````
---
title: "01_Clean_Data_Plot"
author: "Callum Weinberg"
date: "November 26, 2021"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

## Libraries

```{r}
library(dplyr, warn.conflicts = FALSE) #Using
library(tidyr) #Using
library(knitr) #Using
library(lubridate, warn.conflicts = FALSE) #Using
library(ggplot2) #Using
library(MASS) #Uncertain
library(qpcR) #Using
library(forecast) #Using
library(cowplot) #Using
library(TSA) #Using
```

###### Import and Clean the Data #####

```{r}
## Load the Data
landings_full = read.csv("Data/SB_Red_Sea_Urchin_Landings_2008_2019.csv")

## Clean the Data
# Rename the Sea-Urchin Landings variable to pounds
# for simplicity in analysis
names(landings_full)[3] = "pounds"

# Create a Monthly Date Variable
landings_full$date = as.Date(with(landings_full,
                             paste0(as.character(landings_full$Year),"-",
                                    as.character(landings_full$Month),"-01"), "%Y-%m-%d"))

# Create a Separate Dataset for 2008-2018
landings = landings_full[1:132,]
```



##### Plot the Sea-Urching Landings Data #####

```{r}
## Plot the Original Data
full_plot = ggplot(data = landings, mapping = aes(x = date, y = pounds/1000)) +
  geom_line() +
  labs(x = "Date", y = "Thousands of Pounds") +
  #labs(title = "Red Sea Urchin Landings in the Santa Barbara Area\nMeasured in Thounsands of Pounds\nMonthly, 2008-
2018") +
  scale_x_date(breaks = scales::breaks_pretty(10)) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
full_plot

## Plots for report
png(filename = "Images/2008_2018_plot.png", width = 960, height = 480)
full_plot
dev.off()
```

###### Save the Cleaned Data Out #####

```{r}
save(landings,file="Data/landings.Rdata")
save(landings_full,file="Data/landings_full.Rdata")
```
````

~~~~~
##### ACF of Original Data #####

```{r}
## Sample ACF
sample_acf_list = acf(landings$pounds, plot = FALSE, lag.max = 100)

# Put into Dataframe
sample_acf = as.data.frame(do.call(cbind, sample_acf_list))

# Confidence Interval Line
conf.level = 0.95
ciline = qnorm((1 - conf.level)/2)/sqrt(length(landings_transformed_season_only$pounds_transformed))

# Plot
ACF_original= ggplot(data = sample_acf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "ACF") +
  #labs(title = "") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

ACF_original

## Plots for report
png(filename = "Images/ACF_original.png", width = 960, height = 480)
ACF_original
dev.off()
````
~~~~~

````
---
title: "02_Variance_Stabilization_and_Differencing"
author: "Callum Weinberg"
date: "December 1, 2021"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

## Libraries

```{r}
library(dplyr, warn.conflicts = FALSE) #Using
library(tidyr) #Using
library(knitr) #Using
library(lubridate, warn.conflicts = FALSE) #Using
library(ggplot2) #Using
library(MASS) #Uncertain
library(qpcR) #Using
library(forecast) #Using
library(cowplot) #Using
library(TSA) #Using
```

## Part 2a: Load Data from 01

```{r}
load(file="Data/landings.Rdata")
```


##### Check if the Variance is Stable with a Histogram #####

```{r}
# Histogram Pounds of Red Sea Urchin
pounds_histogram = ggplot(landings, aes(x = pounds/1000)) +
  geom_histogram(binwidth = 50) +
  geom_density(aes(y = 50* ..count..),alpha = 0.05, fill = "red") +
  labs(x = "Thousands of Pounds of Red Sea Urchin", y = "Frequency") +
  #scale_x_continuous(label = comma) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 15),
    legend.text = element_text(size = 15),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
pounds_histogram

# Display Variance
var(landings$pounds)/100000

## Plots for report
png(filename = "Images/histogram_raw_data.png", width = 960, height = 480)
pounds_histogram
dev.off()
```

##### Stabilize Variance #####

```{r}
## Stabalize Variance

## Apply Transformation, Homoskedastic
#find optimal lambda for Box-Cox transformation
t = 1:length(landings$pounds)
bcTransform = boxcox((landings$pounds) ~ t,plotit = TRUE)

lambda = bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
print(lambda)

# Use .8 for simplicity
lambda_chosen = .8

# Box Cox Transformation, Log and Sqrt don't work well
# Confidence Interval Includes 1, using the Original Data
# and no Transformation
````
````

````
landings =
  landings %>%
  #mutate(pounds.bc = (1/lambda_chosen)*((pounds)^lambda_chosen-1))
  #mutate(pounds.bc = log(pounds))
  mutate(pounds.bc = pounds) # Note, if no transformation used in final just get rid of this

## Plot Box-Cox for report
png(filename = "Images/box_cox.png", width = 480, height = 480)
bcTransform = boxcox((landings$pounds) ~ t,plotit = TRUE)
dev.off()
```


#####  Difference at Lag 12 to Remove Seasonality #####

```{r}
# Try 12, Since Data Retrieved Monthly

# Difference
landings_diff12 = landings[13:132,]
landings_diff12$pounds.bc_diff12 = diff(landings$pounds.bc,lag = 12)

# Plot Time Series after Differencing
full_plot_diff12 = ggplot(data = landings_diff12, mapping = aes(x = date, y = pounds.bc_diff12/1000)) +
  geom_line() +
  geom_smooth(method='lm', formula= y~x, se = FALSE) +
  labs(x = "Date", y = "Thousands of Pounds of Red Sea Urchin\nDifferenced at Lag 12") +
  #labs(title = "Red Sea Urching Landings 2008-2018\nDifferenced at Lag 12") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
full_plot_diff12

# Plot Histogram after Differencing
pounds_histogram_diff12 =
  ggplot(landings_diff12, aes(x = pounds.bc_diff12/1000)) +
  geom_histogram(binwidth = 50) +
  geom_density(aes(y = 50* ..count..),alpha = 0.05, fill = "red") +
  labs(x = "Thousands of Pounds of Red Sea Urchin\nDifferenced at Lag 12", y = "Frequency") +
  #scale_x_continuous(label = comma) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 15),
    legend.text = element_text(size = 15),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
pounds_histogram_diff12

# Display Variance after This Difference
var(landings_diff12$pounds.bc_diff12)/100000


## Save the Transformed Data Out
landings_transformed_season_only = landings_diff12
i = ncol(landings_transformed_season_only)
names(landings_transformed_season_only)[i] = "pounds_transformed"
save(landings_transformed_season_only,file="Data/landings_transformed_season_only.Rdata")


## Plots for Report
png(filename = "Images/2008_2018_diff12.png", width = 960, height = 480)
plot_grid(full_plot_diff12,pounds_histogram_diff12, labels = NULL, label_size = 12, ncol = 2, nrow = 1)
dev.off()
```



