Analiza szeregów czasowych

Krawiec Piotr

12/06/2021

Spis treści

1 Szereg - Rozwój biznesu

Na szereg ten składają się dane po chodzące ze strony FRED.Dane zbierane są przez U.S Census Bureau, obejmują lata 2006-2021. Zbierane są w tygodniowych odstępach i dotyczą ilości wniosków o wydanie identyfikatora EAN (Employer Identyfication Number). Każdy pracodawna, koropracja, organizacja non-profit itp muszą posiadać takie numery, aby móc rozliczać się z podatku. Jest to zatem dobry wskaźnik tego ile nowych biznesów powstaje.

Do korzyści jakie przyniesie prognoza należy przewidywanie rozwoju gospodarki, gdyż nowo powstające biznesy mogą świadczyć o tym że w kraju panują korzystne warunki do rozwoju biznesu. Analiza szeregu pozwoli też przewidzieć jak ludzie postrzegają obecny stan gospodarki - czy są w stanie zaryzykować inwestując we własny biznes.

1.1 Wczytanie danych

W tym etapie wczytałem dane oraz uzupełniłem brakujące wartości średnimi.

```
## Warning: NAs introduced by coercion
```

```
##         DATE BUSAPPWNSAUS
## 1 2006-01-07        39580
## 2 2006-01-14        36920
## 3 2006-01-21        63300
```

```
## 4 2006-01-28        51910
## 5 2006-02-04        61430
## 6 2006-02-11        62890
```

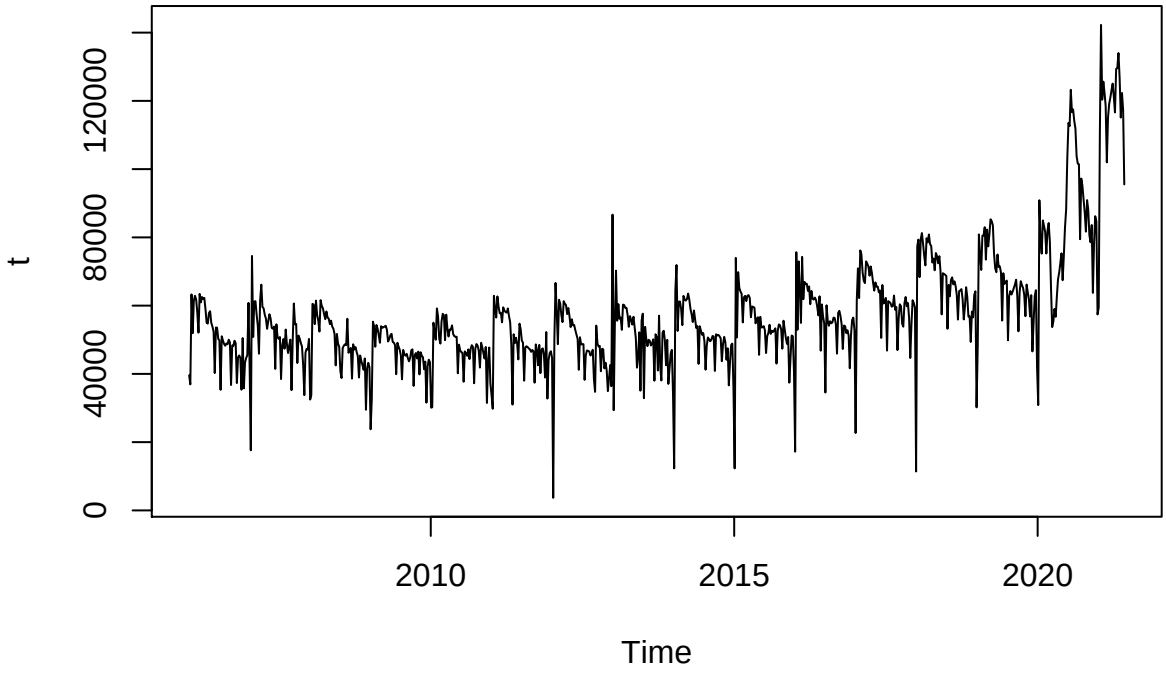
1.2 Główne cechy analizowanych danych

Tak prezentuje się wykres ilości wniosków w czasie:

```
library("forecast")
```

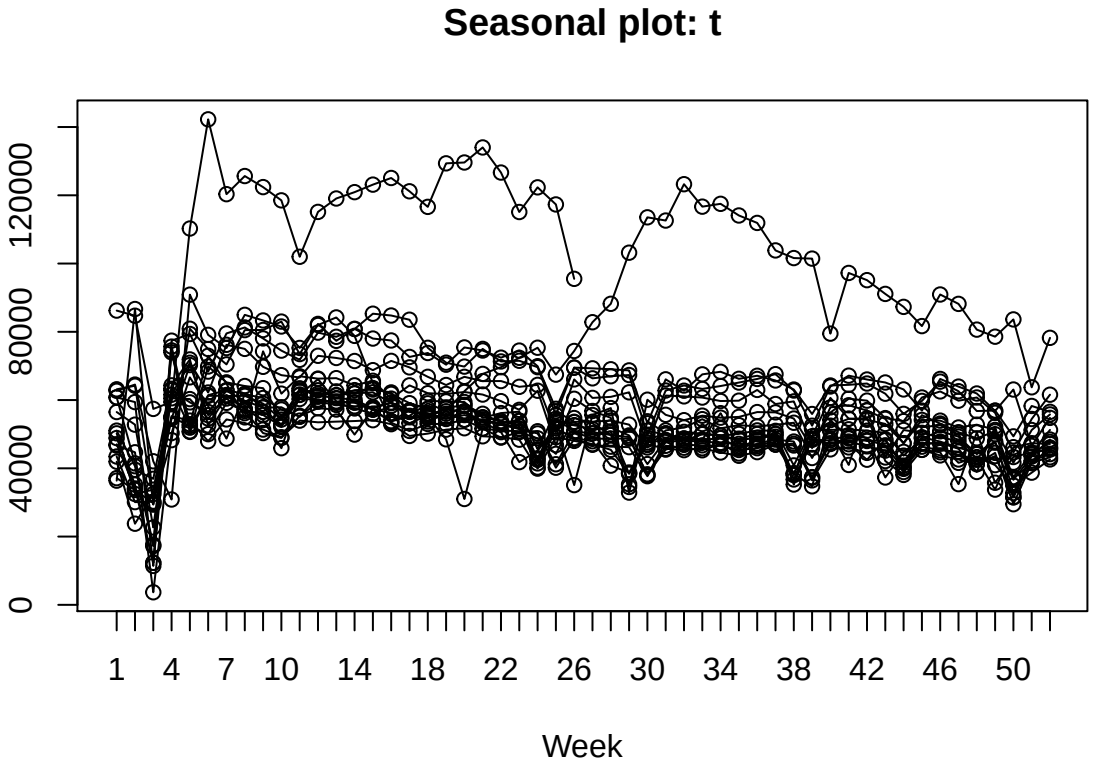
```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
t <- ts(d$BUSAPPWNSAUS, freq = 365.25/7, start =  2006 + 7/365.25)
plot(t)
```



