Group 3 Final Report

Kamran Shirazi & Taylor Kirk

2025-11-14

Table of contents

Introduction

The Industrial Revolution has been a boon to humanity, leading to longer life spans, increased wealth, easy access to clean water, it gave us Netflix…but with every benefit there comes a cost, and the main cost with expansion of human flourishing has been what we now call climate change. This term encapsulates a host of changes to our environment such as rising sea levels, more extreme weather events, reduction in biodiversity, etc. The climate changing isn't particularly new, the climate is always changing. What's unique about this period in Earth's history is the rate at which everything is changing. And the consensus is that rate of change is driven by human activity dumping CO2 emissions into the atmosphere, driving up CO2 concentration in the atmosphere, causing the global temperature to rise and leading to these rapid ecosystem alterations.

Project Problem/Statement (might need to generalize this since CO2 emissions came about trough testing different predictors)

This project was inspired by a desire to get involved in understanding the scope of the problem and hopefully finding a way to solve or mitigate it. Specifically we wanted to find a way to accurately isolate the impact of CO2 emissions on the increasing temperature trend, and by extension an individuals (defined as nation, state, city, organization etc) impact on the warming trend. The belief is that by quantifying impact, people will be better equipped to take action on changing their behavior and upgrading systems in ways that slow or reverse the warming trends.

Data Overview

```r
df <- read_csv("data/global_temp.csv")

glimpse(df, width = 45)
```

```
Rows: 2,109
Columns: 12
$ Year              <dbl> 1850, 1850, 1850~
$ Month             <dbl> 1, 2, 3, 4, 5, 6~
$ Monthly_Anomaly   <dbl> -0.469, -0.660, ~
$ Monthly_Unc       <dbl> 0.662, 0.456, 0.~
$ Annual_Anomaly    <dbl> NA, NA, NA, NA, ~
$ Annual_Unc        <dbl> NA, NA, NA, NA, ~
$ FiveYear_Anomaly  <dbl> NA, NA, NA, NA, ~
$ FiveYear_Unc      <dbl> NA, NA, NA, NA, ~
$ TenYear_Anomaly   <dbl> NA, NA, NA, NA, ~
$ TenYear_Unc       <dbl> NA, NA, NA, NA, ~
$ TwentyYear_Anomaly <dbl> NA, NA, NA, NA, ~
$ TwentyYear_Unc    <dbl> NA, NA, NA, NA, ~
```

```r
summary(df[3:6], digits = 2)
```

```
 Monthly_Anomaly    Monthly_Unc     Annual_Anomaly      Annual_Unc   
 Min.   :-0.8130   Min.   :0.020   Min.   :-0.5640   Min.   :0.015  
 1st Qu.:-0.2850   1st Qu.:0.069   1st Qu.:-0.2780   1st Qu.:0.037  
 Median :-0.0870   Median :0.156   Median :-0.1060   Median :0.096  
 Mean   : 0.0069   Mean   :0.182   Mean   : 0.0052   Mean   :0.110  
 3rd Qu.: 0.1990   3rd Qu.:0.262   3rd Qu.: 0.1797   3rd Qu.:0.168  
 Max.   : 1.4490   Max.   :0.770   Max.   : 1.3050   Max.   :0.360  
                                   NA's   :11        NA's   :11     
```

The data from **Berkely Earth** (insert ref) comes in the form of monthly anomalies. They use a baseline period (1950-1980) and record temperature observations as differences from the average temp (by month) of this period. They include the uncertainties, and 5, 10, and 20 year lag periods. For our purposes, we're only concerned with the Monthly_Monthly_Anomaly column

Convert to Tsibble and Temperatures

```r
Anomaly_data <- df |> mutate(
  Month = month.abb[Month],
  Date = yearmonth(paste(Month, Year)),
  Monthly_Anomaly = replace_na(Monthly_Anomaly, mean(Monthly_Anomaly, na.rm = T))) |>
  select(c(Date, Monthly_Anomaly)) |>
  as_tsibble(index = Date)

baseline_vector <- c(
  '1' = 12.30, '2' = 12.50, '3' = 13.13, '4' = 14.06,
  '5' = 15.00, '6' = 15.66, '7' = 15.90, '8' = 15.75,
  '9' = 15.18, '10' = 14.27, '11' = 13.28, '12' = 12.57
)

converted_df <- Anomaly_data |>
  mutate(
    month_char = as.character(month(Date)),
    baseline_temp = baseline_vector[month_char],
    actual_temp = baseline_temp + Monthly_Anomaly
    ) |>
  select(c(Date, actual_temp))
```

```r
converted_df |>
  head() |>
  kable()
```

Date	actual_temp
1850 Jan	11.831
1850 Feb	11.840
1850 Mar	12.703
1850 Apr	13.250
1850 May	14.625
1850 Jun	15.298

Reporting in monthly anomalies is common for publications given that the main interest is usually in the overall trend of warming. However for our purposes, since we are attempting to isolate the human effect on the warming trend, we also need to be able to accurately model temperature separate of human impact. The seasonality of global temperatures is a big part of that and using only the monthly Monthly_Anomaly measurements largely strips out that component. Both sets of data will be explored and modeled however to see which one is most useful and it may turn out that each is valuable in different capacities.

Exploratory Data Analysis of Global Anomalies

```r
glimpse(df, width = 50)
```

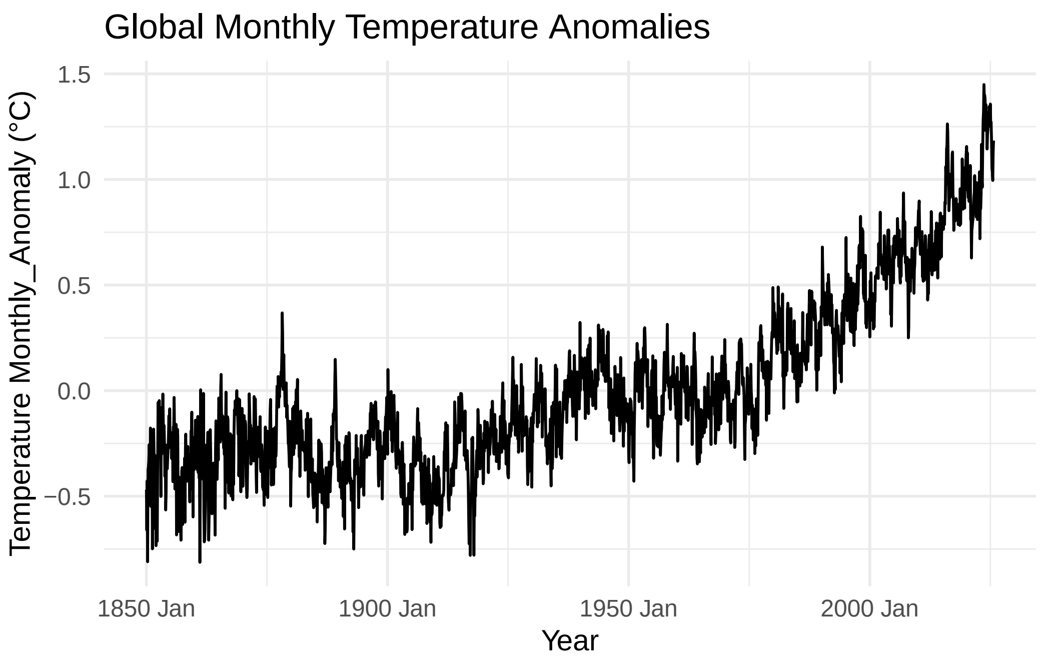
```
Rows: 2,109
Columns: 12
$ Year               <dbl> 1850, 1850, 1850, 185~
$ Month              <dbl> 1, 2, 3, 4, 5, 6, 7, ~
$ Monthly_Anomaly    <dbl> -0.469, -0.660, -0.42~
$ Monthly_Unc        <dbl> 0.662, 0.456, 0.611, ~
$ Annual_Anomaly     <dbl> NA, NA, NA, NA, NA, -~
$ Annual_Unc         <dbl> NA, NA, NA, NA, NA, 0~
$ FiveYear_Anomaly   <dbl> NA, NA, NA, NA, NA, N~
$ FiveYear_Unc       <dbl> NA, NA, NA, NA, NA, N~
$ TenYear_Anomaly    <dbl> NA, NA, NA, NA, NA, N~
$ TenYear_Unc        <dbl> NA, NA, NA, NA, NA, N~
$ TwentyYear_Anomaly <dbl> NA, NA, NA, NA, NA, N~
$ TwentyYear_Unc     <dbl> NA, NA, NA, NA, NA, N~
```

```r
names(df)
```

```
 [1] "Year"               "Month"              "Monthly_Anomaly"   
 [4] "Monthly_Unc"        "Annual_Anomaly"     "Annual_Unc"        
 [7] "FiveYear_Anomaly"   "FiveYear_Unc"       "TenYear_Anomaly"   
[10] "TenYear_Unc"        "TwentyYear_Anomaly" "TwentyYear_Unc"    
```

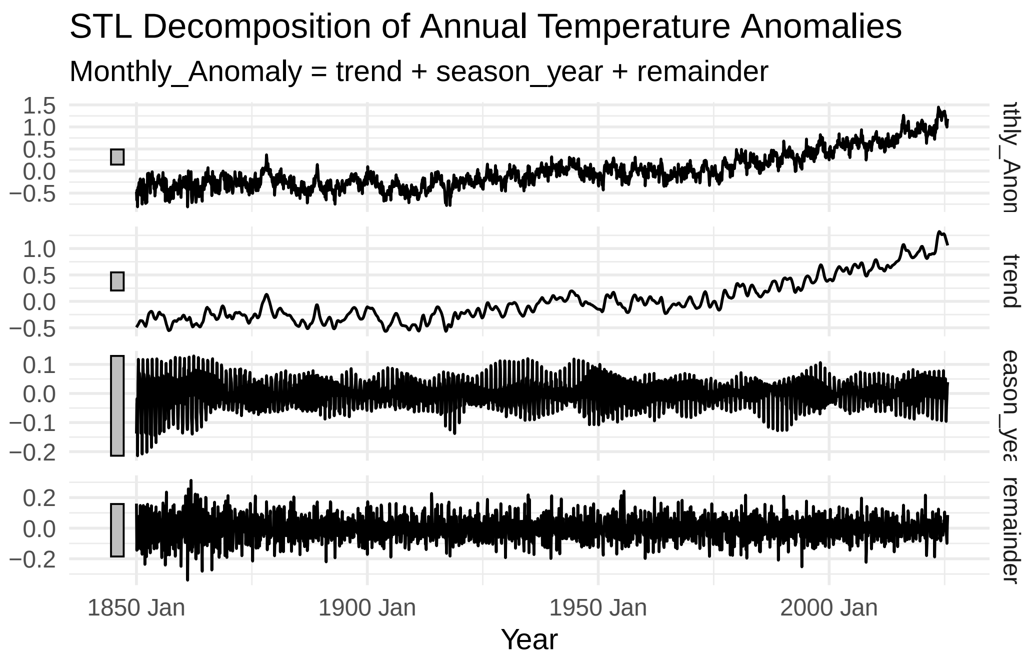
This dataset contains 2,109 monthly observations from 1850–2025, with global temperature anomalies and uncertainties, where **Monthly__Monthly__Anomaly** is the primary usable series and most multi-year Monthly_Anomaly fields contain NA values except at the end of their smoothing windows.

```r
train <- Anomaly_data|> filter(Date < yearmonth("2020 Jan"))
test  <- Anomaly_data|> filter(Date >= yearmonth("2020 Jan"))

autoplot(Anomaly_data, Monthly_Anomaly) +
  labs(title = "Global Monthly Temperature Anomalies",
       x = "Year", y = "Temperature Monthly_Anomaly (°C)") +
  theme_minimal()
```



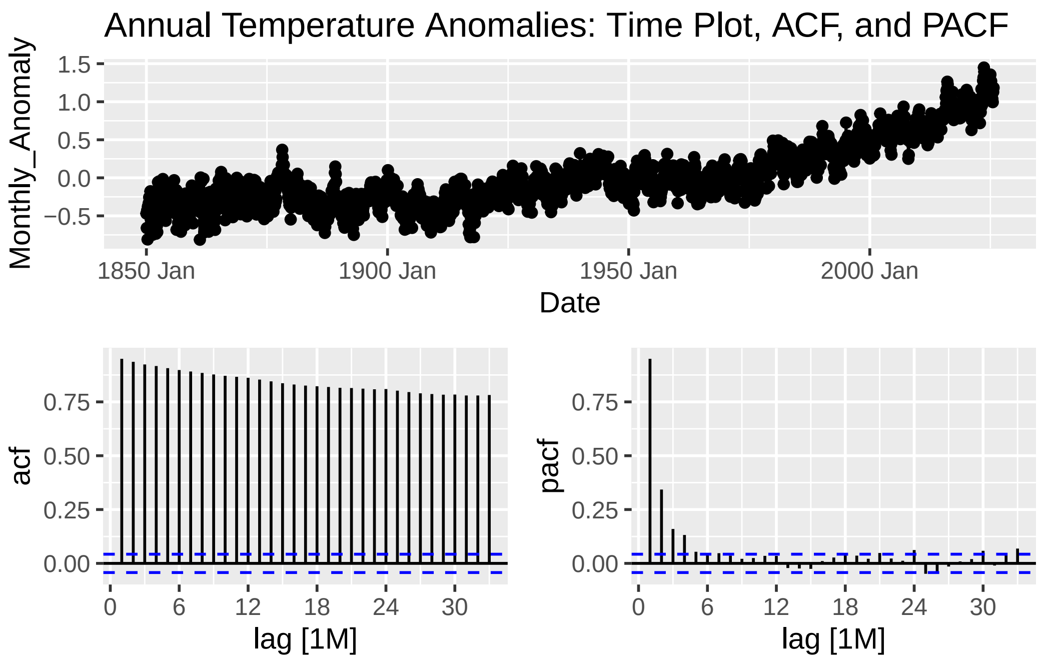
The plot shows that what was once mostly cooler-than-average (average as defined as the period between 1950 - 1980) global temperatures has transitioned into a new normal of sustained warming, particularly over the last 40 years. The sharp upward trend in recent decades signals the accelerating pace of climate change.

```r
climate_stl <- Anomaly_data |>
  model(
    STL(Monthly_Anomaly ~ trend(window = 15))
  ) |>
  components()

autoplot(climate_stl) +
  labs(
    title = "STL Decomposition of Annual Temperature Anomalies",
    x = "Year"
  ) +
  theme_minimal()
```



The STL results highlight how the underlying warming signal has grown stronger over time, while the remainder shows noisy but relatively modest deviations from the trend. This reinforces that annual variability exists but is dwarfed by the long-term increase in global temperatures.

```r
Anomaly_data |>
  gg_tsdisplay(Monthly_Anomaly, plot_type = "partial") +
  labs(
    title = "Annual Temperature Anomalies: Time Plot, ACF, and PACF"
  )
```



