Back transformation before forecasting

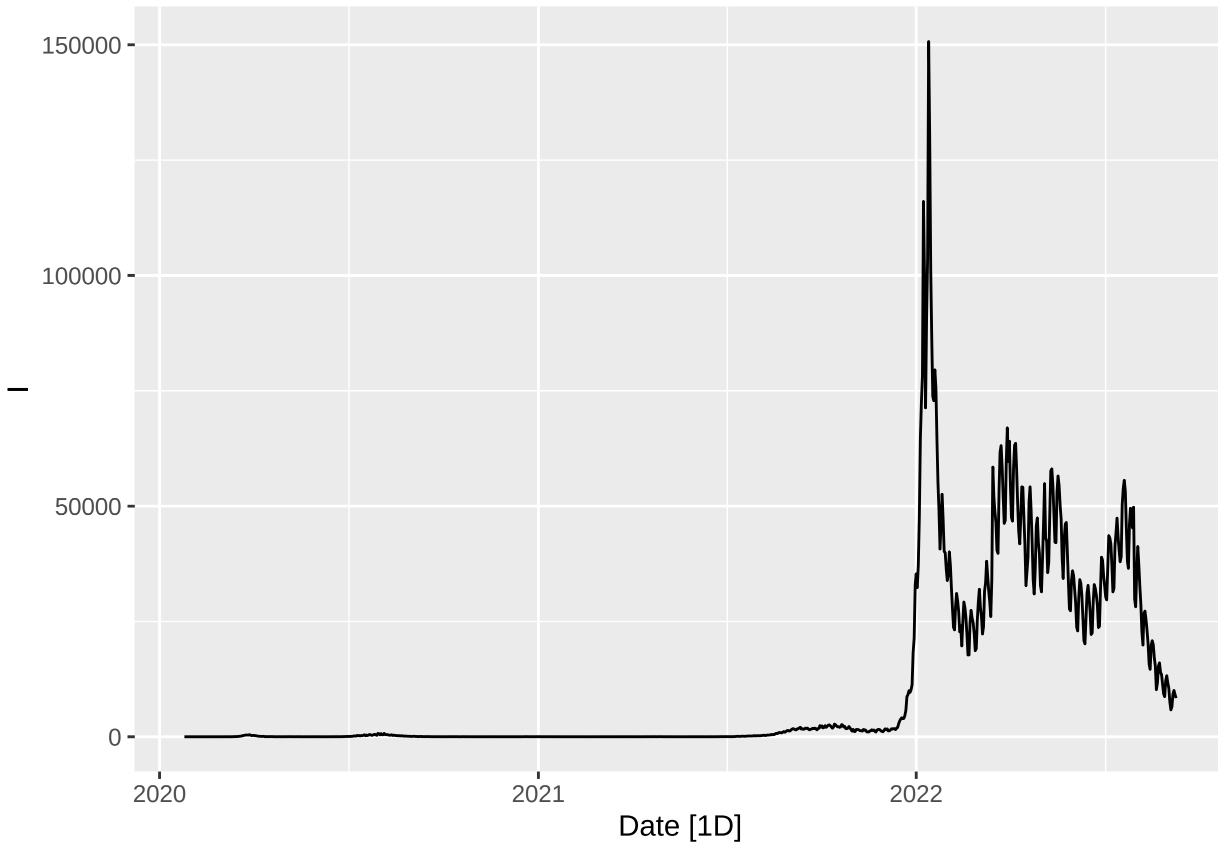
```r
library(fpp3)
library(tidyverse)
library(EpiEstim)
```

```r
read_csv("covidlive_data_2022-09-12.csv") %>%
  select(-...1) %>%
  filter(date_confirmation <= '2022-09-09') -> covidlive_ll

covidlive_ll %>%
  group_by(date_confirmation) %>%
  summarise(daily_case = sum(daily_notification)) -> full_data

colnames(full_data) <- c("Date", 'I')
```

Orginal data

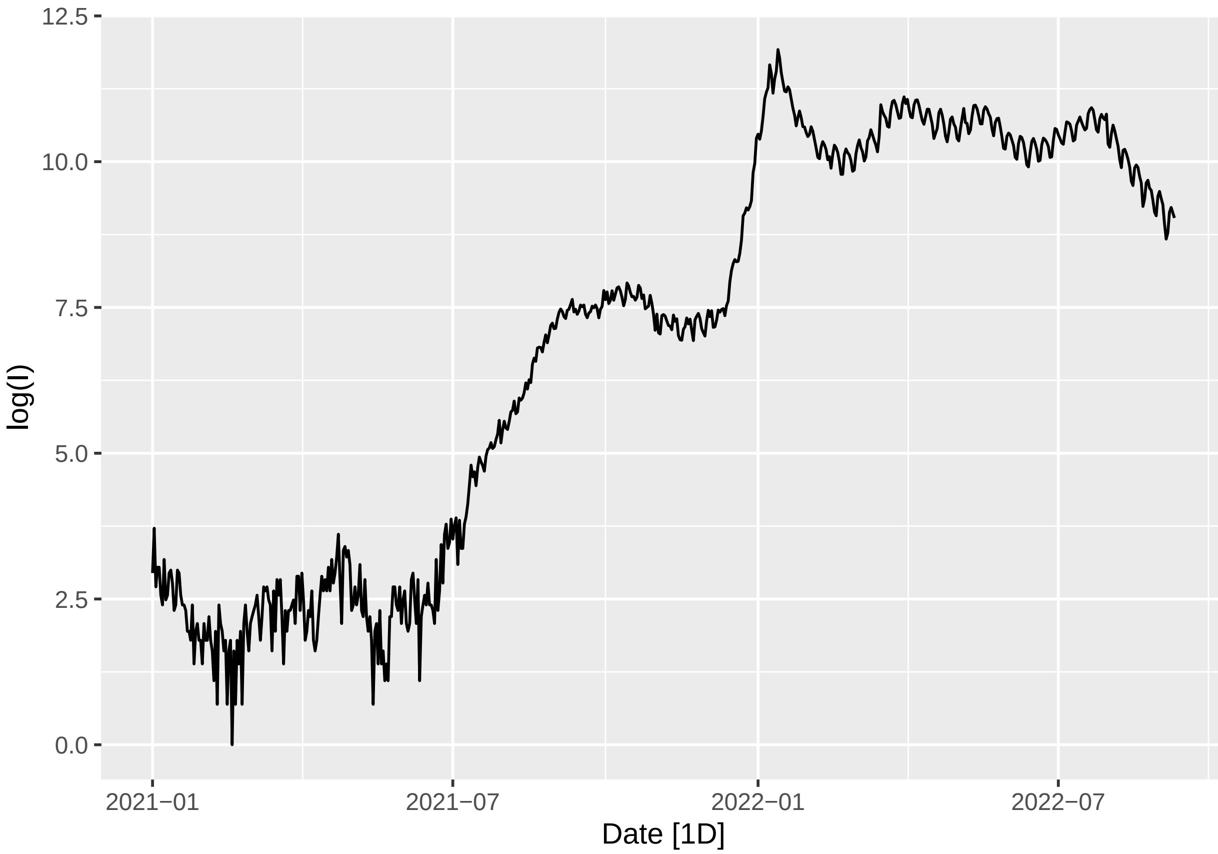
```r
full_data %>%
  as_tsibble(index = Date) -> data_ts
```

```r
data_ts %>%
  autoplot()
```