######  Difference to Remove Trend After Seasonality Difference #####

# NOT USED IN MAIN REPORT
```{r}
# Difference at Lag 1
landings_diff12_diff1 = landings_diff12[2:120,]
````

````
landings_diff12_diff1$pounds.bc_diff12_1 = diff(landings_diff12$pounds.bc_diff12,lag = 1)


# Plot the Data
full_plot = ggplot(data = landings_diff12_diff1, mapping = aes(x = date, y = pounds.bc_diff12_1)) +
  geom_line() +
  geom_smooth(method='lm', formula= y~x) +
  labs(x = "Date", y = "Pounds Transformed",
       title = "Red Sea Urching Landings 2008-2018\nDifferenced at Lag 12 and at Lag 1") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
full_plot

# Calculate the Variance of the Differenced Data
var(landings_diff12_diff1$pounds.bc_diff12_1)/100000

# Histogram Pounds of Red Sea Urchin for Transformed Data
pounds_histogram =
  ggplot(landings_diff12_diff1, aes(x = pounds.bc_diff12_1)) +
  geom_histogram(aes(y = ..density..)) +
  geom_density(alpha = 0.1, fill = "red") +
  #labs(title = "By ZCTA") +
  labs(x = "Pounds of Red Sea Urchin", y = "Frequency") +
  #scale_x_continuous(label = comma) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 15),
    legend.text = element_text(size = 15),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
pounds_histogram

## Save the Transformed Data Out
landings_transformed = landings_diff12_diff1
i = ncol(landings_transformed)
names(landings_transformed)[i] = "pounds_transformed"
save(landings_transformed,file="Data/landings_transformed.Rdata")

```



######  De-Trend Without The Seasonality Difference #####

# NOT USED IN MAIN REPORT
```{r}
# Consider Other Possibilities in Code Below (i.e. difference twice no season, )
landings_diff12_diff1 = landings[2:132,]
landings_diff12_diff1$pounds.bc_diff12_1 = diff(landings$pounds.bc,lag = 1)

# Consider a second difference, but increases the variance
#landings_diff12_diff1_1 = landings[2:131,] # NEED TO SHOW IN FINAL CODE THAT THIS INCREASES THE VARIANCE
#landings_diff12_diff1_1$pounds.bc_diff12_1_1 = diff(landings_diff12_diff1$pounds.bc_diff12_1,lag = 1)


# Plot the Data
full_plot = ggplot(data = landings_diff12_diff1, mapping = aes(x = date, y = pounds.bc_diff12_1)) +
  geom_line() +
  geom_smooth(method='lm', formula= y~x) +
  labs(x = "Date", y = "Pounds Transformed",
       title = "Red Sea Urching Landings 2008-2018\nDifferenced at Lag 12 and at Lag 1") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
full_plot

# Calculate the Variance of the Differenced Data
var(landings_diff12_diff1$pounds.bc_diff12_1)/100000
````

```r
# Histogram Pounds of Red Sea Urchin for Transformed Data
pounds_histogram =
  ggplot(landings_diff12_diff1, aes(x = pounds.bc_diff12_1)) +
  geom_histogram(aes(y = ..density..)) +
  geom_density(alpha = 0.1, fill = "red") +
  #labs(title = "By ZCTA") +
  labs(x = "Pounds of Red Sea Urchin", y = "Frequency") +
  #scale_x_continuous(label = comma) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 15),
    legend.text = element_text(size = 15),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
pounds_histogram

## Save the Transformed Data Out
landings_transformed_no_season = landings_diff12_diff1
i = ncol(landings_transformed_no_season)
names(landings_transformed_no_season)[i] = "pounds_transformed"
save(landings_transformed_no_season,file="Data/landings_transformed_no_season.Rdata")
```

```
---
title: "03_ACF_PACF_Plots"
author: "Callum Weinberg"
date: "December 1, 2021"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

Libraries

```{r}
library(dplyr, warn.conflicts = FALSE) #Using
library(tidyr) #Using
library(knitr) #Using
library(lubridate, warn.conflicts = FALSE) #Using
library(ggplot2) #Using
library(MASS) #Uncertain
library(qpcR) #Using
library(forecast) #Using
library(cowplot) #Using
library(TSA) #Using
```

Seasonal (Lag 12) Only Differenced Data

```{r}
load(file="Data/landings_transformed_season_only.Rdata")
```

```{r}
## Sample ACF
sample_acf_list = acf(landings_transformed_season_only$pounds_transformed, plot = FALSE, lag.max = 60)

# Put into Dataframe
sample_acf = as.data.frame(do.call(cbind, sample_acf_list))

# Confidence Interval Line
conf.level = 0.95
ciline = qnorm((1 - conf.level)/2)/sqrt(length(landings_transformed_season_only$pounds_transformed))

# Plot
ACF_Sample_Graph = ggplot(data = sample_acf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "ACF") +
  #labs(title = "Sample Autocorrelation Function\nfor De-Trended De-Seasoned Red Sea Urchin Landings\nMonthly for 2008-
2018") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))


## Sample PACF
sample_pacf_list = pacf(landings_transformed_season_only$pounds_transformed, plot = FALSE, lag.max = 60)

# Put into Dataframe
sample_pacf = as.data.frame(do.call(cbind, sample_pacf_list))

# Plot
PACF_Sample_Graph = ggplot(data = sample_pacf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "Partial ACF") +
  #labs(title = "Sample Partial Autocorrelation Function\nfor De-Trended, De-Seasoned Red Sea Urchin Landings\nMonthly
for 2008-2018") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
```
```

````
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

## Plots for Report
png(filename = "Images/acf_pacf.png", width = 960, height = 480)
plot_grid(ACF_Sample_Graph,PACF_Sample_Graph, labels = NULL, label_size = 12, ncol = 2, nrow = 1)
dev.off()
```