The climate series exhibits a persistent warming trend, and the ACF's slow decay underscores how each year's temperature is tightly linked to previous years. The PACF's immediate drop-off reinforces that this warming signal creates strong year-to-year inertia in global temperatures.

```
# KPSS
Anomaly_data |>
  features(Monthly_Anomaly, unitroot_kpss) |>
  kable(digits = 4, align = "c")
```

kpss_stat	kpss_pvalue
17.4329	0.01

The KPSS test confirms what the plots already suggest: the Monthly_Anomaly series isn't stationary, with the low p-value indicating that the warming trend dominates over random fluctuations.

```r
# Non-seasonal differencing needed?
Anomaly_data |>
  features(Monthly_Anomaly, c(unitroot_ndiffs, unitroot_nsdiffs)) |>
  kable(digits = 4, align = "c")
```

ndiffs	nsdiffs
1	0

```r
Anomaly_data |>
  features(Monthly_Anomaly, guerrero) |>
  kable(digits = 4, align = "c")
```

lambda_guerrero
0.9662

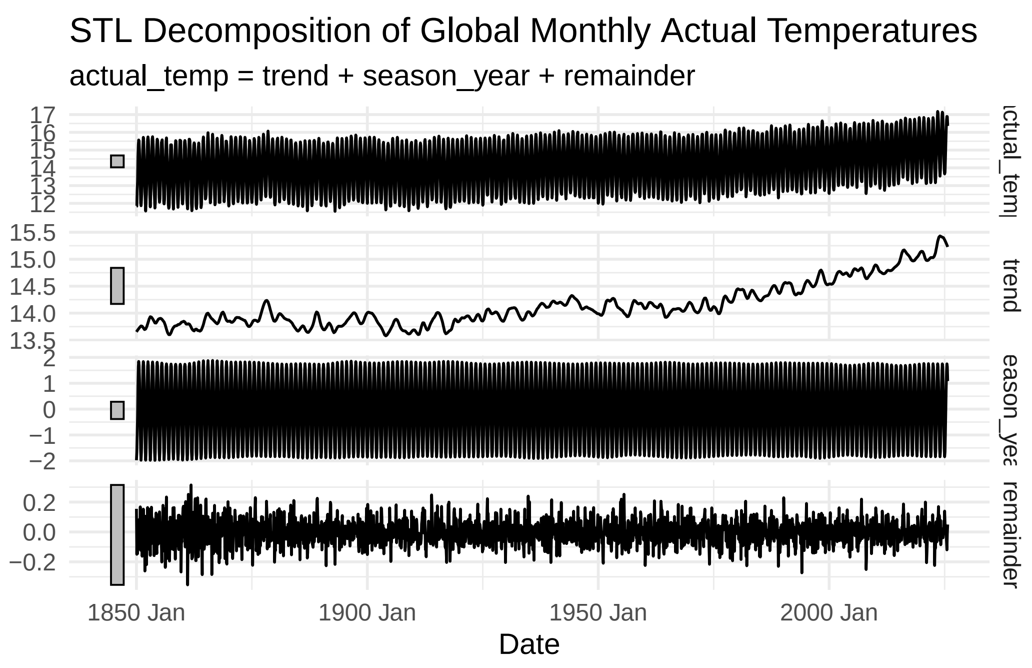
$\lambda \approx 0.966 \rightarrow$ little to no Box–Cox transformation needed; variance is already stable. Unitroot test suggests differencing may be needed

Exploratory Data Analysis of Global Temperatures

```r
summary(converted_df[2])
```

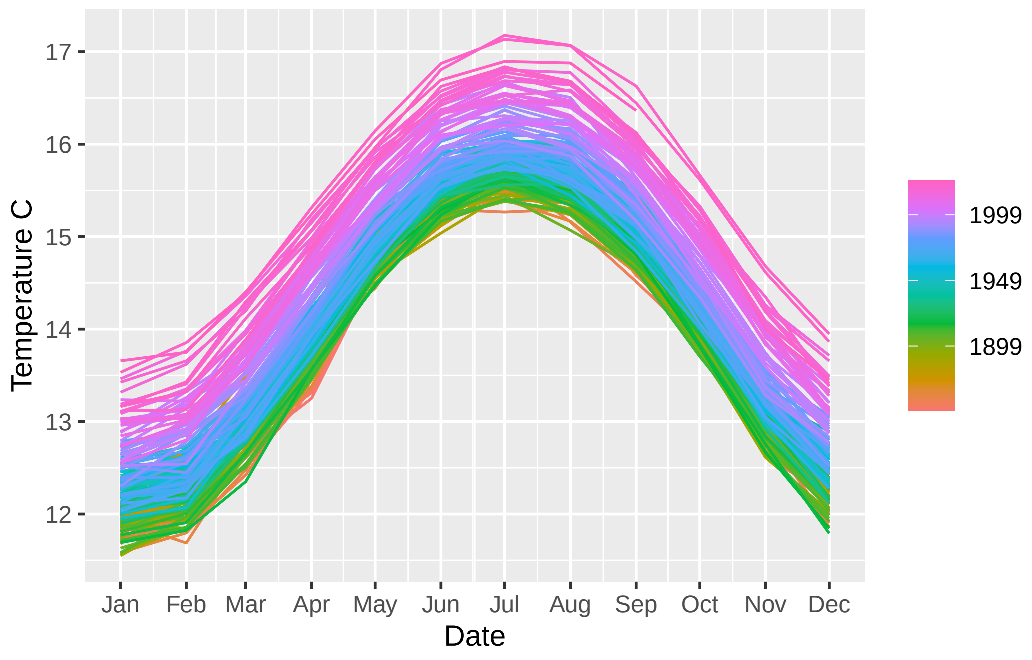
```
  actual_temp   
 Min.   :11.55  
 1st Qu.:12.89  
 Median :14.17  
 Mean   :14.14  
 3rd Qu.:15.40  
 Max.   :17.18  
```

```r
converted_df |>
  model(STL(actual_temp)) |>
  components() |>
  autoplot() +
  labs(title = "STL Decomposition of Global Monthly Actual Temperatures") +
  theme_minimal()
```



Decomposition of global temperatures show seasonality to mainly be constant with some compression in later years (could be a result of more precise measurements, could also be a result of warming trends flattening temperature swings each season). The main driver of the changing level is the trend which stayed fairly constant until 1920 when the trend starts to increase, then really takes off after 1980

```r
converted_df |>
  ggtime::gg_season() +
  labs(y = "Temperature C")
```



```r
theme_minimal()
```

```
<theme> List of 144
 $ line                            : <ggplot2::element_line>
  ..@ colour       : chr "black"
  ..@ linewidth    : num 0.5
  ..@ linetype     : num 1
  ..@ lineend      : chr "butt"
  ..@ linejoin     : chr "round"
  ..@ arrow        : logi FALSE
  ..@ arrow.fill   : chr "black"
  ..@ inherit.blank: logi TRUE
 $ rect                            : <ggplot2::element_rect>
  ..@ fill         : chr "white"
  ..@ colour       : chr "black"
  ..@ linewidth    : num 0.5
  ..@ linetype     : num 1
  ..@ linejoin     : chr "round"
  ..@ inherit.blank: logi TRUE
 $ text                            : <ggplot2::element_text>
  ..@ family       : chr ""
  ..@ face         : chr "plain"
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : chr "black"
  ..@ size         : num 11
  ..@ hjust        : num 0.5
  ..@ vjust        : num 0.5
  ..@ angle        : num 0
  ..@ lineheight   : num 0.9
  ..@ margin       : <ggplot2::margin> num [1:4] 0 0 0 0
  ..@ debug        : logi FALSE
  ..@ inherit.blank: logi TRUE
 $ title                            : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : NULL
  ..@ size         : NULL
  ..@ hjust        : NULL
  ..@ vjust        : NULL
  ..@ angle        : NULL
  ..@ lineheight   : NULL
  ..@ margin       : NULL
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ point                            : <ggplot2::element_point>
  ..@ colour       : chr "black"
  ..@ shape        : num 19
  ..@ size         : num 1.5
  ..@ fill         : chr "white"
  ..@ stroke       : num 0.5
  ..@ inherit.blank: logi TRUE
 $ polygon                          : <ggplot2::element_polygon>
  ..@ fill         : chr "white"
  ..@ colour       : chr "black"
  ..@ linewidth    : num 0.5
  ..@ linetype     : num 1
  ..@ linejoin     : chr "round"
  ..@ inherit.blank: logi TRUE
 $ geom                             : <ggplot2::element_geom>
  ..@ ink        : chr "black"
  ..@ paper      : chr "white"
  ..@ accent     : chr "#3366FF"
  ..@ linewidth  : num 0.5
  ..@ borderwidth: num 0.5
  ..@ linetype   : int 1
  ..@ bordertype : int 1
  ..@ family     : chr ""
  ..@ fontsize   : num 3.87
  ..@ pointsize  : num 1.5
  ..@ pointshape : num 19
  ..@ colour     : NULL
  ..@ fill       : NULL
 $ spacing                          : 'simpleUnit' num 5.5points
  ..- attr(*, "unit")= int 8
 $ margins                          : <ggplot2::margin> num [1:4] 5.5 5.5 5.5 5.5
 $ aspect.ratio                     : NULL
 $ axis.title                       : NULL
 $ axis.title.x                     : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : NULL
  ..@ size         : NULL
  ..@ hjust        : NULL
  ..@ vjust        : num 1
  ..@ angle        : NULL
  ..@ lineheight   : NULL
  ..@ margin       : <ggplot2::margin> num [1:4] 2.75 0 0 0
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ axis.title.x.top                 : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : NULL
  ..@ size         : NULL
  ..@ hjust        : NULL
  ..@ vjust        : num 0
  ..@ angle        : NULL
```

```
  ..@ lineheight   : NULL
  ..@ margin       : <ggplot2::margin> num [1:4] 0 0 2.75 0
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ axis.title.x.bottom            : NULL
 $ axis.title.y                   : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : NULL
  ..@ size         : NULL
  ..@ hjust        : NULL
  ..@ vjust        : num 1
  ..@ angle        : num 90
  ..@ lineheight   : NULL
  ..@ margin       : <ggplot2::margin> num [1:4] 0 2.75 0 0
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ axis.title.y.left              : NULL
 $ axis.title.y.right             : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : NULL
  ..@ size         : NULL
  ..@ hjust        : NULL
  ..@ vjust        : num 1
  ..@ angle        : num -90
  ..@ lineheight   : NULL
  ..@ margin       : <ggplot2::margin> num [1:4] 0 0 0 2.75
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ axis.text                      : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : chr "#4D4D4DFF"
  ..@ size         : 'rel' num 0.8
  ..@ hjust        : NULL
  ..@ vjust        : NULL
  ..@ angle        : NULL
  ..@ lineheight   : NULL
  ..@ margin       : NULL
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ axis.text.x                    : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : NULL
  ..@ size         : NULL
  ..@ hjust        : NULL
  ..@ vjust        : num 1
  ..@ angle        : NULL
  ..@ lineheight   : NULL
  ..@ margin       : <ggplot2::margin> num [1:4] 2.2 0 0 0
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ axis.text.x.top                : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : NULL
  ..@ size         : NULL
  ..@ hjust        : NULL
  ..@ vjust        : NULL
  ..@ angle        : NULL
  ..@ lineheight   : NULL
  ..@ margin       : <ggplot2::margin> num [1:4] 0 0 4.95 0
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ axis.text.x.bottom             : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : NULL
  ..@ size         : NULL
  ..@ hjust        : NULL
  ..@ vjust        : NULL
  ..@ angle        : NULL
  ..@ lineheight   : NULL
  ..@ margin       : <ggplot2::margin> num [1:4] 4.95 0 0 0
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ axis.text.y                     : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : NULL
  ..@ size         : NULL
  ..@ hjust        : num 1
  ..@ vjust        : NULL
  ..@ angle        : NULL
  ..@ lineheight   : NULL
  ..@ margin       : <ggplot2::margin> num [1:4] 0 2.2 0 0
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ axis.text.y.left                : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : NULL
  ..@ size         : NULL
  ..@ hjust        : NULL
  ..@ vjust        : NULL
  ..@ angle        : NULL
  ..@ lineheight   : NULL
  ..@ margin       : <ggplot2::margin> num [1:4] 0 4.95 0 0
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ axis.text.y.right               : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
```

```
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : NULL
  ..@ size         : NULL
  ..@ hjust        : NULL
  ..@ vjust        : NULL
  ..@ angle        : NULL
  ..@ lineheight   : NULL
  ..@ margin       : <ggplot2::margin> num [1:4] 0 0 0 4.95
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ axis.text.theta                : NULL
 $ axis.text.r                    : <ggplot2::element_text>
  ..@ family       : NULL
  ..@ face         : NULL
  ..@ italic       : chr NA
  ..@ fontweight   : num NA
  ..@ fontwidth    : num NA
  ..@ colour       : NULL
  ..@ size         : NULL
  ..@ hjust        : num 0.5
  ..@ vjust        : NULL
  ..@ angle        : NULL
  ..@ lineheight   : NULL
  ..@ margin       : <ggplot2::margin> num [1:4] 0 2.2 0 2.2
  ..@ debug        : NULL
  ..@ inherit.blank: logi TRUE
 $ axis.ticks                     : <ggplot2::element_blank>
 $ axis.ticks.x                   : NULL
 $ axis.ticks.x.top               : NULL
 $ axis.ticks.x.bottom            : NULL
 $ axis.ticks.y                   : NULL
 $ axis.ticks.y.left              : NULL
 $ axis.ticks.y.right             : NULL
 $ axis.ticks.theta               : NULL
 $ axis.ticks.r                   : NULL
 $ axis.minor.ticks.x.top         : NULL
 $ axis.minor.ticks.x.bottom      : NULL
 $ axis.minor.ticks.y.left        : NULL
 $ axis.minor.ticks.y.right       : NULL
 $ axis.minor.ticks.theta         : NULL
 $ axis.minor.ticks.r             : NULL
```

```
$ axis.ticks.length               : 'rel' num 0.5
$ axis.ticks.length.x             : NULL
$ axis.ticks.length.x.top         : NULL
$ axis.ticks.length.x.bottom      : NULL
$ axis.ticks.length.y             : NULL
$ axis.ticks.length.y.left        : NULL
$ axis.ticks.length.y.right       : NULL
$ axis.ticks.length.theta         : NULL
$ axis.ticks.length.r             : NULL
$ axis.minor.ticks.length         : 'rel' num 0.75
$ axis.minor.ticks.length.x       : NULL
$ axis.minor.ticks.length.x.top   : NULL
$ axis.minor.ticks.length.x.bottom: NULL
$ axis.minor.ticks.length.y       : NULL
$ axis.minor.ticks.length.y.left  : NULL
$ axis.minor.ticks.length.y.right : NULL
$ axis.minor.ticks.length.theta   : NULL
$ axis.minor.ticks.length.r       : NULL
$ axis.line                       : <ggplot2::element_blank>
$ axis.line.x                     : NULL
$ axis.line.x.top                 : NULL
$ axis.line.x.bottom              : NULL
$ axis.line.y                     : NULL
$ axis.line.y.left                : NULL
$ axis.line.y.right               : NULL
$ axis.line.theta                 : NULL
$ axis.line.r                     : NULL
$ legend.background               : <ggplot2::element_blank>
$ legend.margin                   : NULL
$ legend.spacing                  : 'rel' num 2
$ legend.spacing.x                : NULL
$ legend.spacing.y                : NULL
$ legend.key                      : <ggplot2::element_blank>
$ legend.key.size                 : 'simpleUnit' num 1.2lines
 ..- attr(*, "unit")= int 3
$ legend.key.height               : NULL
$ legend.key.width                : NULL
$ legend.key.spacing              : NULL
$ legend.key.spacing.x            : NULL
$ legend.key.spacing.y            : NULL
$ legend.key.justification        : NULL
$ legend.frame                    : NULL
$ legend.ticks                    : NULL
$ legend.ticks.length              : 'rel' num 0.2
$ legend.axis.line                 : NULL
$ legend.text                      : <ggplot2::element_text>
 ..@ family       : NULL
 ..@ face         : NULL
 ..@ italic       : chr NA
 ..@ fontweight   : num NA
 ..@ fontwidth    : num NA
 ..@ colour       : NULL
 ..@ size         : 'rel' num 0.8
 ..@ hjust        : NULL
 ..@ vjust        : NULL
 ..@ angle        : NULL
 ..@ lineheight   : NULL
 ..@ margin       : NULL
 ..@ debug        : NULL
 ..@ inherit.blank: logi TRUE
$ legend.text.position             : NULL
$ legend.title                     : <ggplot2::element_text>
 ..@ family       : NULL
 ..@ face         : NULL
 ..@ italic       : chr NA
 ..@ fontweight   : num NA
 ..@ fontwidth    : num NA
 ..@ colour       : NULL
 ..@ size         : NULL
 ..@ hjust        : num 0
 ..@ vjust        : NULL
 ..@ angle        : NULL
 ..@ lineheight   : NULL
 ..@ margin       : NULL
 ..@ debug        : NULL
 ..@ inherit.blank: logi TRUE
$ legend.title.position            : NULL
$ legend.position                  : chr "right"
$ legend.position.inside           : NULL
$ legend.direction                 : NULL
$ legend.byrow                     : NULL
$ legend.justification             : chr "center"
$ legend.justification.top         : NULL
$ legend.justification.bottom      : NULL
$ legend.justification.left        : NULL
$ legend.justification.right       : NULL
```

```
 $ legend.justification.inside     : NULL
  [list output truncated]
 @ complete: logi TRUE
 @ validate: logi TRUE
```