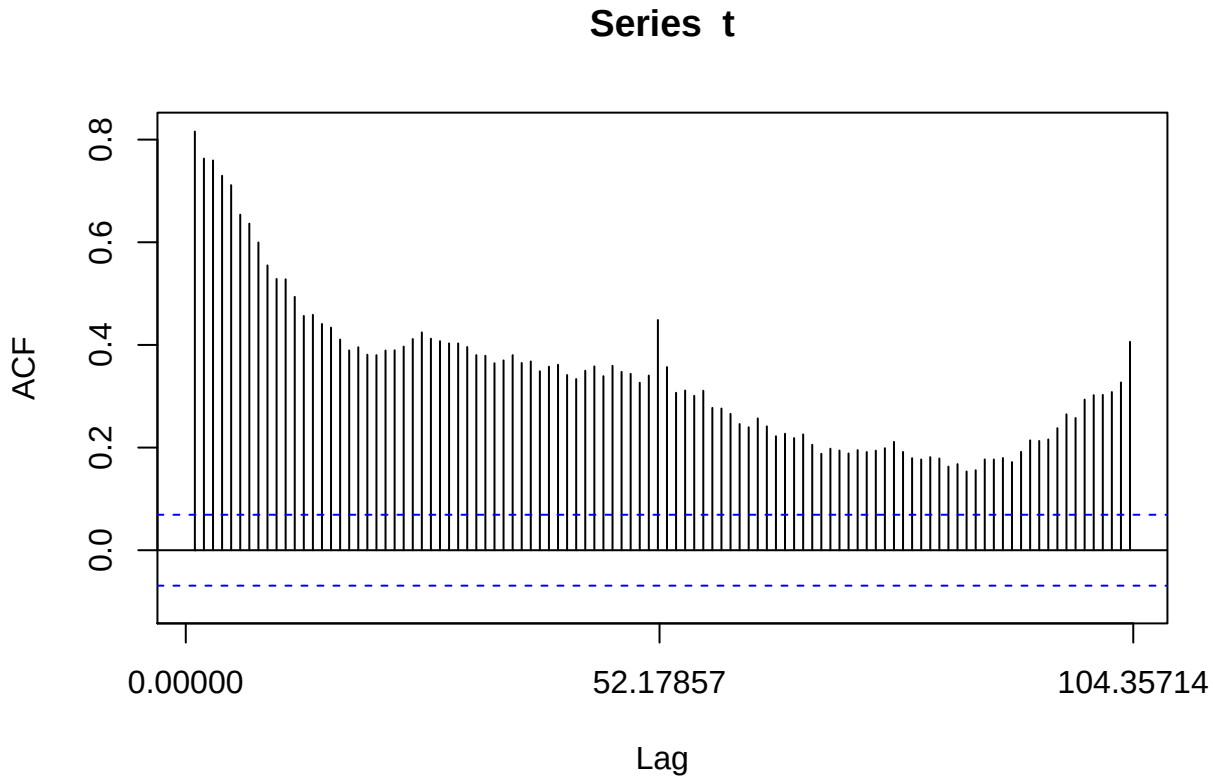
Z wykresu wywnioskować możemy że szereg ten posiada dużą sezonowość, pojawia się tu charakterystyczny wzorzec (odstające szpilki). Widać także niewielki dodatni trend, który gwałtownie rośnie na początku roku 2020.

```
seasonplot(t)
```



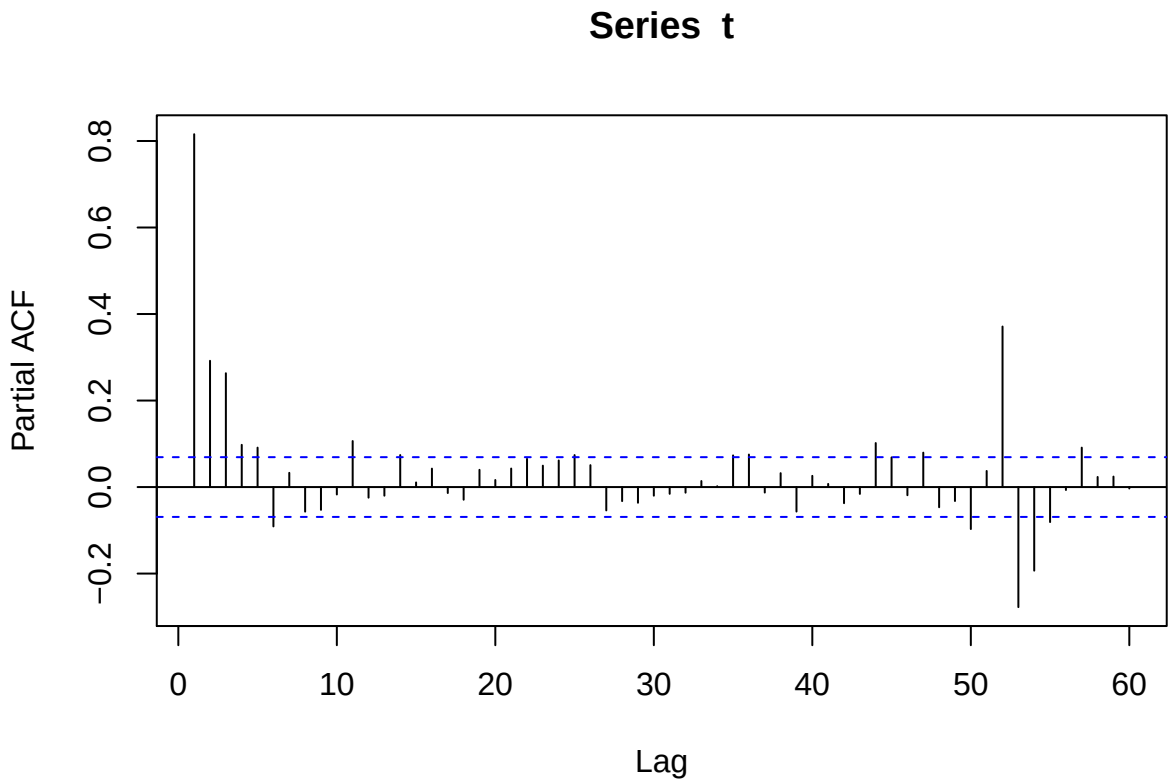
Porównując kolejne roczne sezony między sobą, sezonowość widać jeszcze dokładniej. Pojawia się też rok 2020, który znacznie odstaje wartościami, lecz kształtem nadal przypomina poprzednie sezony.

```
Acf(t)
```



Powolny spadek dodatnich wartości funkcji Acf wskazuje dodatni trend w szeregu.

```
Pacf(t, lag.max = 60)
```



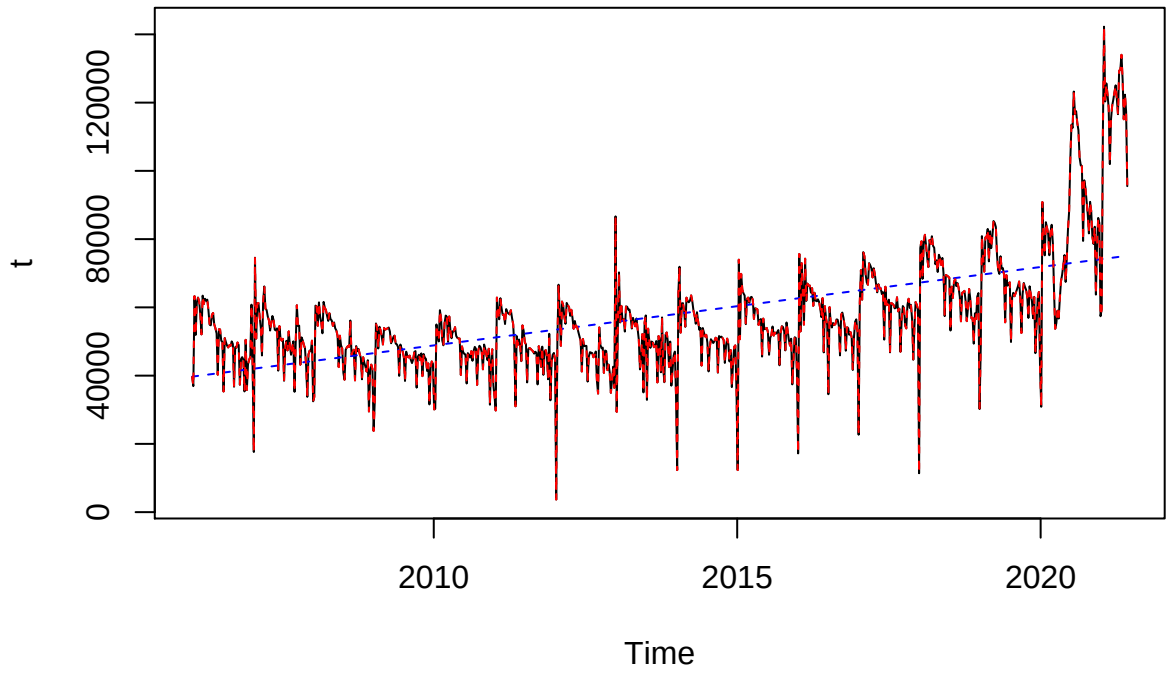
Na wykresie pojawia się wartość znacząca przy Lag=52, ponieważ dane są tygodniowe oznacza to korelację z danymi z poprzednich lat.