###### Seasonal AND Trend-Differenced Data #####

# NOT USED IN REPORT
```{r}
load(file="Data/landings_transformed.Rdata")
```

```{r}
## Sample ACF
sample_acf_list = acf(landings_transformed$pounds_transformed, plot = FALSE, lag.max = 100)

# Put into Dataframe
sample_acf = as.data.frame(do.call(cbind, sample_acf_list))

# Confidence Interval Line
conf.level = 0.95
ciline = qnorm((1 - conf.level)/2)/sqrt(length(landings_transformed$pounds_transformed))

# Plot
ACF_Sample_Graph = ggplot(data = sample_acf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "ACF",
       title = "Sample Autocorrelation Function\nfor De-Trended De-Seasoned Red Sea Urchin Landings\nMonthly for 2008-
2018") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))


## Sample PACF
sample_pacf_list = pacf(landings_transformed$pounds_transformed, plot = FALSE, lag.max = 100)

# Put into Dataframe
sample_pacf = as.data.frame(do.call(cbind, sample_pacf_list))

# Plot
PACF_Sample_Graph = ggplot(data = sample_pacf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "Partial ACF",
       title = "Sample Partial Autocorrelation Function\nfor De-Trended, De-Seasoned Red Sea Urchin Landings\nMonthly
for 2008-2018") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

## Plot Both
ACF_Sample_Graph
PACF_Sample_Graph
```
````
````

````
###### Option 3: Trend-Only Differenced Data #####

# NOT USED IN THE REPORT
```{r}
load(file="Data/landings_transformed_no_season.Rdata")
```

NOT USED IN REPORT
```{r}
## Sample ACF
sample_acf_list = acf(landings_transformed_no_season$pounds_transformed, plot = FALSE, lag.max = 100)

# Put into Dataframe
sample_acf = as.data.frame(do.call(cbind, sample_acf_list))

# Confidence Interval Line
conf.level = 0.95
ciline = qnorm((1 - conf.level)/2)/sqrt(length(landings_transformed_no_season$pounds_transformed))

# Plot
ACF_Sample_Graph = ggplot(data = sample_acf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "ACF",
       title = "Sample Autocorrelation Function\nfor De-Trended De-Seasoned Red Sea Urchin Landings\nMonthly for 2008-
2018") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))


## Sample PACF
sample_pacf_list = pacf(landings_transformed_no_season$pounds_transformed, plot = FALSE, lag.max = 100)

# Put into Dataframe
sample_pacf = as.data.frame(do.call(cbind, sample_pacf_list))

# Plot
PACF_Sample_Graph = ggplot(data = sample_pacf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "Partial ACF",
       title = "Sample Partial Autocorrelation Function\nfor De-Trended, De-Seasoned Red Sea Urchin Landings\nMonthly
for 2008-2018") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

## Plot Both
ACF_Sample_Graph
PACF_Sample_Graph
```
````

````
---
title: "04_Fitting_Models"
author: "Callum Weinberg"
date: "December 1, 2021"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

## Libraries

```{r}
library(dplyr, warn.conflicts = FALSE) #Using
library(tidyr) #Using
library(knitr) #Using
library(lubridate, warn.conflicts = FALSE) #Using
library(ggplot2) #Using
library(MASS) #Uncertain
library(qpcR) #Using
library(forecast) #Using
library(cowplot) #Using
library(TSA) #Using
```



###### MODEL WITH ONLY SEASONAL DIFFERENCE (Lag 12) #######

```{r}
load(file="Data/landings_transformed_season_only.Rdata")
landing_ts_so = landings_transformed_season_only$pounds_transformed
```

## Model 40
```{r}
model40 = arima(landing_ts_so, order=c(2,0,2), seasonal = list(order = c(1,1,1), period = 12),
                method = "ML",fixed = c(NA,NA,0,NA,NA,NA))
model40
AICc(model40)


# Phi (corresponsing to AR)
AR = polyroot(c(1,-.1195,-.8804))
AR_df = data.frame(Root = c("AR1","AR2"),
                   real = Re(AR), im = Im(AR))
AR_df =
  AR_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest AR Root on Complex Plane: ",min(AR_df$z))

# Theta (Corresponding to MA)
MA = polyroot(c(1,0,-0.8638))
MA_df = data.frame(Root = c("MA1","MA2"),
                   real = Re(MA), im = Im(MA))
MA_df =
  MA_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest MA Root on Complex Plane: ",min(MA_df$z))

# PHI (corresponsing to SAR)
SAR = polyroot(c(1,0.4485))
SAR_df = data.frame(Root = c("SAR1"),
                   real = Re(SAR), im = Im(SAR))
SAR_df =
  SAR_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest SAR Root on Complex Plane: ",min(SAR_df$z))

# THETA (Corresponding to SMA)
SMA = polyroot(c(1,-e0.9806))
SMA_df = data.frame(Root = c("SMA1"),
                   real = Re(SMA), im = Im(SMA))
SMA_df =
  SMA_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest SMA Root on Complex Plane: ",min(SMA_df$z))

kable(rbind(AR_df,MA_df,SAR_df,SMA_df), caption = "Phi(B) Roots and Theta(B) Roots")
````
````

````
```

# Model 41

```{r}
model41 = arima(landing_ts_so, order=c(2,0,2), seasonal = list(order = c(1,1,1), period = 12),
                method = "ML",fixed = c(NA,NA,NA,NA,NA,NA))
model41
AICc(model41)


# Phi (corresponsing to AR)
#AR = polyroot(c(1,0.5245,-0.5261,-0.9983)) for 303 x 111
AR = polyroot(c(1,-.1195,-.8804))
AR_df = data.frame(Root = c("AR1","AR2"),
                   real = Re(AR), im = Im(AR))
AR_df =
  AR_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest AR Root on Complex Plane: ",min(AR_df$z))


# MA = polyroot(c(1,-.6319,0.3399,.8839))  for 303 x 111
MA = polyroot(c(1,0,0.8638))
MA_df = data.frame(Root = c("MA1","MA2"),
                   real = Re(MA), im = Im(MA))
MA_df =
  MA_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest MA Root on Complex Plane: ",min(MA_df$z))

kable(rbind(AR_df,MA_df), caption = "Phi(B) Roots and Theta(B) Roots")
```



## Model 42
```{r}
model42 = arima(landing_ts_so, order=c(3,0,3), seasonal = list(order = c(1,1,1), period = 12),
                method = "ML",fixed = c(NA,NA,0,NA,NA,NA,NA,NA))
model42
AICc(model42)

# Theta_1 = 1, not invertible
```



# Model 43

```{r}
# Model 43
model43 = arima(landing_ts_so, order=c(1,0,1), seasonal = list(order = c(1,1,1), period = 12),
                method = "ML",fixed = c(NA,NA,NA,NA))
model43
AICc(model43)

# Phi (corresponsing to AR)
AR = polyroot(c(1,-0.9999))
AR_df = data.frame(Root = c("AR1"),
                   real = Re(AR), im = Im(AR))
AR_df =
  AR_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest AR Root on Complex Plane: ",min(AR_df$z))


# Theta (corresponding to MA)
MA = polyroot(c(1,-0.8988))
MA_df = data.frame(Root = c("MA1"),
                   real = Re(MA), im = Im(MA))
MA_df =
  MA_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest MA Root on Complex Plane: ",min(MA_df$z))

# PHI (corresponsing to SAR)
SAR = polyroot(c(1,0.4546))
SAR_df = data.frame(Root = c("SAR1"),
                   real = Re(SAR), im = Im(SAR))
SAR_df =
  SAR_df %>%
````

````
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest SAR Root on Complex Plane: ",min(SAR_df$z))

# THETA (Corresponding to SMA)
SMA = polyroot(c(1,-0.9822))
SMA_df = data.frame(Root = c("SMA1"),
                   real = Re(SMA), im = Im(SMA))
SMA_df =
  SMA_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest SMA Root on Complex Plane: ",min(SMA_df$z))

kable(rbind(AR_df,MA_df,SAR_df,SMA_df), caption = "Phi(B) Roots and Theta(B) Roots")

# Stationary and Invertible, although the ar1 term is only just stationary
```