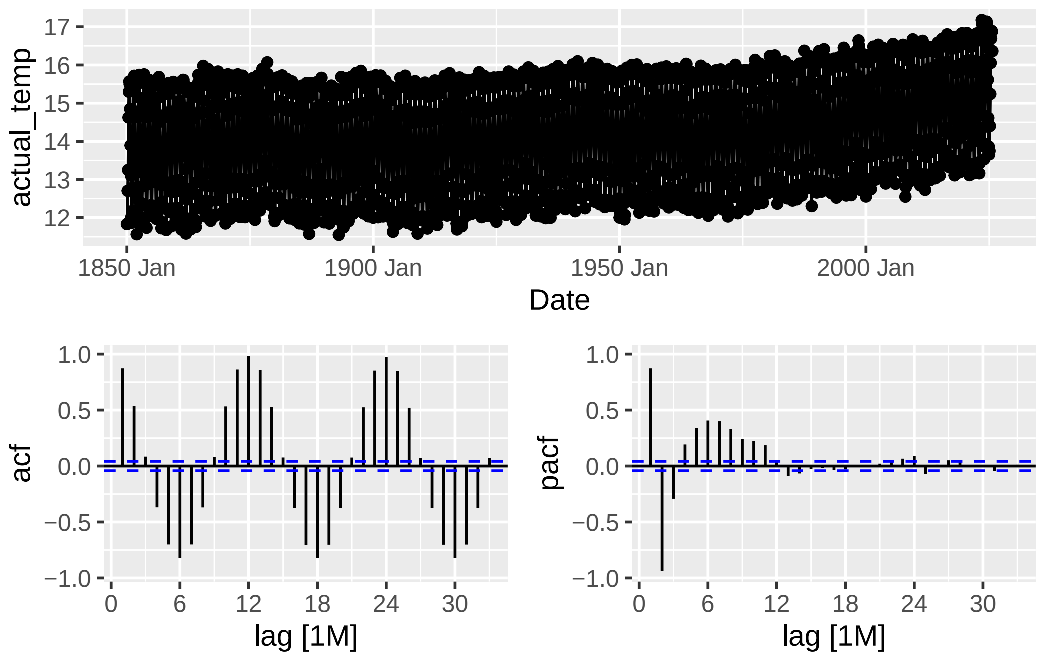
Clearer picture of the increasing trend over time, occurring across all periods

```
converted_df |>
  features(actual_temp, guerrero) |>
  kable(digits = 4, align = "c")
```

lambda_guerrero
1.4379

$\lambda \approx 1.44$, so a power transformation may be helpful

```
gg_tsdisplay(converted_df, actual_temp, plot_type = 'partial')
```



```
converted_df |>
  features(actual_temp, c(unitroot_kpss, unitroot_ndiffs, unitroot_nsdiffs)) |>
  kable(digits = 4, align = "c")
```

kpss_stat	kpss_pvalue	ndiffs	nsdiffs
8.8813	0.01	1	1

Unitroot tests suggest at least one difference and one seasonal difference. Using `gg_tsdisplay`, the seasonal autocorrelation is obvious, with the summer and winter months grouping together on opposite sides of the line. The pacf shows that the seasonal effects are strong. All of this shows the data is not stationary

Modeling / Forecasting

Monthly_Anomaly Modeling

```
# Fit ETS on the training data

fit_ets <- train |>
model(ETS(Monthly_Anomaly))

# Print model details

report(fit_ets)
```

```
Series: Monthly_Anomaly
Model: ETS(A,N,A)
  Smoothing parameters:
    alpha = 0.4616505
    gamma = 0.0001007547

  Initial states:
      l[0]        s[0]        s[-1]      s[-2]       s[-3]      s[-4]      s[-5]
 -0.504211 -0.01987071 -0.01378345 0.03587174 -0.01243867 0.01035226 0.02029624
      s[-6]      s[-7]        s[-8]       s[-9]     s[-10]      s[-11]
 0.03041389 0.01181375 -0.004461374 -0.01081024 -0.0249877 -0.02239573
```

```
  sigma^2:  0.013

     AIC     AICc      BIC 
6708.789 6709.026 6793.099 
```

```
fc_ets <- fit_ets |>
  forecast(new_data = test)

fc_ets_zoom <- fc_ets|>
  filter_index("2020 Jan" ~ "2025 Dec")

autoplot(
  fc_ets_zoom,
  Anomaly_data |> filter_index("2020 Jan" ~ "2025 Dec")
  ) +
  labs(
    title = "Auto-ETS Forecast vs Actuals (2020-2025)",
    subtitle = "Automatically selected ETS model",
    x = "Year",
    y = "Temperature Monthly_Anomaly (°C)"
    )
```

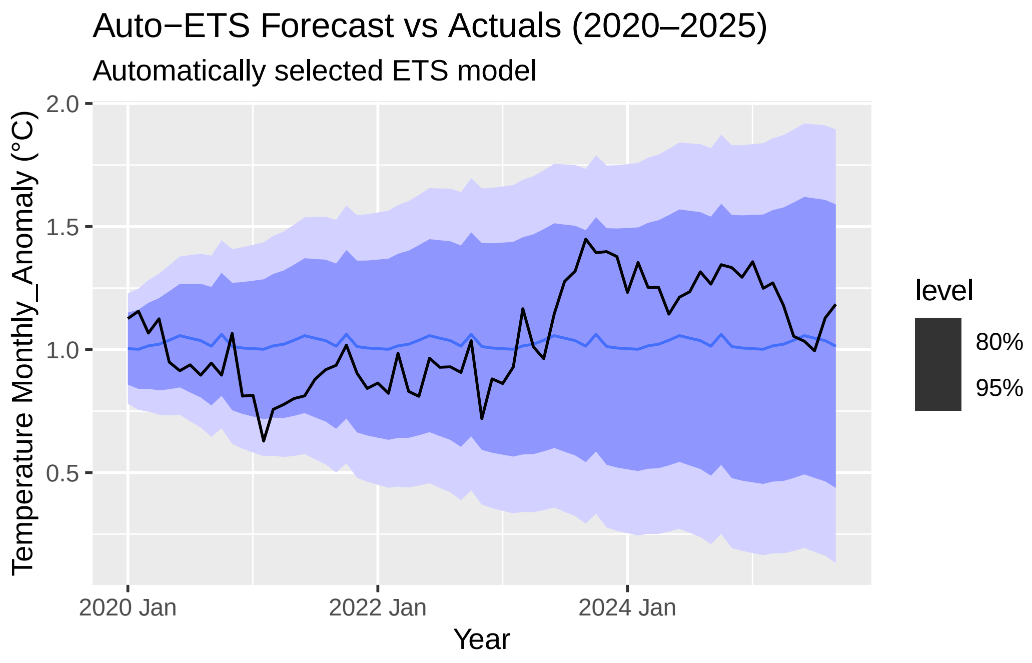


Figure 1: Auto-ETS forecast vs actuals (2020–2025).

The ETS model anticipates a relatively stable Monthly_Anomaly level, but the actual values frequently climb toward the upper edge of the 80% and even 95% prediction intervals. This pattern indicates that recent warming spikes are occurring faster and more intensely than the model expects based on historical structure. When observations consistently press against the top of the interval bands, it suggests the model may be underestimating both the trend strength and the volatility of contemporary climate behavior.

Temperature Modeling

```r
cv_data <- converted_df |>
  stretch_tsibble(.init = 1506, .step = 60)

cv_trn <- cv_data |>
  group_by(.id) |>
  slice(1:(n() - 60)) |>
  ungroup()

cv_valid <- cv_data |>
  group_by(.id) |>
  slice_tail(n = 60) |>
  ungroup()
```

Modeling on all temp data with cross validation sets created from the first roughly 125 years of data (1850 - 1930) and rolling forward by 5 years (60 months). This creates 11 cv splits covering the range of available observations

ETS

```r
ets_fit <- cv_trn |>
  model(
    ets_auto = ETS(),
    additive = ETS(actual_temp ~ error("A") + trend("A") + season("A")),
    multiplicative = ETS(actual_temp ~ error("M") + trend("A") + season("M")),
    damped = ETS(actual_temp ~ error("A") + trend("Ad") + season("A")),
    mult_damp = ETS(actual_temp ~ error("M") + trend("Ad") + season("M")),
    log_auto = ETS(log(actual_temp)),
    box_cox_auto = ETS(box_cox(actual_temp, lambda = 1.5))
  )
```

```r
ets_fit |>
  accuracy() |>
  group_by(.model) |>
  summarise(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(RMSE) |>
  kable(digits = 4, align = "c")
```

.model	RMSE	MAE	ME
box_cox_auto	0.1146	0.0893	5e-04
ets_auto	0.1158	0.0907	8e-04
damped	0.1158	0.0909	7e-04
additive	0.1166	0.0917	-2e-04
mult_damp	0.1195	0.0942	-1e-04
log_auto	0.1198	0.0947	1e-04
multiplicative	0.1249	0.0988	-4e-04

```r
# Smoothing parameters
ets_fit |>
  tidy() |>
  group_by(.model, term) |>
  summarise(across(.cols = estimate, .fns = mean)) |>
  pivot_wider(names_from = term, values_from = estimate) |>
  select(.model, alpha, beta, gamma, phi) |>
  kable(digits = 4, align = "c")
```

.model	alpha	beta	gamma	phi
additive	0.4026	0.0001	0.0285	NA
box_cox_auto	0.4386	0.0006	0.0001	0.9727
damped	0.4325	0.0002	0.0178	0.9787
ets_auto	0.4304	0.0003	0.0109	0.9800
log_auto	0.4139	0.0001	0.0343	0.9754
mult_damp	0.4469	0.0001	0.0295	0.9786
multiplicative	0.2812	0.0057	0.0755	NA

```r
# Report on top performing model
ets_fit |>
  filter(.id == 10) |>
  select(log_auto) |>
  report()
```

```
Series: actual_temp
Model: ETS(A,A,A)
Transformation: log(actual_temp)
  Smoothing parameters:
    alpha = 0.408761
    beta  = 0.0001000166
    gamma = 0.0549685

  Initial states:
     l[0]         b[0]       s[0]       s[-1]      s[-2]      s[-3]     s[-4]
 2.615744 6.255243e-05 -0.1166191 -0.05617134 0.01551034 0.07698261 0.1148955
     s[-5]    s[-6]      s[-7]        s[-8]      s[-9]     s[-10]     s[-11]
 0.1262374 0.111805 0.06813664 -0.001073793 -0.0717811 -0.1219244 -0.1459979

  sigma^2:  1e-04

      AIC      AICc       BIC
-3724.686 -3724.375 -3629.590
```

All versions of ETS performed well on the training data with box_cox_auto performing the best. Auto selection chose an additive model with a damped trend. Looking the parameters, the `alpha` parameter values ranged from .2812 to .4469. This shows that the level is fairly stable and perhaps indicates that the recent warming trend is being masked by the long tail of relatively stable temperatures. This idea is bolstered by the the really small, almost 0 values of the `beta` parameter which indicates a stable trend. The `gamma` parameters are all very small as showing, suggesting that the ETS models are correctly reading that the seasonal pattern is relatively stable.

