The Changing Import/Export Dynamic Between the U.S. and China

Hannah Goldstein, Nicolas Renaud, Brendan Hennessey, Jarrett Rennie

Abstract -

Imports and Exports between the world's two largest economies and global superpowers are a critical indicator of not only trade relations between the two nations, but also hold implications for global economic well-being. Our investigation of two time series, one of imports to the U.S. from China, and other exports. IT covers a range of important political and economic events from 1985 to the present day, several of which are visible as changes in the graph's appearance like the 2009 recession and China's entrance into the WTO. We discovered annual seasonality in both series, pointing to a late fall maximum. Statistical analysis pointed to non-linear structure, and we were able to develop a neural network model for imports as well as a bagged model for exports that satisfy our accuracy checks. Still, questions remain about the predictive power of this model given the very new nature of tariffs enacted by both countries' since 2018. If anything, these forecast models can be used as a reference for what could have been, shoudl the bilateral relationship deteriorate in the face of tariffs.

Index Terms - economics, trade relations, imports/exports, ETS, ARIMA, neural network

1 Introduction/Motivation

There has been a flurry of regulatory activity between the United States and China over the past 20-30 years. As tensions have grown worse recently, the economies of the two countries are more likely than ever to be negatively impacted. Through the various tariffs, threats of tariffs, and other political strong-arming, the most likely place where the impact would manifest is in the imports/exports between the U.S. and China. Ultimately, we hypothesize that the trade dynamic between the U.S. and China has shifted significantly in recent years.

In the sections below, we'll explore the tenability of this hypothesis. We'll first briefly touch on the source of the dataset, its frequency, and any peculiarities. Then, with the 35 years of import/export data collected, we'll conduct an exploratory data analysis (EDA) to better

Hannah Goldstein is an MSBA major, Bentley University,
Nicolas Renaud is an MSBA major, Bentley University,
Brendan Hennessey is an MSBA major, Bentley University,
Jarrett Rennie is an MSBA major, Bentley University

understand each time series. In this EDA, we'll examine the trend and seasonality of imports and exports over the entire time period, as well as discern any other patterns that can be deduced from the graphs. We'll also compare the data to known historical events in this initial analysis.

After conducting an EDA, we'll dive deeper into each time series. We'll use various decomposition methods to break down imports and exports on a more granular level. Next we'll perform various tests on each series, checking for things such as linearity and stationarity. The results of the decomposition and tests will inform the modeling, and any transformations necessary prior to fitting.

With EDA and the deep dive complete, we'll attempt to fit various linear and nonlinear models to each time series. Here, we'll leverage modeling techniques such as ETS (where the smoothing parameter could inform us about the anomaly of the most recent time periods), ARIMA, neural networks, and bagging. With the various types of models, and their respective results, we'll be able to discern whether or not it's possible to fit an accurate model for simulating (train) and forecasting (test) imports and exports.

2 The Data Set

Both the "imports" [1] and "exports" [2] data sets used to conduct this analysis originated from the U.S. Bureau of Economic Analysis and U.S. Census Bureau. The data sets were retrieved and downloaded from the Federal Reserve Bank of St. Louis (FRED), a group that conducts various types of global economic research.

Both data sets are time series, capturing monthly imports/exports in millions of U.S. dollars. They each span from January 1985 to September 2019. There are no missing observations or major outliers in either time series.

Given each data set is captured at the macro level, representing the entire U.S., the information is not inherently sensitive. This data is publicly available and published/cited in numerous places. If we were instead looking at a given company's imports/exports, then one could conclude that the data contains sensitive and/or confidential information.

Additionally, because the frequency of each time series is monthly, and over thirty years of data is captured, it is safe to conclude that the volume is sufficient for the purpose of this analysis.

3 Exploratory Data Analysis

Let's first explore ordinary time plots of the imports and exports time series themselves. These two series, along with a trend line for each, may be examined in Figure 1 below.

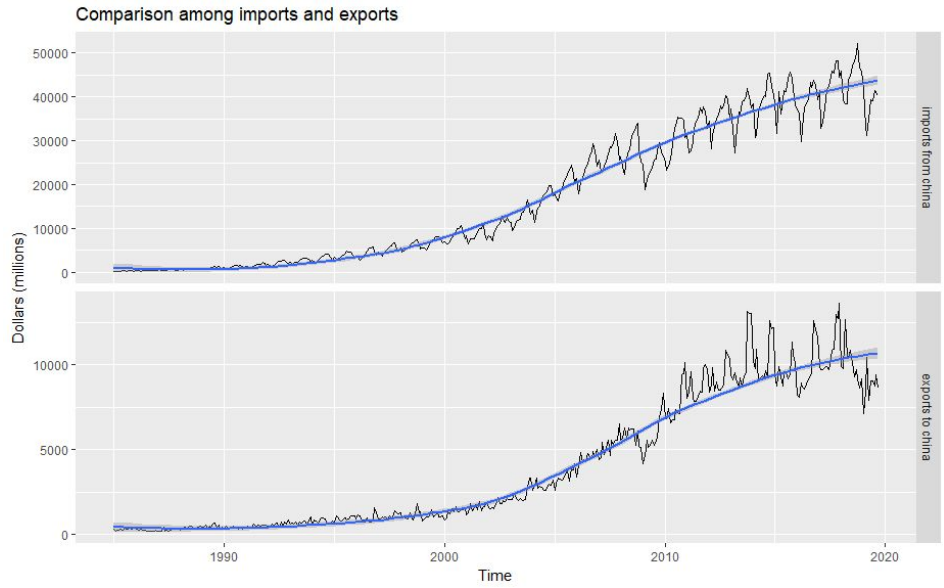


Figure 1 - Plotting the Imports and Exports Time Series

Each series appears to have strong trend and some (nonzero) seasonality. From the year 2001 onward, each series get's especially interesting where it appears that U.S. business with China grew exponentially. To better understand this uptick, the Council on Foreign Relations' timeline of U.S. Relations with China [3] was referenced. From the timeline, it appears that in 2001, China officially joined the World Trade Organization, which is a perfect explanation for the massive growth.

It was important to confirm that the catalyst could be attributed to something specific, rather than being unexplainable. If unexplainable, a concern would be that perhaps the method for measuring imports/exports changed around this timeframe, which would make it difficult to properly compare values before/after 2001.

Along with the 2001 event, several other significant trade relations events (2009 recession, 2018-2019 increased tariff activity) were pulled out of the Council on Foreign Relations' timeline. These events are in fact visible in the time plots, and thus are called out in Figure 2 below to add some additional color to the variances.

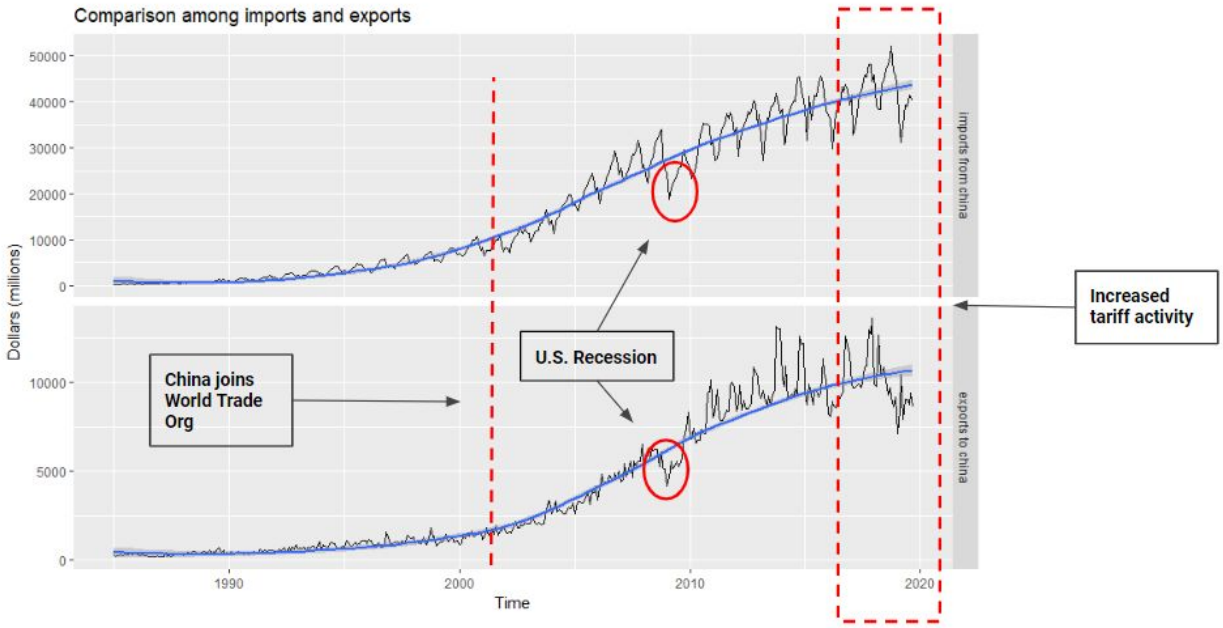


Figure 2 - Imports and Exports with Significant Trade Relations Events

Next, we'll dive deeper into the trend and seasonality of each time series. In order to quantify the relative strength of the trend and seasonality, we calculate the F(T) and F(S) coefficients. The coefficients were determined using a classical additive decomposition. The values may be examined in Table 1 below.

	Imports	**Exports**
$\mathbf{F_T}$	0.9884271	0.9816865
$\mathbf{F_S}$	0.5088625	0.2667869

Table 1 - Trend Strength and Seasonality Strength of Imports and Exports

Both series have trend strength very close to one, indicating a strong trend. The seasonality is less strong in both, but higher for the imports series. This means that both series have seasonality, but it's more recognizable in the imports series. To further study the trends and seasonalities, Polarmaps (Figure 3 and 4) and Lag Plots (Figure 5 and 6) were produced below. Through these other graphical means, we should be able to perceive any other trend/seasonality intricacies.