1.3 Dekompozycja szeregu

1.3.1 Modele regresji z trendem liniowym i sezonowością

Poniższy wykres przedstawia dopasowanie dwóch modeli liniowych trendu, z czego jeden z nich uwzględnia sezonowość.

```
ti <- t
tT <- tslm(t ~ trend) # Model regrasji z trendem liniowym
tTS <- tslm(t ~  trend + season) # Model regresji z trendem liniowym i sezonowością
plot(t)
lines(fitted(tT), col = "blue", lty = 2)
lines(fitted(tTS), col = "red", lty = 2)
```



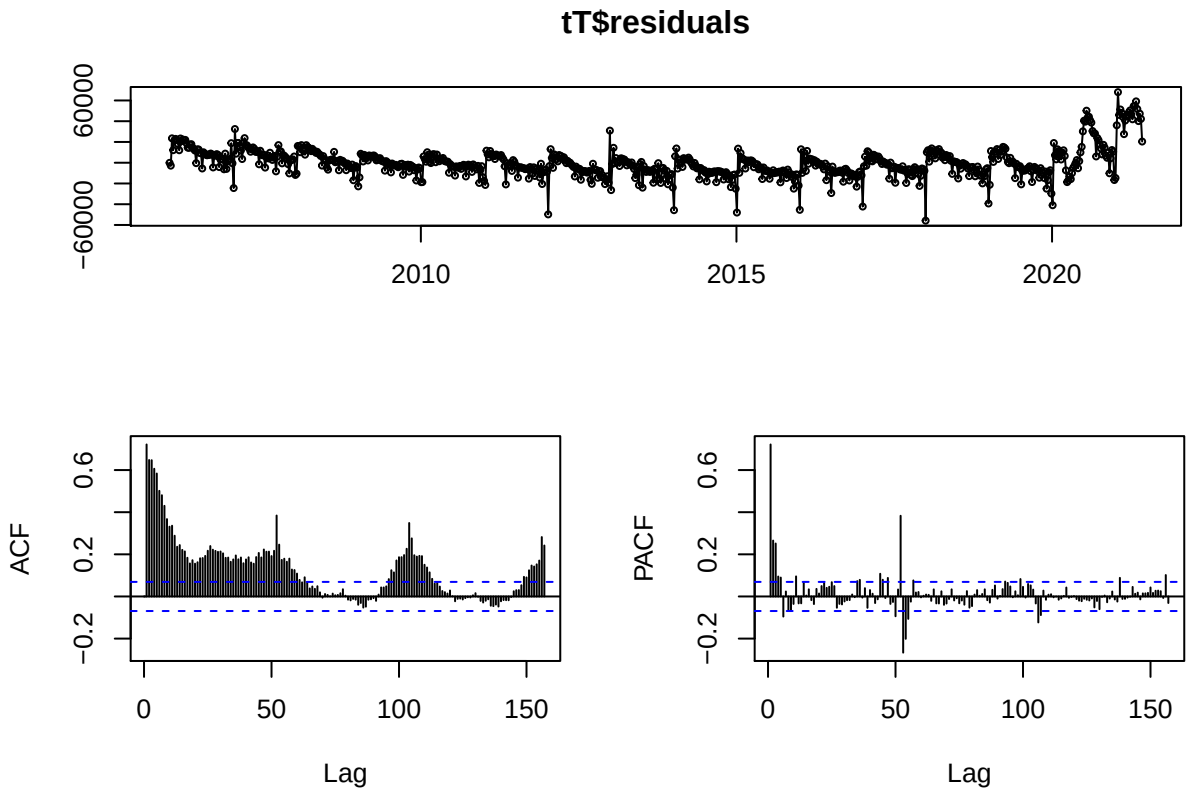
Model czerwony, uwzględniający sezonowość, został bardzo dobrze dopasowany do szeregu. Wręcz za dobrze (gdyż mogło dojść do przeuczenia), gdyż wektor reszt jest wektorem samych zer.

```
head(tTS$residuals)
```

```
## Time Series:
## Start = 2006.01916495551
## End = 2006.11498973306
## Frequency = 52.1785714285714
## [1] 0 0 0 0 0 0
```

Poniżej model uwzględniający wyłącznie trend liniowy. Sezonowość nadal występuje. Widać też niewielki trend po roku 2020.

