#ARIMA/SARIMA

```
library(plyr)
library(fpp3)
```

```
## -- Attaching packages -------------------------------------------- fpp3 0.4.0 --

## v tibble      3.1.2      v tsibble     1.0.1
## v dplyr       1.0.7      v tsibbledata 0.3.0
## v tidyr       1.1.3      v feasts      0.2.2
## v lubridate   1.7.10     v fable       0.3.1
## v ggplot2     3.3.5

## -- Conflicts ------------------------------------------------- fpp3_conflicts --
## x dplyr::arrange()     masks plyr::arrange()
## x dplyr::count()       masks plyr::count()
## x lubridate::date()    masks base::date()
## x dplyr::failwith()    masks plyr::failwith()
## x dplyr::filter()      masks stats::filter()
## x dplyr::id()          masks plyr::id()
## x tsibble::intersect() masks base::intersect()
## x tsibble::interval()  masks lubridate::interval()
## x dplyr::lag()         masks stats::lag()
## x dplyr::mutate()      masks plyr::mutate()
## x dplyr::rename()      masks plyr::rename()
## x tsibble::setdiff()   masks base::setdiff()
## x dplyr::summarise()   masks plyr::summarise()
## x dplyr::summarize()   masks plyr::summarize()
## x tsibble::union()     masks base::union()
```

```
library(tsibble)
library(forecast)
```

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

```
library(zoo)
```

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

## The following object is masked from 'package:tsibble':
##
##     index

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

```r
#read in the interpolated data
data <- readr::read_csv(file = 'data/data_interpolated_with_lags.csv') %>%
  mutate(yw = yearweek(yw)) %>%
  select(-X1) %>%
  as_tsibble(key = c(Mode,ORegionDAT, DRegionDAT), index = yw)
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_double(),
##   yw = col_character(),
##   Mode = col_character(),
##   ORegionDAT = col_character(),
##   DRegionDAT = col_character()
## )
## i Use `spec()` for the full column specifications.
```

```r
#make into univariate approx_cost series
data <- data %>%
  select(Mode, ORegionDAT, DRegionDAT, yw, approx_cost, tmax_lag_12, tmax_lag_2, prcp_lag_12, prcp_lag_
  filter(Mode == "R", DRegionDAT == "IL_CHI")
```

```r
#trim leading and trailing na's
data <- drop_na(data)
```

```r
#create training set - up through 2020 of the time series
train <- data %>%
  filter_index(~ "2018 W52")
```

There are many possible arima models - based on choice of hyperparameters and whether to include seasonality or not. The ARIMA() function automatically chooses the best hyperparameters.

```r
fit <- train %>%
  model(ARIMA(approx_cost ~ tmax_lag_12 + tmax_lag_2 + prcp_lag_12 + prcp_lag_2 + diesel_price + new_dea
```

```
## Warning: Provided exogenous regressors are rank deficient, removing regressors:
## `new_deaths`, `pandemic`
```

```r
#see what the automatically chosen arima models were.
report(fit)
```

```
## Series: approx_cost
## Model: LM w/ ARIMA(2,0,0) errors
##
## Coefficients:
##          ar1      ar2  tmax_lag_12  tmax_lag_2  prcp_lag_12  prcp_lag_2
##       1.0859  -0.2193       0.0008      0.0018      -0.1615     -0.0764
## s.e.  0.1030   0.1044       0.0055      0.0033       1.2465      0.2509
```