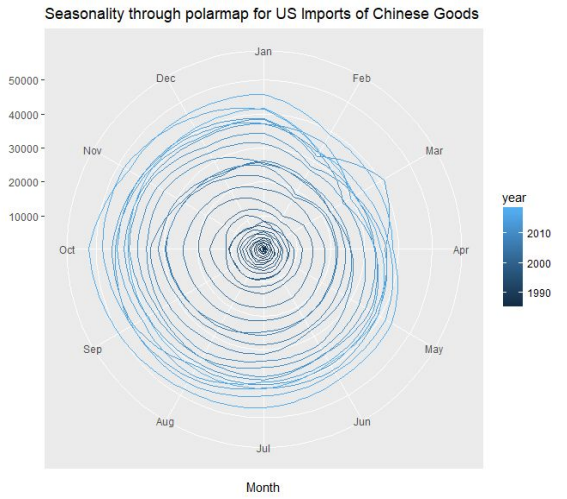


Figure 3 - Polarmap of U.S. Imports of Chinese Goods

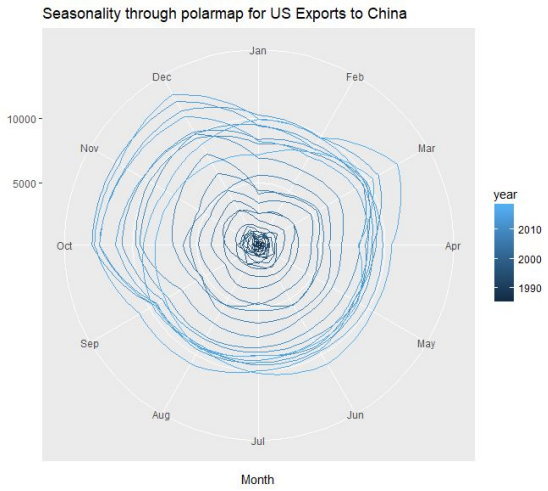


Figure 4 - Polarmap of U.S. Exports to China

If we examine both the imports' and exports' Polarmaps (Figures 3 and 4), we can see a clear trend from the fact that the circles (generally) grow larger as time goes on. In the imports' Polarmap, the peak tends to happen around October, where the dip happens somewhere in the February-March months. Generally Q3 and Q4 have larger values, where Q1 and Q2 are lower. This relationship by nature is seasonal, but not perfectly clear cut, which supports the lower seasonality strength. In the exports' Polarmap, peaks happen in October-December, where dips occur in January and February. Generally here, Q4 has the largest values, Q2 and Q3 have moderate values, and Q1 has the lowest values. This explains why the seasonality strength is a much smaller value for this series, as it's much less straightforward. Let's see if the Lag plots of each series can tell us more.

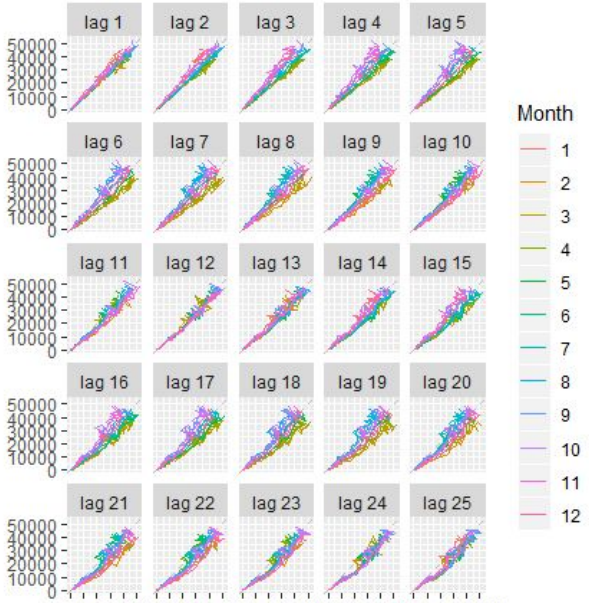


Figure 5 - Lag Plots for Imports.

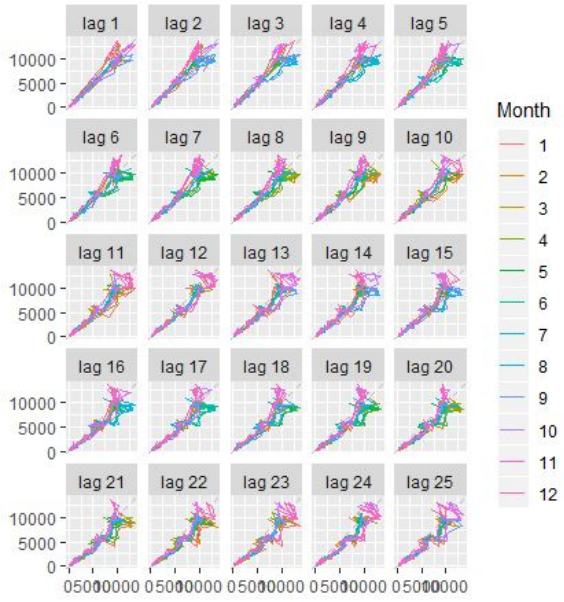


Figure 6 - Lag plots for Exports

The lag plots for imports and exports (Figure 5 and 6) both display a clustering on the diagonal at Lag 12. This is consistent with the time plot investigation (Figure 1) of the data which indicated a general yearly cyclicality. Other than Lag 12, no other lags offer insights that are different than those gleaned from the other graphical methods employed above.

With a solid grasp on each series' trend and seasonality, let's look at their entropies.

```
SampEn(imp.ts)

## [1] 0.1134533

SampEn(exp.ts)

## [1] 0.1184296
```

Figure 7 - Entropy of Each Time Series

It's first clear that imports and exports have similar entropy. Given the low entropy values, we know that there's limited randomness to the data, and that each data point has some relationship with previous data points.

Finally, we'll apply decomposition methods to set the stage for modeling. Given the 2001 spikes in both series we saw in Figures 1 and 2, we know that multiplicative decomposition must be used. We cannot use additive because the seasonality magnitude/trend variations change with the average level of the series. In Figure 8 (imports) and Figure 9 (exports) below, these decompositions are shown.

```
autoplot(decomp.imp.mult)
```

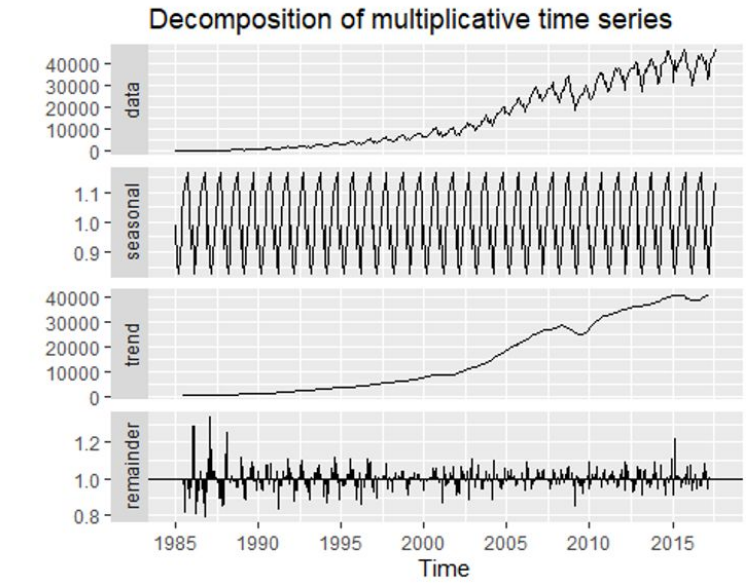


Figure 8 - Multiplicative Decomposition of Imports

```
autoplot(decomp.exp.mult)
```

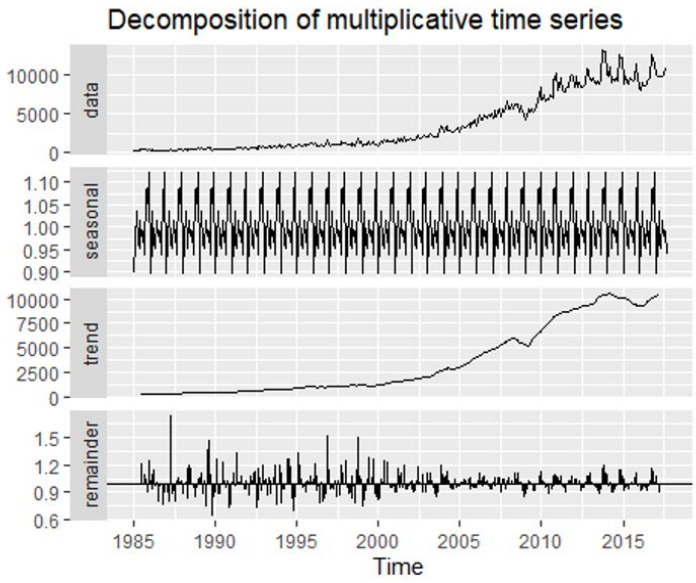


Figure 9 - Multiplicative Decomposition of Exports

In both decompositions, we can see that the random components have slightly larger variances prior to 2001, which is expected due to the event called out above. An interesting take-away from each decomposition is that although some seasonality is present, it is not much larger than the random component. This means that it's really the trend driving the most significant change over the time period examined. We'll see how this affects modeling in section 5 below.

4 Theory and Methods

Our analysis consisted of initial data formatting using Microsoft Excel. Following this, we utilized R software analysis and associated packages to conduct all further testing and visualization.

The first step was to conduct the previously displayed descriptive exploration of the two time series, using graphical methods such as polar plots and lag plots. Through this the core features of the set become clearer, which then informs the modelling process. Critically, it allows us to have a logical basis for determining if our forecast model makes sense in reality given the behavior of the underlying data. In our case, we know we have a non-stationary increasing trend, with annual seasonality and likely non-linear nature. Even still, we wanted to test linear methods first to see if they may actually perform well for this case study. As such, we plugged

the model into the corresponding functions to produce the best ETS and best SARIMA models possible. Following this, we also conducted non-linearity testing and model-building using neural networks and bagging. Lastly, we reviewed all the available accuracy metrics across all models to choose the best ones and draw conclusions.