# Model 44

```{r}
model44 = arima(landing_ts_so, order=c(2,0,3), seasonal = list(order = c(1,1,1), period = 12),
                method = "ML",fixed = c(0,NA,0,0,NA,NA,NA))
model44
AICc(model44)

# Phi (corresponsing to AR)
AR = polyroot(c(1,0,-0.2517))
AR_df = data.frame(Root = c("AR1","AR2"),
                   real = Re(AR), im = Im(AR))
AR_df =
  AR_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest AR Root on Complex Plane: ",min(AR_df$z))

# Theta (corresponding to MA)
MA = polyroot(c(1,0,0,0.2789))
MA_df = data.frame(Root = c("MA1","MA2","MA3"),
                   real = Re(MA), im = Im(MA))
MA_df =
  MA_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest MA Root on Complex Plane: ",min(MA_df$z))

# PHI (corresponsing to SAR)
SAR = polyroot(c(1,0.4122))
SAR_df = data.frame(Root = c("SAR1"),
                   real = Re(SAR), im = Im(SAR))
SAR_df =
  SAR_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest SAR Root on Complex Plane: ",min(SAR_df$z))

# THETA (Corresponding to SMA)
SMA = polyroot(c(1,-0.9016))
SMA_df = data.frame(Root = c("SMA1"),
                   real = Re(SMA), im = Im(SMA))
SMA_df =
  SMA_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest SMA Root on Complex Plane: ",min(SMA_df$z))

kable(rbind(AR_df,MA_df,SAR_df,SMA_df), caption = "Phi(B) Roots and Theta(B) Roots")

```
````
````

MODELS WITH SEASONAL DIFFERENCE AND TREND DIFFERENCE

NOT USED IN THE REPORT

```{r}
load(file="Data/landings_transformed.Rdata")
landing_ts = landings_transformed$pounds_transformed
```

0 Model

SARIMA (1,1,0)x(1,1,0) 12

This is viable: seems not great based on the PACF, but not sure

```{r}
model0 = arima(landing_ts, order=c(1,1,0), seasonal = list(order = c(1,1,0), period = 12), method = "ML")
model0
AICc(model0)


# Phi (corresponsing to AR)
AR = polyroot(c(1,-0.7046))
roots_AR = c("Root 1")
root_model0 = data.frame(Root = roots_AR, Value = AR)
kable(root_model0, caption = "Phi(B) Roots")
```

Model 1

Double Check Stationarity

```{r}
model1 = arima(landing_ts, order=c(2,1,0), seasonal = list(order = c(1,1,0), period = 12), method = "ML")
model1
AICc(model1)


# Phi (corresponsing to AR)
AR = polyroot(c(1,1.1017,.5427))
roots_AR = c("Root 1", "Root 2")
root_model1 = data.frame(Root = roots_AR, Value = AR)
kable(root_model1, caption = "Phi(B) Roots")
```

Model 2

SARIMA (3,1,0)x(1,1,0) 12

NON-STATIONARY: Non complex roots less than 1

```{r}
model2 = arima(landing_ts, order=c(3,1,0), seasonal = list(order = c(1,1,0), period = 12), method = "ML")
model2
AICc(model2)

# Phi (corresponsing to AR)
AR = polyroot(c(1,1.3384,1.0118,.4160))
roots_AR = c("Root 1", "Root 2","Root 3")
root_model2 = data.frame(Root = roots_AR, Value = AR)
kable(root_model2, caption = "Phi(B) Roots")
```

Model 3

SARIMA (14,1,0)x(0,1,0) 12

NON-STATIONARY: Need to include this in the code to show that the I tried it

```{r}
model3 = arima(landing_ts, order=c(14,1,0), seasonal = list(order = c(0,1,0), period = 12), method = "ML", fixed =
c(NA,NA,NA,NA,NA,NA,NA,0,0,0,0,0,0,NA))
model3
AICc(model3)

# Phi (corresponsing to AR)
AR = polyroot(c(1,-1.5022,-1.7596,-1.5529,-1.3036,-1.0724,-0.7035,-0.3185,0,0,0,0,0,0,-0.2220))
roots_AR = c("Root 1", "Root 2","Root 3","Root 4", "Root 5","Root 6","Root 7", "Root 8","Root 9","Root 10", "Root
11","Root 12","Root 13", "Root 14")
root_model3 = data.frame(Root = roots_AR, Value = AR)
```

````
kable(root_model3, caption = "Phi(B) Roots")
```

# Model 4

SARIMA (27,1,0)x(0,1,0) 12

```{r}
model4 = arima(landing_ts, order=c(27,1,0), seasonal = list(order = c(0,1,0), period = 12), method = "ML")
model4
AICc(model4)
```

##### MODELS WITH ONLY TREND DIFFERENCE


## NOT USED IN THE REPORT

```{r}
load(file="Data/landings_transformed_no_season.Rdata")
landing_ts_ns = landings_transformed_no_season$pounds_transformed
```

MODEL 21
```{r}
# THIS WORKS, BUT PROBABLY NOT A GREAT MODEL
model21 = arima(landing_ts_ns, order=c(2,1,1), method = "ML")
model21
AICc(model21)

# Phi (corresponsing to AR)
AR = polyroot(c(1,-0.6605))
roots_AR = c("Root 1")
root_model21 = data.frame(Root = roots_AR, Value = AR)
kable(root_model21, caption = "Phi(B) Roots")
```
MODEL 22
```{r}
# Revised Model based on Residual PACF
model22 = arima(landing_ts_ns, order=c(15,1,0), method = "ML")
model22
AICc(model22)

# Phi (corresponsing to AR)
AR =
polyroot(c(1,1.7032,2.0569,2.1340,2.1507,2.2627,2.4141,2.5125,2.4864,2.3683,2.1207,1.7952,1.3713,0.9685,0.6628,0.2976))
AR_df = data.frame(Root = seq(1,15,by=1),
                   real = Re(AR), im = Im(AR))
AR_df =
  AR_df %>%
  mutate(z = sqrt(real^2 + im^2))
paste0("Smallest Root on Complex Plane: ",min(AR_df$z))
kable(AR_df, caption = "Phi(B) Roots")
```