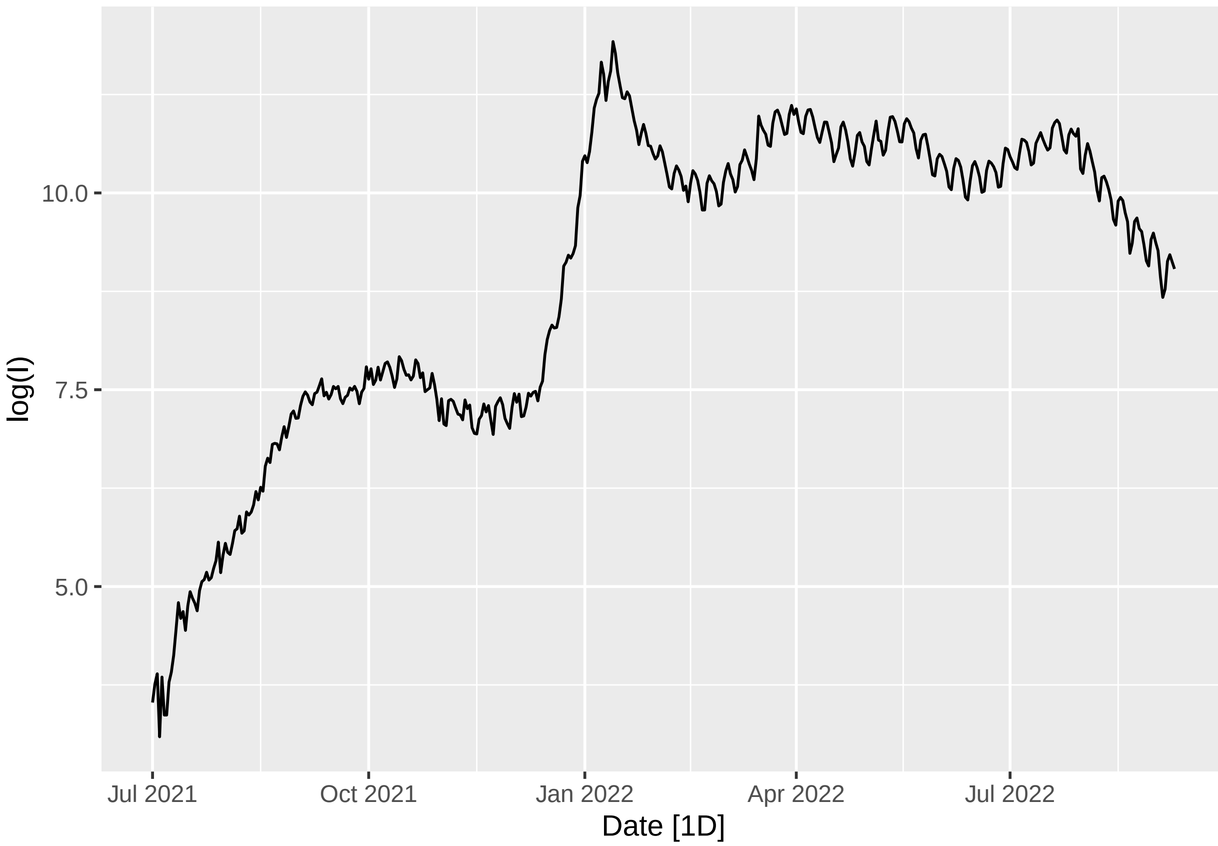


log transformation

```
data_ts %>%
  filter(Date >= ymd("2021-01-01")) %>%
  autoplot(log(I))
```