5 Modeling

ETS

We began by trying to fit the best ETS model to our data. For both imports and exports, an ETS (M,Ad,M) was presented as the best ETS model. This is using multiplicative error, additive trend and multiplicative seasonality. Interpreting this model did not move very far; the Ljung-Box test showed evidence that the residuals for both datasets were dependent. This means that the models are not recommended to move forward with.

data	method	model	lambda	sigma**2	subset	mape	mase	residuals
import	ets	M,Ad,M	N/A	0.0763	train	5.494712	0.3703234	dependent
					test	7.537235	1.763118	
export	ets	M,Ad,M	N/A	0.1551	train	10.37534	0.5103051	dependent
					test	15.89129	2.8818206	

SARIMA

SARIMA was the next model that was run with our data. Knowing that the data was seasonal led us to using the SARIMA instead of ARIMA. The first SARIMA model for import data has two auto regressive terms, one differencing term, one moving average term, two seasonal auto regressive terms, one seasonal differencing term, no seasonal moving average terms, a seasonality of twelve and does not include a transformation.The second model for imports includes the same terms but includes a transformation. In the figure below, the models with better training and testing data are highlighted but the residuals were found to be dependent, with very small Ljung-Box p-values.

data	method	model	lambda	sigma**2	significant parameters	subset	mape	mase	residuals
import	sarima	ARIMA(2,1,1)(2,1,0)[12]	N/A	1180427	YES	train	5.356059	0.3627931	dependent
						test	6.900162	1.6583602	
import	sarima	ARIMA(2,1,1)(2,1,0)[12]	0.1637399	0.07768	YES	train	4.977908	0.3433272	dependent
						test	7.287937	1.7651253	
export	sarima	ARIMA(2,1,1)(0,1,1)[12]	N/A	119334	YES	train	11.35756	0.4513094	independent
						test	19.70742	3.5138966	
export	sarima	ARIMA(2,1,1)(2,0,0)[12] with drift	0.220129	0.4778	YES	train	10.81068	0.4775985	independent
						test	22.87962	4.1109044	

The first SARIMA model for export data has two auto regressive terms, one differencing term, one moving average term, two seasonal auto regressive terms, no seasonal differencing terms, no seasonal moving average terms, a seasonality of twelve and does not include a transformation.The second model for imports includes the same terms but includes a

transformation and drift. Above the highlighted training and testing accuracy figures are highlighted to show that overall the model without the transformation performs better. The export models' residuals also proved to be independent. As discussed before though, we predict that our model will be nonlinear and therefore will perform nonlinear tests.

Nonlinear Modeling

The nonlinear tests make it clear that this data should not be modeled by a linear model and that some nonlinear model should instead be used for both the import and export time series. We looked at neural network models with and without a Box Cox transformation, as well as, bagged models with and without a Box Cox transformation for both the import and export data.

data	method	model	lambda	sigma**2	subset	mape	mase
import	neural	NNAR(4,1,3)[12]	N/A	1618448	train	7.23142	0.4292306
					test	8.17489	2.02708
import	neural	NNAR(1,1,2)[12]	0.1637399	0.1308	train	6.597035	0.5082465
					test	8.48026	2.0782421
export	neural	NNAR(1,1,2)[12]	N/A	195589	train	12.53733	0.5520151
					test	19.45795	3.5122214
export	neural	NNAR(2,1,2)[12]	0.220129	0.5041	train	11.04153	0.5366247
					test	16.06305	2.9532921

In the above figure are the results of the neural network models for import and export data. The first neural network model for import data does not include a Box Cox transformation, recalls 4 ordinary lags, 1 seasonal lag, uses 3 hidden nodes, and has seasonality of order 12. The second neural network model for import data does include a Box Cox transformation, recalls 1 ordinary lag, 1 seasonal lag, uses 2 hidden nodes, and has seasonality of order 12. Because these two models use different data, one has been transformed using Box Cox, the best way to compare the two is using mean absolute scaled error (mase) in order to avoid the the problem mean absolute percentage error (mape) has with favoring underestimation as well as the ambiguity of percentages. Between these 2 models the non transformed model performs better in terms of mase.

The first neural network model for export data does not include a Box Cox transformation, recalls 1 ordinary lag, 1 seasonal lag, uses 2 hidden nodes, and has seasonality of order 12. The second neural network model for export data does include a Box Cox transformation, recalls 2 ordinary lags, 1 seasonal lags, uses 2 hidden nodes, and has seasonality of order 12. Again the best way to compare these 2 models is using mase; however, in the case of exports the transformed model performs better.

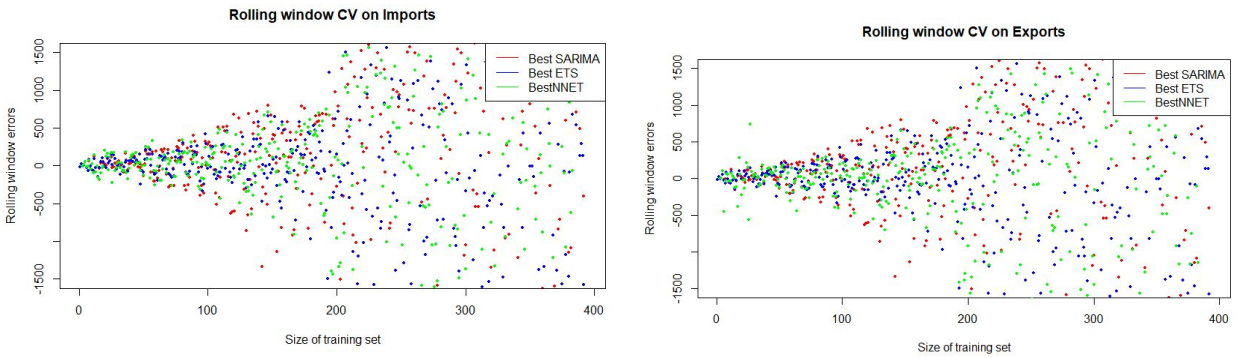
data	method	lambda	subset	mape	mase
import	bagged	N/A	train	5.215448	0.3438924
			test	8.673755	2.0504201
import	bagged	0.1637399	train	60220.0962	22417.894
			test	147209.9	62004.77
export	bagged	N/A	train	9.4366	0.4757732
			test	15.04068	2.7782196
export	bagged	0.220129	train	13916.27189	3543.666
			test	35757.59	10691.656

In the above figure are the results of the bagged models on imports and exports with and without a Box Cox transformation applied. Out of the transformed and non transformed bagged models for the import data the non transformed model performs better. This pattern continues for the export data where again the non transformed model outperforms the transformed model.

Overall, the best performing nonlinear model for import data we found to be the non transformed neural network model. The best performing nonlinear model for export data we found to be the non transformed bagged model. Both of the these models may not perform as well as some of the SARIMA and ETS models however it would not make sense to model a nonlinear time series using one of those linear methods.

Rolling Window Cross-validation

Another method to check the performance of a time series model is rolling-window cross-validation. This is a method that checks the predictive performance of the model across a sequence of training sets of different sizes. It checks for the error on the prediction for the very next point, relative to the real data. We conducted this analysis for the ETS, SARIMA, and Neural models for each of the two time series and compared graphically. The bagged model was not completing it's run in R; we would have liked to include it here as well - though the neural network does provide sufficient non-linear representation for imports especially.



The resulting figures can be interpreted by looking for which series cluster well around 0. In our instance, there isn't a major discernible winner in the sets across any size of training set - though there is a noticeable widening of the error margins with a bigger training set. Our

hypothesis for this is that the highly non-stationary nature of our entire dataset makes “closer” training sets(i.e.) small have such close values to the “tomorrow” value that they naturally perform better, even with less history to draw from.

Rolling window is not one of our more important accuracy checks, however. This is because we do not envision this sort of time series data being important for monthly economic forecasting - this length of data would be more useful for forecasting on the order of years, and as such we lean more heavily on our sigma, MAPE, and MASE metrics to pick among competing non-linear models.

Final Model Selection and Comparison

We conducted a final comparison of all of the top-contender models we produced by producing a tabular comparison of the ETS, ARIMA, and non-linear models for each of imports and exports. The ETS models are quite easily disqualified due to the dependence of residuals and the failure of the nonlinearity tests. The SARIMA suffers the same linearity concern, and dependent residuals for the import series. The export series has an otherwise quite solid SARIMA (2,1,1)(0,1,1)[12] model which was the ARIMA accuracy winner for MAPE and MASE in all areas except the training MAPE.

For the overall imports winner, the best neural network model had better MAPE and MASE values than the bagged version in every category except training MAPE. We were also satisfied with the number of coefficients, though the p=4 value is a little higher than desired. The bagged model for exports had the lowest MAPE and MASE in all respects among the models we made on the series. Interestingly, the lambda BoxCox transformation vrsions of the various models did not improve the results. The winning models are marked with green coloring in the table below. The forecast overlay graphs (Figure x) shows that the models stay quite centered in the range, especially on the export model where the orange line noting the bag model noticeably lacks many of the extreme points that plague the ETS and ARIMA model (Figure x). The best export SARIMA has a more narrow PI (judged from the sigma^2) value than the neural networks, but because the bagged model doesn’t provide prediction intervals of the same kind it can’t be compared in this manner.

data	method	model	lambda	sigma**2	subset	mape	mase	residuals
import	ets	M,Ad,M	N/A	0.0763	train	5.494712	0.3703234	dependent
					test	7.537235	1.763118	
export	ets	M,Ad,M	N/A	0.1551	train	10.37534	0.5103051	dependent
					test	15.89129	2.8818206	

data	method	model	lambda	sigma**2	significant parameters	subset	mape	mase	residuals
import	sarima	ARIMA(2,1,1)(2,1,0)[12]	N/A	1180427	YES	train	5.356059	0.3627931	dependent
						test	6.900162	1.6583602	
import	sarima	ARIMA(2,1,1)(2,1,0)[12]	0.1637399	0.07768	YES	train	4.977908	0.3433272	dependent
						test	7.287937	1.7651253	
export	sarima	ARIMA(2,1,1)(0,1,1)[12]	N/A	119334	YES	train	11.35756	0.4513094	independent
						test	19.70742	3.5138966	
export	sarima	ARIMA(2,1,1)(2,0,0)[12] with drift	0.220129	0.4778	YES	train	10.81068	0.4775985	independent
						test	22.87962	4.1109044	

	data	method	model	lambda	sigma**2	subset	mape	mase
imports best nonlinear model->	import	neural	NNAR(4,1,3)[12]	N/A	1618448	train	7.23142	0.4292306
						test	8.17489	2.02708
	import	neural	NNAR(1,1,2)[12]	0.1637399	0.1308	train	6.597035	0.5082465
						test	8.48026	2.0782421
	export	neural	NNAR(1,1,2)[12]	N/A	195589	train	12.53733	0.5520151
						test	19.45795	3.5122214
	export	neural	NNAR(2,1,2)[12]	0.220129	0.5041	train	11.04153	0.5366247
						test	16.06305	2.9532921

	data	method	lambda	subset	mape	mase
	import	bagged	N/A	train	5.215448	0.3438924
				test	8.673755	2.0504201
	import	bagged	0.1637399	train	60220.0962	22417.894
				test	147209.9	62004.77
exports best nonlinear model->	export	bagged	N/A	train	9.4366	0.4757732
				test	15.04068	2.7782196
	export	bagged	0.220129	train	13916.27189	3543.666
				test	35757.59	10691.656