MODEL 23
```{r}
model23 = auto.arima(
  landing_ts_ns,
  d = 1,
  max.p = 20,
  max.q = 20,
  max.P = 2,
  max.Q = 2,
  max.order = 50,
  max.D = 1,
  start.p = 0,
  start.q = 0,
  start.P = 0,
  start.Q = 0,
  stationary = FALSE)

model23
````

````
```

MODEL 24
```{r}
# NOT STATIONARY
model24 = arima(landing_ts_ns, order=c(3,1,0), method = "ML")
model24
AICc(model24)


# Phi (corresponsing to AR)
AR = polyroot(c(1,1.2063,0.8988,0.3598))
roots_AR = c("Root 1", "Root 2","Root 3")
root_model8 = data.frame(Root = roots_AR, Value = AR)
kable(root_model8, caption = "Theta(B) Roots")

acf(residuals(model24))
pacf(residuals(model24))
```


```{r}
# Non Stationary
model9 = arima(landing_ts, order=c(2,1,0), method = "ML")
model9
AICc(model9)

# Phi (corresponsing to AR)
AR = polyroot(c(1,-1.0559,-0.5322))
roots_AR = c("Root 1","Root2")
root_model9 = data.frame(Root = roots_AR, Value = AR)
kable(root_model9, caption = "Phi(B) Roots")
```
````
````

```
---
title: "05_Diagnostic_Tests"
author: "Callum Weinberg"
date: "December 1, 2021"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

Libraries

```{r}
library(dplyr, warn.conflicts = FALSE) #Using
library(tidyr) #Using
library(knitr) #Using
library(lubridate, warn.conflicts = FALSE) #Using
library(ggplot2) #Using
library(MASS) #Uncertain
library(qpcR) #Using
library(forecast) #Using
library(cowplot) #Using
library(TSA) #Using
```

QQplot Function
```{r}
#Function for QQ Plot in GGPLOT
# Source: https://stackoverflow.com/questions/4357031/qqnorm-and-qqline-in-ggplot2
qqplot_residuals <- function (vec) # argument: vector of numbers
{
  # following four lines from base R's qqline()
  y <- quantile(vec[!is.na(vec)], c(0.25, 0.75))
  x <- qnorm(c(0.25, 0.75))
  slope <- diff(y)/diff(x)
  int <- y[1L] - slope * x[1L]

  d <- data.frame(resids = vec)

  ggplot(d, aes(sample = resids)) +
    stat_qq(color = "blue") +
    geom_abline(slope = slope, intercept = int) +
  theme(plot.title = element_text(hjust = 0.5))
}
```

SEASONAL ONLY MODELS

```{r}
load(file="Data/landings_transformed_season_only.Rdata")
landing_ts_so = landings_transformed_season_only$pounds_transformed
```

```{r}
# Model 40 AKA Model 1 in the Report
model40 = arima(landing_ts_so, order=c(2,0,2), seasonal = list(order = c(1,1,1), period = 12),
                method = "ML",fixed = c(NA,NA,0,NA,NA,NA))
model40
AICc(model40)


# Diagnostics
Box.test(residuals(model40),lag = 11, type = ("Box-Pierce"), fitdf = 5) # Only 5 since one term fixed at 0
Box.test(residuals(model40),lag = 11, type = ("Ljung-Box"), fitdf = 5)
Box.test(residuals(model40)^2,lag = 11, type = ("Ljung-Box"), fitdf = 0)
shapiro.test(residuals(model40))


## acf
acf_resid = acf(residuals(model40),main = "Autocorrelation", lag.max = 60)
# Put into Dataframe
sample_acf = as.data.frame(do.call(cbind, acf_resid))
# Confidence Interval Line
conf.level = 0.95
ciline_resid = qnorm((1 - conf.level)/2)/sqrt(132)

# Plot
ACF_residual_mod1= ggplot(data = sample_acf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
```
```

````
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline_resid), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline_resid), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "ACF") +
  #labs(title = "") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

## pacf
pacf_resid = pacf(residuals(model40),main = "Autocorrelation", lag.max = 60)
# Put into Dataframe
sample_pacf = as.data.frame(do.call(cbind, pacf_resid))
# Plot
PACF_residual_mod1= ggplot(data = sample_pacf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline_resid), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline_resid), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "PACF") +
  #labs(title = "") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

# Histogram
hist_df_mod = data.frame(x = residuals(model40))
histogram_resid1 = ggplot(data = hist_df, aes(x = x/1000)) +
  geom_histogram(aes(y = ..density..)) +
  geom_density(alpha = 0.1, fill = "red") +
  labs(x = "Residuals (Divided by 1000)", y = "Frequency") +
  #scale_x_continuous(label = comma) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 15),
    legend.text = element_text(size = 15),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
histogram_resid

# q-q plot
qq_resid_mod1 = qqplot_residuals(residuals(model40))

# Plot All Together
#png(filename = "Images/diagnostics_mod1.png", width = 960, height = 960)
model1_grid = plot_grid(qq_resid_mod1,histogram_resid1,ACF_residual_mod1,PACF_residual_mod1, labels = NULL, label_size =
12, ncol = 2, nrow = 2)
#dev.off()

```

```{r}
# Model 43 AKA Model 3 in the Report
model43 = arima(landing_ts_so, order=c(1,0,1), seasonal = list(order = c(1,1,1), period = 12),
                method = "ML",fixed = c(NA,NA,NA,NA))
model43
AICc(model43)


# Diagnostics
Box.test(residuals(model43),lag = 11, type = ("Box-Pierce"), fitdf = 4)
Box.test(residuals(model43),lag = 11, type = ("Ljung-Box"), fitdf = 4)
Box.test(residuals(model43)^2,lag = 11, type = ("Ljung-Box"), fitdf = 0)
shapiro.test(residuals(model43))

## acf
acf_resid = acf(residuals(model43),main = "Autocorrelation", lag.max = 60)
# Put into Dataframe
````

```
sample_acf = as.data.frame(do.call(cbind, acf_resid))
# Confidence Interval Line
conf.level = 0.95
ciline_resid = qnorm((1 - conf.level)/2)/sqrt(132)

# Plot
ACF_residual_mod3= ggplot(data = sample_acf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline_resid), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline_resid), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "ACF") +
  #labs(title = "") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

## pacf
pacf_resid = pacf(residuals(model43),main = "Autocorrelation", lag.max = 60)
# Put into Dataframe
sample_pacf = as.data.frame(do.call(cbind, pacf_resid))
# Plot
PACF_residual_mod3= ggplot(data = sample_pacf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline_resid), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline_resid), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "PACF") +
  #labs(title = "") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

# Histogram
hist_df_mod = data.frame(x = residuals(model43))
histogram_resid3 = ggplot(data = hist_df, aes(x = x/1000)) +
  geom_histogram(aes(y = ..density..)) +
  geom_density(alpha = 0.1, fill = "red") +
  labs(x = "Residuals (Divided by 1000)", y = "Frequency") +
  #scale_x_continuous(label = comma) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 15),
    legend.text = element_text(size = 15),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

# q-q plot
qq_resid_mod3 = qqplot_residuals(residuals(model43))

# Plot All Together
png(filename = "Images/diagnostics_mod3.png", width = 960, height = 960)
plot_grid(qq_resid_mod3,histogram_resid3,ACF_residual_mod3,PACF_residual_mod3, label_size = 12, ncol = 2, nrow = 2)
dev.off()