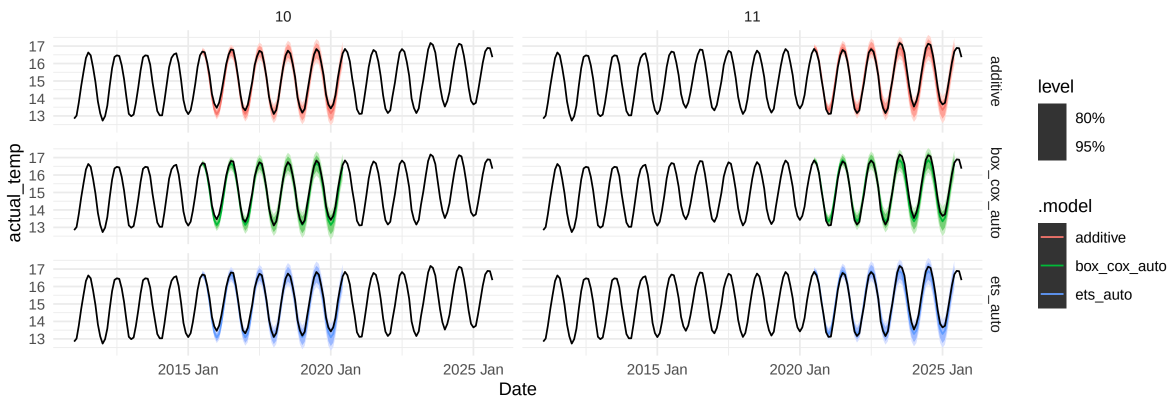
```r
ets_fc <- ets_fit |>
  forecast(new_data = cv_valid)

ets_fc |>
  accuracy(cv_valid) |>
  group_by(.model) |>
  summarise(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
additive	0.1724	0.1412	-0.0013
ets_auto	0.1726	0.1409	0.0239

.model	RMSE	MAE	ME
box_cox_auto	0.1730	0.1409	0.0183
damped	0.1735	0.1419	0.0208
log_auto	0.1758	0.1441	0.0033
mult_damp	0.1763	0.1442	0.0261
multiplicative	0.1919	0.1627	-0.0179

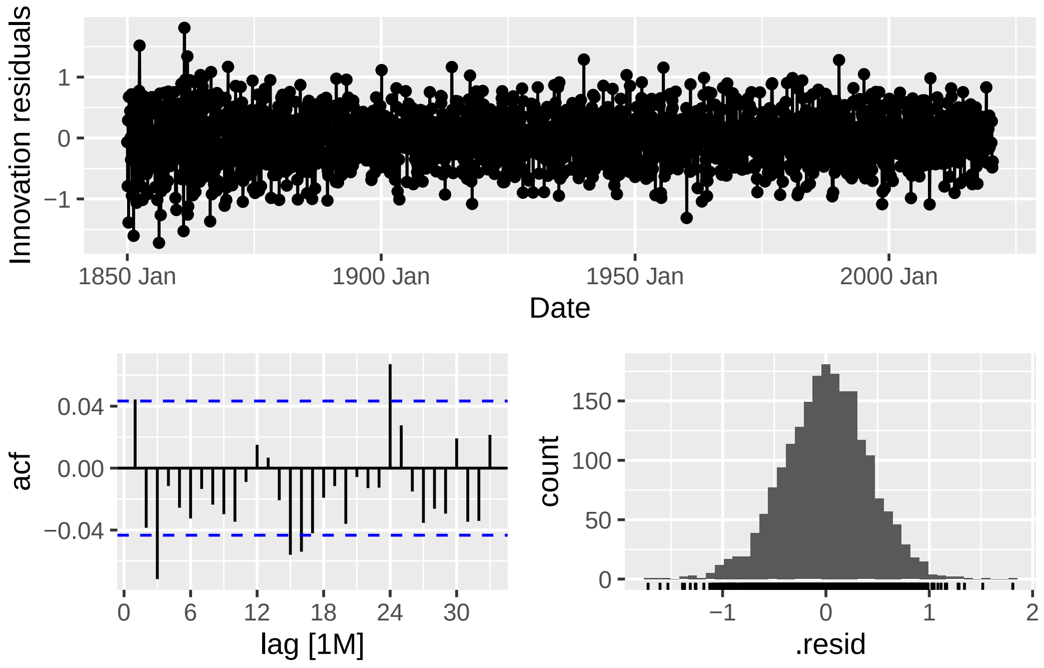
The additive model performs the best on the validation data but the out performance is negligible to ets_auto and box_cox_auto

```r
ets_fc |>
  filter(.id %in% c(10, 11), .model %in% c('additive', 'box_cox_auto', 'ets_auto')) |>
  autoplot() +
  autolayer(converted_df |> filter(year(Date) > 2010), actual_temp) +
  facet_grid(.model ~ .id) +
  theme_minimal()
```



Visualizing the top 3 models illustrates how ETS models do just fine at predicting the fall and spring months, but consistently undershoot the seasonal peaks and exhibit wide prediction intervals around those areas. ETS models have trouble capturing the increasing trend

```r
ets_fit |>
  select(box_cox_auto) |>
  tail(n = 1) |>
  ggtime::gg_tsresiduals()
```



Using box cox and checking the residuals of the model fitted on the most data, the residuals show a mostly normal distribution. Slight skew to the right. There are autocorrelations at lag 3 and lag 24. We can confirm that the residuals aren't white noise with a Ljung-Box test. There are certainly factors that influence temperature that ETS isn't capturing

```
ets_fit |>
  augment() |>
  filter(.model == "box_cox_auto") |>
  features(.innov, ljung_box, lag = 24) |>
  summarize(across(.cols = lb_pvalue, .fns = mean)) |>
  kable(digits = 4, align = "c")
```

lb_pvalue
0.0041

TSLM

```
tslm_fit <- cv_trn |>
  model(
    tslm_auto = TSLM(actual_temp),
```

```r
    tslm_trend = TSLM(actual_temp ~ trend()),
    tslm_trend_season = TSLM(actual_temp ~ trend() + season()),
    tslm_fourier = TSLM(actual_temp ~ trend() + fourier(K = 2)),
    tslm_log = TSLM(log(actual_temp) ~ trend() + season()),
    tslm_box_cox = TSLM(box_cox(actual_temp, lambda = 1.5) ~ trend() + season()),
    tslm_piecewise = TSLM(actual_temp ~ trend(knots = c(1920, 1975)) + season())
  )
```

Testing Fourier terms for a simpler model as opposed to using season dummy variables, and piece-wise to capture changes in the trend at those points

```r
tslm_fit |>
  accuracy() |>
  group_by(.model) |>
  summarise(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
tslm_piecewise	0.1861	0.1492	0.0000
tslm_trend_season	0.1861	0.1492	0.0000
tslm_box_cox	0.1861	0.1493	-0.0006
tslm_log	0.1874	0.1501	0.0013
tslm_fourier	0.1880	0.1508	0.0000
tslm_trend	1.3262	1.1887	0.0000
tslm_auto	1.3409	1.1979	0.0000

```r
tslm_fit |>
  select(tslm_trend_season) |>
  tail(n = 1) |>
  report()
```

```
Series: actual_temp
Model: TSLM

Residuals:
     Min       1Q   Median       3Q      Max
-0.63061 -0.15153 -0.01237  0.14543  0.79855
```

```
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)    1.172e+01  1.870e-02 626.999   <2e-16 ***
trend()        5.171e-04  8.187e-06  63.157   <2e-16 ***
season()year2  2.096e-01  2.365e-02   8.861   <2e-16 ***
season()year3  8.370e-01  2.365e-02  35.387   <2e-16 ***
season()year4  1.779e+00  2.365e-02  75.227   <2e-16 ***
season()year5  2.735e+00  2.365e-02 115.630   <2e-16 ***
season()year6  3.411e+00  2.365e-02 144.184   <2e-16 ***
season()year7  3.636e+00  2.369e-02 153.494   <2e-16 ***
season()year8  3.479e+00  2.369e-02 146.845   <2e-16 ***
season()year9  2.890e+00  2.369e-02 122.000   <2e-16 ***
season()year10 2.024e+00  2.369e-02  85.447   <2e-16 ***
season()year11 9.840e-01  2.369e-02  41.538   <2e-16 ***
season()year12 2.759e-01  2.369e-02  11.645   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2187 on 2033 degrees of freedom
Multiple R-squared: 0.9743, Adjusted R-squared: 0.9742
F-statistic:  6435 on 12 and 2033 DF, p-value: < 2.22e-16
```

The trend_season, box_cox and piecewise models all perform the same on the training data. Looking at the model report for the trend_season model we can see that all parameters are highly significant

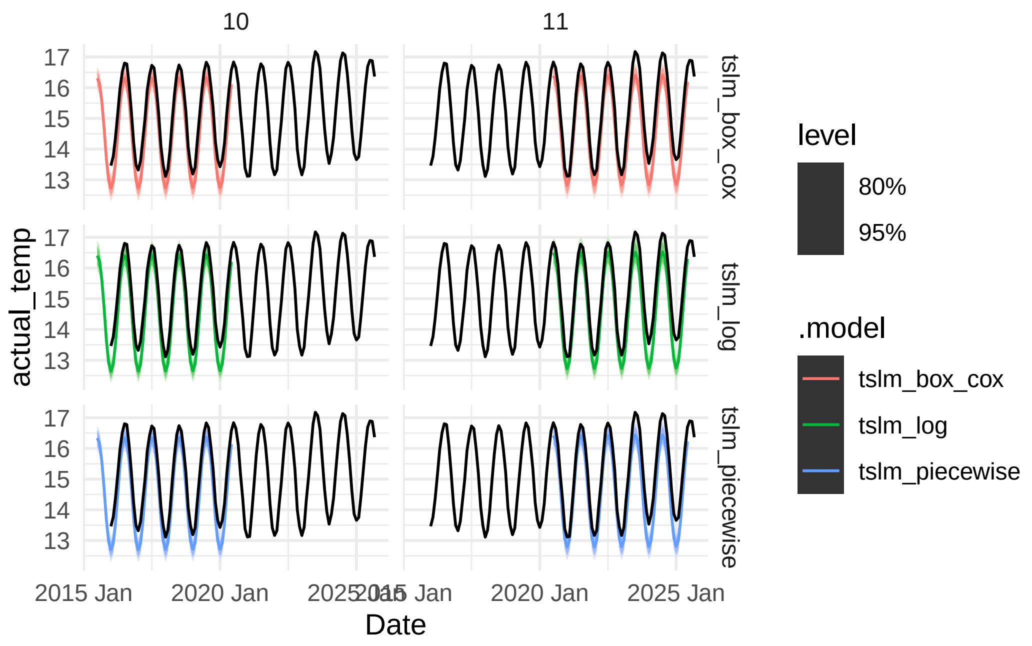
```r
tslm_fc <- tslm_fit |>
  forecast(new_data = cv_valid)

tslm_fc |>
  accuracy(cv_valid) |>
  group_by(.model) |>
  summarise(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(MAE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
tslm_log	0.3198	0.2881	0.2660
tslm_piecewise	0.3212	0.2933	0.2724
tslm_trend_season	0.3212	0.2933	0.2724
tslm_fourier	0.3222	0.2939	0.2723

.model	RMSE	MAE	ME
tslm_box_cox	0.3212	0.2940	0.2731
tslm_trend	1.3361	1.1945	0.2735
tslm_auto	1.4553	1.2584	0.6108

TSLM log performs the best on the validation data but significantly under-perform ETS models

```r
tslm_fc |>
  filter(.id %in% c(10, 11), .model %in% c('tslm_log', 'tslm_piecewise', 'tslm_box_cox')) |>
  autoplot() +
  autolayer(converted_df |> filter(year(Date) > 2015), actual_temp) +
  facet_grid(.model ~ .id) +
  theme_minimal()
```



It's pretty clear from the visuals that TSLM models are failing even harder than the ETS models at capturing the warming trend.

ARIMA

```r
trn_data <- converted_df |>
  slice(1:(n() - 60))

valid_data <- converted_df |>
  slice_tail(n = 60)
```

ARIMA models take too long to converge on CV splits and often time out, so testing them first on normal splits

```r
arima_fit <- trn_data |>
  model(
    arima_auto = ARIMA(actual_temp),
    arima_box = ARIMA(box_cox(actual_temp, lambda = 1.5)),
    arima_log = ARIMA(log(actual_temp))
  )

arima_fit |>
  accuracy() |>
  select(.model, RMSE, MAE, ME) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
arima_box	0.1191	0.0931	-4e-04
arima_auto	0.1192	0.0932	0e+00
arima_log	0.1318	0.1027	2e-04

ARIMA box_cox and auto performed the same on the training data

```r
arima_fit |>
  select(arima_box) |>
  report()
```

```
Series: actual_temp
Model: ARIMA(1,0,0)(0,1,1)[12] w/ drift
Transformation: box_cox(actual_temp, lambda = 1.5)

Coefficients:
         ar1     sma1  constant
```

```
      0.7088  -0.8825    0.0079
s.e.  0.0176   0.0126    0.0012

sigma^2 estimated as 0.1955:  log likelihood=-1235.64
AIC=2479.28   AICc=2479.3   BIC=2501.76
```

```r
arima_fit |>
  select(arima_auto) |>
  report()
```

```
Series: actual_temp
Model: ARIMA(1,0,0)(0,1,1)[12] w/ drift

Coefficients:
         ar1     sma1  constant
      0.7053  -0.8840    0.0021
s.e.  0.0176   0.0126    0.0003

sigma^2 estimated as 0.01431:  log likelihood=1426.97
AIC=-2845.94   AICc=-2845.92   BIC=-2823.47
```

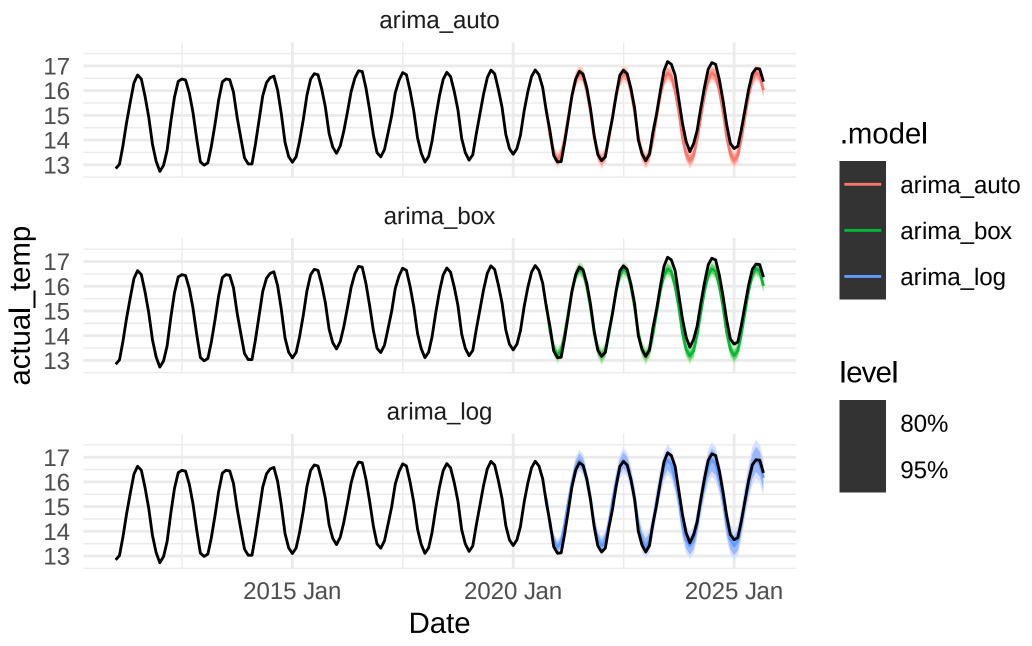
Surprisingly the ARIMA models only selected a seasonal differencing instead of both a seasonal and non-seasonal. 1 non-seasonal AR component was selected and 1 seasonal moving average. So the auto selected models are using the previous observation and the previous seasons forecast error to make its predictions.

```r
arima_fc <- arima_fit |>
  forecast(new_data = valid_data)

arima_fc |>
  accuracy(valid_data) |>
  group_by(.model) |>
  summarise(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
arima_log	0.2174	0.1823	0.0214
arima_box	0.2929	0.2365	0.2020
arima_auto	0.2929	0.2357	0.2035

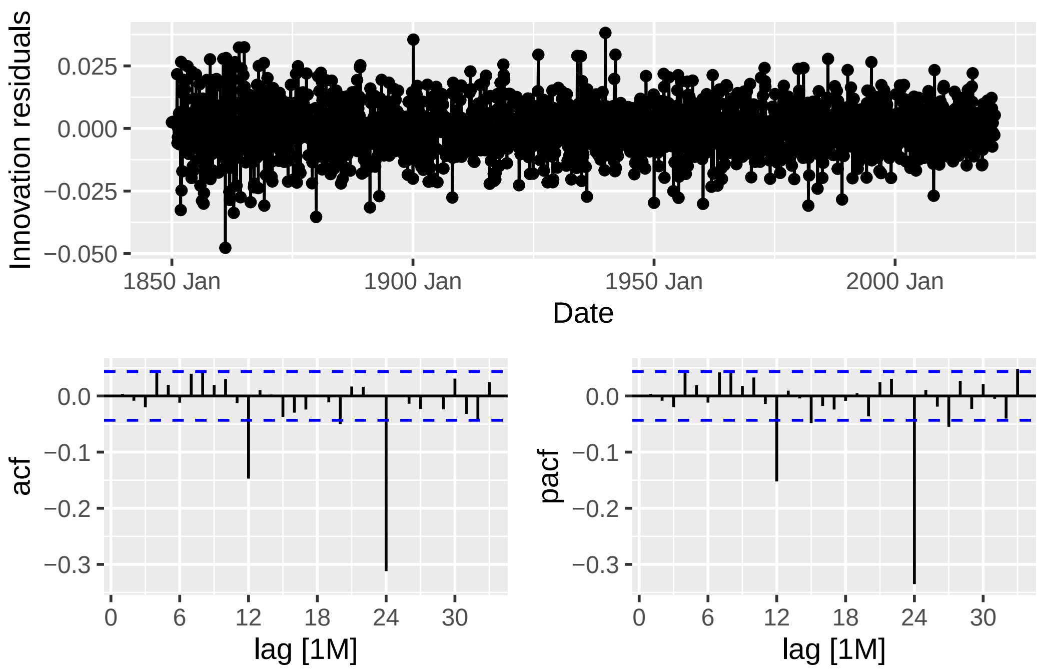
Arima log does best on the validation data