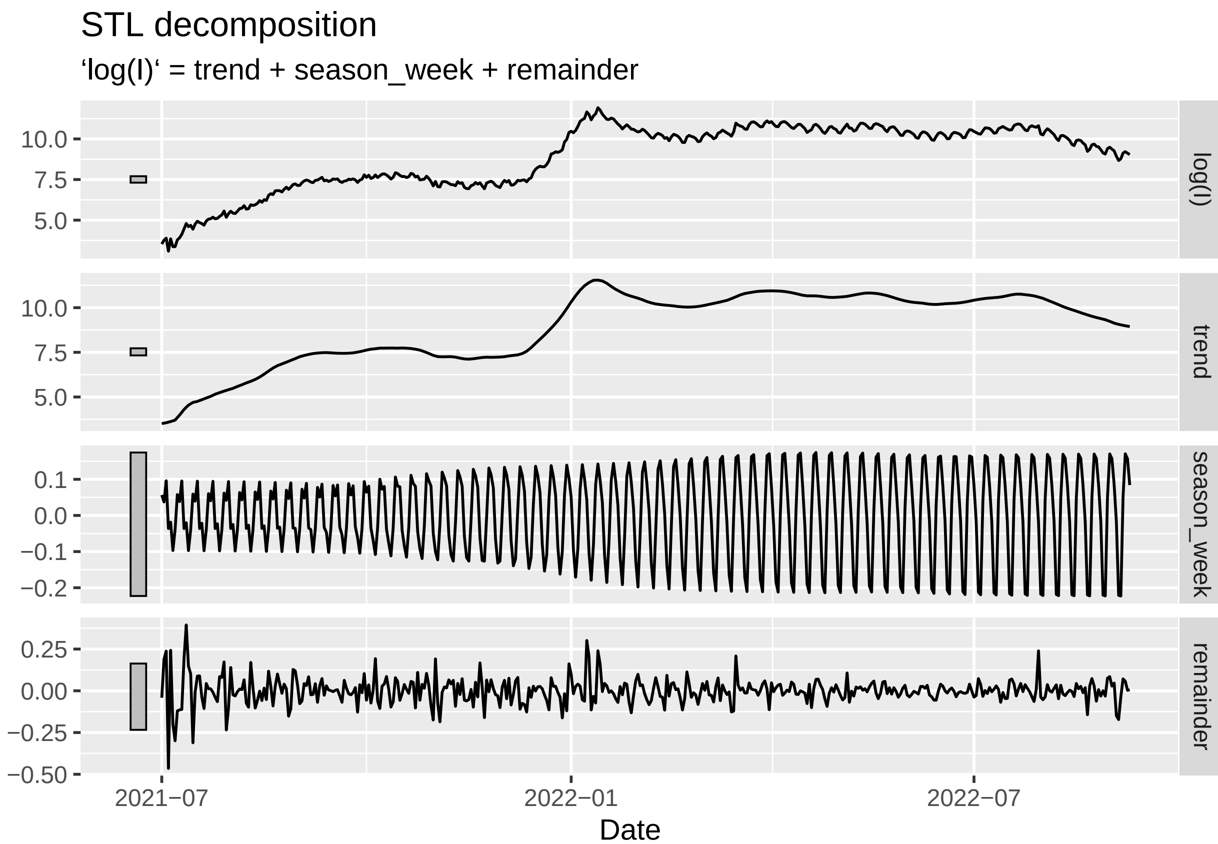



```
data_ts %>%
  filter(Date >= ymd("2021-07-01")) %>%
  autoplot(log(I))
```