```

```{r}
# Model 44 AKA Model 2 in Report
model44 = arima(landing_ts_so, order=c(2,0,3), seasonal = list(order = c(1,1,1), period = 12),
                method = "ML",fixed = c(0,NA,0,0,NA,NA,NA))
model44
AICc(model44)

# Diagnostics
Box.test(residuals(model44),lag = 11, type = ("Box-Pierce"), fitdf = 4) # Only 4 coefficients, since 3 are set to 0
Box.test(residuals(model44),lag = 11, type = ("Ljung-Box"), fitdf = 4)
```

```
Box.test(residuals(model44)^2,lag = 11, type = ("Ljung-Box"), fitdf = 4)
shapiro.test(residuals(model44))


## acf
acf_resid = acf(residuals(model44),main = "Autocorrelation", lag.max = 60)
# Put into Dataframe
sample_acf = as.data.frame(do.call(cbind, acf_resid))
# Confidence Interval Line
conf.level = 0.95
ciline_resid = qnorm((1 - conf.level)/2)/sqrt(132)

# Plot
ACF_residual_mod4= ggplot(data = sample_acf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline_resid), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline_resid), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "ACF") +
  #labs(title = "") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

## pacf
pacf_resid = pacf(residuals(model44),main = "Autocorrelation", lag.max = 60)
# Put into Dataframe
sample_pacf = as.data.frame(do.call(cbind, pacf_resid))
# Plot
PACF_residual_mod4= ggplot(data = sample_pacf, mapping = aes(x = as.numeric(lag), y = as.numeric(acf))) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = as.numeric(lag), yend = 0)) +
  geom_hline(aes(yintercept = ciline_resid), linetype = 2, color = 'darkblue') +
  geom_hline(aes(yintercept = -ciline_resid), linetype = 2, color = 'darkblue') +
  labs(x = "lag", y = "PACF") +
  #labs(title = "") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

# Histogram
hist_df_mod = data.frame(x = residuals(model44))
histogram_resid4 = ggplot(data = hist_df, aes(x = x/1000)) +
  geom_histogram(aes(y = ..density..)) +
  geom_density(alpha = 0.1, fill = "red") +
  labs(x = "Residuals (Divided by 1000)", y = "Frequency") +
  #scale_x_continuous(label = comma) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 15),
    legend.text = element_text(size = 15),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

# q-q plot
qq_resid_mod4 = qqplot_residuals(residuals(model44))

# Plot All Together
model2_grid = plot_grid(qq_resid_mod4,histogram_resid4,ACF_residual_mod4,PACF_residual_mod4, labels = NULL, label_size =
12, ncol = 2, nrow = 2)

png(filename = "Images/diagnostics_mod1_2.png", width = 960, height = 720)
plot_grid(model1_grid,model2_grid, label_size = 12, ncol = 2, nrow = 1, labels = c("Model1","Model2"), label_x = .2)
dev.off()
```

```
##### SEASONAL and TREND MODELS - Not used in report #####

## Model 0

```{r}
model0 = arima(landing_ts, order=c(1,1,0), seasonal = list(order = c(1,1,0), period = 12), method = "ML")
model0
AICc(model0)



# Diagnostics
Box.test(residuals(model0),lag = 11, type = ("Box-Pierce"), fitdf = 1)

plot(residuals(model0))

acf(residuals(model0))
pacf(residuals(model0))

```


Model 1

```{r}
model1 = arima(landing_ts, order=c(2,1,0), seasonal = list(order = c(1,1,0), period = 12), method = "ML")
model1
AICc(model1)



# Diagnostics
Box.test(residuals(model1),lag = 11, type = ("Box-Pierce"), fitdf = 3)

plot(residuals(model1))

acf(residuals(model1))
pacf(residuals(model1))

```


TREND DIFFERENCE ONLY MODELS - Not used in report

```{r}
# THIS WORKS, BUT PROBABLY NOT A GREAT MODEL
model21 = arima(landing_ts_ns, order=c(1,1,0), method = "ML")
model21
AICc(model21)

# Phi (corresponsing to AR)
AR = polyroot(c(1,-0.6605))
roots_AR = c("Root 1")
root_model5 = data.frame(Root = roots_AR, Value = AR)
kable(root_model5, caption = "Phi(B) Roots")


# Diagnostics
Box.test(residuals(model21),lag = 11, type = ("Box-Pierce"), fitdf = 1)

plot(residuals(model21))

acf(residuals(model21), lag.max = 50)
pacf(residuals(model21), lag.max = 50)
```
````

```{r}
# Rerun the Model (defined in 04)
model22 = arima(landing_ts_ns, order=c(15,1,0), method = "ML")
model22
AICc(model22)


# Diagnostics
Box.test(residuals(model22),lag = 11, type = ("Box-Pierce"), fitdf = 15) # Fails, not enough df
Box.test(residuals(model22),lag = 11, type = ("Ljung-Box"), fitdf = 15) # Fails, not enough df
Box.test(residuals(model22)^2,lag = 11, type = ("Ljung-Box"), fitdf = 0) # Fine, no evidence of non-linear dependence

# Residuals
plot(residuals(model22))

## QQPlot
qqnorm(residuals(model22))
qqline(residuals(model22),col ="blue")

# Plot diagnostics of residuals
par(mfrow=c(1,2),oma=c(0,0,2,0))
op <- par(mfrow=c(2,2))
# acf
acf(residuals(model22),main = "Autocorrelation")
# pacf
pacf(residuals(model22),main = "Partial Autocorrelation")
# Histogram
hist(residuals(model22),main = "Histogram")
# q-q plot
qqnorm(residuals(model22))
qqline(residuals(model22),col ="blue")
```

```{r}
# Try a different version
model5c = arima(landing_ts_ns, order=c(7,1,0), method = "ML")
model5c
AICc(model5c)


# Diagnostics
Box.test(residuals(model5c),lag = 11, type = ("Box-Pierce"), fitdf = 7)


Box.test(residuals(model5c)^2,lag = 11, type = ("Ljung-Box"), fitdf = 0)


plot(residuals(model5c))

# Plot diagnostics of residuals
par(mfrow=c(1,2),oma=c(0,0,2,0))
op <- par(mfrow=c(2,2))
# acf
acf(residuals(model5c),main = "Autocorrelation")
# pacf
pacf(residuals(model5c),main = "Partial Autocorrelation")
# Histogram
hist(residuals(model5c),main = "Histogram")
# q-q plot
qqnorm(residuals(model5c))
qqline(residuals(model5c),col ="blue")
```

```
---
title: "06_Forecasting"
author: "Callum Weinberg"
date: "December 2, 2021"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

Libraries