```r
arima_fc |>
  autoplot() +
  autolayer(converted_df |> filter(year(Date) > 2010), actual_temp) +
  facet_wrap(~ .model, ncol = 1) +
  theme_minimal()
```



We see that the ARIMA log model does slightly better at capturing the peaks than the other two, but all still don't quite capture the increasing trend

```r
arima_fit |>
  select(arima_log) |>
  ggtime::gg_tsresiduals(plot_type = 'partial')
```



```r
arima_fit |>
  augment() |>
  filter(.model == "arima_box") |>
  features(.innov, ljung_box, lag = 24) |>
  kable(digits = 4, align = "c")
```

.model	lb_stat	lb_pvalue
arima_box	229.746	0

Residuals aren't even close to stationary. With ARIMA we can experiment with other parameter values

```r
arima_fit <- trn_data |>
  model(
    arima_diff = ARIMA(actual_temp ~ pdq(d = 1)),
    arima_box_diff = ARIMA(box_cox(actual_temp, lambda = 1.5) ~ pdq(d = 1)),
    arima_box_ar_two = ARIMA(box_cox(actual_temp, lambda = 1.5) ~ pdq(p = 2)),
    arima_custom = ARIMA(actual_temp ~ 0 + pdq(3, 1, 2) + PDQ(1, 1, 2))
  )

arima_fc <- arima_fit |>
```

```r
  forecast(new_data = valid_data)

arima_fc |>
  accuracy(valid_data) |>
  group_by(.model) |>
  summarise(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
arima_custom	0.2110	0.1714	0.0801
arima_diff	0.2169	0.1814	-0.1230
arima_box_diff	0.2183	0.1818	-0.1199
arima_box_ar_two	0.2212	0.1851	0.0201

Adding non-seasonal differencing, another AR term and a seasonal MA term to deal with the autocorrelation at lag 24 helped improved model results on the validation data

```r
arima_fit |>
  augment() |>
  filter(.model == "arima_custom") |>
  features(.innov, ljung_box, lag = 24) |>
  kable(digits = 4, align = "c")
```

.model	lb_stat	lb_pvalue
arima_custom	21.0759	0.6342

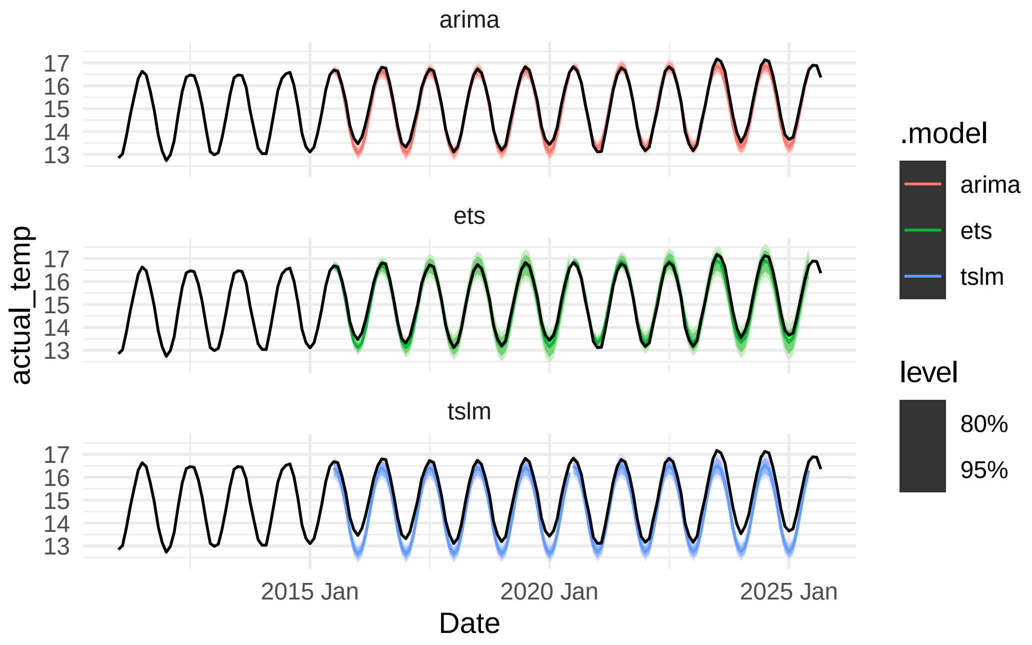
A high p-value tells us that the residuals are now resembling white noise

Compare Best Models

```r
compare_fit <- cv_trn |>
  model(
    ets = ETS(actual_temp ~ error("A") + trend("A") + season("A")),
    tslm = TSLM(log(actual_temp) ~ trend() + season()),
    arima = ARIMA(actual_temp ~ 0 + pdq(3, 1, 2) + PDQ(1, 1, 2))
  )

compare_fc <- compare_fit |>
  forecast(new_data = cv_valid)

compare_fc |>
  accuracy(cv_valid) |>
  filter(.id != 6) |>
  group_by(.model) |>
  summarise(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
arima	0.1634	0.1323	0.0482
ets	0.1748	0.1429	-0.0004
tslm	0.3191	0.2880	0.2637

```
compare_fc |>
  filter(.id %in% c(10, 11)) |>
  autoplot() +
  autolayer(converted_df |> filter(year(Date) > 2010), actual_temp) +
  facet_wrap(~ .model, ncol = 1) +
  theme_minimal()
```



The metrics based on RMSE show that ARIMA does slightly better than ETA and both do much better than TSLM. Comparing ARIMA to ETS visually, we see much more uncertainty in the ETS predictions. Both models still seem to under shoot the peaks which is a strong indicator that other factors are at play that will need to be considered. In addition, looking at the `mable` for the arima models shows a null result on split 6 so there may be some instability in the chosen parameters

```r
compare_fit |>
  select(arima)
```

```
# A mable: 11 x 2
# Key:     .id [11]
     .id                     arima
   <int>                   <model>
 1     1 <ARIMA(3,1,2)(1,1,2)[12]>
 2     2 <ARIMA(3,1,2)(1,1,2)[12]>
 3     3 <ARIMA(3,1,2)(1,1,2)[12]>
 4     4 <ARIMA(3,1,2)(1,1,2)[12]>
 5     5 <ARIMA(3,1,2)(1,1,2)[12]>
 6     6              <NULL model>
 7     7 <ARIMA(3,1,2)(1,1,2)[12]>
 8     8 <ARIMA(3,1,2)(1,1,2)[12]>
 9     9 <ARIMA(3,1,2)(1,1,2)[12]>
```

```
10     10 <ARIMA(3,1,2)(1,1,2)[12]>
11     11 <ARIMA(3,1,2)(1,1,2)[12]>
```

TSLM External Predictors

According to the Intergovernmental Panel on Climate Change (IPCC, 2021), the main factors that influence global temperature are greenhouse gas concentrations, natural weather and ocean-atmosphere patterns, and variations in solar irradiance. To see which drivers best serve our forecasting purpose, we can model these factors individually and compare their predictive performance. Intergovernmental Panel on Climate Change (IPCC). *Climate Change 2021: The Physical Science Basis.*. Of the below predictors, accurate monthly estimates are available across varying time periods. The latest are for CO2 (1958), CH4 and N2O (1977). CO2 is a more significant predictor than the others so the cutoff date for the training data used was 1958. Techniques used to simulate the monthly estimates for N2O and CH4 between 1958 and 1977 will be discussed below

https://www.ipcc.ch/report/ar6/wg1/

TSI:

ENSO Index:

CO2:

RF_CO2:

Volcanic Activity:

CH4:

N2O:

```
predictor_modeling <- read_rds("data/lagged_external_predictors.rds")
```

Here go into detail about interpolating and lags

There are over 100 total combinations of external predictors, so instead of trying all of them, a handful of complementary ones are tested and reviewed to see if any other combinations are worth exploring. For example, current co2_ppm combined with la_nina and el_nino events are compared against the same but with a 10yr lag on the co2_ppm variable. If the lagged version of this model performs better, then lagged trends are worth looking further into

```r
cv_data <- predictor_modeling |>
  filter(year(Date) > 1957) |>
  stretch_tsibble(.init = 573, .step = 60)

cv_trn <- cv_data |>
  group_by(.id) |>
  slice(1:(n() - 60)) |>
  ungroup()

cv_valid <- cv_data |>
  group_by(.id) |>
  slice_tail(n = 60) |>
  ungroup()

tslm_regressor_fit <- cv_trn |>
  model(
    tslm_trend_season = TSLM(actual_temp ~ trend() + season()),
    tslm_log = TSLM(log(actual_temp) ~ trend() + season()),
    tslm_box_cox = TSLM(box_cox(actual_temp, lambda = 1.5) ~ trend() + season()),
    tslm_all = TSLM(actual_temp ~ trend() + season() + el_nino + la_nina + co2_ppm + ch4_ppb
    tslm_all_box = TSLM(box_cox(actual_temp, lambda = 1.5) ~ trend() + season() + el_nino + l
    tslm_ch4_co2_nino = TSLM(actual_temp ~ trend() + season() + el_nino + la_nina + co2_ppm +
    tslm_co2_nino = TSLM(actual_temp ~ trend() + season() + el_nino + la_nina + co2_ppm),
    tslm_nino = TSLM(actual_temp ~ trend() + season() + el_nino + la_nina),
    tslm_enso = TSLM(actual_temp ~ trend() + season() + ENSO),
    tslm_enso_smooth = TSLM(actual_temp ~ trend() + season() + enso_smooth_12),
    tslm_co2_nino_lag = TSLM(actual_temp ~ trend() + season() + el_nino + la_nina + co2_lag1)
  )

tslm_regressor_fit |>
  accuracy() |>
  group_by(.model) |>
  summarise(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
tslm_all_box	0.1188	0.0935	-2e-04
tslm_all	0.1190	0.0936	0e+00
tslm_ch4_co2_nino	0.1214	0.0958	0e+00
tslm_co2_nino	0.1242	0.0981	0e+00

.model	RMSE	MAE	ME
tslm_co2_nino_lag	0.1260	0.0993	0e+00
tslm_enso_smooth	0.1299	0.1023	0e+00
tslm_nino	0.1332	0.1066	0e+00
tslm_enso	0.1368	0.1107	0e+00
tslm_trend_season	0.1461	0.1176	0e+00
tslm_box_cox	0.1465	0.1178	-4e-04
tslm_log	0.1479	0.1191	8e-04

Training metrics show that all the predictors being added leads to the best model, whereas models without any predictors underperform

```
tslm_regressor_fit |>
  glance() |>
  group_by(.model) |>
  summarize(across(.cols = c(AIC, AICc, BIC), .fns = mean)) |>
  arrange(AICc)
```

```
# A tibble: 11 x 4
   .model               AIC   AICc    BIC
   <chr>              <dbl>  <dbl>  <dbl>
 1 tslm_log          -5722. -5721. -5659.
 2 tslm_co2_nino     -2606. -2605. -2530.
 3 tslm_co2_nino_lag -2588. -2587. -2513.
 4 tslm_enso_smooth  -2555. -2555. -2489.
 5 tslm_nino         -2519. -2518. -2448.
 6 tslm_enso         -2490. -2489. -2423.
 7 tslm_trend_season -2407. -2407. -2345.
 8 tslm_all          -2365. -2363. -2274.
 9 tslm_ch4_co2_nino -2349. -2348. -2271.
10 tslm_all_box       -868.  -866.  -777.
11 tslm_box_cox       -732.  -731.  -670.
```

