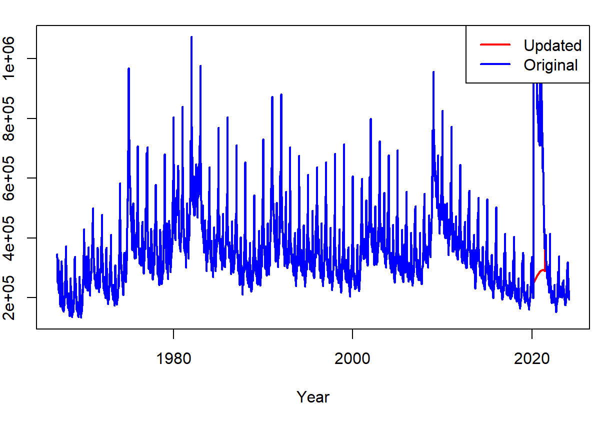
Comparison of Time Series (spar = 0.3)



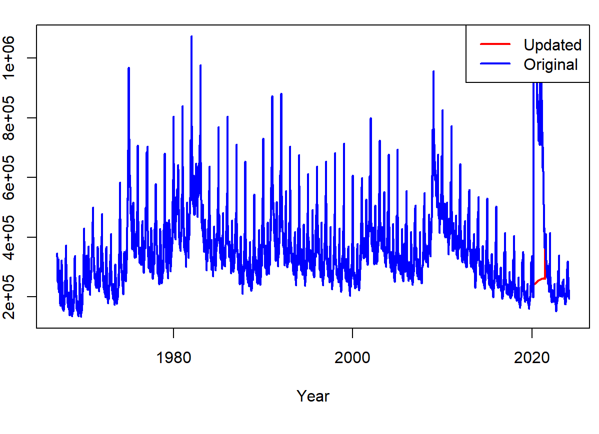
Comparison of Time Series (spar = 0.4)



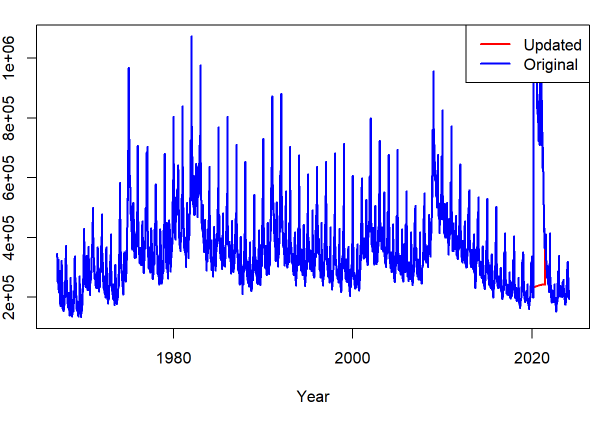
Comparison of Time Series (spar = 0.5)



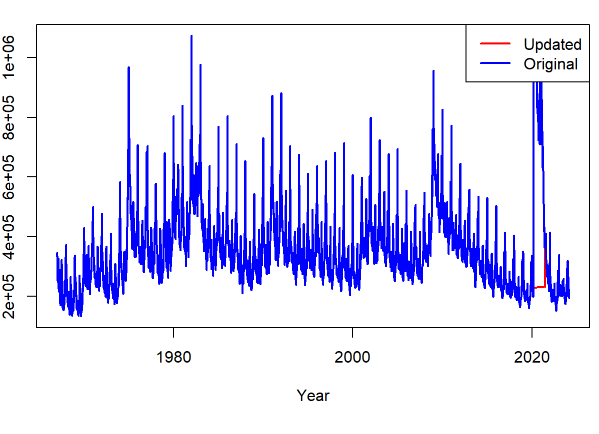
Comparison of Time Series (spar = 0.6)



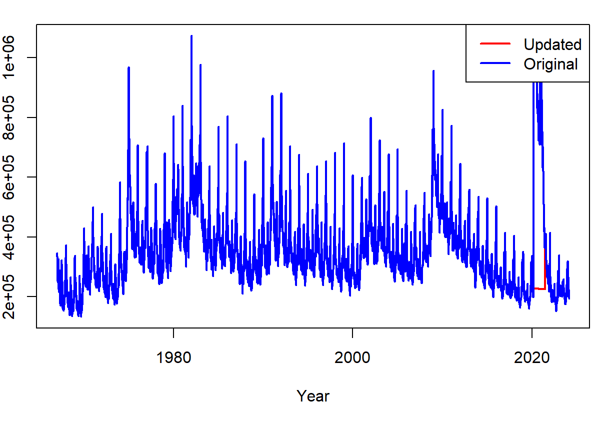
Comparison of Time Series (spar = 0.7)