```{r}
library(dplyr, warn.conflicts = FALSE) #Using
library(tidyr) #Using
library(knitr) #Using
library(lubridate, warn.conflicts = FALSE) #Using
library(ggplot2) #Using
library(MASS) #Uncertain
library(qpcR) #Using
library(forecast) #Using
library(cowplot) #Using
library(TSA) #Using
```


SEASON ONLY MODEL

```{r}
load(file="Data/landings_transformed_season_only.Rdata")
landing_ts_so = landings_transformed_season_only$pounds_transformed
```

Used in the Report
```{r forecast}
# Model 43 - Model Chosen for Forecasting Final Report. Corresponds to Model 3 of the report.
model43 = arima(landing_ts_so, order=c(1,0,1), seasonal = list(order = c(1,1,1), period = 12),
                method = "ML",fixed = c(NA,NA,NA,NA))
model43
AICc(model43)

# Perform Prediction
mypred = predict(model43, n.ahead=12)
#mypred = forecast(model40,h=12, level=c(95))

# Code from Lecture Notes, doing with GGPLOT instead
#ts.plot(landings, xlim=c(0,144), ylim=c(-300000,900000))
#points(133:144,mypred$pred)
#lines(133:144,mypred$pred+1.96*mypred$se,lty=2)
#lines(133:144,mypred$pred-1.96*mypred$se,lty=2)


# Update the Full Dataset with the data for the 12 predicted rows
#landings_forecast = landings
landings_pred_2019 = data.frame(Year = rep(2019,12),
                                Month = seq(1,12,by=1),
                                pounds = mypred$pred,
                                upper = mypred$pred+1.96*mypred$se,
                                lower = mypred$pred-1.96*mypred$se,
                                pounds.bc = rep(NA,12))

landings_pred_2019$date =  as.Date(with(landings_pred_2019,
                                paste0(as.character(landings_pred_2019$Year),"-",
                                as.character(landings_pred_2019$Month),"-01"), "%Y-%m-%d"))


## Plot the Original Data with the Forecast
forecast_plot = ggplot() +
  geom_line(data = landings_full, mapping = aes(x = date, y = pounds/1000)) +
  geom_line(data = landings_pred_2019, mapping = aes(x = date, y = pounds/1000), color = "red", linetype = "dashed") +
  geom_line(data = landings_pred_2019, mapping = aes(x = date, y = upper/1000), color = "blue", linetype = "twodash",
size = .8) +
  geom_line(data = landings_pred_2019, mapping = aes(x = date, y = lower/1000), color = "blue", linetype = "twodash",
size = .8) +
  labs(x = "Date", y = "Thousands of Pounds") +
  scale_x_date(breaks = scales::breaks_pretty(15)) +
  scale_y_continuous(limits = c(-500,900)) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
```
```

````
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
forecast_plot

# Plot Forecast
png(filename = "Images/forecast.png", width = 960, height = 480)
forecast_plot
dev.off()
```



# Not Used in the Report #
```{r forecast}
# Model 40 - Not Used in Final Report
model40 = Arima(landing_ts_so, order=c(2,0,2), seasonal = list(order = c(1,1,1), period = 12),
                method = "ML",fixed = c(NA,NA,0,NA,NA,NA))
model40
AICc(model40)

# Perform Prediction
mypred = predict(model40, n.ahead=12)
#mypred = forecast(model40,h=12, level=c(95))

# Code from Lecture Notes, doing with GGPLOT instead
#ts.plot(landings, xlim=c(0,144), ylim=c(-300000,900000))
#points(133:144,mypred$pred)
#lines(133:144,mypred$pred+1.96*mypred$se,lty=2)
#lines(133:144,mypred$pred-1.96*mypred$se,lty=2)


# Update the Full Dataset with the data for the 12 predicted rows
#landings_forecast = landings
landings_pred_2019 = data.frame(Year = rep(2019,12),
                                Month = seq(1,12,by=1),
                                pounds = mypred$pred,
                                pounds.bc = rep(NA,12))

landings_pred_2019$date =  as.Date(with(landings_pred_2019,
                                paste0(as.character(landings_pred_2019$Year),"-",
                                as.character(landings_pred_2019$Month),"-01"), "%Y-%m-%d"))

landings_forecast = rbind(landings,landings_pred_2019)



## Plot the Original Data with the Forecast
forecast_plot = ggplot(data = landings_forecast, mapping = aes(x = date, y = pounds/1000)) +
  geom_line() +
  labs(x = "Date", y = "Thousands of Pounds", title = "Forecast") +
  scale_x_date(breaks = scales::breaks_pretty(10)) +
  scale_y_continuous(limits = c(-200,900)) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
forecast_plot


## Plot the Actual Data
load(file="Data/landings_full.Rdata")
full_plot = ggplot(data = landings_full, mapping = aes(x = date, y = pounds/1000)) +
  geom_line() +
  labs(x = "Date", y = "Thousands of Pounds", title = "Actual Data") +
  scale_x_date(breaks = scales::breaks_pretty(10)) +
  scale_y_continuous(limits = c(-200,900)) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
````

```
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
full_plot