Interestingly, tslm_log is the best model by far on the AICc metric, so the gains may not be all that much by adding predictors

```
tslm_regressor_fit |>
  select(tslm_all_box) |>
  tail(n = 1) |>
  report()
```

```
Series: actual_temp
Model: TSLM
Transformation: box_cox(actual_temp, lambda = 1.5)

Residuals:
    Min      1Q  Median      3Q     Max
-1.3498 -0.2808 -0.0138  0.3062  1.1927

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)     -1.829e+02  6.784e+01  -2.696 0.007188 **
trend()         -6.771e-03  1.535e-03  -4.409 1.21e-05 ***
season()year2    6.992e-01  8.594e-02   8.136 2.00e-15 ***
season()year3    2.899e+00  8.647e-02  33.525  < 2e-16 ***
season()year4    6.328e+00  8.815e-02  71.788  < 2e-16 ***
season()year5    9.818e+00  8.898e-02 110.341  < 2e-16 ***
season()year6    1.257e+01  8.739e-02 143.882  < 2e-16 ***
season()year7    1.360e+01  8.564e-02 158.751  < 2e-16 ***
season()year8    1.322e+01  8.679e-02 152.368  < 2e-16 ***
season()year9    1.096e+01  8.932e-02 122.746  < 2e-16 ***
season()year10   7.592e+00  8.963e-02  84.702  < 2e-16 ***
season()year11   3.581e+00  8.759e-02  40.887  < 2e-16 ***
season()year12   1.017e+00  8.645e-02  11.766  < 2e-16 ***
el_nino          1.346e-01  4.439e-02   3.032 0.002526 **
la_nina         -3.062e-01  4.695e-02  -6.523 1.37e-10 ***
co2_ppm          7.708e-02  7.967e-03   9.675  < 2e-16 ***
ch4_ppb          1.690e-03  5.032e-04   3.358 0.000831 ***
n2o_ppb          9.750e-03  1.197e-02   0.815 0.415562
TSI              1.334e-01  4.994e-02   2.672 0.007721 **
volcano_forcing -9.692e-02  2.823e-02  -3.434 0.000633 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4599 on 666 degrees of freedom
Multiple R-squared: 0.9921, Adjusted R-squared: 0.9919
F-statistic:  4417 on 19 and 666 DF, p-value: < 2.22e-16
```

All predictors on the best performing training model show as significant except for N2O

```r
tslm_fc <- tslm_regressor_fit |>
  forecast(new_data = cv_valid)
```

```r
tslm_fc |>
  accuracy(cv_valid) |>
  group_by(.model) |>
  summarise(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
tslm_all_box	0.1471	0.1172	-0.0497
tslm_co2_nino_lag	0.1476	0.1166	-0.0234
tslm_all	0.1493	0.1196	-0.0526
tslm_co2_nino	0.1519	0.1203	-0.0409
tslm_enso_smooth	0.1519	0.1254	0.0976
tslm_ch4_co2_nino	0.1531	0.1229	-0.0627
tslm_enso	0.1578	0.1297	0.0987
tslm_nino	0.1705	0.1382	0.1017
tslm_trend_season	0.1746	0.1429	0.0990
tslm_box_cox	0.1761	0.1463	0.1004
tslm_log	0.1766	0.1398	0.0904

TSLM with a box_cox transformation and all of the predictors performed the best on the validation data. The `enso_smooth` variable also performed better than either `ENSO` or the nino dummy variables. These results indicate that other model combinations are worth exploring.

Adding a year lag to `co2_ppm` provided a bit of a boost on the validation data so some of the lagged variables may have more predictive power and are worth looking into

Interesting to note that adding `ch4_ppb` actually reduced model performance compared to just using `co2_ppm`

```r
tslm_second_fit <- cv_trn |>
  model(
    tslm_all_lag_box = TSLM(box_cox(actual_temp, lambda = 1.5) ~ trend() + season() + el_ninc
    tslm_nino_five_lag = TSLM(actual_temp ~ trend() + season() + el_nino + la_nina + co2_lag5
    tslm_enso_co2_lag = TSLM(actual_temp ~ trend() + season() + ENSO + co2_lag1),
    tslm_enso_smooth_lag_all = TSLM(actual_temp ~ trend() + season() + enso_smooth_12 + co2_l
    tslm_all_lag = TSLM(actual_temp ~ trend() + season() + el_nino + la_nina + co2_lag1 + n2c
    tslm_nino_10_lag = TSLM(box_cox(actual_temp, lambda = 1.5) ~ trend() + season() + el_ninc
    tslm_enso_smooth_lag = TSLM(box_cox(actual_temp, lambda = 1.5) ~ trend() + season() + ens
  )
```

```
tslm_second_fc <- tslm_second_fit |>
  forecast(new_data = cv_valid)

tslm_second_fc |>
  accuracy(cv_valid) |>
  group_by(.model) |>
  summarise(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
tslm_enso_smooth_lag	0.1217	0.0950	-0.0063
tslm_enso_smooth_lag_all	0.1298	0.1028	-0.0476
tslm_enso_co2_lag	0.1374	0.1112	-0.0329
tslm_nino_10_lag	0.1413	0.1101	-0.0029
tslm_nino_five_lag	0.1445	0.1133	0.0057
tslm_all_lag_box	0.1464	0.1144	-0.0394
tslm_all_lag	0.1475	0.1172	-0.0420

The target variable undergoing a box_cox transformation using the smoothed enso index over 6 months and a 10 year lag in CO2 concentration performs the best out of all the models by a fair margin

```
tslm_second_fit |>
  select(tslm_enso_smooth_lag) |>
  tail(n = 1) |>
  report()
```

```
Series: actual_temp
Model: TSLM
Transformation: box_cox(actual_temp, lambda = 1.5)

Residuals:
      Min        1Q    Median        3Q       Max
-1.448628 -0.292156  0.003149  0.296378  1.450508

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)    1.297e+01  1.246e+00  10.410   <2e-16 ***
trend()        2.270e-04  4.573e-04   0.496     0.62
```

```
season()year2  7.171e-01  8.137e-02   8.813   <2e-16 ***
season()year3  2.934e+00  8.152e-02  35.994   <2e-16 ***
season()year4  6.402e+00  8.197e-02  78.097   <2e-16 ***
season()year5  9.928e+00  8.222e-02 120.741   <2e-16 ***
season()year6  1.263e+01  8.189e-02 154.285   <2e-16 ***
season()year7  1.362e+01  8.142e-02 167.267   <2e-16 ***
season()year8  1.316e+01  8.157e-02 161.305   <2e-16 ***
season()year9  1.087e+01  8.234e-02 132.039   <2e-16 ***
season()year10 7.474e+00  8.282e-02  90.241   <2e-16 ***
season()year11 3.528e+00  8.216e-02  42.944   <2e-16 ***
season()year12 1.024e+00  8.177e-02  12.526   <2e-16 ***
enso_smooth_6  3.182e-01  2.166e-02  14.694   <2e-16 ***
co2_lag10      4.747e-02  4.146e-03  11.450   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4564 on 738 degrees of freedom
Multiple R-squared: 0.9921, Adjusted R-squared: 0.992
F-statistic:  6628 on 14 and 738 DF, p-value: < 2.22e-16
```

And we can see from the report that all the predictors used are significant except trend. It could be that smoothing out the enso index and the 10 yr lag of co2_ppm captures most of the trend component

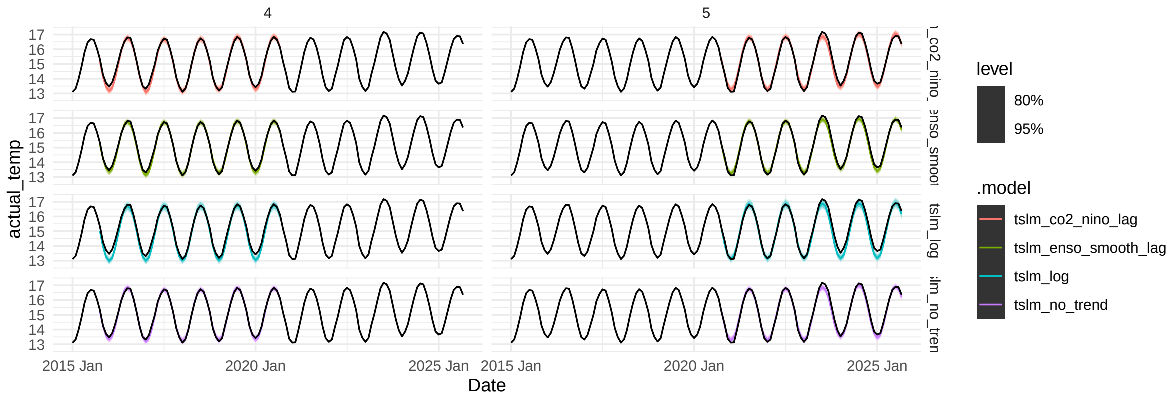
```r
tslm_best_fit <- cv_trn |>
  model(
    tslm_enso_6 = TSLM(box_cox(actual_temp, lambda = 1.5) ~ trend() + season() + enso_smooth_
    tslm_enso_12 = TSLM(box_cox(actual_temp, lambda = 1.5) ~ trend() + season() + enso_smoot
    tslm_no_trend = TSLM(box_cox(actual_temp, lambda = 1.5) ~ season() + enso_smooth_12 + co
    tslm_volcanic = TSLM(box_cox(actual_temp, lambda = 1.5) ~ season() + enso_smooth_12 + co
  )

tslm_best_fc <- tslm_best_fit |>
  forecast(new_data = cv_valid)

tslm_best_fc |>
  accuracy(cv_valid) |>
  group_by(.model) |>
  summarise(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
tslm_no_trend	0.1183	0.0927	-0.0032
tslm_volcanic	0.1191	0.0936	0.0013
tslm_enso_12	0.1211	0.0942	-0.0031
tslm_enso_6	0.1217	0.0950	-0.0063

```r
tslm_best_fit |>
  select(tslm_no_trend) |>
  tail(n = 1) |>
  report()
```

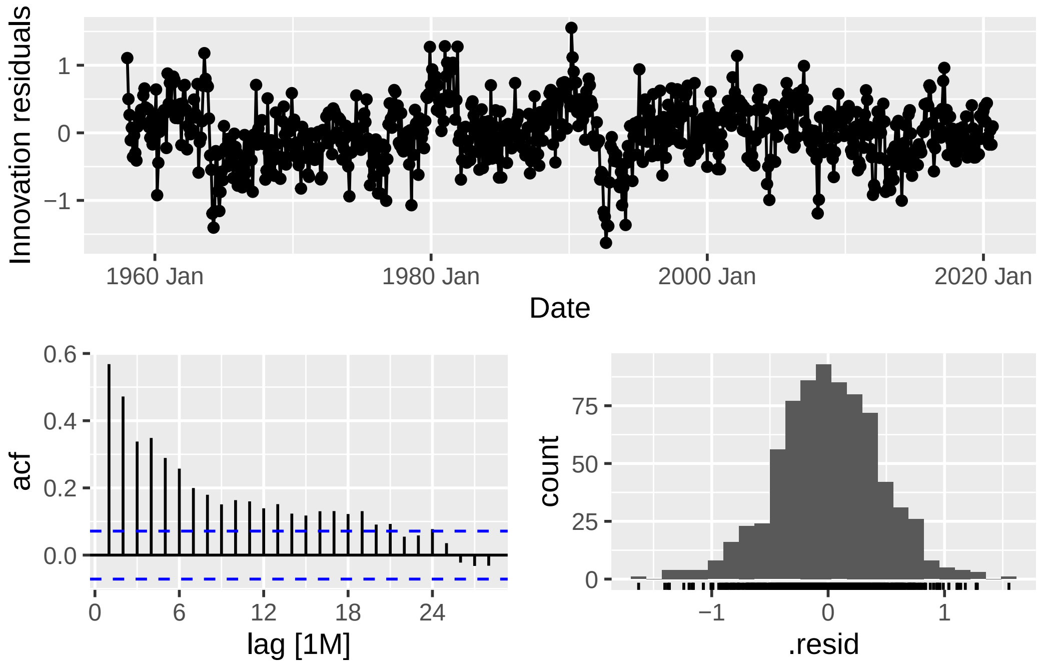
```
Series: actual_temp
Model: TSLM
Transformation: box_cox(actual_temp, lambda = 1.5)

Residuals:
      Min        1Q    Median        3Q       Max
-1.628084 -0.287697  0.001792  0.303473  1.553132

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)    1.233e+01  2.424e-01  50.849   <2e-16 ***
season()year2  7.143e-01  8.076e-02   8.844   <2e-16 ***
season()year3  2.923e+00  8.076e-02  36.191   <2e-16 ***
season()year4  6.380e+00  8.078e-02  78.985   <2e-16 ***
season()year5  9.896e+00  8.079e-02 122.495   <2e-16 ***
season()year6  1.260e+01  8.078e-02 155.962   <2e-16 ***
season()year7  1.358e+01  8.076e-02 168.156   <2e-16 ***
season()year8  1.312e+01  8.076e-02 162.517   <2e-16 ***
season()year9  1.085e+01  8.077e-02 134.317   <2e-16 ***
season()year10 7.461e+00  8.111e-02  91.985   <2e-16 ***
season()year11 3.522e+00  8.109e-02  43.432   <2e-16 ***
season()year12 1.020e+00  8.108e-02  12.581   <2e-16 ***
enso_smooth_12 3.785e-01  2.500e-02  15.142   <2e-16 ***
co2_lag10      4.966e-02  6.909e-04  71.876   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4532 on 739 degrees of freedom
Multiple R-squared: 0.9922, Adjusted R-squared: 0.9921
F-statistic:  7239 on 13 and 739 DF, p-value: < 2.22e-16
```

```r
tslm_fc |>
  bind_rows(tslm_second_fc) |>
  bind_rows(tslm_best_fc) |>
  filter(.id %in% c(4, 5), .model %in% c("tslm_no_trend", "tslm_enso_smooth_lag", "tslm_nino_
  autoplot() +
  autolayer(predictor_modeling |> filter(year(Date) >= 2015), actual_temp) +
  facet_grid(.model ~ .id) +
  theme_minimal()
```



Comparing best models to the under-performing ones, we see that adding predictors, especially ones that are lagged, closes the gap in the increasing trend that other models seem to miss. **Talk more in depth about the AICc scores**

```r
tslm_best_fit |>
  select(tslm_no_trend) |>
  tail(n = 1) |>
  ggtime::gg_tsresiduals()
```



The biggest issue is that there is significant autocorrelation in the residuals, indicating there is some pattern the TSLM model doesn't catch. We'll see if the ARIMA models are able to accurately account for these factors

ARIMA External Predictors

Using a non-cv split initially to speed up fit time

```
arima_trn <- predictor_modeling |>
  filter(year(Date) > 1957) |>
  slice(1:(n() - 60))

arima_valid <- predictor_modeling |>
  filter(year(Date) > 1957) |>
  slice_tail(n = 60)
```