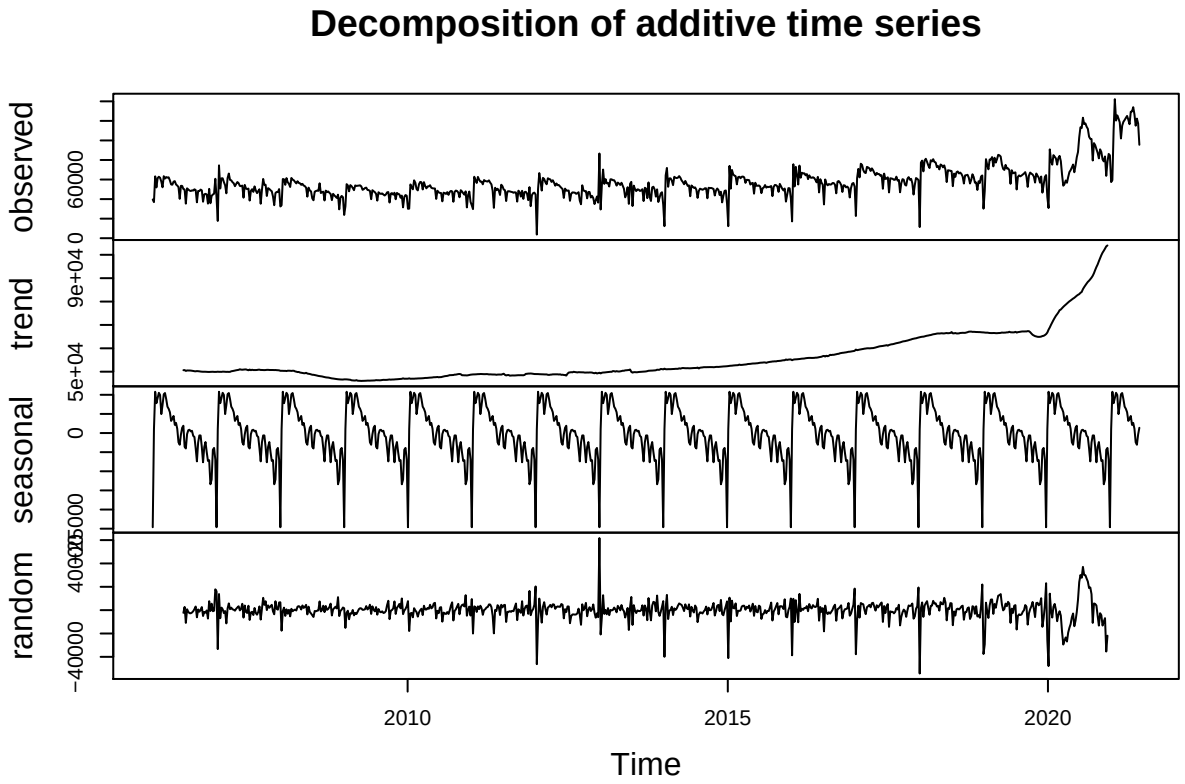
```
tsdisplay(tT$residuals)
```



1.3.2 Model addytywny

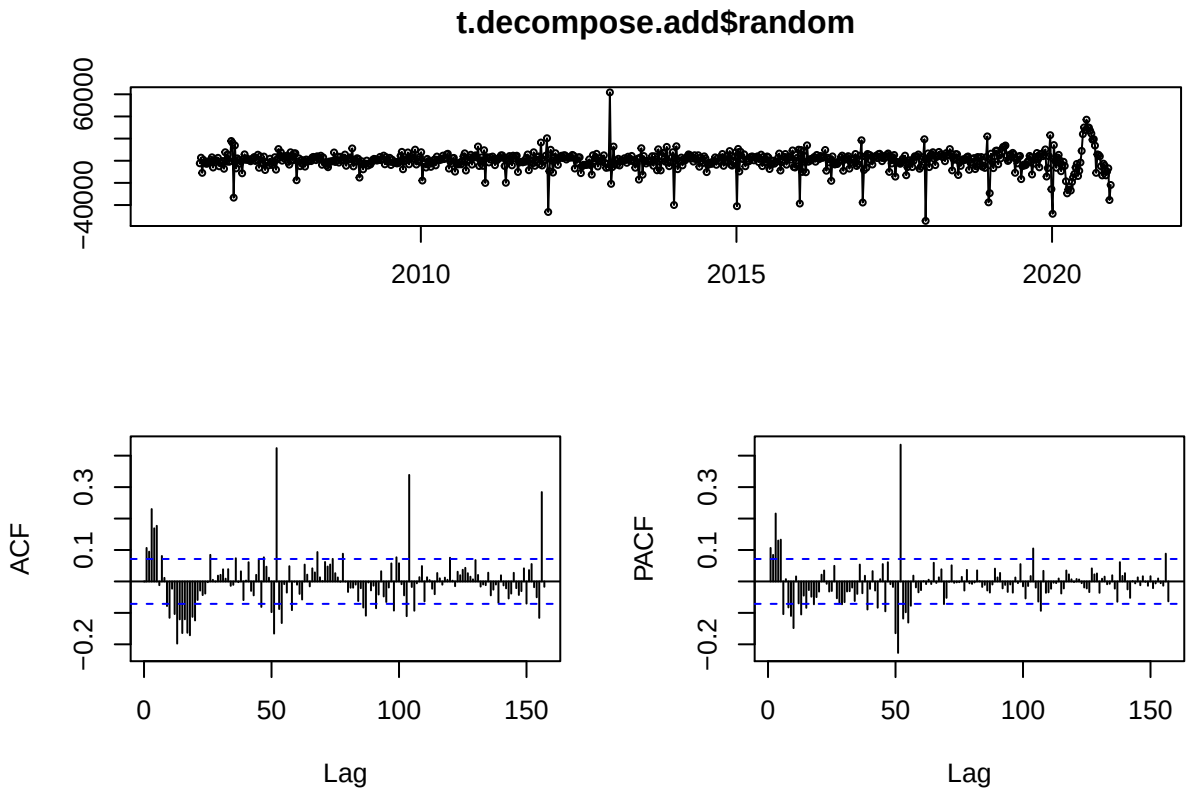
Ze względu na to , że wariancja sezonowa nie zmienia się w czasie (z wyjątkiem lat 2020 i w wzwyż), zastosowałem dekompozycję addytywną.

```
t.decompose.add <- decompose(t)
plot(t.decompose.add)
```



Szereg został rozłożony na swoje składowe, wyraźnie widać sezonowość. Trend najbardziej widoczny jest po roku 2015.

```
tsdisplay(t.decompose.add$random)
```

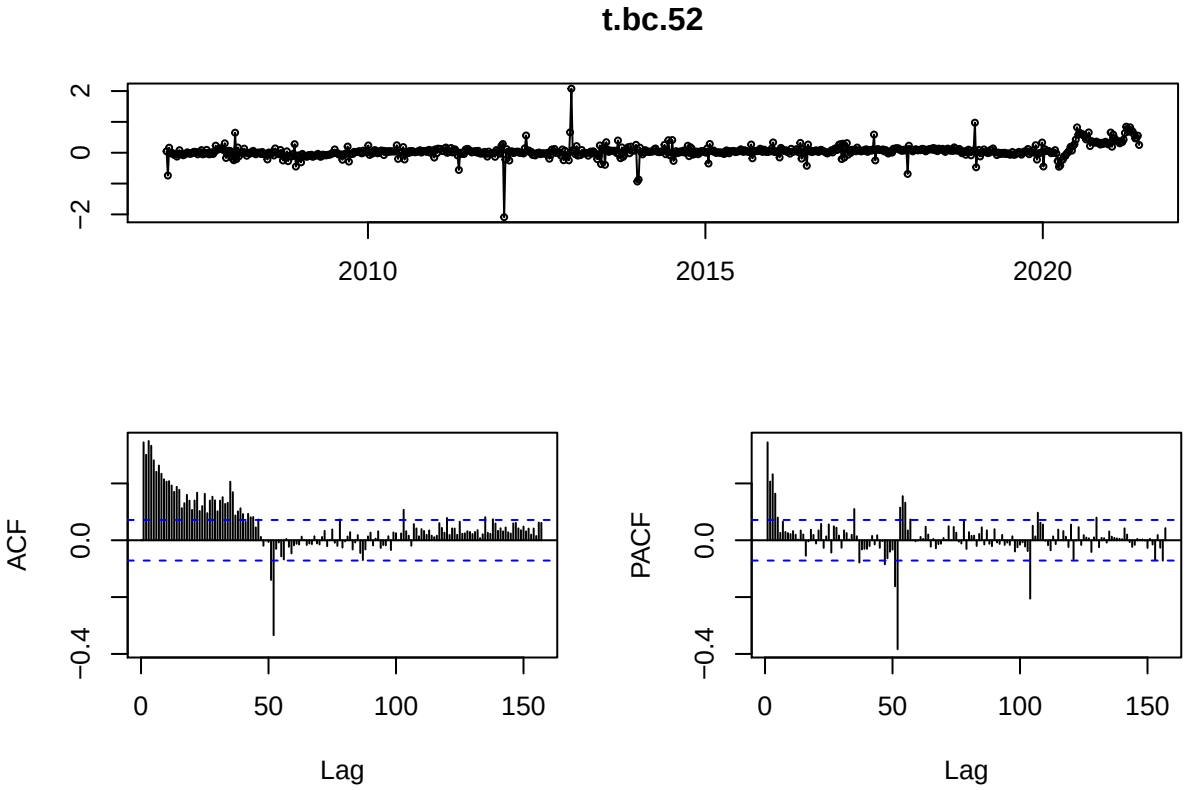


Z wykresów funkcji ACF i PACF odczytać możemy, że cała sezonowość nie została usunięta z szeregu(PACF posiada wartość odstającą ~52).

1.4 Eliminacja trendu i sezonowości

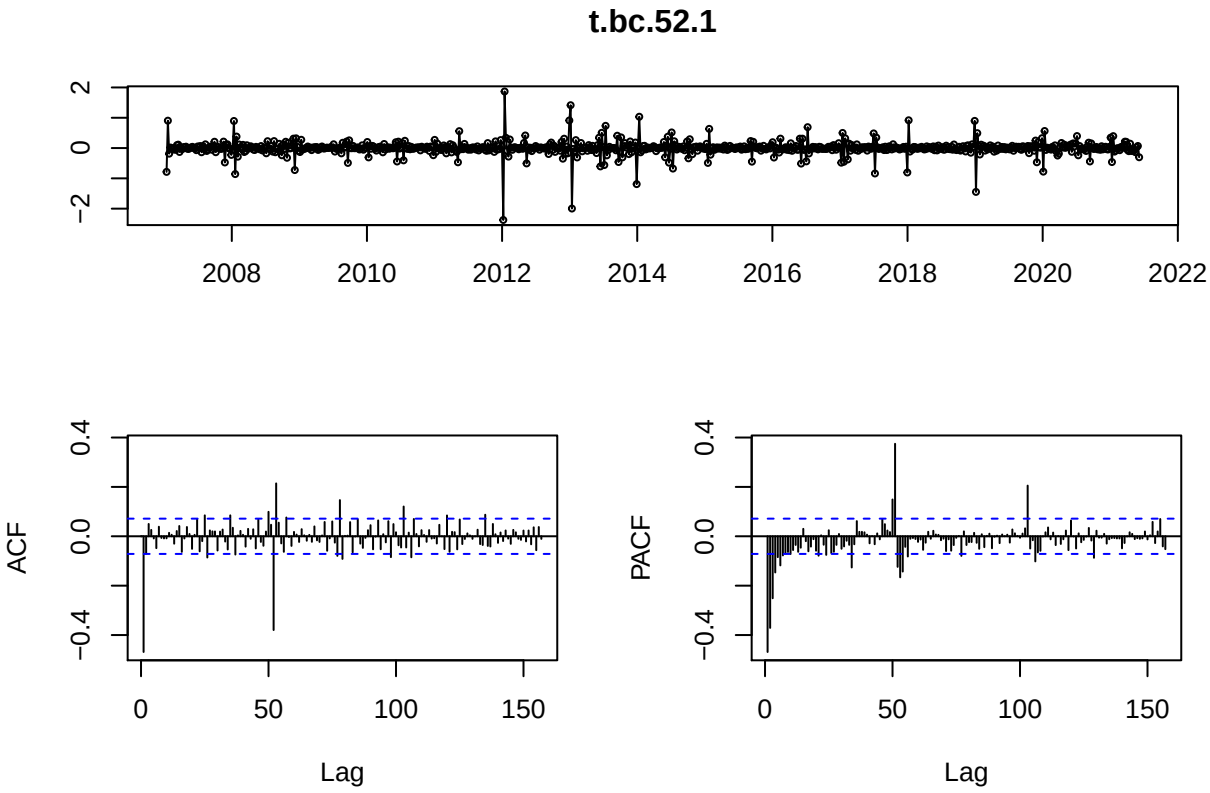
Z poprzednich wykresów wiem, że szereg charakteryzuje się wyraźnym trendem i sezonowością, którą należy wyeliminować. Dodatkowo, aby pozbyć się gwałtownej zmiany wariancji z początku roku 2020, zastosuję transformację logarytmiczną Boxa-Coxa.

```
t.bc <- BoxCox(t, lambda = 0)
t.bc.52 <- diff(t.bc, lag = 52)
tsdisplay(t.bc.52)
```