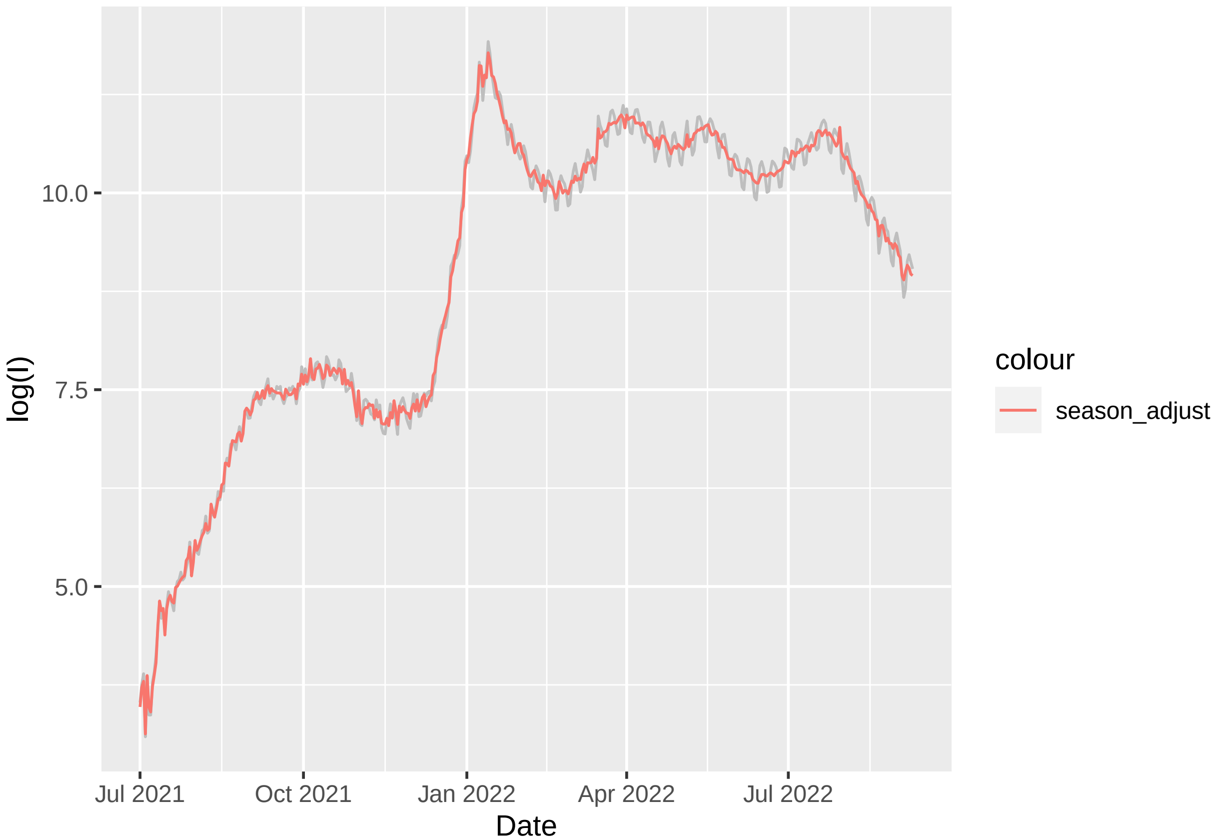


Time Decomposition

```
decomp %>%
  components() %>%
  autoplot()
```



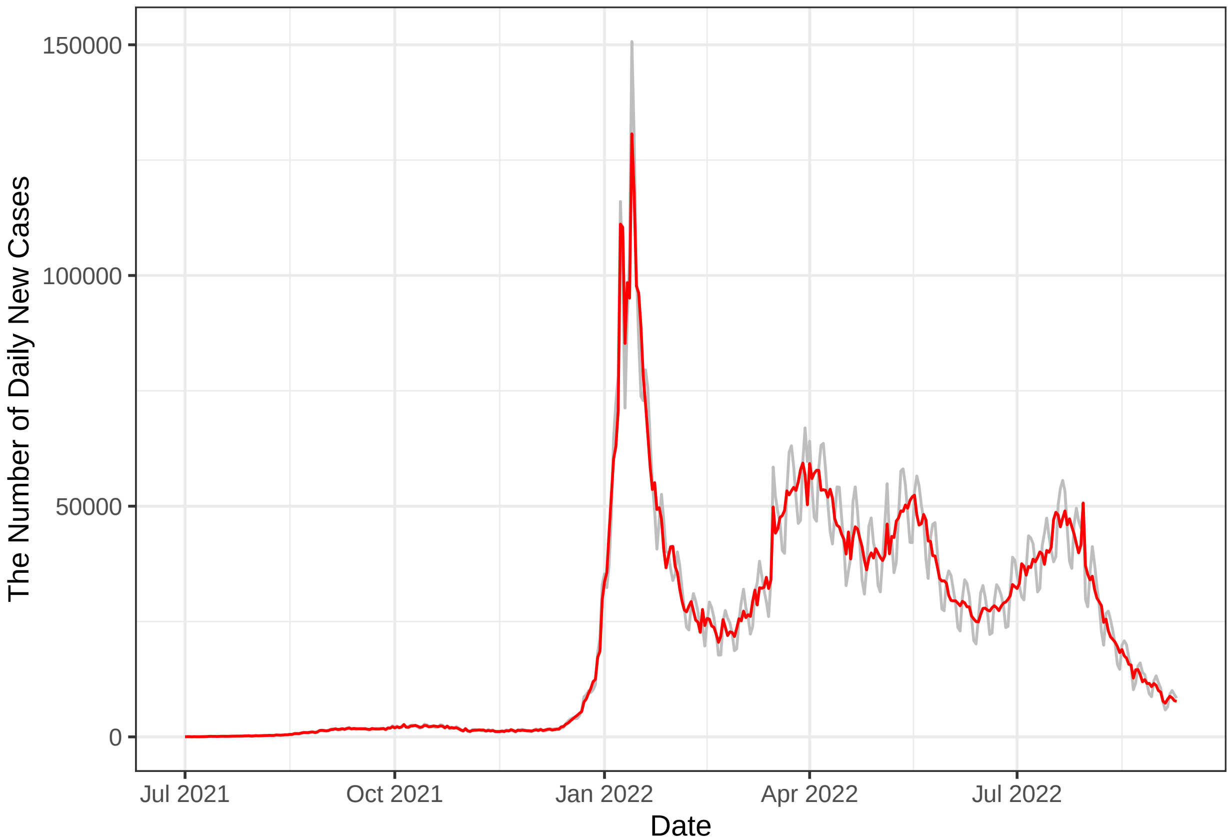

```
data_ts %>%
  filter(Date >= ymd("2021-07-01")) -> df
components(decomp) -> decomp_comp

ggplot() +
  geom_line(aes(x= Date, y = log(I)), color = "grey", data = df) +
  geom_line(aes(x = Date, y = season_adjust, color = "season_adjust"), data = decomp_comp) +
  ylab("log(I)")
```



Back transformation

```
decomp_comp %>%
  select(Date, season_adjust) %>%
  mutate(season_adjust = exp(season_adjust)) -> full_data_adj
```

```
ggplot() +
  geom_line(aes(x= Date, y = I), color = "grey", data = df) +
  geom_line(aes(x = Date, y = season_adjust), color = "red",
            data = full_data_adj) +
  ylab("The Number of Daily New Cases") +
  theme_bw()
```




```
colnames(full_data_adj) <- c("Date", "I")

data.frame(full_data_adj) -> data
```

Function

```
r_estiamte <- function(df, start, end, mean, std){
  output <- estimate_R(df, method = "parametric_si",
                  config = make_config(list(mean_si = mean, std_si = std,
                                            t_start = start,
                                            t_end = end)))
  output$dates <- data$Date

  return(output)
}

find_r <- function(date, r_df, full_df){
  r_matrix <- matrix(NA, nrow = length(date))
  for (i in 1:length(date)) {
    date_index <-which(full_df$Date == date[i])
    r_index <- which(r_df$R$t_end == date_index)
    r_matrix[i,] <- r_df$R$`Mean(R)`[r_index]
  }
  date_r = tibble("Date" = range, "R" = r_matrix[,1])
  return(date_r)
}
```

```
forecast_i <- function(r_date, full_df, r_df){

  output_df <- tibble("Date"= as.Date(NA),
                      "I" = as.numeric(NA),
                      "Week" = as.numeric(NA))

  for (i in 1:dim(r_date)[1]) {

    I_renew<-full_df$I[which(full_df$Date <= r_date$Date[i])]
    I_lambda <-I_renew[(length(I_renew) - 99):length(I_renew)]
    data.frame(r_df$si_distr)[,1][1:100] -> si
    predict_w1 <- matrix(NA, nrow = 7, ncol = 1)

    for (j in 1:7) {
      element <-  overall_infectivity(I_lambda, si)[100+j-1] * r_date$R[i]
      predict_w1[j,1] <- element
      I_lambda <- append(I_lambda, element)
      si <- append(si, 0)

      temp <- tibble("Date" = seq(ymd(r_date$Date[i]), ymd(r_date$Date[i])+6, "day"),
                     "I" = predict_w1[,1],
                     "Week" = i)
    }

    output_df <- bind_rows(output_df, temp)

  }

  output_df %>%
    drop_na() %>%
    mutate(Week = paste0("Period", Week)) -> output_df

  return(output_df)

}


add_season_pattern <- function(f_data, seasona_data){

  f_data$Date - 7 -> season_date

  seasona_data %>%
    filter(Date %in% season_date) -> seasona_df

  f_data$I <- f_data$I + seasona_df$season_week
  return(f_data)
}
```

set windows

```
#09/01 708
# one week
t_one <- seq(60, nrow(data)-7)
te_one <- t_one + 7
```

```r
# two week
t_two <- seq(60, nrow(data)-14)
te_two <- t_two + 14

# three week
t_three <- seq(60, nrow(data)-21)
te_three <- t_three + 21

# four week
t_four <- seq(60, nrow(data)-28)
te_four <- t_four + 28

# five week
t_five <- seq(60, nrow(data)-35)
te_five <- t_five + 35
```