Table:

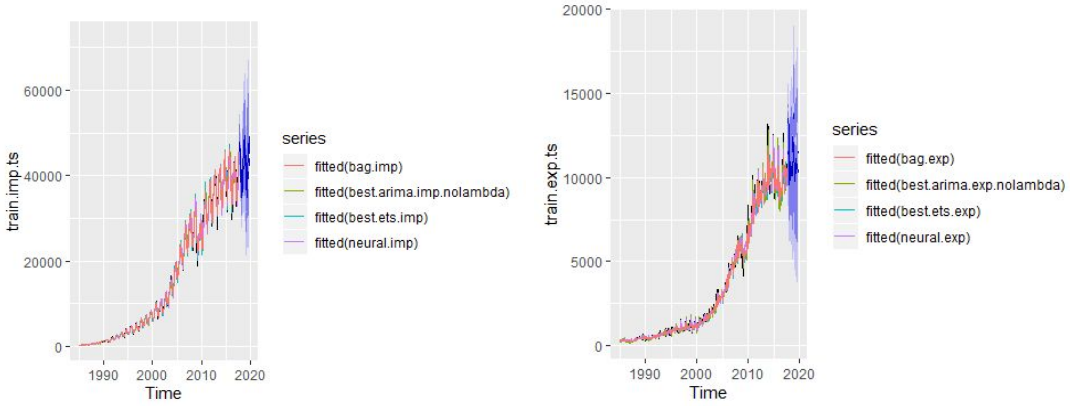


Figure: Category-best model Imports and Exports forecast Overlays

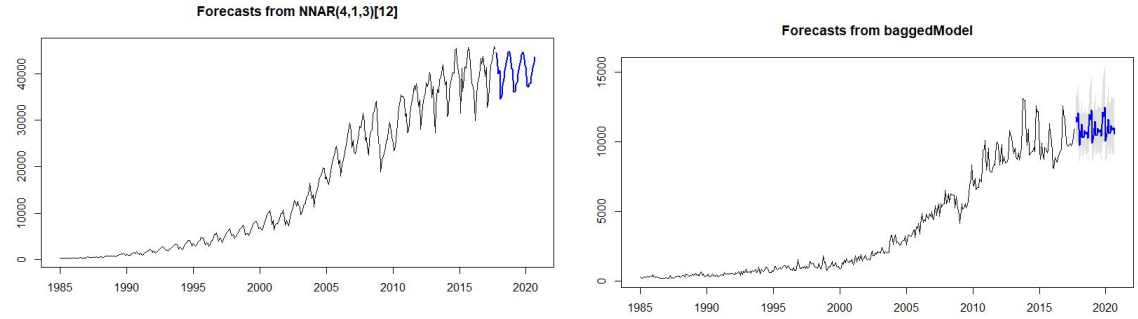


Figure: Forecasts of the model winners

When graphing the individual forecast results of the import NNET and the bagged exports, we were pleased to encounter forecasts that were very visually pleasing and consistent with our expectation. They maintained the slight upward trend (though less steep than the early 2000's), and seasonal effect was clearly evident.

6 Conclusions and Future Work

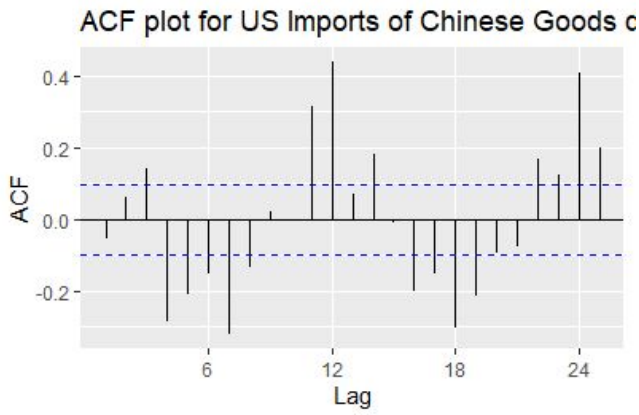
Initially we set out to answer several key questions, and we have succeeded in at least addressing them all (or determining a follow-up question to ask). Firstly, we were curious whether this data would be conducive to time series forecasting at all. It ended up being low entropy data, which satisfied this concern. Another key question was simply whether the data would align with certain common-knowledge current events between the U.S. and China that we knew about - critically the early 2000's boom, the Great Recession, and the most recent round of tariffs. All three of these events are apparent visually in the main graph of the time series.

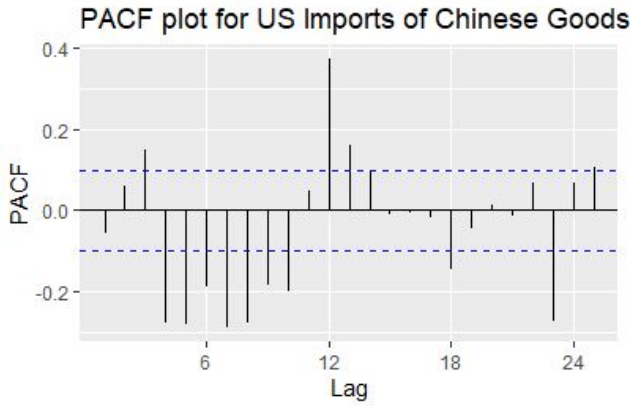
Another takeaway from this investigation, following our model-making process, was that this data was non-linear. It would be interesting to replicate this investigation with other bilateral trade relationships - perhaps the U.S. and another high volume trade partner like Canada, to see if that was still the case. Yet another future investigation we would like to do may have to wait a bit. The recent tariffs fell within the testing window for our model-making, and as such we suspect that their effect and the overall testing accuracy of the models suffers as a result. We would be interested to replicate the study with perhaps a shorter time frame, and with more tariff-active months in the series overall. A logical extension of looking at imports and exports separately would be to difference them and investigate trade balance. The application of this series may be more political than economic, but has been the focus of much of the recent tariff discussions in the United States nonetheless.

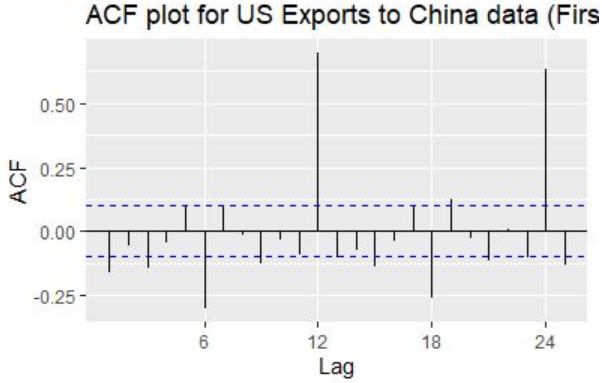
Our last future study of interest would be to add regressors into the data, to determine if there are explanatory factors related to GDP, population growth, or other main economic drivers that could enhance the predictive capability of the models.

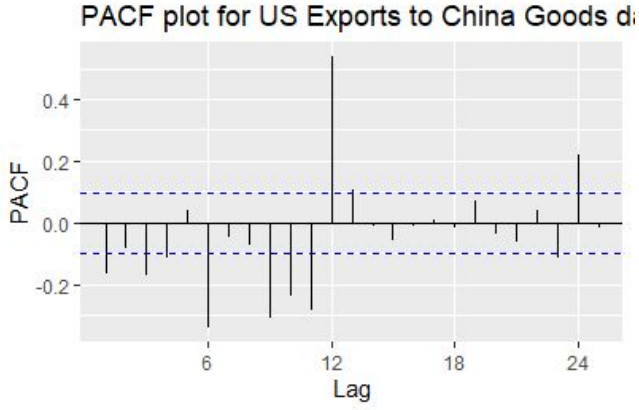
7 Appendix

```
ndiffs(train.imp.ts)

## [1] 1

ndiffs(train.exp.ts)

## [1] 1
```


ACF plot for US Imports of Chinese Goods c
ACF
0.4
0.2
0.0
-0.2
6
12
18
24
Lag


PACF plot for US Imports of Chinese Goods
PACF
0.4
0.2
0.0
-0.2
6
12
18
24
Lag


ACF plot for US Exports to China data (Firs
ACF
0.50
0.25
0.00
-0.25
6
12
18
24
Lag