Po usunięciu sezonowości i zastosowaniu transformacji Boxa-Coxa, nadal pozostał silny trend - wykres funkcji ACF jest dodatni i stopniowo maleje.

```
t.bc.52.1 <- diff(t.bc.52, lag = 1)
tsdisplay(t.bc.52.1)
```



Szereg ten nie jest realizacją szumu białego. Widać to po znaczących wartościach odstających dla lag=52. Stacjonarność szeregu sprawdzę korzystając z biblioteki urca, dla ufności $\alpha = 0.05$. Zawiera ona test na stacjonarność szeregu: H_0 - szereg jest stacjonarny,wobec hipotezy alternatywnej: szereg nie jest stacjonarny.

```
library(urca)
t.bc.52.1 %>% ur.kpss() %>% summary()
```

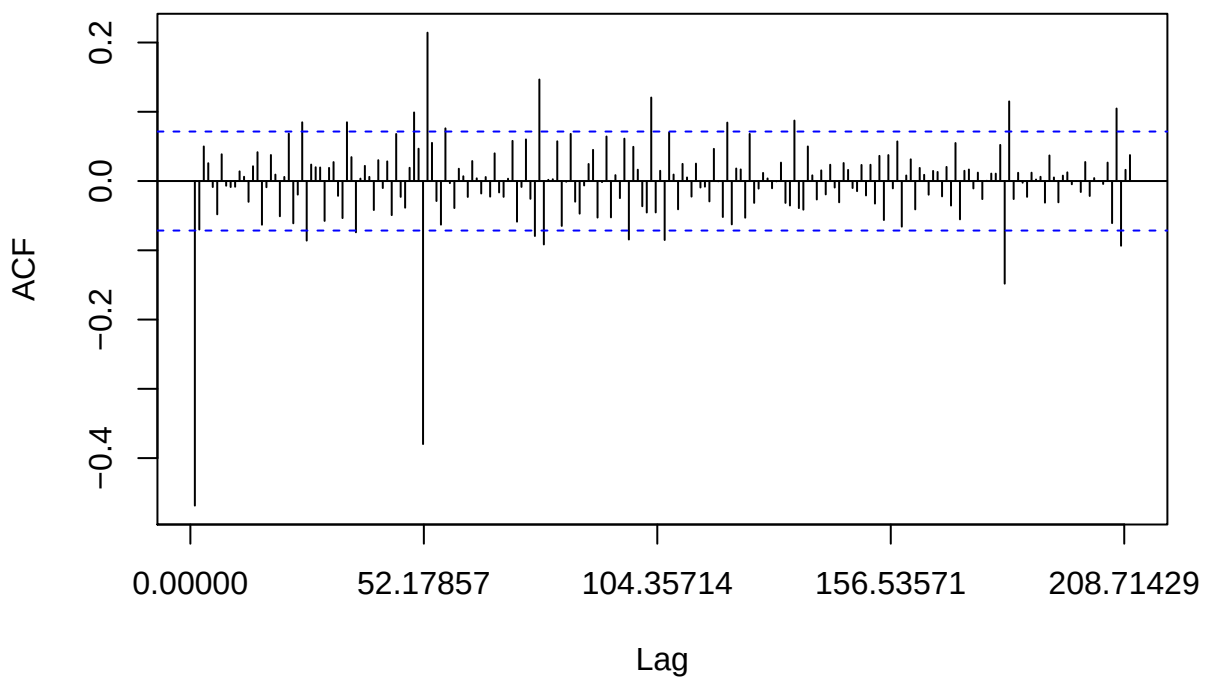
```
##
## ########################
## # KPSS Unit Root Test #
## ########################
##
## Test is of type: mu with 6 lags.
##
## Value of test-statistic is: 0.0072
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

Wartość statystyki jest bardzo mała, wynosi 0.0072, co jest poniżej wartości krytycznej dla zadanego poziomu ufności. Zatem brak podstaw do odrzucenia hipotezy o stacjonarności szeregu.

1.5 Wyznaczenie rzędu MA

Do wyznaczenia parametrów skorzystam z funkcji Acf. Rząd modelu dobiorę na podstawie wartości odstających.

```
Acf(t.bc.52.1, lag.max = 210)
```



Do wyboru mam rzędy MA równe:

```
t.bc.52.1.acf <- Acf(t.bc.52.1, plot = FALSE, lag.max = 210)
t.bc.52.1.acf$lag[which(abs(t.bc.52.1.acf$acf)>1.96/sqrt(t.bc.52.1.acf$n.used))] # Wszystkie lag poza p
```

```
##  [1]   0   1  25  26  35  37  50  52  53  57  77  78  79  98 103 106 120 135 182
## [20] 183 207 208
```

Obliczam współczynniki MA(52) i MA(26):

```
st <- t.bc.52.1 # szereg stacjonarny
st.ma52 <- Arima(st, order = c(0,0,52))
st.ma26 <- Arima(st, order = c(0,0,26))
```

```
st.ma26
```

```
## Series: st
## ARIMA(0,0,26) with non-zero mean
##
## Coefficients:
##           ma1      ma2     ma3     ma4      ma5      ma6     ma7      ma8
##       -0.8599  -0.0496  0.0758  0.0052  -0.0504  -0.0530  0.0479  -0.0241
## s.e.   0.0367   0.0495  0.0503  0.0499   0.0500   0.0498  0.0499   0.0503
##           ma9     ma10    ma11    ma12     ma13    ma14    ma15     ma16
##       -0.0158  -0.0104  0.0416  0.0012  -0.0417  0.0401  0.0056  -0.0946
## s.e.   0.0502   0.0500  0.0502  0.0500   0.0511  0.0555  0.0487   0.0517
##          ma17    ma18     ma19     ma20    ma21    ma22     ma23    ma24    ma25
##       0.0245  0.0530  -0.0132  -0.0528  0.0531  0.0544  -0.0855  0.0212  0.1067
## s.e.  0.0506  0.0523   0.0508   0.0483  0.0509  0.0481   0.0520  0.0490  0.0574
##          ma26   mean
##       -0.0465  9e-04
## s.e.   0.0377  9e-04
```