Estimate R

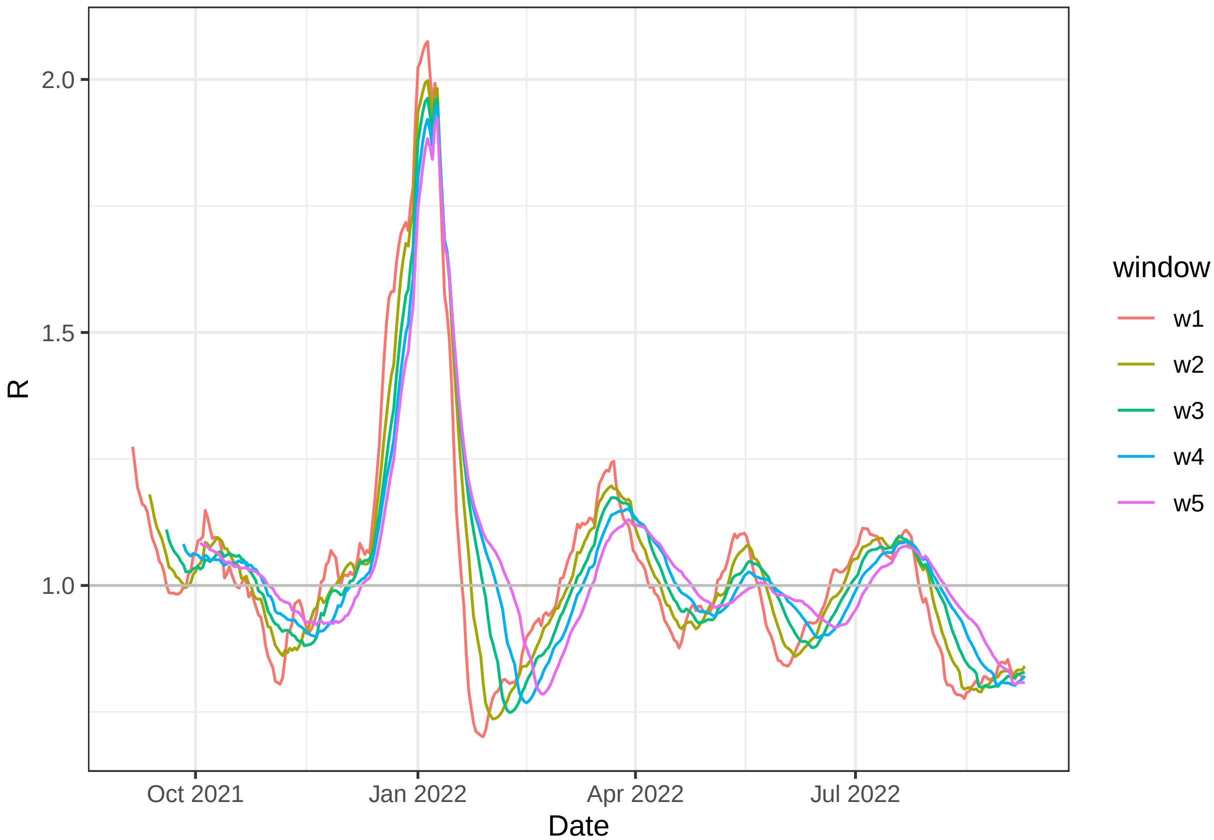
Using SI mean 4.7, std 2.9

```r
res_w1 <- r_estiamte(df = data, start = t_one, end = te_one, mean = 4.7, std = 2.9)
res_w2 <- r_estiamte(df = data, start = t_two, end = te_two, mean = 4.7, std = 2.9)
res_w3 <- r_estiamte(df = data, start = t_three, end = te_three, mean = 4.7, std = 2.9)
res_w4 <- r_estiamte(df = data, start = t_four, end = te_four, mean = 4.7, std = 2.9)
res_w5 <- r_estiamte(df = data, start = t_five, end = te_five, mean = 4.7, std = 2.9)
```

```r
tibble("Date" = data$Date[te_one],"R" = res_w1$R$`Mean(R)`, "window" = "w1") -> r_w1
tibble("Date" = data$Date[te_two],"R" = res_w2$R$`Mean(R)`, "window" = "w2") -> r_w2
tibble("Date" = data$Date[te_three],"R" = res_w3$R$`Mean(R)`, "window" = "w3") -> r_w3
tibble("Date" = data$Date[te_four],"R" = res_w4$R$`Mean(R)`, "window" = "w4") -> r_w4
tibble("Date" = data$Date[te_five],"R" = res_w5$R$`Mean(R)`, "window" = "w5") -> r_w5
```

```r
bind_rows(r_w1, r_w2) %>%
  bind_rows(r_w3) %>%
  bind_rows(r_w4) %>%
  bind_rows(r_w5) -> df_f
```

```r
df_f %>%
  ggplot() +
  geom_line(aes(x = Date, y = R, color = window)) +
  geom_hline(aes(yintercept = 1), color = "grey") +
  theme_bw()
```