PACF plot for US Exports to China Goods d
PACF
0.4
0.2
0.0
-0.2
6
12
18
24
Lag

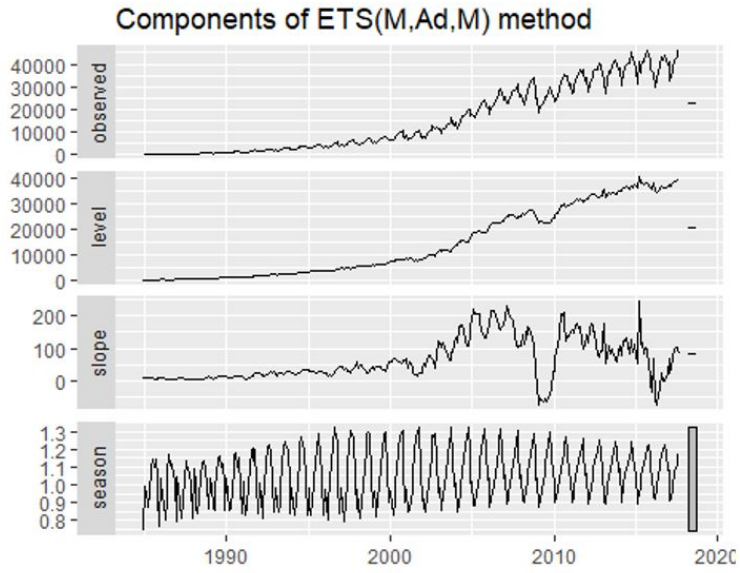
```
nonlinearityTest(train.imp.ts)
```

```
##       ** Teraesvirta's neural network test  **
##       Null hypothesis: Linearity in "mean"
##       X-squared =  14.54697  df =  2  p-value =  0.0006936909
##
##       ** White neural network test  **
##       Null hypothesis: Linearity in "mean"
##       X-squared =  14.70372  df =  2  p-value =  0.0006413997
##
##       ** Keenan's one-degree test for nonlinearity  **
##       Null hypothesis: The time series follows some AR process
##       F-stat =  20.80022  p-value =  7.11869e-06
##
##       ** McLeod-Li test  **
##       Null hypothesis: The time series follows some ARIMA process
##       Maximum p-value =  0
##
##       ** Tsay's Test for nonlinearity **
##       Null hypothesis: The time series follows some AR process
##       F-stat =  110.6  p-value =  3.2e-15
##
##       ** Likelihood ratio test for threshold nonlinearity **
##       Null hypothesis: The time series follows some AR process
##       Alternativce hypothesis: The time series follows some TAR process
##       X-squared =  111.4864  p-value =  1.121134e-10
```

```
nonlinearityTest(train.exp.ts)
```

```
##       ** Teraesvirta's neural network test  **
##       Null hypothesis: Linearity in "mean"
##       X-squared =  39.29675  df =  2  p-value =  2.929673e-09
##
##       ** White neural network test  **
##       Null hypothesis: Linearity in "mean"
##       X-squared =  44.29962  df =  2  p-value =  2.401379e-10
##
##       ** Keenan's one-degree test for nonlinearity  **
##       Null hypothesis: The time series follows some AR process
##       F-stat =  6.013017  p-value =  0.01470146
##
##       ** McLeod-Li test  **
##       Null hypothesis: The time series follows some ARIMA process
##       Maximum p-value =  0
##
##       ** Tsay's Test for nonlinearity **
##       Null hypothesis: The time series follows some AR process
##       F-stat =  9.493  p-value =  1.722e-06
##
##       ** Likelihood ratio test for threshold nonlinearity **
##       Null hypothesis: The time series follows some AR process
##       Alternativce hypothesis: The time series follows some TAR process
##       X-squared =  62.49455  p-value =  0.002001879
```

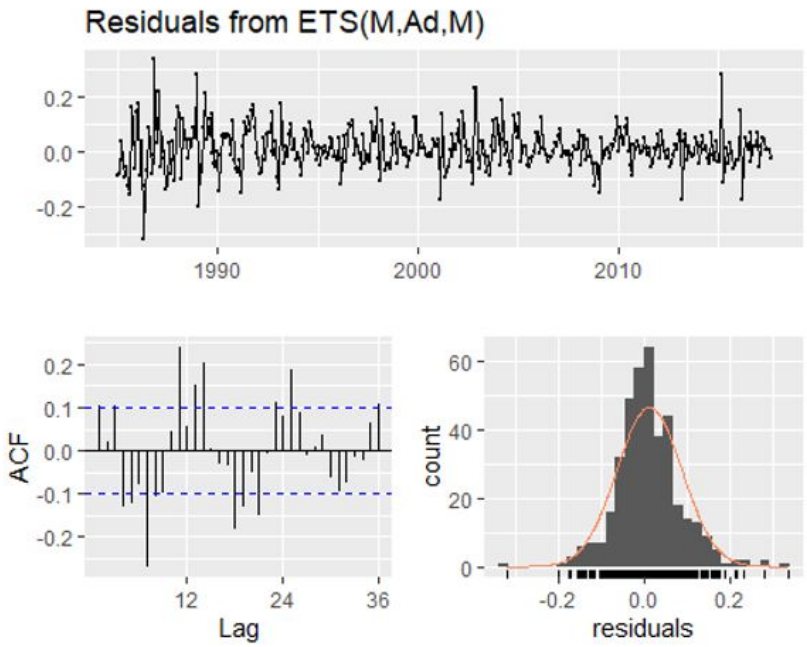
```r
best.ets.imp <- ets(train.imp.ts, damped = T)
autoplot(best.ets.imp)
```




```r
accuracy(forecast(best.ets.imp),imp.ts)
```

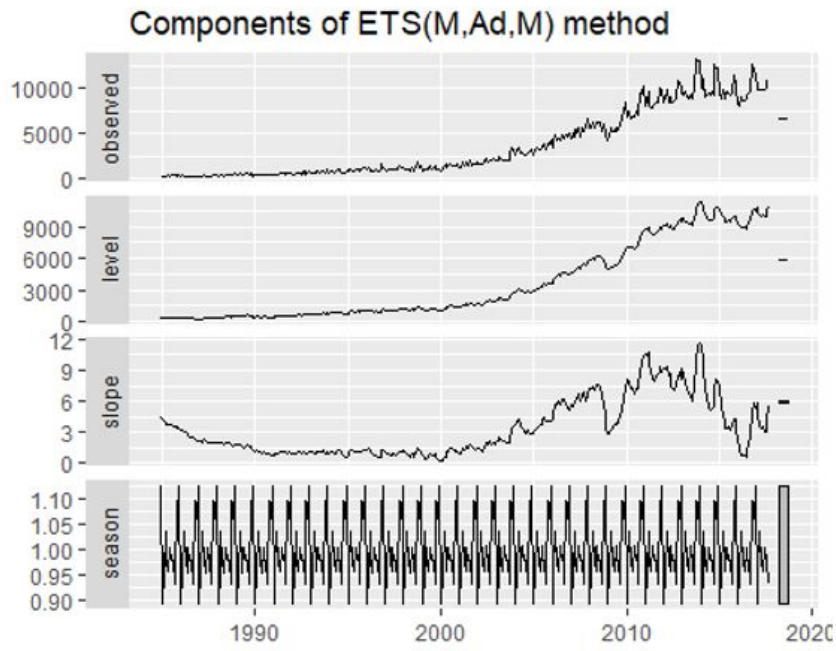
```
##                      ME     RMSE       MAE         MPE     MAPE      MASE
## Training set   54.9893 1162.354  644.2055  0.5429829 5.494712 0.3703234
## Test set     -348.9521 3524.005 3067.0773 -1.5712144 7.537235 1.7631180
##                      ACF1 Theil's U
## Training set -0.06901977        NA
## Test set      0.75490701  1.092759
```

```
checkresiduals(forecast(best.ets.imp))
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,Ad,M)
## Q* = 154.31, df = 7, p-value < 2.2e-16
##
## Model df: 17.   Total lags used: 24
```

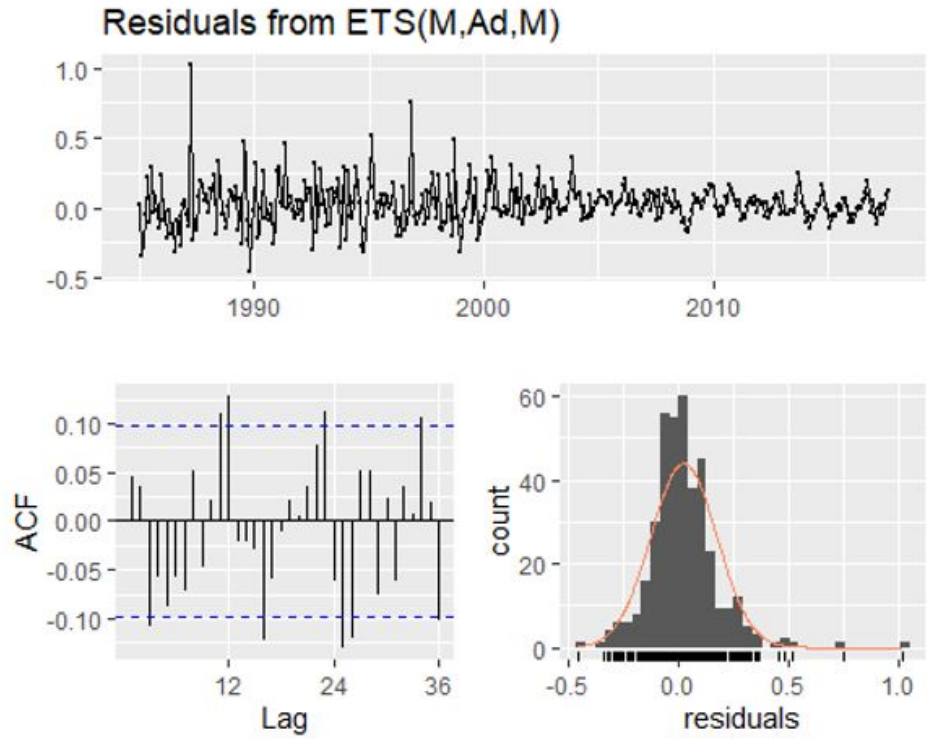
```
best.ets.exp <- ets(train.exp.ts)
autoplot(best.ets.exp)
```




```
accuracy(forecast(best.ets.exp),exp.ts)
```

```
##                       ME     RMSE       MAE          MPE     MAPE      MASE
## Training set    62.92264  420.841  257.0866   0.1297779 10.37534 0.5103051
## Test set     -1045.83546 1742.102 1451.8324 -12.7413460 15.89129 2.8818206
##                    ACF1 Theil's U
## Training set 0.4627870        NA
## Test set     0.6846683  1.369482
```

```
checkresiduals(forecast(best.ets.exp))
```