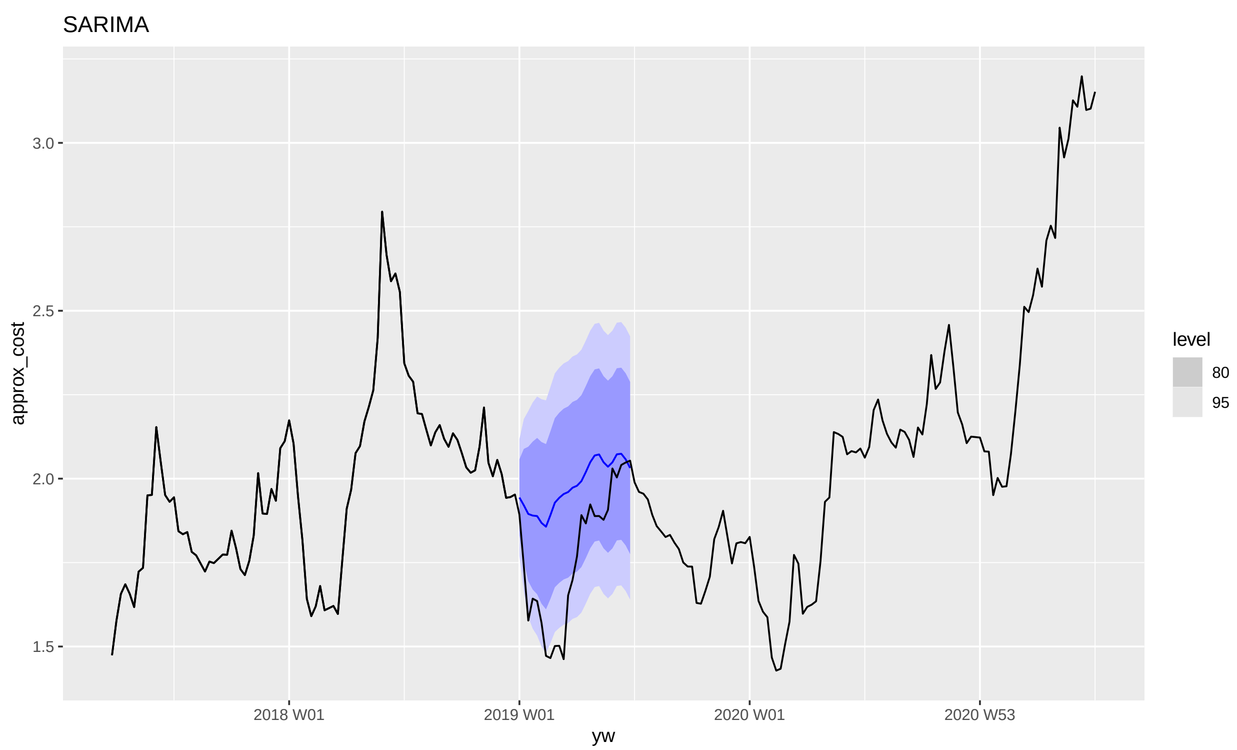
```
##       diesel_price  volume
##             0.5823   6e-04
## s.e.        0.1149   1e-03
##
## sigma^2 estimated as 0.007878:  log likelihood=95.61
## AIC=-173.22   AICc=-171.03   BIC=-150.53
```

It looks like for the first 3 time series ARIMA() automatically picked up on the period 52 seasonality. For Boston V data, it did not. Also it chose a different set of hyperparameters for each time series.

```r
#forecast
#in order to produce the forecast of approx_cost, we need to feed in a forecast of tmax, prcp, and dies
future_data <- data %>%
  filter_index("2019 W01"~"2019 W26") %>%
  select(Mode, ORegionDAT, DRegionDAT, yw, approx_cost, tmax_lag_12, tmax_lag_2, prcp_lag_12, prcp_lag_
fc <- fit %>% forecast(future_data)
```

```r
#plot
fc %>%
  autoplot(train) +
  autolayer(data, colour = "black") +
  labs(title="SARIMA")
```

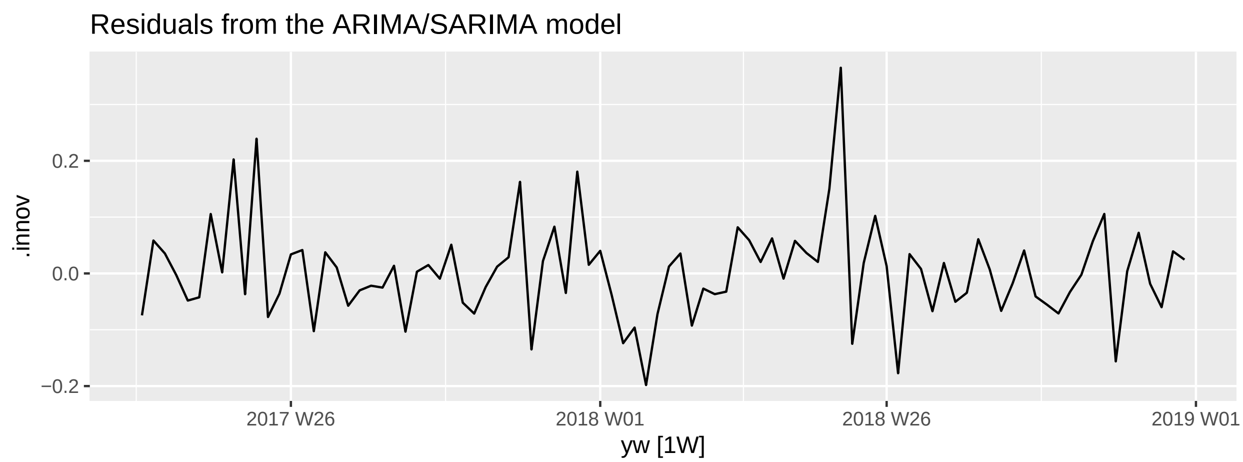
```
## Plot variable not specified, automatically selected `.vars = approx_cost`
```