Forecast

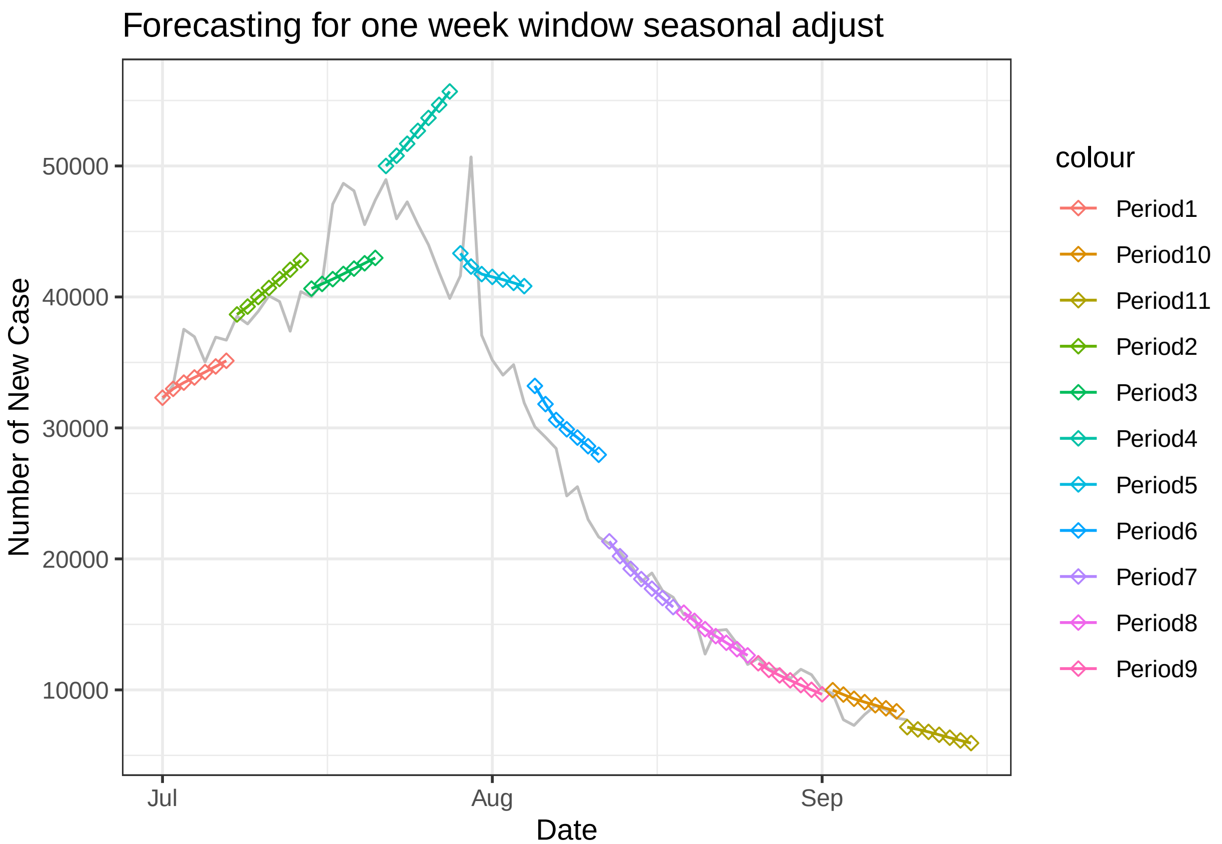
```
start_date <- ymd("2022-07-01")
end_date <- ymd("2022-09-09")
range <- seq(start_date, end_date,"week")

window_w1 <- find_r(date = range, r_df = res_w1, full_df = full_data_adj)
window_w2 <- find_r(date = range, r_df = res_w2, full_df = full_data_adj)
window_w3 <- find_r(date = range, r_df = res_w3, full_df = full_data_adj)
window_w4 <- find_r(date = range, r_df = res_w4, full_df = full_data_adj)
window_w5 <- find_r(date = range, r_df = res_w5, full_df = full_data_adj)
```

```
window1_predict <- forecast_i(r_date = window_w1, full_df = full_data_adj, r_df = res_w1)
window2_predict <- forecast_i(r_date = window_w2, full_df = full_data_adj, r_df = res_w2)
window3_predict <- forecast_i(r_date = window_w3, full_df = full_data_adj, r_df = res_w3)
window4_predict <- forecast_i(r_date = window_w4, full_df = full_data_adj, r_df = res_w4)
window5_predict <- forecast_i(r_date = window_w5, full_df = full_data_adj, r_df = res_w5)
```

Graph for seasonal adjust

```
decomp_comp %>%
  filter(Date >= "2022-07-01") %>%
  ggplot() +
  geom_line(aes(x = Date, y = exp(season_adjust) ,color = "Actual"),color = "grey") +
```

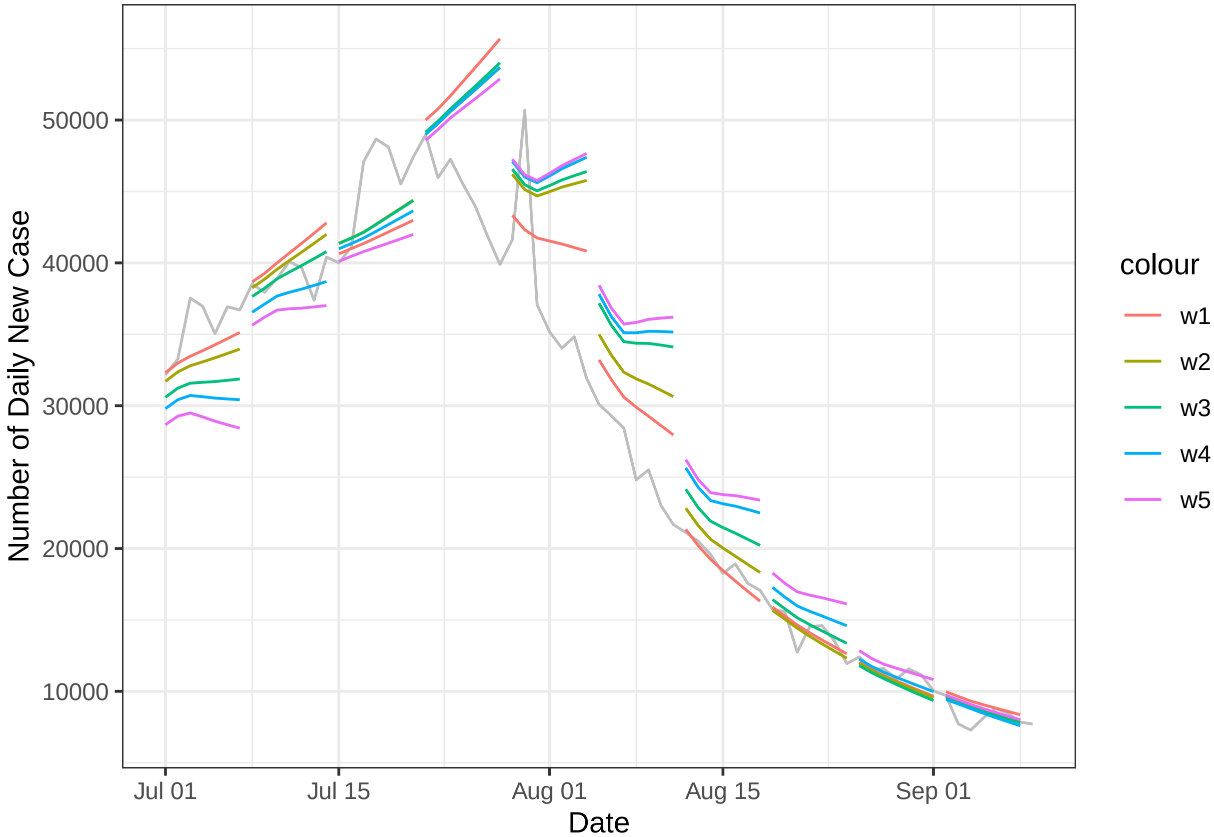
```
  geom_point(aes(x = Date, y = I,color = Week), shape = 5,data = window1_predict) +
  geom_line(aes(x = Date, y = I,color = Week),data = window1_predict) +
  theme_bw() +
  ylab("Number of New Case") +
  ggtitle("Forecasting for one week window seasonal adjust")
```




```
predict_df <- tibble("Date" = window1_predict$Date,
                     "w1" = window1_predict$I,
                     "w2" = window2_predict$I,
                     "w3" = window3_predict$I,
                     "w4" = window4_predict$I,
                     "w5" = window5_predict$I,
                     "week" = window1_predict$Week)
predict_df %>%
  pivot_longer(cols = -c("Date", "week"), names_to = "Windows",values_to = "I") %>%
  mutate(type = paste0(week, Windows)) %>%
  filter()-> predict_long
```

```
predict_long %>%
  filter(Date <= "2022-09-09") -> predict_long

decomp_comp %>%
  filter(Date >= "2022-07-01") %>%
  ggplot() +
  geom_line(aes(x = Date,  y = exp(season_adjust),color = "Actual"),color = "grey") +
  geom_line(aes(x = Date, y = I,group = interaction(week, Windows),
              color = Windows) ,data = predict_long)+
```

```
  #ggtitle("Forecasting value seasonal adjust") +
  ylab("Number of Daily New Case") +
  theme_bw()
```