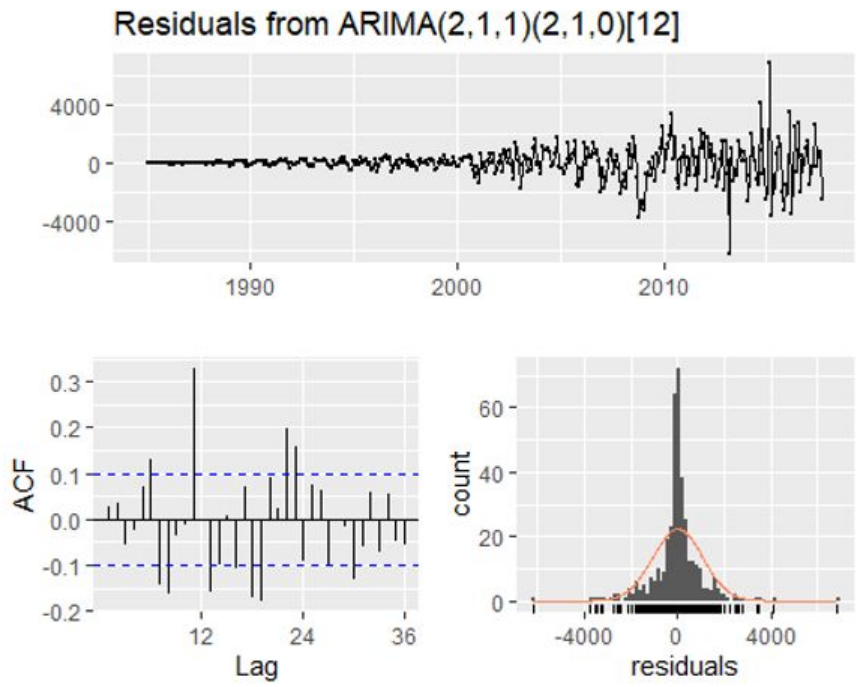

```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,Ad,M)
## Q* = 45.992, df = 7, p-value = 8.773e-08
##
## Model df: 17.   Total lags used: 24
```

```
summary(best.arima.imp.nolambda)

## Series: train.imp.ts
## ARIMA(2,1,1)(2,1,0)[12]
##
## Coefficients:
##           ar1      ar2     ma1     sar1     sar2
##       -0.9614  -0.5410  0.4616  -0.7938  -0.5975
## s.e.   0.0731   0.0462  0.0799   0.0466   0.0512
##
## sigma^2 estimated as 1180427:  log likelihood=-3200.48
## AIC=6412.97   AICc=6413.19   BIC=6436.61
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 8.237035 1061.302 631.1061 -0.0213644 5.356059 0.3627931
##                     ACF1
## Training set 0.02655289
```

```
accuracy(forecast(best.arima.imp.nolambda),imp.ts)
```

```
##                      ME     RMSE       MAE         MPE     MAPE      MASE
## Training set   8.237035 1061.302  631.1061 -0.0213644 5.356059 0.3627931
## Test set     390.503477 3280.559 2884.8431  0.2099417 6.900162 1.6583602
##                    ACF1 Theil's U
## Training set 0.02655289        NA
## Test set     0.64841666 0.9497607
```

```
checkresiduals(best.arima.imp.nolambda)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,1)(2,1,0)[12]
## Q* = 154.92, df = 19, p-value < 2.2e-16
##
## Model df: 5.   Total lags used: 24
```

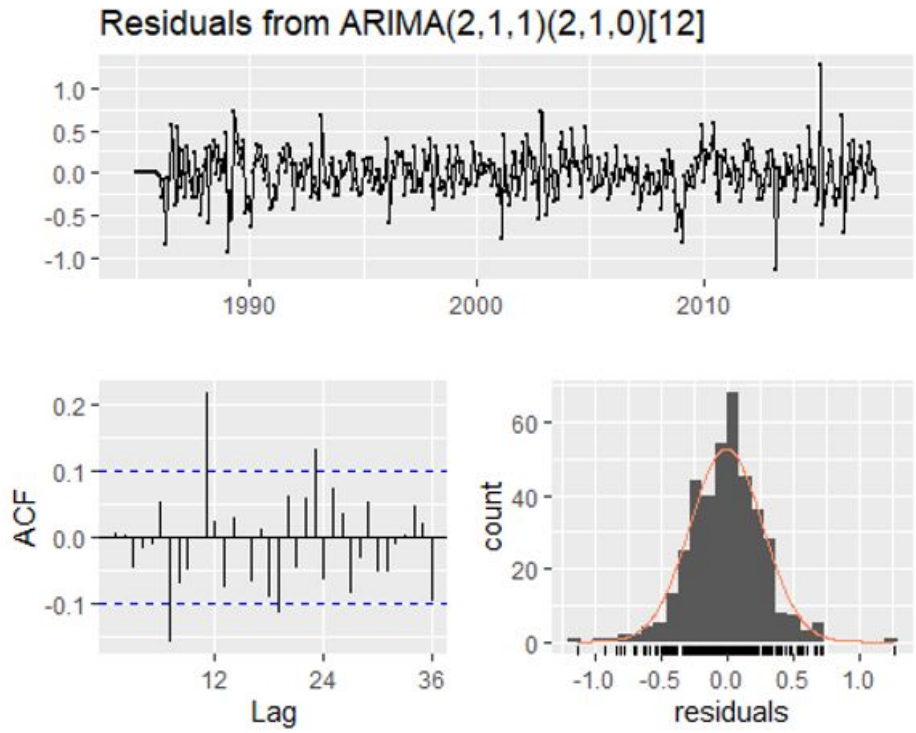
```r
summary(best.arima.imp.yeslambda)
```

```
## Series: train.imp.ts
## ARIMA(2,1,1)(2,1,0)[12]
## Box Cox transformation: lambda= 0.1637399
##
## Coefficients:
##           ar1      ar2     ma1     sar1     sar2
##       -0.8298  -0.4428  0.3331  -0.7430  -0.4380
## s.e.   0.1037   0.0538  0.1119   0.0477   0.0519
##
## sigma^2 estimated as 0.07768:  log likelihood=-55.83
## AIC=123.66   AICc=123.88   BIC=147.3
##
## Training set error measures:
##                      ME    RMSE      MAE        MPE     MAPE      MASE
## Training set -34.37358 1076.89 597.2436 -0.4262426 4.977908 0.3433272
##                      ACF1
## Training set -0.04637189
```

```r
accuracy(forecast(best.arima.imp.yeslambda),imp.ts)
```

```
##                      ME     RMSE       MAE        MPE     MAPE      MASE
## Training set -34.37358 1076.890  597.2436 -0.4262426 4.977908 0.3433272
## Test set     643.62276 3440.545 3070.5692  0.8346843 7.287937 1.7651253
##                      ACF1 Theil's U
## Training set -0.04637189        NA
## Test set      0.70960777 0.9910815
```

```
checkresiduals(best.arima.imp.yeslambda)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,1)(2,1,0)[12]
## Q* = 60.985, df = 19, p-value = 2.7e-06
##
## Model df: 5.   Total lags used: 24
```

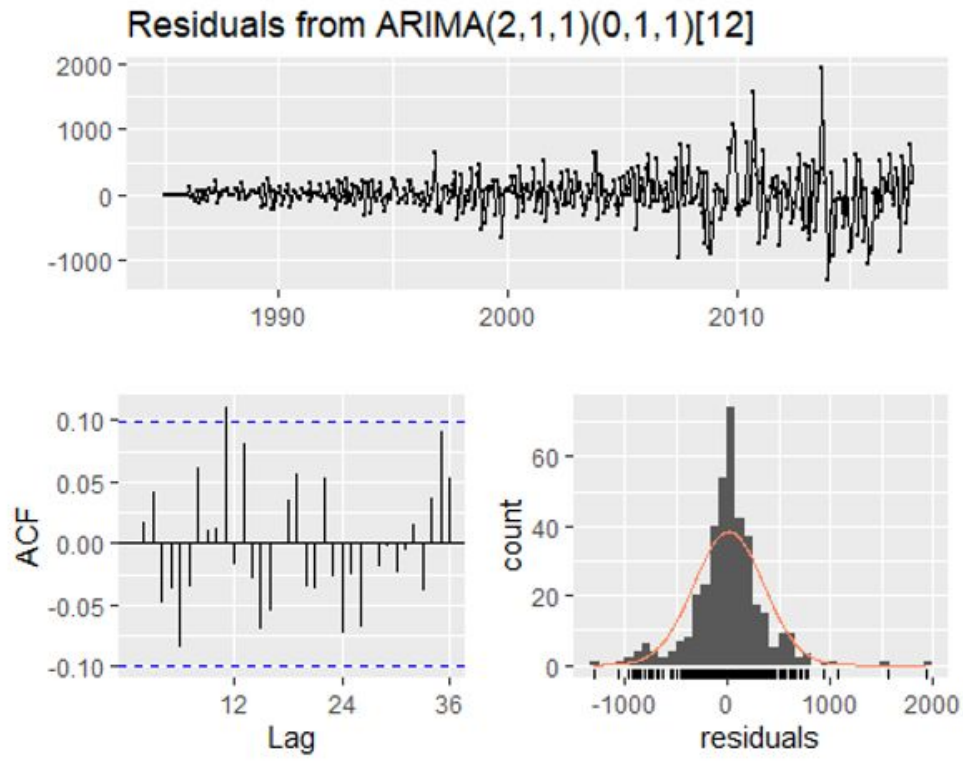
```
summary(best.arima.exp.nolambda)
```

```
## Series: train.exp.ts
## ARIMA(2,1,1)(0,1,1)[12]
##
## Coefficients:
##          ar1     ar2      ma1     sma1
##       0.7114  0.1283  -0.9646  -0.5424
## s.e.  0.0574  0.0559   0.0268   0.0442
##
## sigma^2 estimated as 119334:  log likelihood=-2760.92
## AIC=5531.83   AICc=5531.99   BIC=5551.53
##
## Training set error measures:
##                     ME    RMSE      MAE        MPE     MAPE      MASE
## Training set 11.03621 337.893 227.3651 -0.4317361 11.35756 0.4513094
##                      ACF1
## Training set 0.0001588779
```

```
accuracy(forecast(best.arima.exp.nolambda),exp.ts)
```

```
##                       ME     RMSE       MAE         MPE     MAPE      MASE
## Training set    11.03621  337.893  227.3651  -0.4317361 11.35756 0.4513094
## Test set     -1482.88331 2239.336 1770.2659 -17.4212900 19.70742 3.5138966
##                     ACF1 Theil's U
## Training set 0.0001588779        NA
## Test set     0.6795136848  1.767939
```

```
checkresiduals(best.arima.exp.nolambda)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,1)(0,1,1)[12]
## Q* = 25.256, df = 20, p-value = 0.1918
##
## Model df: 4.   Total lags used: 24
```