```
##
## sigma^2 estimated as 0.03392:  log likelihood=217.83
## AIC=-379.66   AICc=-377.41   BIC=-250.22
```

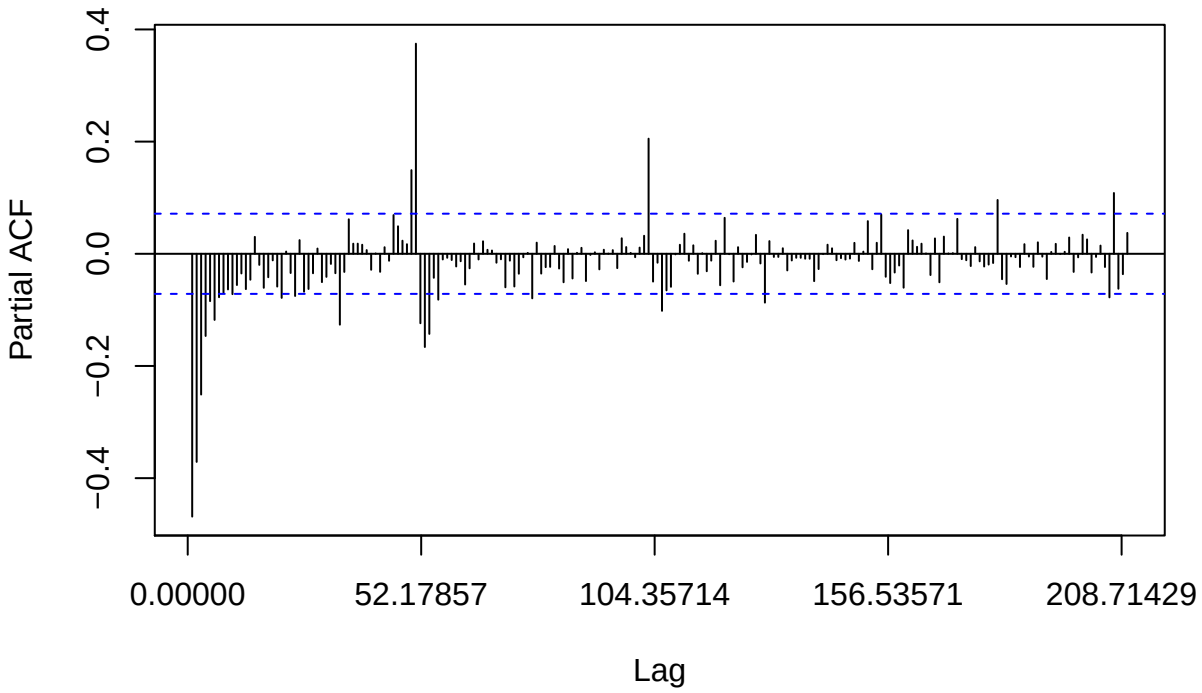
```
st.ma52
```

```
## Series: st
## ARIMA(0,0,52) with non-zero mean
##
## Coefficients:
##           ma1     ma2     ma3      ma4      ma5     ma6     ma7      ma8
##       -1.0682  0.1704  0.2471  -0.0864  -0.1146  0.0529  0.0265  -0.0606
## s.e.   0.0477  0.0638  0.0713   0.0681   0.0662  0.0642  0.0666   0.0643
##           ma9    ma10     ma11    ma12     ma13    ma14     ma15     ma16
##       -0.0424  0.0741  -0.0198  0.0000  -0.0387  0.0861  -0.0409  -0.0961
## s.e.   0.0654  0.0652   0.0639  0.0673   0.0586  0.0626   0.0645   0.0658
##          ma17     ma18     ma19     ma20    ma21     ma22     ma23    ma24
##       0.0900  -0.0054  -0.0694  -0.0106  0.1193  -0.0469  -0.1262  0.1011
## s.e.  0.0665   0.0623   0.0653   0.0632  0.0654   0.0635   0.0627  0.0632
##          ma25     ma26    ma27     ma28     ma29    ma30    ma31     ma32
##       0.1086  -0.2756  0.2672  -0.0979  -0.0778  0.0851  0.0785  -0.1424
## s.e.  0.0613   0.0717  0.0694   0.0642   0.0644  0.0646  0.0684   0.0656
##          ma33    ma34    ma35     ma36    ma37    ma38     ma39     ma40    ma41
##       0.0158  0.0892  0.0510  -0.1232  0.0684  0.0817  -0.0947  -0.0019  0.0150
## s.e.  0.0664  0.0662  0.0665   0.0723  0.0706  0.0709   0.0626   0.0591  0.0748
##          ma42     ma43    ma44    ma45    ma46     ma47    ma48    ma49     ma50
##       0.0705  -0.1183  0.0869  0.0039  0.0218  -0.1265  0.1748  0.0742  -0.2604
## s.e.  0.0660   0.0712  0.0708  0.0693  0.0705   0.0670  0.0705  0.0760   0.0838
##          ma51    ma52   mean
##       -0.2389  0.1238  3e-04
## s.e.   0.0675  0.0496  2e-04
##
## sigma^2 estimated as 0.0275:  log likelihood=291.81
## AIC=-475.62   AICc=-467.09   BIC=-225.99
```

1.6 Wyznaczenie rzędu AR

Do wyznaczenia parametrów skorzystam z funkcji Pacf. Rząd modelu dobiorę na podstawie wartości odstających.

```
Pacf(t.bc.52.1, lag.max = 210)
```



Do wyboru mam rzędy AR równe:

```
t.bc.52.1.pacf <- Pacf(t.bc.52.1, plot = FALSE, lag.max = 210)
t.bc.52.1.pacf$lag[which(abs(t.bc.52.1.pacf$acf)>1.96/sqrt(t.bc.52.1.pacf$n.used))] # Wszystkie lag spo
```

```
##  [1]   1   2   3   4   5   6   7  10  21  24  34  50  51  52  53  54  56  77 103
## [20] 106 129 181 206 207
```

1.6.1 AR(52) i AR(56)

Obliczam współczynniki AR(52), AR(56):

```
st.ar56.yw <-ar(st, order.max = 56, aic = FALSE, method = "yule-walker")
st.ar56.burg <- ar(st, order.max = 56, aic = FALSE, method ="burg")
st.ar52.yw <- ar(st, order.max = 52, aic = FALSE)
st.ar1.yw<- ar(st, order.max = 1, aic = FALSE)
```

Lista obliczonych współczynników:

```
st.ar56 <- Arima(st, order = c(56,0,0))
st.ar52 <- Arima(st, order = c(52,0,0), method = "CSS")
```

```
summary(st.ar56)
```

```
## Series: st
## ARIMA(56,0,0) with non-zero mean
##
## Coefficients:
##           ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##       -0.9487  -0.8261  -0.6259  -0.4941  -0.3921  -0.3649  -0.3333  -0.2995
## s.e.   0.0365   0.0504   0.0581   0.0612   0.0625   0.0644   0.0652   0.0657
##           ar9     ar10     ar11     ar12     ar13     ar14     ar15     ar16
```

```
##       -0.2853  -0.2868  -0.2854  -0.2554  -0.2689  -0.2555  -0.2307  -0.2286
## s.e.   0.0660   0.0664   0.0668   0.0674   0.0679   0.0685   0.0691   0.0696
##          ar17     ar18     ar19     ar20     ar21     ar22     ar23     ar24
##       -0.1952  -0.1663  -0.1577  -0.1866  -0.1817  -0.1472  -0.1583  -0.1486
## s.e.   0.0701   0.0702   0.0703   0.0703   0.0703   0.0704   0.0705   0.0707
##          ar25     ar26     ar27     ar28     ar29     ar30     ar31     ar32
##       -0.0854  -0.1497  -0.0954  -0.0649  -0.0463  -0.0606  -0.0426  -0.0417
## s.e.   0.0709   0.0714   0.0722   0.0723   0.0722   0.0721   0.0718   0.0717
##          ar33    ar34    ar35    ar36    ar37    ar38    ar39    ar40    ar41
##       -0.0264  0.0053  0.1125  0.1557  0.1264  0.1209  0.1003  0.0985  0.0918
## s.e.   0.0715  0.0712  0.0709  0.0708  0.0707  0.0706  0.0705  0.0703  0.0700
##          ar42    ar43    ar44    ar45    ar46    ar47    ar48    ar49    ar50
##       0.1173  0.1256  0.1805  0.1874  0.2154  0.1898  0.1862  0.1981  0.1814
## s.e.  0.0696  0.0692  0.0688  0.0687  0.0685  0.0685  0.0684  0.0686  0.0685
##          ar51     ar52     ar53     ar54     ar55     ar56   mean
##       0.0266  -0.4229  -0.3844  -0.2655  -0.1342  -0.0910  7e-04
## s.e.  0.0678   0.0660   0.0649   0.0613   0.0523   0.0375  7e-04
##
## sigma^2 estimated as 0.02585:  log likelihood=328.37
## AIC=-540.75   AICc=-530.87   BIC=-272.63
##
## Training set error measures:
##                         ME      RMSE      MAE      MPE     MAPE      MASE
## Training set 0.0001153186 0.1545722 0.086313 402.8552 762.8487 0.4199214
##                       ACF1
## Training set -0.0006334146
```