It looks good for seasonal adjust

Add Back Seasonal pattern

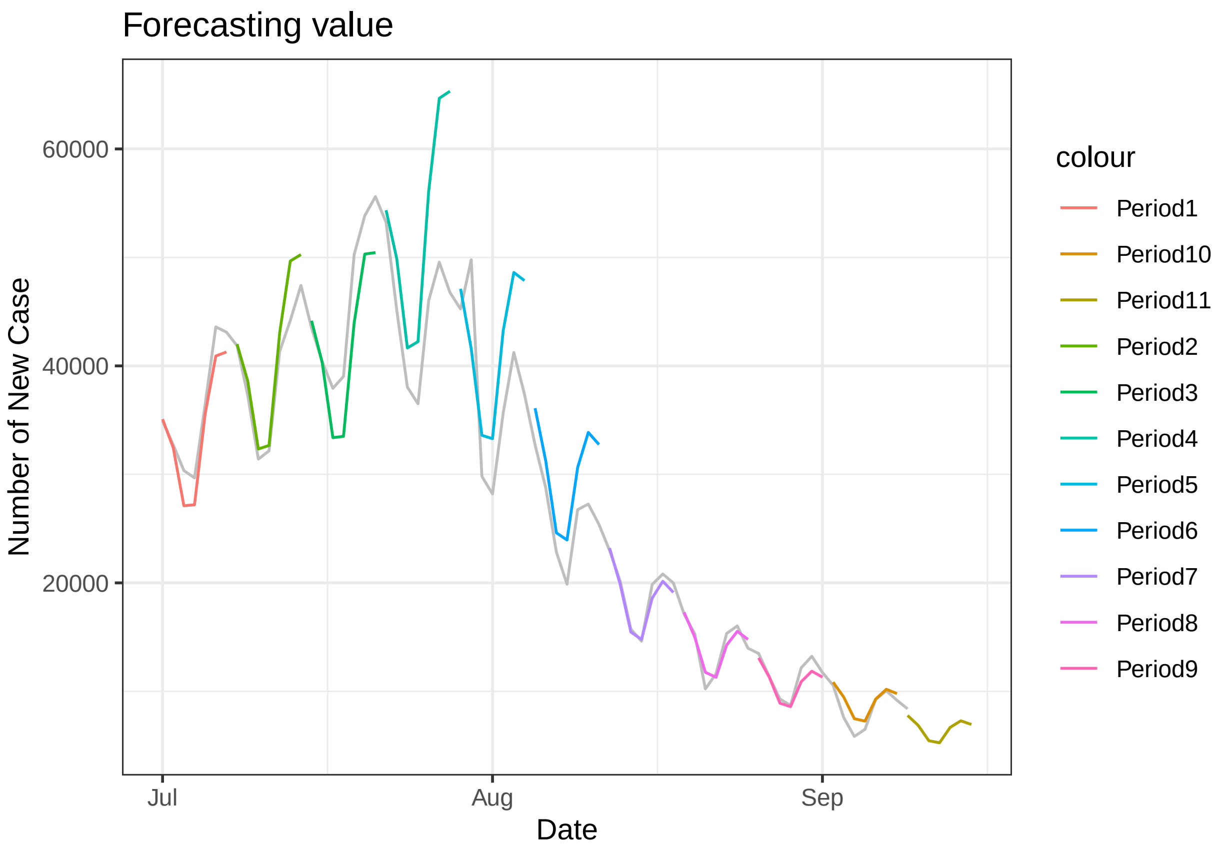
```
add_season_pattern <- function(f_data, seasona_data){

  f_data$Date - 7 -> season_date

  seasona_data %>%
    filter(Date %in% season_date) -> seasona_df

  f_data$I <- f_data$I + seasona_df$season_week
  return(f_data)
}
```

```
window1_predict <- add_season_pattern(f_data = window1_predict, seasona_data = decomp_comp)
window2_predict <- add_season_pattern(f_data = window2_predict, seasona_data = decomp_comp)
window3_predict <- add_season_pattern(f_data = window3_predict, seasona_data = decomp_comp)
window4_predict <- add_season_pattern(f_data = window4_predict, seasona_data = decomp_comp)
window5_predict <- add_season_pattern(f_data = window5_predict, seasona_data = decomp_comp)
```

```
predict_seasonal_df <- tibble("Date" = window1_predict$Date,
                              "w1" = window1_predict$I,
```