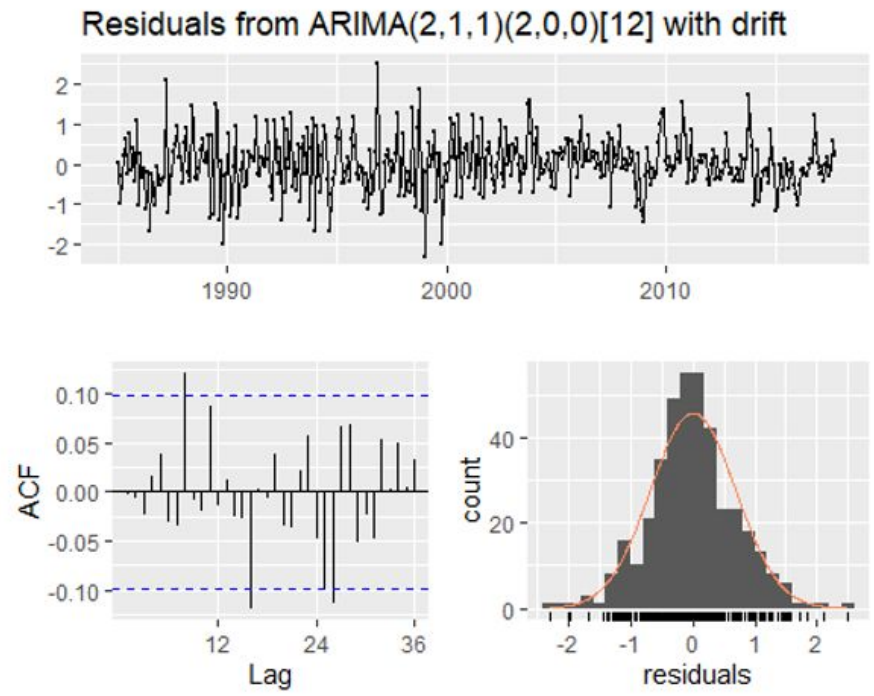
```r
summary(best.arima.exp.yeslambda)
```

```
## Series: train.exp.ts
## ARIMA(2,1,1)(2,0,0)[12] with drift
## Box Cox transformation: lambda= 0.220129
##
## Coefficients:
##          ar1     ar2      ma1    sar1    sar2   drift
##       0.3991  0.2344  -0.9395  0.3487  0.1650  0.0480
## s.e.  0.0606  0.0582   0.0312  0.0518  0.0511  0.0117
##
## sigma^2 estimated as 0.4778:  log likelihood=-410.12
## AIC=834.24   AICc=834.53   BIC=862.04
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 10.36284 402.0039 240.6093 -1.331049 10.81068 0.4775985
##                    ACF1
## Training set 0.2891842
```

```r
accuracy(forecast(best.arima.exp.yeslambda),exp.ts)
```

```
##                       ME      RMSE       MAE        MPE     MAPE      MASE
## Training set    10.36284  402.0039  240.6093  -1.331049 10.81068 0.4775985
## Test set     -1610.97316 2434.7656 2071.0324 -19.342475 22.87962 4.1109044
##                    ACF1 Theil's U
## Training set 0.2891842        NA
## Test set     0.6678145   1.93414
```

```
checkresiduals(best.arima.exp.yeslambda)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,1)(2,0,0)[12] with drift
## Q* = 21.647, df = 18, p-value = 0.248
##
## Model df: 6.   Total lags used: 24
```

```
neural.imp <- nnetar(train.imp.ts)
neural.imp

## Series: train.imp.ts
## Model:  NNAR(4,1,3)[12]
## Call:   nnetar(y = train.imp.ts)
##
## Average of 20 networks, each of which is
## a 5-3-1 network with 22 weights
## options were - linear output units
##
## sigma^2 estimated as 1618448
```

```
accuracy(forecast(neural.imp),imp.ts)
```

```
##                       ME     RMSE       MAE       MPE    MAPE      MASE
## Training set    1.004195 1272.183  746.6791 -2.172185 7.23142 0.4292306
## Test set     2113.309300 3970.815 3526.2591  4.157140 8.17489 2.0270800
##                   ACF1 Theil's U
## Training set 0.1262050        NA
## Test set     0.6928315  1.100258
```

```
neural.imp.boxcox <- nnetar(train.imp.ts, lambda =
BoxCox.lambda(train.imp.ts))
neural.imp.boxcox
```

```
## Series: train.imp.ts
## Model:  NNAR(1,1,2)[12]
## Call:   nnetar(y = train.imp.ts, lambda = BoxCox.lambda(train.imp.ts))
##
## Average of 20 networks, each of which is
## a 2-2-1 network with 9 weights
## options were - linear output units
##
## sigma^2 estimated as 0.1319
```

```
accuracy(forecast(neural.imp.boxcox),imp.ts)
```

```
##                     ME     RMSE       MAE        MPE     MAPE      MASE
## Training set   39.81596 1514.976  884.1333 -0.4315061 6.597035 0.5082465
## Test set     1795.17829 4444.743 3615.2595  3.0381427 8.485026 2.0782421
##                  ACF1 Theil's U
## Training set 0.1137020        NA
## Test set     0.7510175  1.264006
```

```
neural.exp <- nnetar(train.exp.ts)
neural.exp
```

```
## Series: train.exp.ts
## Model:  NNAR(1,1,2)[12]
## Call:   nnetar(y = train.exp.ts)
##
## Average of 20 networks, each of which is
## a 2-2-1 network with 9 weights
## options were - linear output units
##
## sigma^2 estimated as 195589
```

```
accuracy(forecast(neural.exp),exp.ts)
```

```
##                         ME      RMSE       MAE         MPE     MAPE
## Training set     0.2292767  442.2549  278.0997  -3.358741 12.53733
## Test set     -1196.3518931 2091.7454 1769.4220 -14.909501 19.45795
##                    MASE      ACF1 Theil's U
## Training set 0.5520151 0.2018676        NA
## Test set     3.5122214 0.6016065  1.652535
```

```
neural.exp.boxcox <- nnetar(train.exp.ts, lambda =
BoxCox.lambda(train.exp.ts))
neural.exp.boxcox

## Series: train.exp.ts
## Model:  NNAR(2,1,2)[12]
## Call:   nnetar(y = train.exp.ts, lambda = BoxCox.lambda(train.exp.ts))
##
## Average of 20 networks, each of which is
## a 3-2-1 network with 11 weights
## options were - linear output units
##
## sigma^2 estimated as 0.5098
```

```
accuracy(forecast(neural.exp.boxcox),exp.ts)

##                      ME      RMSE       MAE         MPE     MAPE      MASE
## Training set   15.52981  457.1013  270.3462  -1.369548 11.04153 0.5366247
## Test set     -789.80332 1744.3656 1487.8390 -10.532445 16.06305 2.9532921
##                   ACF1 Theil's U
## Training set 0.2351623        NA
## Test set     0.5333260  1.342548
```

data	method	model	lambda	sigma**2	subset	mape	mase
import	neural	NNAR(4,1,3)[12]	N/A	1618448	train	7.23142	0.4292306
					test	8.17489	2.02708
import	neural	NNAR(1,1,2)[12]	0.1637399	0.1308	train	6.597035	0.5082465
					test	8.48026	2.0782421
export	neural	NNAR(1,1,2)[12]	N/A	195589	train	12.53733	0.5520151
					test	19.45795	3.5122214
export	neural	NNAR(2,1,2)[12]	0.220129	0.5041	train	11.04153	0.5366247
					test	16.06305	2.9532921

```
accuracy(forecast(bag.imp), imp.ts)

##                     ME     RMSE       MAE         MPE     MAPE      MASE
## Training set  9.890236 1001.747  598.2268  0.1107069 5.215448 0.3438924
## Test set     69.670790 4186.723 3566.8610 -0.5733451 8.673755 2.0504201
##                   ACF1 Theil's U
## Training set 0.1788646        NA
## Test set     0.7304160   1.24933
```

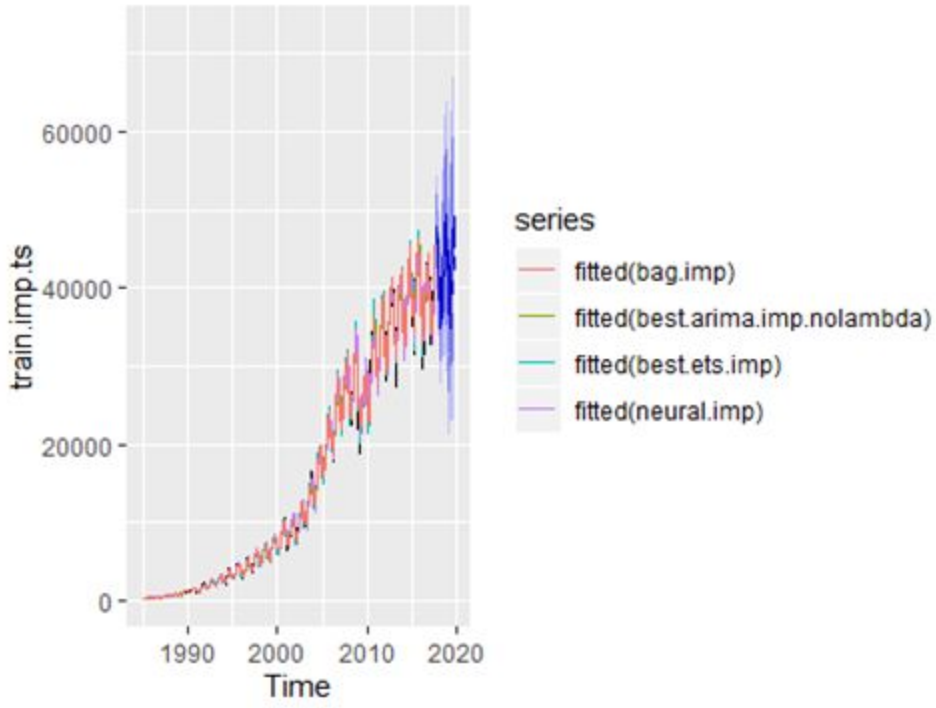
```
bag.imp.wlambda <- baggedModel(BoxCox(train.imp.ts,lambda =
BoxCox.lambda(train.imp.ts)), bootstrapped_series =
bld.mbb.bootstrap(train.imp.ts, 100))
accuracy(forecast(bag.imp.wlambda), BoxCox(imp.ts,lambda =
BoxCox.lambda(train.imp.ts)))

##                     ME     RMSE      MAE        MPE     MAPE     MASE
## Training set -15365.52 21156.07 15365.52  -60220.1  60220.1 22417.89
## Test set     -42498.90 42738.15 42498.90 -147209.9 147209.9 62004.77
##                   ACF1 Theil's U
## Training set 0.9869763        NA
## Test set     0.6993608  87036.79
```