#Looking at fitted values and residuals

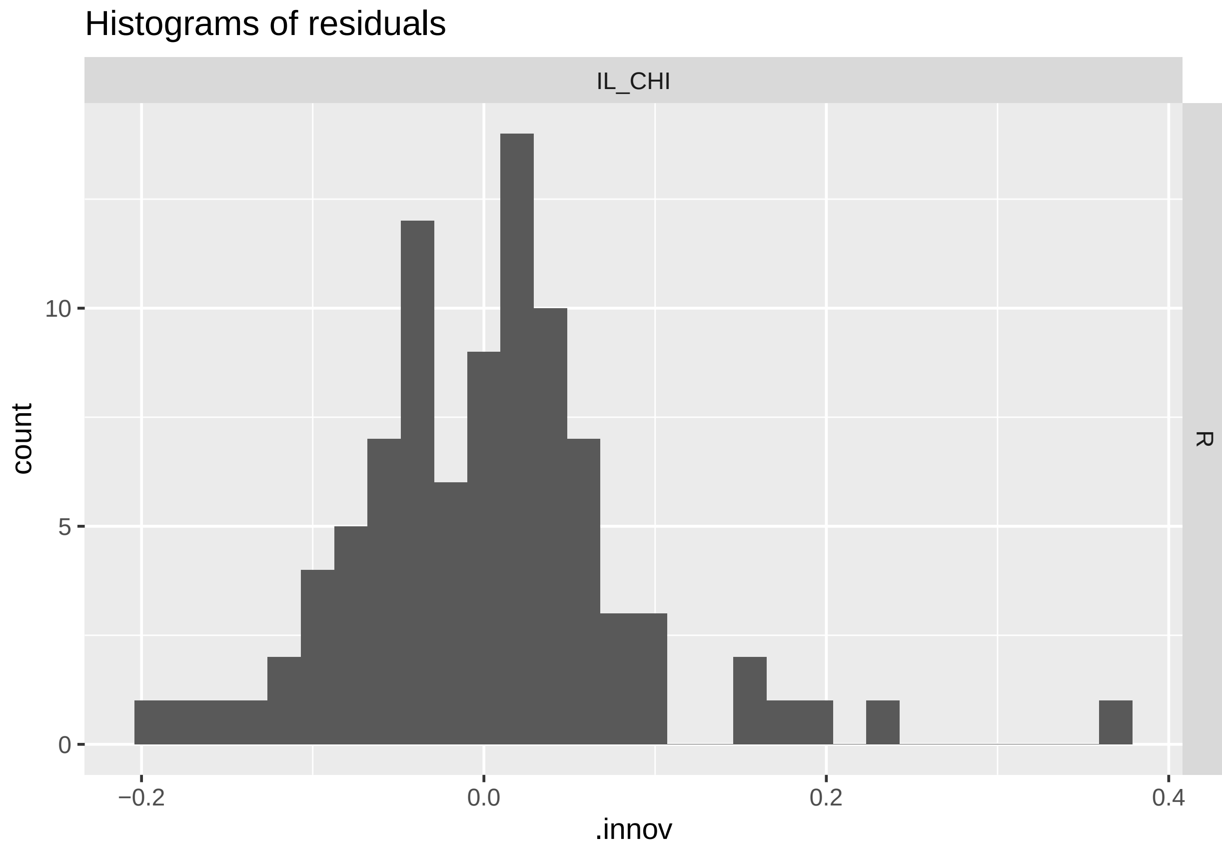
```r
#get fitted values and residuals
aug = augment(fit)
```

```r
#autoplot them
autoplot(aug, .innov) +
  labs(title = "Residuals from the ARIMA/SARIMA model")
```