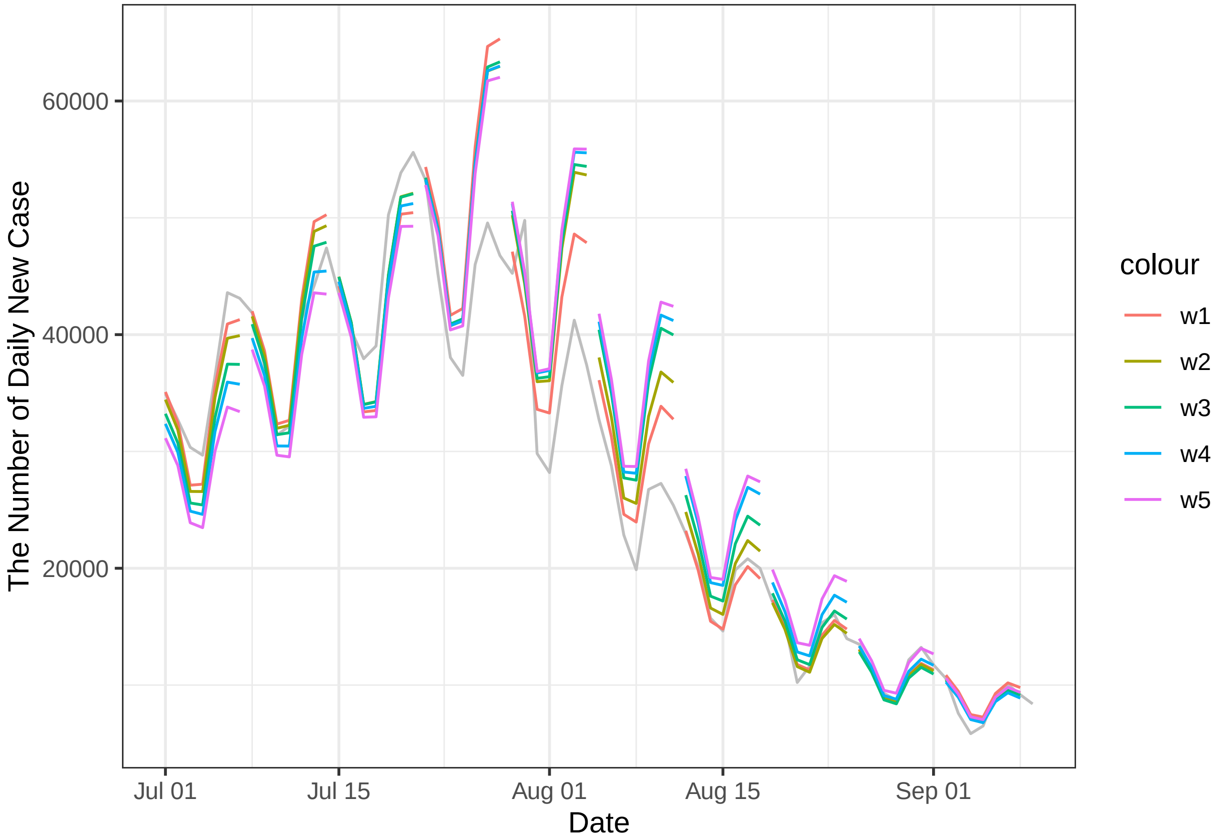
```
                     "w2" = window2_predict$I,
                     "w3" = window3_predict$I,
                     "w4" = window4_predict$I,
                     "w5" = window5_predict$I,
                     "week" = window1_predict$Week)
predict_seasonal_df %>%
  pivot_longer(cols = -c("Date", "week"), names_to = "Windows",values_to = "I") %>%
  mutate(type = paste0(week, Windows)) -> predict_season_long
```

```
predict_season_long %>%
  filter(Date <= "2022-09-09") -> predict_season_long

full_data %>%
  filter(Date >= "2022-07-01") %>%
  ggplot() +
  geom_line(aes(x = Date,  y = I,color = "Actual"),color = "grey") +
  geom_line(aes(x = Date, y = I, color = Week) ,data = window1_predict)+
  ggtitle("Forecasting value") +
  ylab("Number of New Case") +
  theme_bw()
```




```
full_data %>%
  filter(Date >= "2022-07-01") %>%
  ggplot() +
  geom_line(aes(x = Date,  y = I,color = "Actual"),color = "grey") +
  geom_line(aes(x = Date, y = I,group = interaction(week, Windows),
             color = Windows) ,data = predict_season_long)+
```

```
  #ggtitle("Forecasting value") +
  ylab("The Number of Daily New Case") +
  theme_bw()
```



I use back transformed seasonal adjust into my model. Then, I add seasonal pattern back. For the seasonal pattern, I do exp for seasonal_week from decomp__comp for. However, it is not very significant

Accuary

```
full_data %>%
  filter(Date %in% window1_predict$Date) -> fc_actual

window1_predict %>%
  filter(Date <= "2022-09-09") %>%
  mutate(resid = I -fc_actual$I,
         p = resid/I) %>%
  summarise(MAE = mean(abs(resid)),
            RMSE = sqrt(mean(resid^2)),
            MAPE = mean(abs(p)))
```

```
## # A tibble: 1 x 3
##     MAE  RMSE   MAPE
##   <dbl> <dbl>  <dbl>
## 1 2841. 4517. 0.0838
```

```
window2_predict %>%
  filter(Date <= "2022-09-09") %>%
```

```r
  mutate(resid = I -fc_actual$I,
         p = resid/I) %>%
  summarise(MAE = mean(abs(resid)),
            RMSE = sqrt(mean(resid^2)),
            MAPE = mean(abs(p)))
```

```
## # A tibble: 1 x 3
##     MAE  RMSE   MAPE
##   <dbl> <dbl>  <dbl>
## 1 3239. 5013. 0.0945
```

```r
window3_predict %>%
  filter(Date <= "2022-09-09") %>%
  mutate(resid = I -fc_actual$I,
         p = resid/I) %>%
  summarise(MAE = mean(abs(resid)),
            RMSE = sqrt(mean(resid^2)),
            MAPE = mean(abs(p)))
```

```
## # A tibble: 1 x 3
##     MAE  RMSE  MAPE
##   <dbl> <dbl> <dbl>
## 1 3846. 5713. 0.114
```

```r
window4_predict %>%
  filter(Date <= "2022-09-09") %>%
  mutate(resid = I -fc_actual$I,
         p = resid/I)%>%
  summarise(MAE = mean(abs(resid)),
            RMSE = sqrt(mean(resid^2)),
            MAPE = mean(abs(p)))
```

```
## # A tibble: 1 x 3
##     MAE  RMSE  MAPE
##   <dbl> <dbl> <dbl>
## 1 4390. 6200. 0.131
```

```r
window5_predict %>%
  filter(Date <= "2022-09-09") %>%
  mutate(resid = I -fc_actual$I,
         p = resid/I) %>%
  summarise(MAE = mean(abs(resid)),
            RMSE = sqrt(mean(resid^2)),
            MAPE = mean(abs(p)))
```

```
## # A tibble: 1 x 3
##     MAE  RMSE  MAPE
##   <dbl> <dbl> <dbl>
## 1 4957. 6655. 0.151
```