```
arima_fit <- arima_trn |>
  model(
    arima_auto = ARIMA(actual_temp ~ co2_ppm + PDQ(D=1)),
    arima_box = ARIMA(box_cox(actual_temp, lambda = 1.5) ~ co2_ppm + PDQ(D=1)),
    arima_log = ARIMA(log(actual_temp) ~ co2_ppm + PDQ(D=1)),
    arima_all = ARIMA(actual_temp ~ co2_ppm + ch4_ppb + n2o_ppb + TSI + el_nino + la_nina +
    arima_lag_one = ARIMA(actual_temp ~ co2_lag1 + PDQ(D=1)),
    arima_lag_one_nino = ARIMA(actual_temp ~ co2_lag1 + el_nino + la_nina + PDQ(D=1)),
    arima_enso = ARIMA(actual_temp ~ ENSO + PDQ(D=1)),
    arima_enso_smooth = ARIMA(actual_temp ~ enso_smooth_6 + PDQ(D=1)),
    arima_emissions = ARIMA(actual_temp ~ aggregate_emissions + PDQ(D=1)),
    arima_forcing = ARIMA(log(actual_temp) ~ rf_co2 + PDQ(D=1)),
    arima_enso_delay = ARIMA(actual_temp ~ enso_delay + PDQ(D=1)),
    arima_lag_3 = ARIMA(actual_temp ~ co2_lag3 + PDQ(D=1)),
    arima_physics_robust = ARIMA(log(actual_temp) ~ rf_co2 + volcano_linear + enso_delay + PI
    arima_volcano_basic = ARIMA(log(actual_temp) ~ ENSO + co2_ppm + volcano_forcing + PDQ(D =
  )

arima_fit |>
  glance() |>
  arrange(AICc) |>
  select(.model, AICc, AIC, BIC) |>
  kable(align = "c")
```

.model	AICc	AIC	BIC
arima_volcano_basic	-5154.6844	-5154.8811	-5118.0171
arima_physics_robust	-5021.0184	-5021.3198	-4975.2398
arima_forcing	-4925.4545	-4925.6073	-4893.3513
arima_log	-4924.9670	-4925.1637	-4888.2997
arima_enso_smooth	-1259.0458	-1259.2425	-1222.3785
arima_enso	-1254.6394	-1254.8361	-1217.9721
arima_enso_delay	-1227.5044	-1227.7011	-1190.8371
arima_lag_one_nino	-1137.8875	-1138.1888	-1092.1088
arima_auto	-1125.8302	-1126.0269	-1089.1629
arima_emissions	-1125.2980	-1125.4947	-1088.6307
arima_lag_one	-1123.8375	-1124.0343	-1087.1703
arima_lag_3	-1123.6502	-1123.8469	-1086.9829
arima_all	-962.7635	-963.2642	-903.3602
arima_box	827.9954	827.7987	864.6627

Looking through the AICc scores, the log transformations seem to result in the best models

```
arima_fc <- arima_fit |>
  forecast(new_data = arima_valid)

arima_fc |>
```

```
  accuracy(arima_valid) |>
  select(.model, RMSE, MAE, ME) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

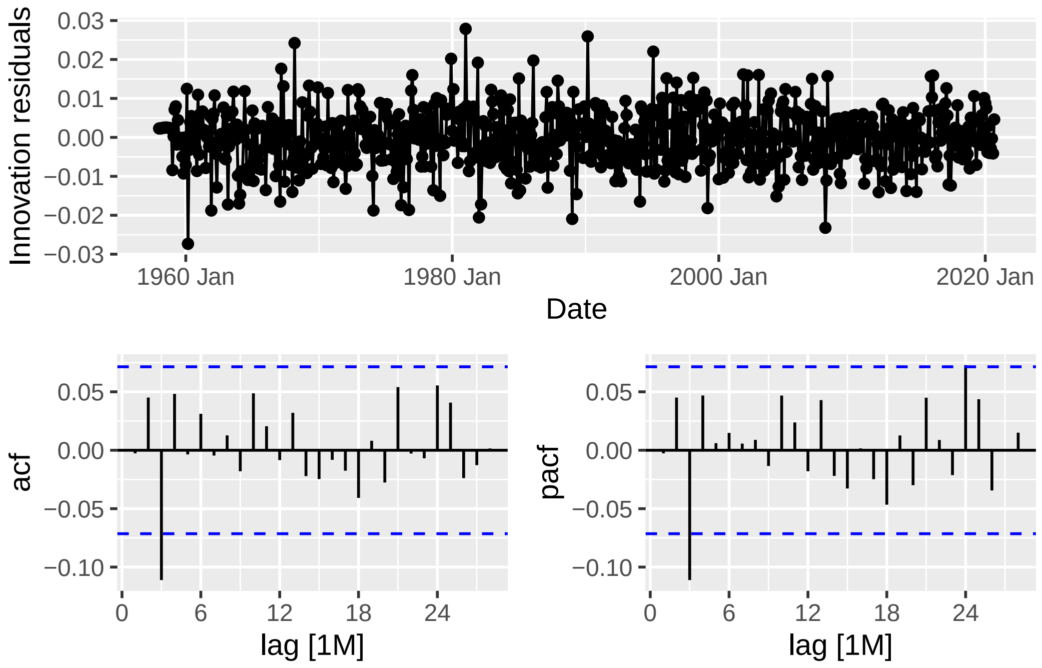
.model	RMSE	MAE	ME
arima_volcano_basic	0.1444	0.1185	0.0340
arima_physics_robust	0.1551	0.1197	0.0121
arima_all	0.1781	0.1387	-0.0221
arima_lag_one_nino	0.1934	0.1549	0.0475
arima_auto	0.1953	0.1630	0.0072
arima_box	0.1967	0.1638	0.0069
arima_emissions	0.1981	0.1662	0.0081
arima_log	0.1995	0.1654	-0.0205
arima_forcing	0.2022	0.1675	-0.0086
arima_lag_3	0.2067	0.1741	0.0328
arima_lag_one	0.2074	0.1742	0.0301
arima_enso	0.2819	0.2302	0.1923
arima_enso_smooth	0.2859	0.2347	0.2114
arima_enso_delay	0.3086	0.2519	0.2180

The model including features for volcanic activity and non-lagged variables for CO2 and ENSO is the best overall by any metric. **include a more robust explanation of why a delay in the enso index and why co2 radiative forcing on it's own aren't strong signals**

```
# spot check models
arima_fit |>
  select(arima_volcano_basic) |>
  report()
```

```
Series: actual_temp
Model: LM w/ ARIMA(1,0,1)(0,1,2)[12] errors
Transformation: log(actual_temp)

Coefficients:
         ar1      ma1      sma1     sma2    ENSO  co2_ppm  volcano_forcing
      0.8344  -0.3830  -0.9473  0.0544  0.0046    7e-04           -9e-04
s.e.  0.0326   0.0552   0.0355  0.0364  0.0007    1e-04            9e-04

sigma^2 estimated as 5.37e-05:  log likelihood=2585.44
AIC=-5154.88   AICc=-5154.68   BIC=-5118.02
```

```r
arima_fit |>
  select(arima_volcano_basic) |>
  ggtime::gg_tsresiduals(plot_type = "partial")
```



The arima_volcano_basic model selected AR(1), MA(1) and sMA(2) with a seasonal difference. The ACF and PACF plots still contain spikes at lag 3. Interestingly, the arima_physics robust model contains a spike at lag 24 but not 3. There might be something in delaying the El Nino effect by a few months that explains the autocorrelation

```r
# parameter check
arima_fit |>
  tidy() |>
  group_by(term) |>
  summarize(across(.cols = c(estimate:`p.value`), .fns = mean)) |>
  arrange(p.value) |>
  kable(digits = 4)
```

term	estimate	std.error	statistic	p.value
sma1	-0.9267	0.0385	-24.1588	0.0000
sar2	-0.3306	0.0350	-9.4517	0.0000
enso_smooth_6	0.0805	0.0122	6.6143	0.0000

term	estimate	std.error	statistic	p.value
ENSO	0.0313	0.0052	6.3122	0.0000
ar2	0.6779	0.0864	7.8968	0.0000
sar1	-0.6278	0.0347	-18.4039	0.0013
ma2	-0.2650	0.0731	-3.6212	0.0032
enso_delay	0.0158	0.0065	3.7552	0.0120
la_nina	-0.0453	0.0207	-2.1940	0.0297
ma1	0.3425	0.1148	2.8495	0.0433
el_nino	0.0333	0.0177	1.8858	0.0606
aggregate_emissions	0.0006	0.0003	1.8391	0.0663
co2_ppm	0.0126	0.0064	3.2146	0.0665
rf_co2	0.0688	0.0373	1.7833	0.0902
sma2	0.0606	0.0367	1.6484	0.1025
ar3	-0.0228	0.0852	0.1827	0.1074
co2_lag1	0.0079	0.0048	1.6714	0.1114
co2_lag3	0.0069	0.0054	1.2846	0.1993
volcano_linear	-0.0054	0.0062	-1.3169	0.2395
TSI	0.0204	0.0191	1.0671	0.2863
intercept	-0.0011	0.0011	-0.9745	0.3301
volcano_forcing	-0.0009	0.0009	-0.9703	0.3322
ar1	0.1208	0.1062	2.3740	0.4273
n2o_ppb	-0.0049	0.0123	-0.3959	0.6923
ch4_ppb	0.0001	0.0009	0.0822	0.9345

Not necessarily the best way to ascertain how important variables are, but informative nonetheless. ENSO related variables and seasonal ARIMA parameters seem to be the most important to modeling this type of data

Now we'll check the best models from above as well as a few other promising predictor combinations across 5 CV splits and see how they hold up

```
arima_second_fit <- cv_trn |>
  model(
    arima_enso_log_co2 = ARIMA(log(actual_temp) ~ ENSO + co2_ppm + PDQ(D = 1)),
    arima_auto_10 = ARIMA(log(actual_temp) ~ ENSO + co2_lag10 + PDQ(D = 1)),
    arima_physics_robust = ARIMA(log(actual_temp) ~ rf_co2 + volcano_linear + enso_delay + P
    arima_forcing = ARIMA(log(actual_temp) ~ rf_co2 + enso_smooth_6 + PDQ(D = 1)),
    arima_auto_10_enso_6 = ARIMA(log(actual_temp) ~ enso_smooth_6 + co2_lag10 + PDQ(D = 1)),
    arima_robust_enso_6 = ARIMA(log(actual_temp) ~ rf_co2 + volcano_linear + enso_smooth_6 +
    arima_co2_10_volcano = ARIMA(log(actual_temp) ~ co2_lag10 + volcano_linear + PDQ(D = 1))
    arima_co2_3_nino = ARIMA(actual_temp ~ co2_lag3 + el_nino + la_nina + PDQ(D = 1)),
    arima_co2_3_nino_tsi = ARIMA(actual_temp ~ co2_lag3 + el_nino + la_nina + TSI + PDQ(D = 
    arima_physics_robust_decay = ARIMA(log(actual_temp) ~ rf_co2 + volcano_forcing + enso_de
    arima_physics_robust_decay_enso_3 = ARIMA(log(actual_temp) ~ rf_co2 + volcano_forcing + 
    arima_lag_3_enso = ARIMA(log(actual_temp) ~ co2_lag3 + ENSO + PDQ(D=1)),
    arima_physics_enso_3 = ARIMA(log(actual_temp) ~ rf_co2 + enso_smooth_3 + PDQ(D=1))
  )


arima_second_fc <- arima_second_fit |>
  forecast(new_data = cv_valid)

arima_second_fc |>
  accuracy(cv_valid) |>
  select(.model, RMSE, ME, MAE) |>
  group_by(.model) |>
  summarize(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
arima_auto_10_enso_6	0.1225	0.0947	-0.0047
arima_physics_robust_decay_enso_3	0.1233	0.0946	0.0005
arima_forcing	0.1266	0.1005	-0.0005
arima_robust_enso_6	0.1273	0.1018	-0.0004
arima_physics_enso_3	0.1304	0.1012	-0.0011
arima_lag_3_enso	0.1322	0.1044	0.0084
arima_auto_10	0.1356	0.1083	-0.0048
arima_enso_log_co2	0.1358	0.1076	-0.0304
arima_physics_robust	0.1376	0.1065	0.0158
arima_physics_robust_decay	0.1379	0.1064	0.0142
arima_co2_10_volcano	0.1597	0.1268	-0.0218
arima_co2_3_nino_tsi	0.1598	0.1275	0.0563
arima_co2_3_nino	0.1608	0.1290	0.0624

Across multiple folds, the forcing models and models that smooth out the enso index and/or utilize a lagged CO2 concentration variable perform the best. All the results are fairly close and would probably trade places over different subsets of the data so choosing the best one will require consideration of other factors

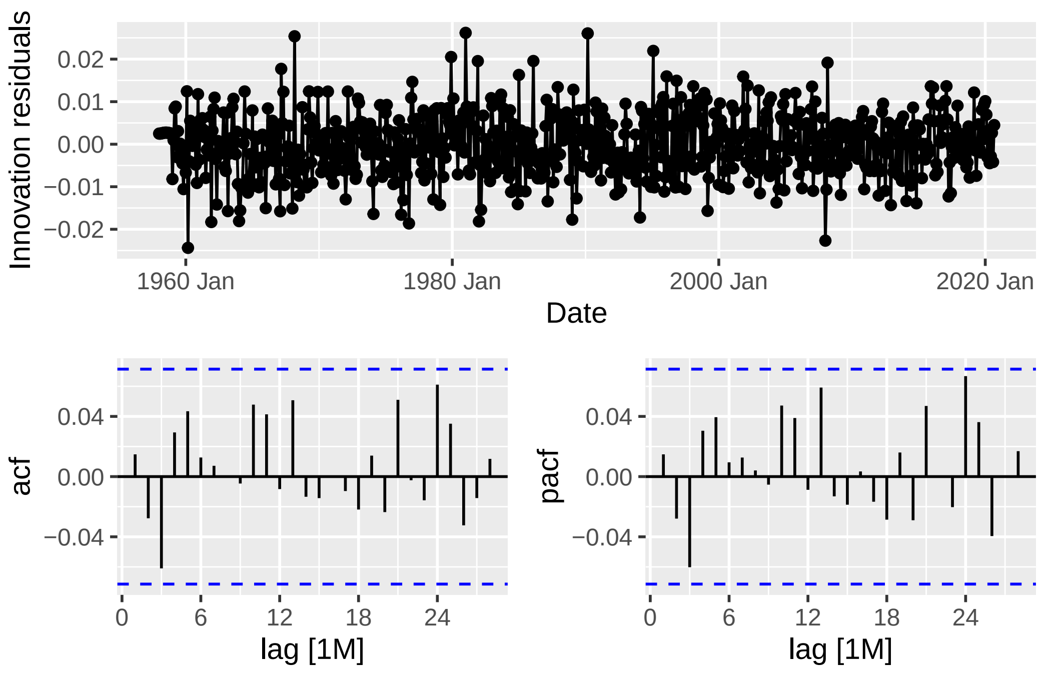
```
arima_second_fit |>
  glance() |>
```

```r
  select(.model, AICc, AIC, BIC) |>
  group_by(.model) |>
  summarize(across(.cols = c(AICc, AIC, BIC), .fns = mean)) |>
  arrange(AICc) |>
  kable(align = "c")
```

.model	AICc	AIC	BIC
arima_physics_robust_decay_enso_3	-4299.9720	-4300.3234	-4256.9483
arima_auto_10	-4272.7611	-4272.9678	-4240.2055
arima_physics_enso_3	-4271.2614	-4271.5471	-4232.5938
arima_enso_log_co2	-4260.9085	-4261.1570	-4225.8582
arima_lag_3_enso	-4257.9412	-4258.1951	-4221.9284
arima_auto_10_enso_6	-4211.9089	-4212.1762	-4175.0569
arima_physics_robust	-4196.0189	-4196.3555	-4153.8237
arima_physics_robust_decay	-4195.2754	-4195.6120	-4153.0802
arima_robust_enso_6	-4191.8260	-4192.1806	-4148.8485
arima_forcing	-4177.1452	-4177.4457	-4137.6491
arima_co2_10_volcano	-4140.9632	-4141.2000	-4105.7470
arima_co2_3_nino_tsi	-929.6644	-930.1066	-881.4664
arima_co2_3_nino	-917.6065	-917.9743	-873.7559

The top model by AICc was second best metric wise. It's also clear from the bottom models that a log transformation of the response variable drastically improves the models