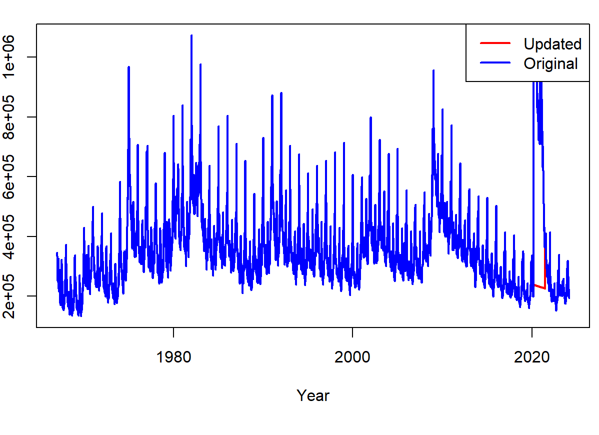
Comparison of Time Series (spar = 0.8)



Comparison of Time Series (spar = 0.9)

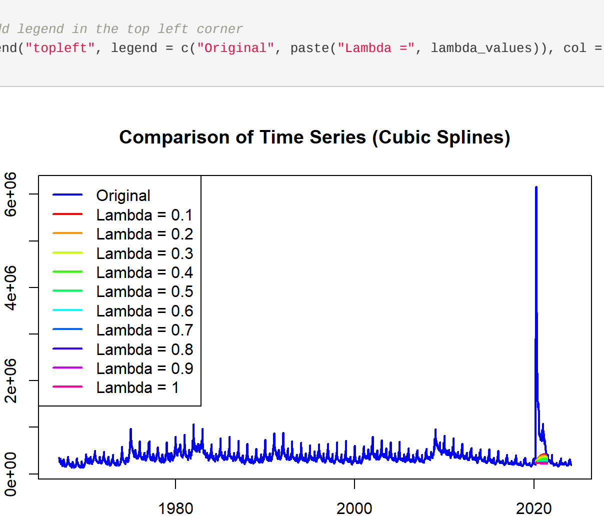


Comparison of Time Series (spar = 1)



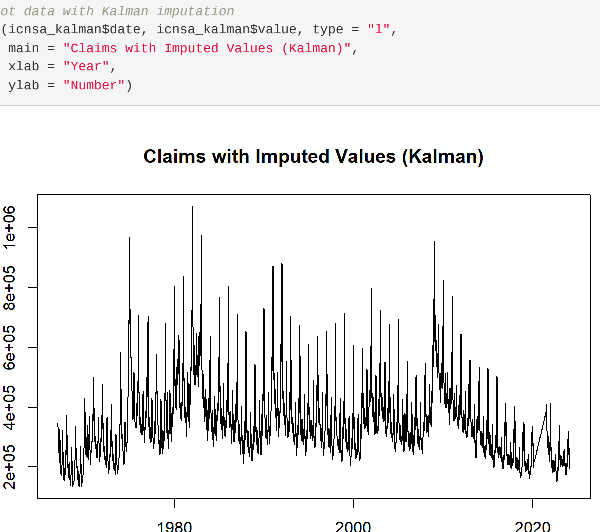

```r
# Create an empty plot
plot(icnsa$date, icnsa$value, type = "l", col = "blue", lwd = 2,
     main = "Comparison of Time Series (Cubic Splines)", xlab = "Year", ylab = "Number")

# Impute missing COVID data using cubic splines
lambda_values <- seq(0.1, 1.0, by = 0.1)
colors <- rainbow(length(lambda_values)) # Generate rainbow colors for each lambda value

for (i in seq_along(lambda_values)) {
  lam <- lambda_values[i]
  color <- colors[i]

  spline_fit <- smooth.spline(x = as.numeric(non_covid$date), y = non_covid$value, spar = lam)
  imputed_values <- predict(spline_fit, x = as.numeric(covid$date))$y

  # Plot the imputed values with different colors
  lines(covid$date, imputed_values, col = color, lwd = 2)
}

# Add legend in the top left corner
legend("topleft", legend = c("Original", paste("Lambda =", lambda_values)), col = c("blue", colors), lty = 1, lwd
= 2)
```

Comparison of Time Series (Cubic Splines)