```

Not Used in the Report

```{r forecast}
# Model 44
model44 = arima(landing_ts_so, order=c(2,0,3), seasonal = list(order = c(1,1,1), period = 12),
                method = "ML",fixed = c(0,NA,0,0,NA,NA,NA))
model44
AICc(model44)

# Perform Prediction
mypred = predict(model44, n.ahead=12)
#mypred = forecast(model40,h=12, level=c(95))

# Code from Lecture Notes, doing with GGPLOT instead
#ts.plot(landings, xlim=c(0,144), ylim=c(-300000,900000))
#points(133:144,mypred$pred)
#lines(133:144,mypred$pred+1.96*mypred$se,lty=2)
#lines(133:144,mypred$pred-1.96*mypred$se,lty=2)


# Update the Full Dataset with the data for the 12 predicted rows
#landings_forecast = landings
landings_pred_2019 = data.frame(Year = rep(2019,12),
                                Month = seq(1,12,by=1),
                                pounds = mypred$pred,
                                pounds.bc = rep(NA,12))

landings_pred_2019$date =  as.Date(with(landings_pred_2019,
                                paste0(as.character(landings_pred_2019$Year),"-",
                                as.character(landings_pred_2019$Month),"-01"), "%Y-%m-%d"))

landings_forecast = rbind(landings,landings_pred_2019)



## Plot the Original Data with the Forecast
forecast_plot = ggplot(data = landings_forecast, mapping = aes(x = date, y = pounds/1000)) +
  geom_line() +
  labs(x = "Date", y = "Thousands of Pounds", title = "Forecast") +
  scale_x_date(breaks = scales::breaks_pretty(10)) +
  scale_y_continuous(limits = c(-200,900)) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
forecast_plot


## Plot the Actual Data
load(file="Data/landings_full.Rdata")
full_plot = ggplot(data = landings_full, mapping = aes(x = date, y = pounds/1000)) +
  geom_line() +
  labs(x = "Date", y = "Thousands of Pounds", title = "Actual Data") +
  scale_x_date(breaks = scales::breaks_pretty(10)) +
  scale_y_continuous(limits = c(-200,900)) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
full_plot
```

```
##### TREND ONLY MODEL - Not used in the Report #####

## Load the Trend-Only Differenced Data

```{r}
load(file="Data/landings_transformed_no_season.Rdata")
landing_ts_ns = landings_transformed_no_season$pounds_transformed
```


Forecast Model 22

```{r}
# Rerun the Model (defined in 04)
model22 = arima(landing_ts_ns, order=c(15,1,0), method = "ML")
model22
AICc(model22)

# Perform Prediction
mypred = predict(model22, n.ahead=12)

# Code from Lecture Notes, doing with GGPLOT instead
#ts.plot(landings, xlim=c(0,144), ylim=c(-300000,900000))
#points(133:144,mypred$pred)
#lines(133:144,mypred$pred+1.96*mypred$se,lty=2)
#lines(133:144,mypred$pred-1.96*mypred$se,lty=2)


# Update the Full Dataset with the data for the 12 predicted rows
#landings_forecast = landings
landings_pred_2019 = data.frame(Year = rep(2019,12),
                                Month = seq(1,12,by=1),
                                pounds = mypred$pred,
                                pounds.bc = rep(NA,12))

landings_pred_2019$date =  as.Date(with(landings_pred_2019,
                                paste0(as.character(landings_pred_2019$Year),"-",
                                as.character(landings_pred_2019$Month),"-01"), "%Y-%m-%d"))

landings_forecast = rbind(landings,landings_pred_2019)



## Plot the Original Data with the Forecast
forecast_plot = ggplot(data = landings_forecast, mapping = aes(x = date, y = pounds/1000)) +
  geom_line() +
  labs(x = "Date", y = "Thousands of Pounds", title = "Forecast") +
  scale_x_date(breaks = scales::breaks_pretty(10)) +
  scale_y_continuous(limits = c(-100,900)) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
forecast_plot
```
```

````
## Plot the Actual Data
load(file="Data/landings_full.Rdata")
full_plot = ggplot(data = landings_full, mapping = aes(x = date, y = pounds/1000)) +
  geom_line() +
  labs(x = "Date", y = "Thousands of Pounds", title = "Actual Data") +
  scale_x_date(breaks = scales::breaks_pretty(10)) +
  scale_y_continuous(limits = c(-100,900)) +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 10),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))
full_plot

```

```
````

````
---
title: "07_Spectral_Analysis"
author: "Callum Weinberg"
date: "December 3, 2021"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

## Libraries

```{r}
library(dplyr, warn.conflicts = FALSE) #Using
library(tidyr) #Using
library(knitr) #Using
library(lubridate, warn.conflicts = FALSE) #Using
library(ggplot2) #Using
library(MASS) #Uncertain
library(qpcR) #Using
library(forecast) #Using
library(cowplot) #Using
library(TSA) #Using
```

##### Load Data #####

```{r}
load(file="Data/landings.Rdata")

load(file="Data/landings_transformed_season_only.Rdata")
landing_ts_so = landings_transformed_season_only$pounds_transformed
```

###### Periodogram of Original Data #####
```{r}
## Periodogram
#periodogram = periodogram(landings$pounds, plot = TRUE)
periodogram = periodogram(landings$pounds, plot = FALSE)

# Put into Dataframe
periodogram_df = data.frame(periodogram = periodogram$spec[1:60], frequency = periodogram$freq[1:60])

# Plot
Periodogram_Graph = ggplot(data = periodogram_df, mapping = aes(x = frequency, y = periodogram)) +
  geom_segment(mapping = aes(xend = as.numeric(frequency), yend = 0)) +
  labs(x = "frequency", y = "periodogram") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

Periodogram_Graph

# Get the Period at the spikes
print(1/periodogram_df[11,2])
print(1/periodogram_df[2,2])

# Fisher Test for Periodicity
library(GeneCycle) #Using
fisher.g.test(landings$pounds)

# Kolmogorov Smirnov Test
cpgram(landings$pounds,main="")



## Plots for report
png(filename = "Images/periodogram.png", width = 960, height = 480)
Periodogram_Graph
dev.off()
```

###### Periodogram of Residuals of Model 3 #####
```{r}
````
````

`````
# Model 43 - Model Chosen for Forecasting Final Report. Corresponds to Model 3 of the report.
model43 = arima(landing_ts_so, order=c(1,0,1), seasonal = list(order = c(1,1,1), period = 12),
                method = "ML",fixed = c(NA,NA,NA,NA))

## Periodogram
#periodogram = periodogram(landings$pounds, plot = TRUE)
periodogram = periodogram(residuals(model43), plot = FALSE)

# Put into Dataframe
periodogram_df = data.frame(periodogram = periodogram$spec[1:60], frequency = periodogram$freq[1:60])

# Plot
Periodogram_Residuals = ggplot(data = periodogram_df, mapping = aes(x = frequency, y = periodogram)) +
  geom_segment(mapping = aes(xend = as.numeric(frequency), yend = 0)) +
  labs(x = "frequency", y = "periodogram") +
  theme(text = element_text(size = 20),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.key.width=unit(1,"cm"),
    axis.text.y = element_text(angle=90, hjust=1, size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(hjust = 0.5, size = 12),
    axis.title=element_text(size=10,face="bold"))

Periodogram_Residuals


## Plots for report
png(filename = "Images/periodogram_residuals.png", width = 960, height = 480)
Periodogram_Residuals
dev.off()
```



###### Tests for Periodicity #####

```{r}
# Fisher Test for Periodicity
library(GeneCycle) #Using
fisher.g.test(residuals(model43))

# Kolmogorov Smirnov Test
png(filename = "Images/KS_test Residuals.png", width = 480, height = 480)
cpgram(residuals(model43),main="")
dev.off()
````
`````
`````