```
summary(st.ar52)
```

```
## Series: st
## ARIMA(52,0,0) with non-zero mean
##
## Coefficients:
##           ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##       -0.9008  -0.8147  -0.6828  -0.5902  -0.5370  -0.5194  -0.4808  -0.4373
## s.e.   0.0364   0.0474   0.0534   0.0568   0.0591   0.0607   0.0618   0.0630
##           ar9     ar10     ar11     ar12     ar13     ar14     ar15     ar16
##       -0.4041  -0.3911  -0.3687  -0.3261  -0.3277  -0.3105  -0.2875  -0.2963
## s.e.   0.0640   0.0651   0.0664   0.0677   0.0686   0.0694   0.0700   0.0703
##          ar17     ar18     ar19     ar20     ar21     ar22     ar23     ar24
##       -0.2752  -0.2521  -0.2263  -0.2368  -0.2145  -0.1683  -0.1744  -0.1576
## s.e.   0.0705   0.0708   0.0713   0.0717   0.0722   0.0726   0.0729   0.0732
##          ar25     ar26     ar27     ar28     ar29     ar30    ar31    ar32
##       -0.0843  -0.1392  -0.0691  -0.0352  -0.0087  -0.0041  0.0134  0.0270
## s.e.   0.0734   0.0735   0.0736   0.0736   0.0735   0.0733  0.0732  0.0728
##          ar33    ar34    ar35    ar36    ar37    ar38    ar39    ar40    ar41
##       0.0502  0.0849  0.2011  0.2521  0.2314  0.2297  0.2147  0.2248  0.2254
## s.e.  0.0724  0.0721  0.0717  0.0714  0.0713  0.0710  0.0707  0.0701  0.0696
##          ar42    ar43    ar44    ar45    ar46    ar47    ar48    ar49    ar50
##       0.2659  0.2803  0.3539  0.3631  0.4054  0.3998  0.4161  0.4537  0.4659
## s.e.  0.0687  0.0679  0.0669  0.0661  0.0649  0.0636  0.0618  0.0595  0.0557
##          ar51     ar52    mean
##       0.3314  -0.0879  0.0007
## s.e.  0.0488   0.0368  0.0011
##
```

```
## sigma^2 estimated as 0.0259:  part log likelihood=307.25
##
## Training set error measures:
##                         ME      RMSE        MAE      MPE     MAPE      MASE
## Training set -4.598934e-06 0.1551525 0.08462721 587.8179 918.4954 0.4117199
##                       ACF1
## Training set -0.005745215
```

Współczynniki dla AR(56) i AR(52) są podobne.

1.6.2 auto.arima

```
au <- auto.arima(st)
```

```
summary(au)
```

```
## Series: st
## ARIMA(1,0,0)(1,0,0)[52] with zero mean
##
## Coefficients:
##           ar1     sar1
##       -0.5210  -0.4427
## s.e.   0.0314   0.0322
##
## sigma^2 estimated as 0.03631:  log likelihood=174.79
## AIC=-343.58   AICc=-343.54   BIC=-329.71
##
## Training set error measures:
##                        ME      RMSE        MAE      MPE     MAPE      MASE
## Training set 0.001074207 0.1903062 0.09991121 263.6516 471.3774 0.4860781
##                      ACF1
## Training set -0.2010561
```

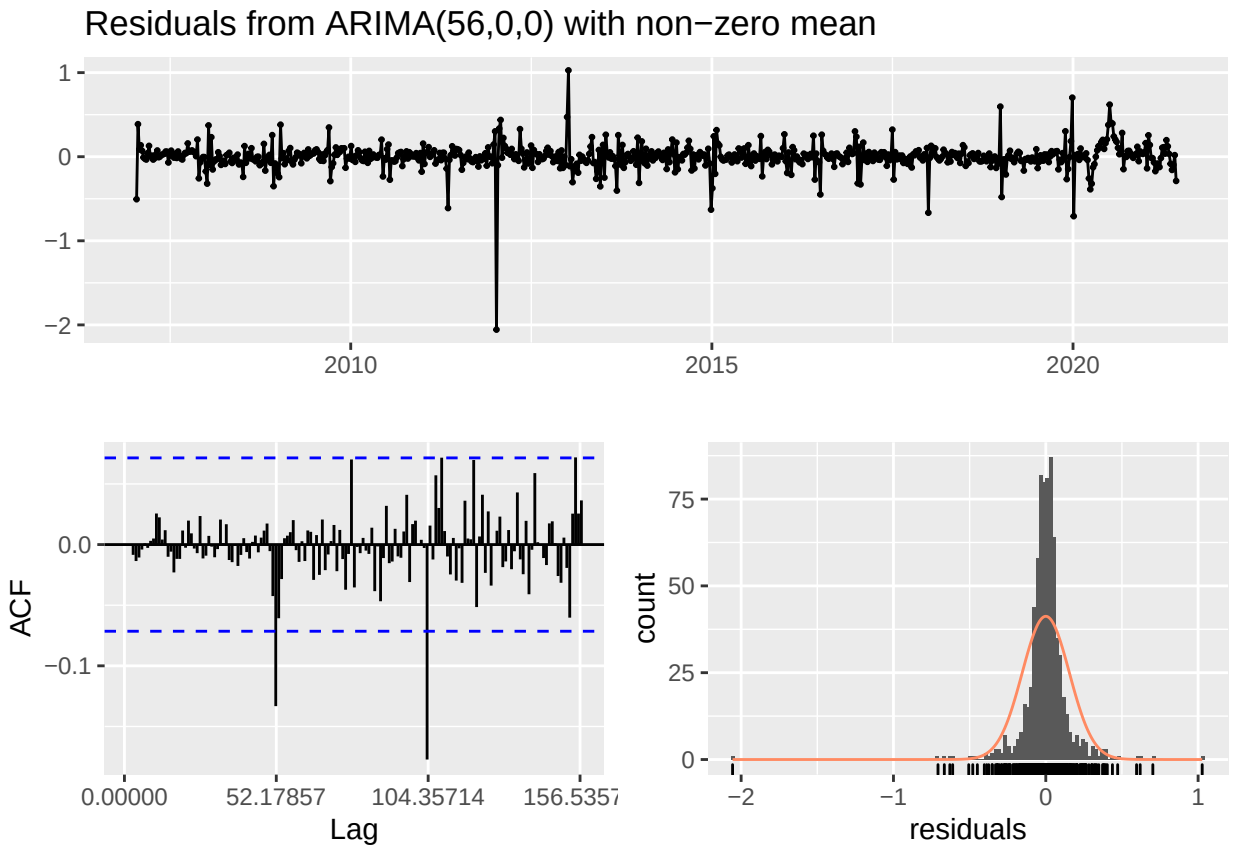
1.7 Porównanie analizowanych modeli

Wszystkie modele korzystały z transformacji Boxa-Coxa więc mogę je porównywać między sobą.

```
# ARIMA(0,0,26)              AIC=-379.66   AICc=-377.41   BIC=-250.22
# ARIMA(0,0,52)              AIC=-475.62   AICc=-467.09   BIC=-225.99
# ARIMA(56,0,0)              AIC=-540.75 + AICc=-530.87 + BIC=-272.63
# ARIMA(1,0,0)               AIC=-181.41   AICc=-181.38   BIC=-167.54
# ARIMA(1,0,0)(1,0,0)[52]    AIC=-343.58   AICc=-343.54   BIC=-329.71  +
```

Ze wszystkich modeli, najlepszym wydaje się ARIMA(56,0,0). Pomimo dużej ilości, parametrów jako jedyny przechodzi test Ljung-Boxa (dla $\alpha = 0.05$). Analiza reszt znajduje się na wykresach poniżej.