```r
arima_second_fit |>
  select(arima_physics_robust_decay_enso_3) |>
  tail(n = 1) |>
  ggtime::gg_tsresiduals(plot_type = "partial")
```

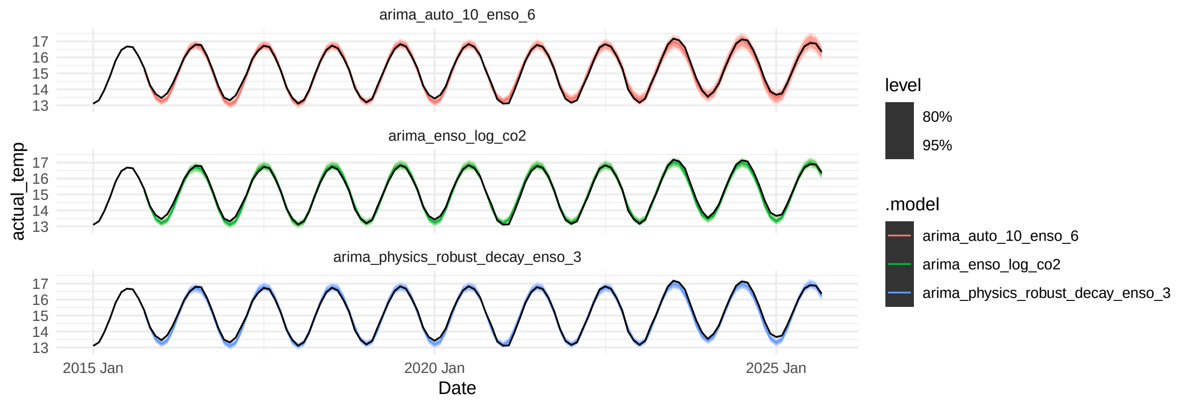


```
arima_second_fit |>
  select(arima_physics_robust_decay_enso_3) |>
  tail(n = 1) |>
  augment() |>
  features(.innov, ljung_box, lag = 24) |>
  kable(digits = 3, align = "c")
```

.id	.model	lb_stat	lb_pvalue
5	arima_physics_robust_decay_enso_3	17.372	0.832

No autocorrelation spikes in the residuals and the Ljung-Box test confirms that the residuals resemble white noise for the arima_physics_robust_decay_enso_3 model. Most other models have a spike at lag 3 or lag 24. The only other top model without any autocorrelation is arima_physics_enso_3 so a 3 month smoothing of the ENSO index seems to take care of it

```
arima_second_fc |>
  filter(.id %in% c(4, 5), .model %in% c("arima_physics_robust_decay_enso_3", "arima_enso_log
  autoplot() +
  autolayer(predictor_modeling |> filter(year(Date) >= 2015), actual_temp) +
  facet_wrap(~ .model, ncol = 1) +
  theme_minimal()
```

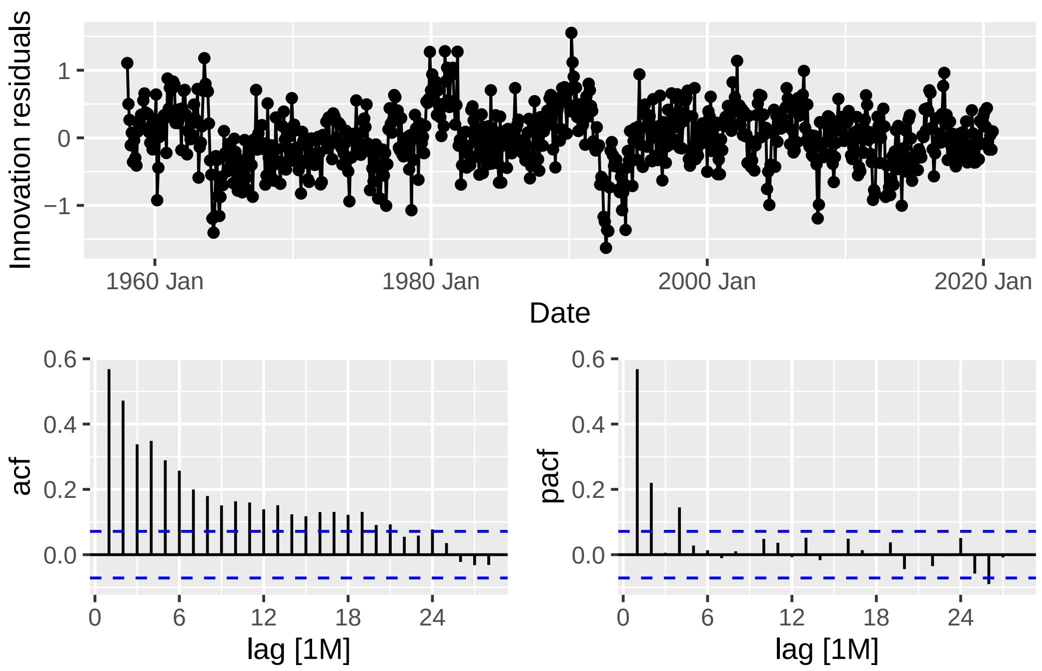


Retrain the top models on the same subset of data and compare

```
final_trn <- predictor_modeling |>
  filter(year(Date) > 1957) |>
  slice(1:(n() - 60))

final_valid <- predictor_modeling |>
  filter(year(Date) > 1957) |>
  slice_tail(n = 60)


final_fit <- cv_trn |>
  model(
    ets_auto = ETS(actual_temp),
    ets_additive = ETS(actual_temp ~ error("A") + trend("A") + season("A")),
    ets_box_auto = ETS(box_cox(actual_temp, lambda = 1.5)),
    tslm_no_trend = TSLM(box_cox(actual_temp, lambda = 1.5) ~ season() + enso_smooth_12 + co
    tslm_volcanic = TSLM(box_cox(actual_temp, lambda = 1.5) ~ season() + enso_smooth_12 + co
    tslm_enso_smooth_lag = TSLM(box_cox(actual_temp, lambda = 1.5) ~ trend() + season() + ens
    arima_auto_10 = ARIMA(log(actual_temp) ~ ENSO + co2_lag10 + PDQ(D = 1)),
    arima_physics_robust_decay_enso_3 = ARIMA(log(actual_temp) ~ rf_co2 + volcano_forcing + 
    arima_physics_enso_3 = ARIMA(log(actual_temp) ~ rf_co2 + enso_smooth_3 + PDQ(D = 1)),
    arima_enso_log_co2 = ARIMA(log(actual_temp) ~ ENSO + co2_ppm + PDQ(D = 1)),
  )

final_fc <- final_fit |>
  forecast(new_data = cv_valid)

final_fc |>
  accuracy(cv_valid) |>
  select(.model, ME:MAPE) |>
  group_by(.model) |>
  summarize(across(.cols = c(RMSE, MAE, ME), .fns = mean)) |>
  arrange(RMSE) |>
  kable(digits = 4)
```

.model	RMSE	MAE	ME
tslm_no_trend	0.1183	0.0927	-0.0032
tslm_volcanic	0.1191	0.0936	0.0013
tslm_enso_smooth_lag	0.1217	0.0950	-0.0063
arima_physics_robust_decay_enso_3	0.1233	0.0946	0.0005
arima_physics_enso_3	0.1304	0.1012	-0.0011
arima_auto_10	0.1356	0.1083	-0.0048
arima_enso_log_co2	0.1358	0.1076	-0.0304
ets_additive	0.1569	0.1272	0.0397
ets_box_auto	0.1621	0.1331	0.0342
ets_auto	0.1656	0.1348	0.0604

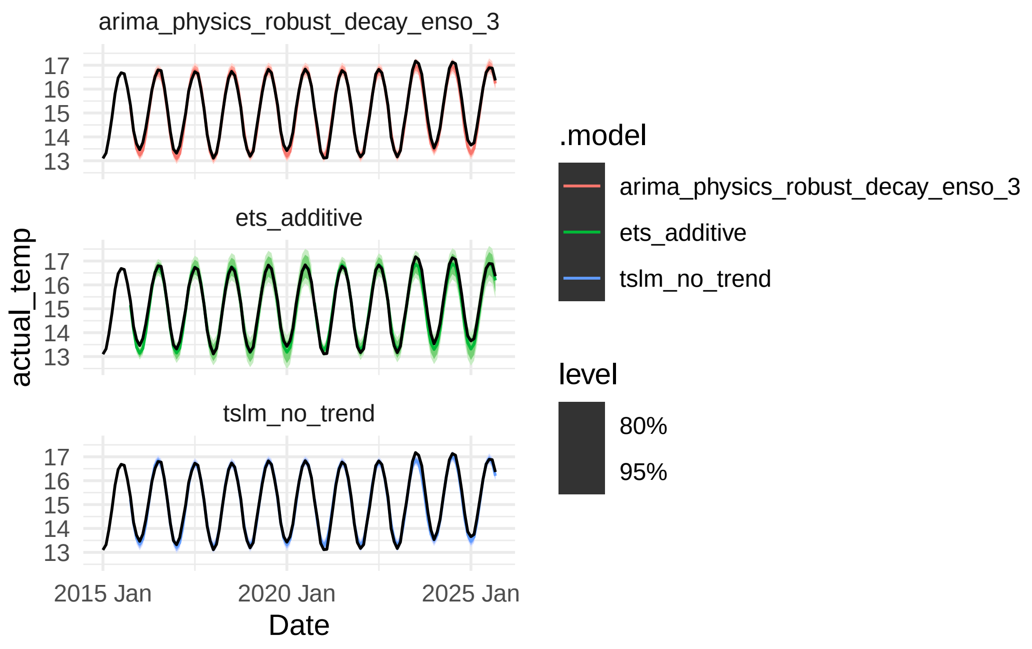
```
final_fit |>
  select(tslm_no_trend) |>
  tail(n = 1) |>
  ggtime::gg_tsresiduals(plot_type = "partial")
```



```
final_fit |>
  select(arima_auto_10) |>
  augment() |>
  features(.innov, ljung_box, lag = 24)
```

```
# A tibble: 5 x 4
    .id .model        lb_stat lb_pvalue
  <int> <chr>           <dbl>     <dbl>
1     1 arima_auto_10    19.0   0.749
2     2 arima_auto_10    45.6   0.00501
3     3 arima_auto_10    22.8   0.530
4     4 arima_auto_10    16.8   0.856
5     5 arima_auto_10    24.7   0.424
```

```
final_fc |>
  filter(.id %in% c(4, 5), .model %in% c("tslm_no_trend", "arima_physics_robust_decay_enso_3
  autoplot() +
  autolayer(predictor_modeling |> filter(year(Date) >= 2015), actual_temp) +
  facet_wrap(~ .model, ncol = 1) +
  theme_minimal()
```



```r
current_baseline <- 3400921

model_data <- predictor_modeling |>
  mutate(
    lag_emissions = lag(aggregate_emissions, 12),
    enso_lag = lag(enso_smooth_12, 4)
  )

temp_model <- model_data |>
  filter(year(Date) > 1957) |>
  model(
    temp_arima = ARIMA(log(actual_temp) ~ 0 + co2_lag10 + enso_smooth_6 + pdq(3, 0, 2) + PDQ
  )

# for the enso index, the average across the whole data set is 0.0470

low_enso <- predictor_modeling |>
  select(ENSO) |>
  slice_tail(n = 600)
# 0.0158

high_enso <- predictor_modeling |>
  select(ENSO) |>
  slice_head(n = 600)

# 0.0997

normal_enso <- predictor_modeling |>
  select(ENSO) |>
  slice(650:1249)

# 0.0469
#
# Then would need smoothed versions of these

co2_model <- model_data |>
  model(
    co2_arima = ARIMA(co2_ppm ~ lag_emissions + enso_lag + pdq(1, 1, 1) + PDQ(0, 1, 1)),
  )

# Model uses a 1 year lag on emissions and enso index is a 12 month smooth and 4 month lag
```

"noticed a spike at 12 and 24 for arima_auto_10 and auto was choosing ARIMA(3, 0, 2)(1, 1, 0) with enso_smooth_6 or ARIMA(2, 0, 2)(1, 1, 0) with enso_smooth_3. So added a sma term and that got rid of the spikes and the sar term showed as not significant so removed that. AICc was slightly higher using enso_smooth_6"

"Arima_auto_10_6 has lower AICc than all and residuals are white noise"

```
scenario_forecasting <- function(growth_rate, years, enso_level) {

  most_recent_observation <- predictor_modeling |> tail(1)
  baseline_aggregate <- most_recent_observation$aggregate_emissions

  annual_rate <- (1 + growth_rate/100)

  year_vector <- 1:years
  month_vector <- years * (1:12)

  forecast <- current_baseline * (annual_rate ** year_sequence)

  month_sequence <- rep(forecast, each = years)

  new_scenarios <- scenarios(
    enso_low = new_data(predictor_modeling, years * 12) |>
      mutate(
        ENSO = low_enso$ENSO,
      ),
    enso_norm = new_data(predictor_modeling, years * 12) |>
      mutate(
        ENSO = normal_enso$ENSO,
      ),
    enso_high = new_data(predictor_modeling, years * 12) |>
      mutate(
        ENSO = high_enso$ENSO,
      ),
    emissions = new_data(predictor_modeling, years * 12) |>
      mutate(
        total_co2 = month_sequence,
        aggregate_emissions = baseline_aggregate + cumsum(total_co2 / 1e6)
      )
  )

  co2_modeling_data <- new_scenarios[[enso_level]] |>
    left_join(new_scenarios$emissions, by = "Date") |>
    bind_rows(predictor_modeling) |>
    mutate(
      lag_emissions = lag(aggregate_emissions, 12),
      enso_smooth_3 = slider::slide_dbl(ENSO, mean, .before = 2),
      enso_smooth_6 = slider::slide_dbl(ENSO, mean, .before = 5),
      enso_smooth_12 = slider::slide_dbl(ENSO, mean, .before = 11),
      enso_lag = lag(enso_smooth_12, 4)
    )|>
    filter(Date >= yearmonth("2025 Oct"))

  co2_ppm_fc <- co2_model |>
    forecast(new_data = co2_modeling_data)

  co2_modeling_data$co2_ppm <- co2_ppm_fc$.mean

  temp_modeling_data <- predictor_modeling |>
    bind_rows(co2_modeling_data) |>
    mutate(
      co2_lag10 = lag(co2_ppm, 120),
    ) |>
    filter(Date >= yearmonth("2025 Oct"))
```

```r
  temperature_forecast <- temp_model |>
    forecast(new_data = temp_modeling_data)

  return(temperature_forecast)
}
```