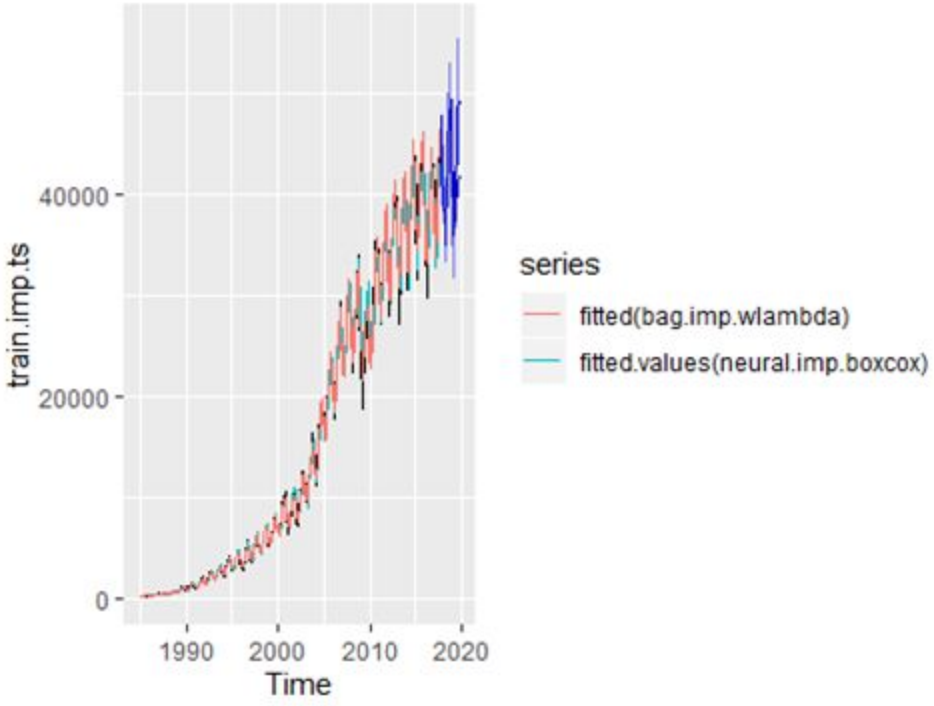
```
accuracy(forecast(bag.exp), exp.ts)

##                     ME      RMSE       MAE         MPE     MAPE      MASE
## Training set   42.11229  391.9716  239.6898 -0.3486852  9.43660 0.4757732
## Test set     -764.05031 1642.8201 1399.6392 -9.9447899 15.04068 2.7782196
##                   ACF1 Theil's U
## Training set 0.5771892        NA
## Test set     0.7268391  1.264623
```

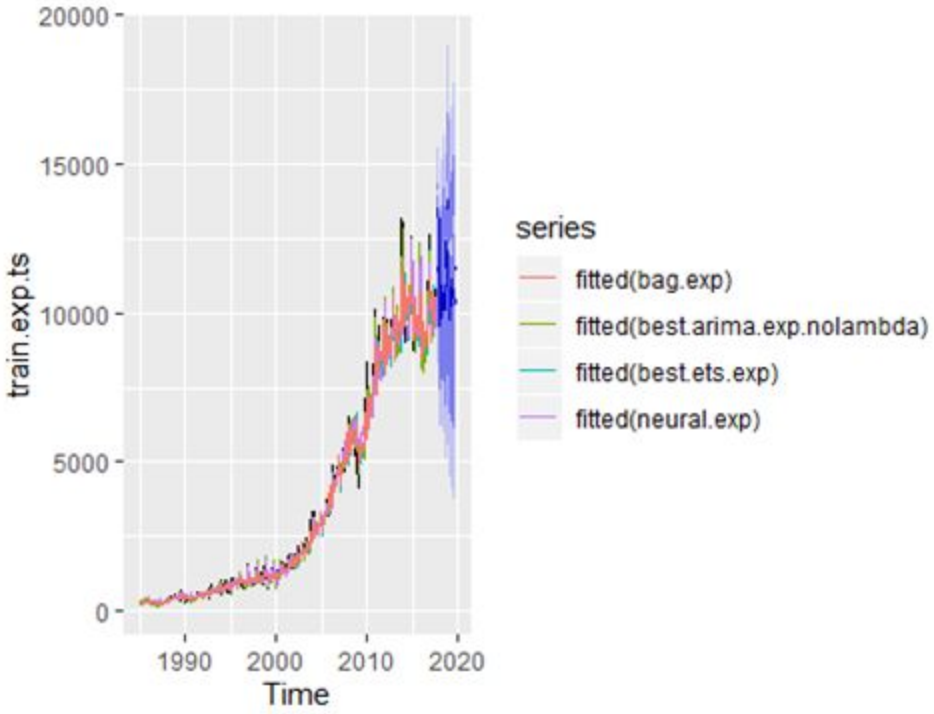
```
bag.exp.wlambda <- baggedModel(BoxCox(train.exp.ts,lambda =
BoxCox.lambda(train.exp.ts)), bootstrapped_series =
bld.mbb.bootstrap(train.exp.ts, 100, block_size = NULL))
accuracy(forecast(bag.exp.wlambda), BoxCox(exp.ts,lambda =
BoxCox.lambda(train.exp.ts)))
```

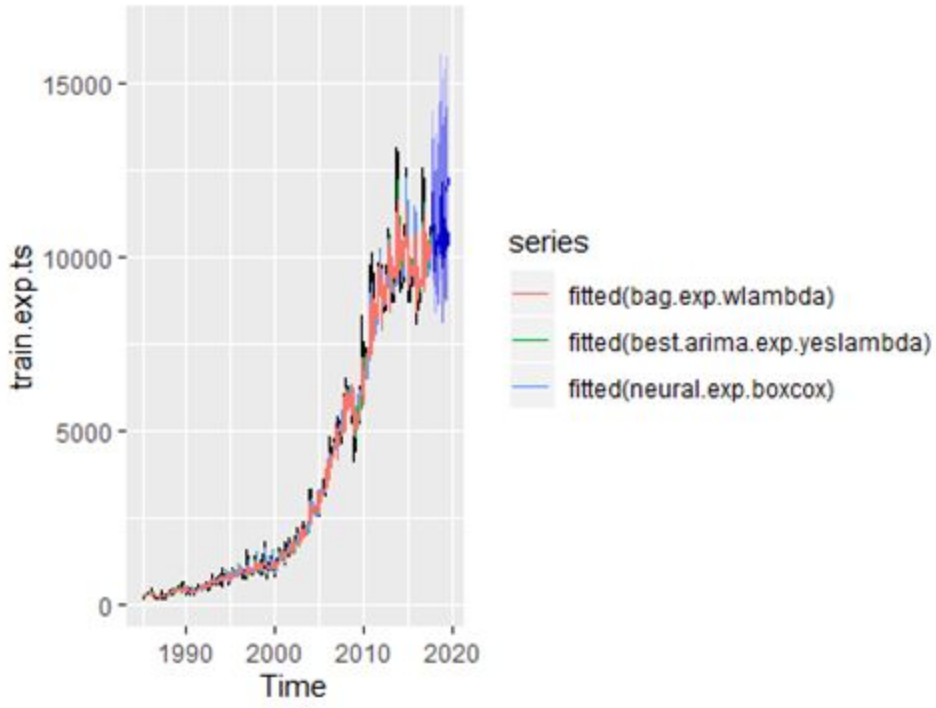
```
##                      ME      RMSE       MAE        MPE     MAPE      MASE
## Training set  -3533.075  5038.574  3533.075 -13916.27 13916.27  3543.666
## Test set     -10659.701 10680.271 10659.701 -35757.59 35757.59 10691.656
##                    ACF1 Theil's U
## Training set 0.98596726        NA
## Test set     0.06434909  9805.401
```

data	method	lambda	subset	mape	mase
import	bagged	N/A	train	5.215448	0.3438924
			test	8.673755	2.0504201
import	bagged	0.1637399	train	60220.0962	22417.894
			test	147209.9	62004.77
export	bagged	N/A	train	9.4366	0.4757732
			test	15.04068	2.7782196
export	bagged	0.220129	train	13916.27189	3543.666
			test	35757.59	10691.656




train.imp.ts
40000
20000
0
1990
2000
2010
2020
Time
series
fitted(bag.imp.wlambda)
fitted.values(neural.imp.boxcox)


train.exp.ts
20000
15000
10000
5000
0
1990
2000
2010
2020
Time
series
fitted(bag.exp)
fitted(best.arima.exp.nolambda)
fitted(best.ets.exp)
fitted(neural.exp)




```
diss(impex.ts, METHOD = "EUCL")

##           Imports
## Exports 358571.2

diss(impex.ts, METHOD = "COR")

##            Imports
## Exports 0.2604179
```

```
tsoutliers(imp.ts, lambda = "auto")

## $index
## [1]  15  22 290 363
##
## $replacements
## [1]   277.6325   519.7730 21335.2673 33007.5785
```

```
tsoutliers(exp.ts, lambda = "auto")

## $index
## integer(0)
##
## $replacements
## numeric(0)
```

References

[1] U.S. Bureau of Economic Analysis and U.S. Census Bureau, U.S. Imports of Goods by Customs Basis from China [IMPCH], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/IMPCH, December 2019

[2] U.S. Bureau of Economic Analysis and U.S. Census Bureau, U.S. Exports of Goods by F.A.S. Basis to China, Mainland [EXPCH], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/EXPCH, December, 2019.

[3] Council on Foreign Relations, U.S. Relations with China 1949-2019, https://www.cfr.org/timeline/us-relations-china