```
checkresiduals(st.ar56)
```

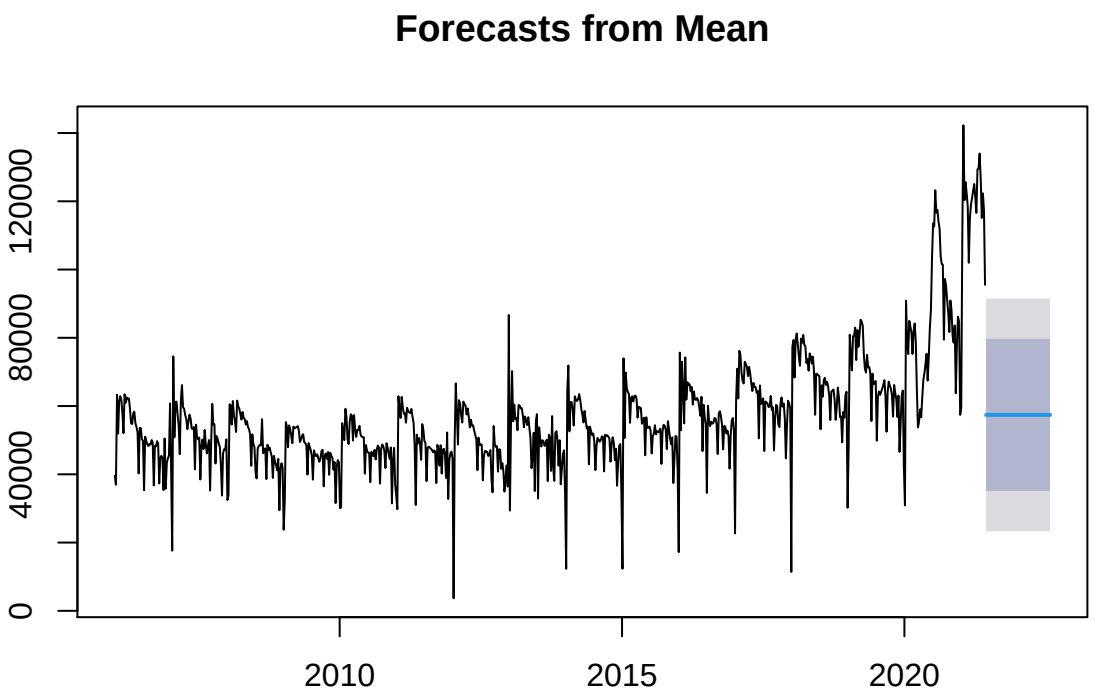



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(56,0,0) with non-zero mean
## Q* = 70.555, df = 47.357, p-value = 0.01601
##
## Model df: 57.   Total lags used: 104.357142857143
```

1.8 Prognozowanie

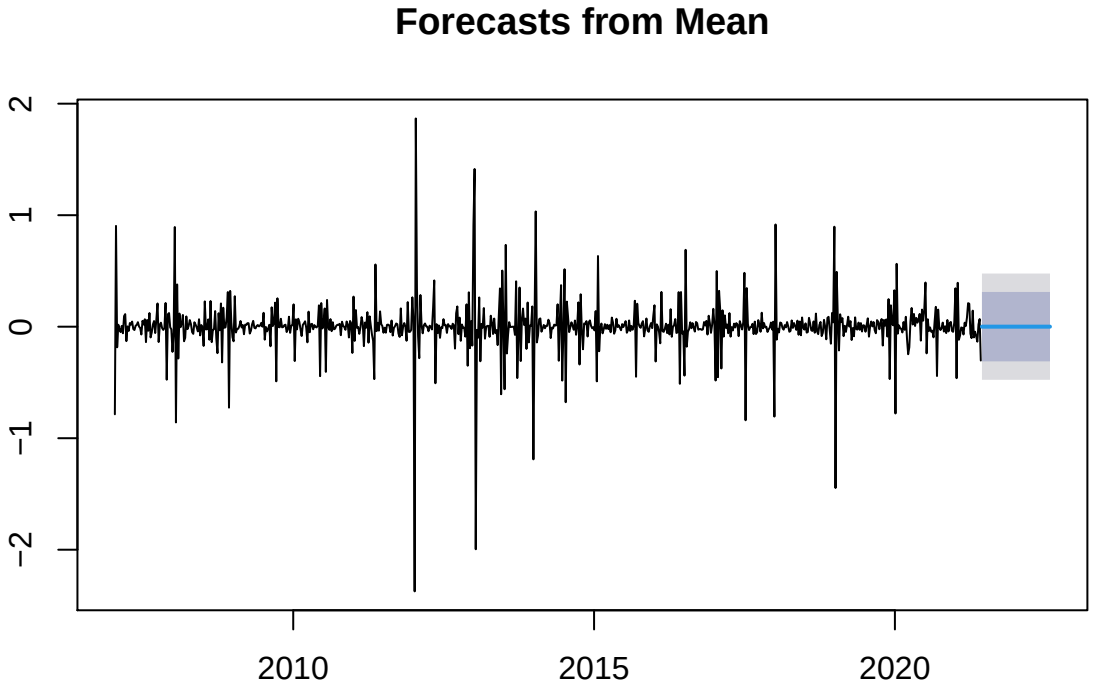
1.8.1 Prognozowanie naiwne metodą średniej

```
t.meanf <- meanf(t, h = 60)
plot(t.meanf)
```



Prognozowanie naiwne metodą średniej nie daje dobrych rezultatów, może być to spowodowane tym iż szereg ten zawiera trend i sezonowość. Prognoza dla szeregu bez trendu i sezonowości:

```
st.meanf <- meanf(st, h = 60)
plot(st.meanf)
```



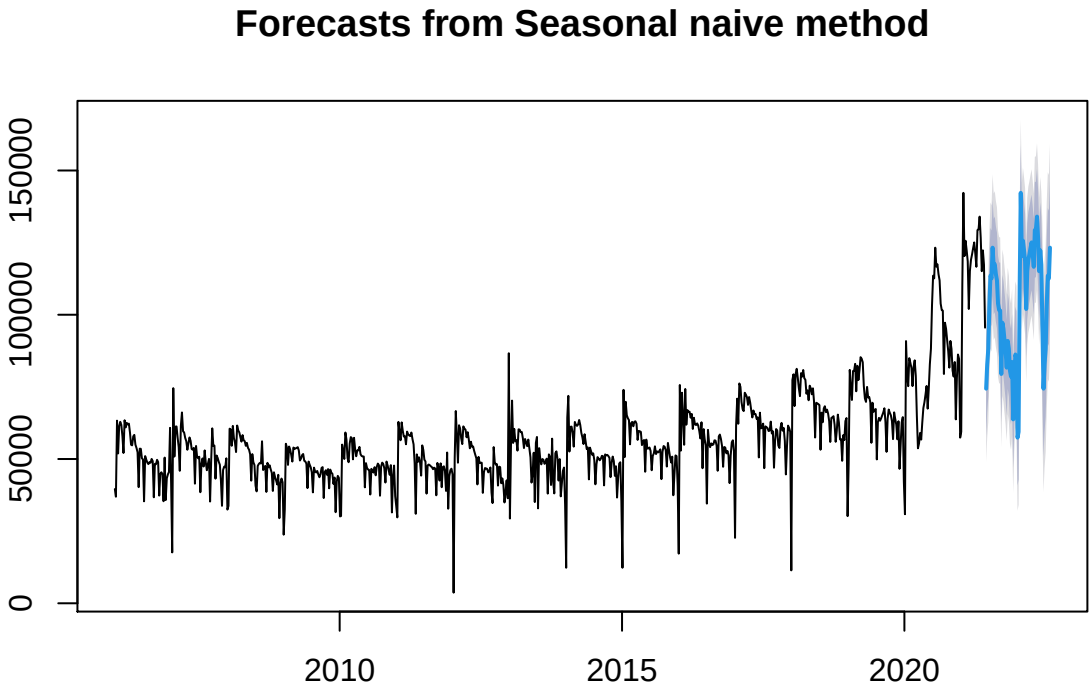
Prognoza ta jest dużo lepsza. Dodając trend i sezonowość moglibyśmy uzyskać nią lepsze przewidywania, niż za pierwszym razem.

1.8.2 Prognozowanie naiwne sezonowe

```
t.snaive <- snaive(t, h = 60)
```

```
## Warning in lag.default(y, -lag): 'k' is not an integer
```

```
plot(t.snaive)
```



Prognoza naiwna sezonowa daje na pierwszy rzut oka najlepsze rezultaty. Uwzględnia ona silną sezonowość szeregu oraz to że w poprzednich latach składowa trendu była dużo większa, jednak nie uwzględnia ona przyszłego wzrostu trendu.

2 - Index cen nieruchomości