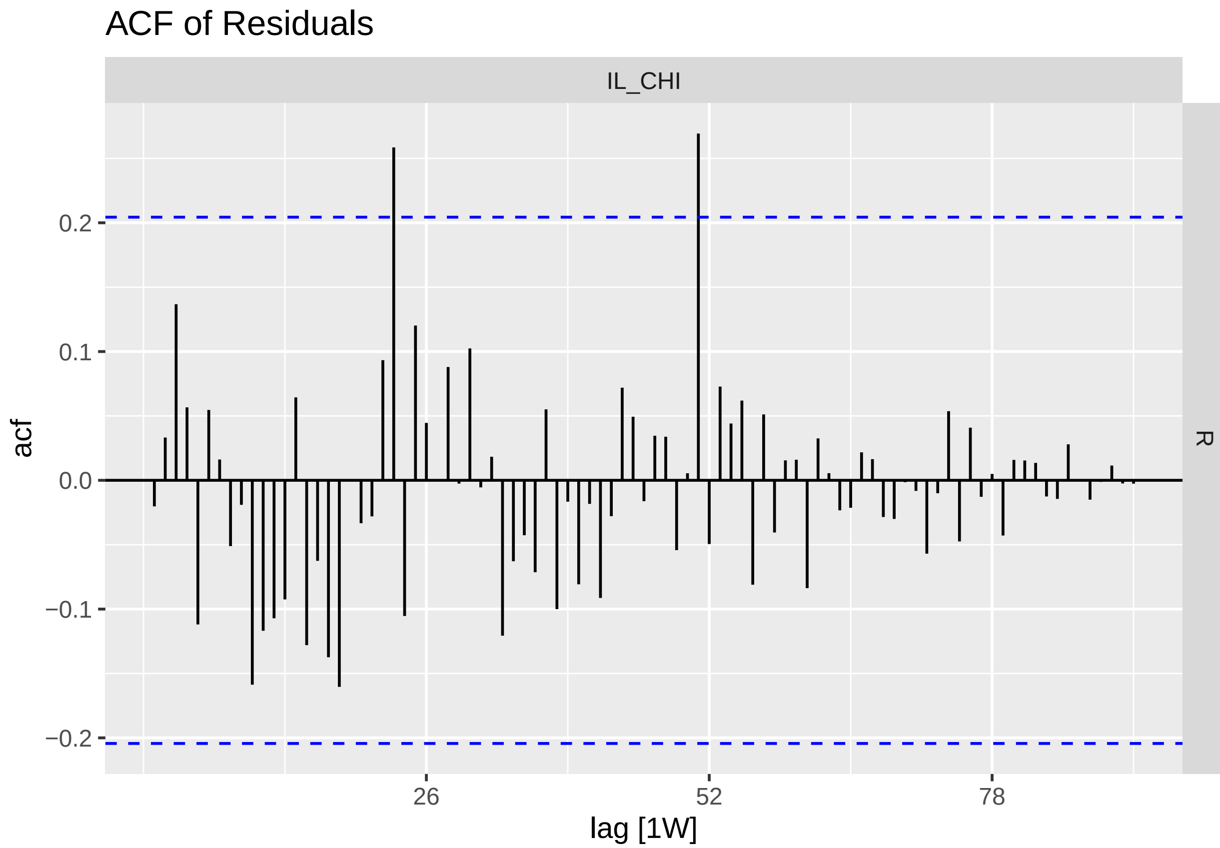



```r
#histograms
aug %>%
  ggplot(aes(x = .innov)) +
  geom_histogram() +
  facet_grid(rows = vars(Mode), cols = vars(DRegionDAT)) +
  labs(title = "Histograms of residuals")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
#acf
aug %>%
  ACF(.innov, lag_max = Inf) %>%
  autoplot() +
  facet_grid(rows = vars(Mode), cols = vars(DRegionDAT)) +
  labs(title = "ACF of Residuals")
```



Looking at forecast errors

```
accuracy(fc, data)
```

```
## # A tibble: 1 x 13
##   .model  Mode  ORegionDAT DRegionDAT .type     ME  RMSE   MAE   MPE  MAPE  MASE
##   <chr>   <chr> <chr>      <chr>      <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 "ARIMA~ R     CA_FRS     IL_CHI     Test  -0.210 0.254 0.212 -12.9  13.0 0.586
## # ... with 2 more variables: RMSSE <dbl>, ACF1 <dbl>
```