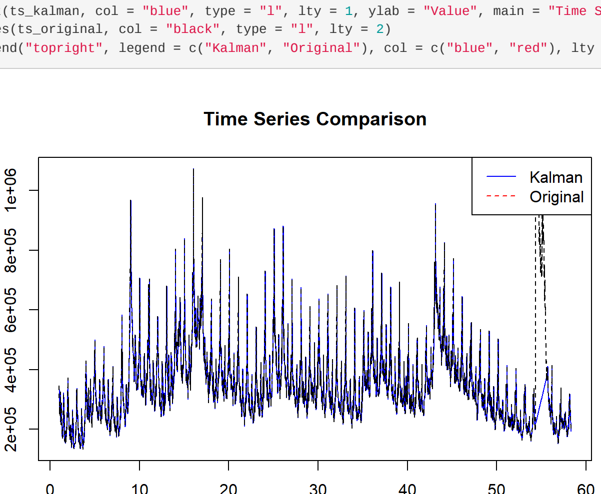

```r
# Impute missing COVID data using Kalman filter
icnsa_kalman <- icnsa
icnsa_kalman$value[icnsa_kalman$date >= covid_start & icnsa_kalman$date <= covid_end] <- NA
imputed_kalman <- na_kalman(icnsa_kalman$value, smooth = TRUE)
icnsa_kalman$value <- imputed_kalman
# Plot data with Kalman imputation
plot(icnsa_kalman$date, icnsa_kalman$value, type = "l",
     main = "Claims with Imputed Values (Kalman)",
     xlab = "Year",
     ylab = "Number")
```

Claims with Imputed Values (Kalman)



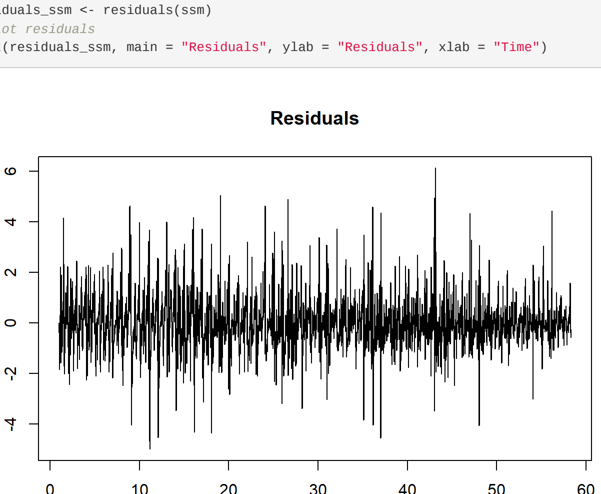

```r
# Create time series objects
ts_original <- ts(icnsa$value, frequency = 52)
ts_kalman <- ts(icnsa_kalman$value, frequency = 52)
# Compare original and Kalman-smoothed series
plot(ts_kalman, col = "blue", type = "l", lty = 1, ylab = "Value", main = "Time Series Comparison")
lines(ts_original, col = "black", type = "l", lty = 2)
legend("topright", legend = c("Kalman", "Original"), col = c("blue", "red"), lty = 1:2)
```

Time Series Comparison




```r
# Fit a structural time series model
ssm <- StructTS(ts_kalman, type = "BSM")
smoothed_data <- tsSmooth(ssm)
residuals_ssm <- residuals(ssm)
# Plot residuals
plot(residuals_ssm, main = "Residuals", ylab = "Residuals", xlab = "Time")
```

Residuals




```r
# Load IURNSA data
iurnsa <- fredr(series_id = "IURNSA")
iurnsa$date <- as.Date(iurnsa$date)

# Merge ICNSA and IURNSA data
merged_data <- merge(icnsa, iurnsa, by = "date", all.x = TRUE)

# Fit ARIMA model with covariate
arima_cov <- auto.arima(merged_data$value.x, xreg = merged_data$value.y, seasonal = TRUE)
```

```r
# Forecast next week's value
forecast_next_week <- forecast(arima_cov, h = 1, xreg = tail(merged_data$value.y, 2))
```

```
## Warning in forecast.forecast_ARIMA(arima_cov, h = 1, xreg =
## tail(merged_data$value.y, : Upper prediction intervals are not finite.
```

```r
print(forecast_next_week)
```

```
##      Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## 2983       190698.2 -13078.38 394474.7 -120951.2 502347.5
## 2984             NA        NA       NA        NA       NA
```