Analiza szeregów czasowych

Krawiec Piotr

18/06/2021

Streszczenie

Celem pracy jest analiza dwóch szeregów czasowych z wykorzystaniem języka R. Pierwszego zawierającego silną sezonowość, drugiego silny trend. Szeregi poddane dekompozycji, usunięciu trendu i sezonowości. Zbadana i potwierdzona została stacjonarność reszt obu szeregów, a ostatecznie zostały modelowane modelem ARIMA. Przewidywania dotyczące dalszego ich przebiegu zostały stworzone z pomocą metod naiwnych uwzględniających dryf, a także sezonowość. Analiza reszt obu szeregów potwierdziła, że stanowią one realizację szumu białego.

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 pracodawca, korporacja, 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
```

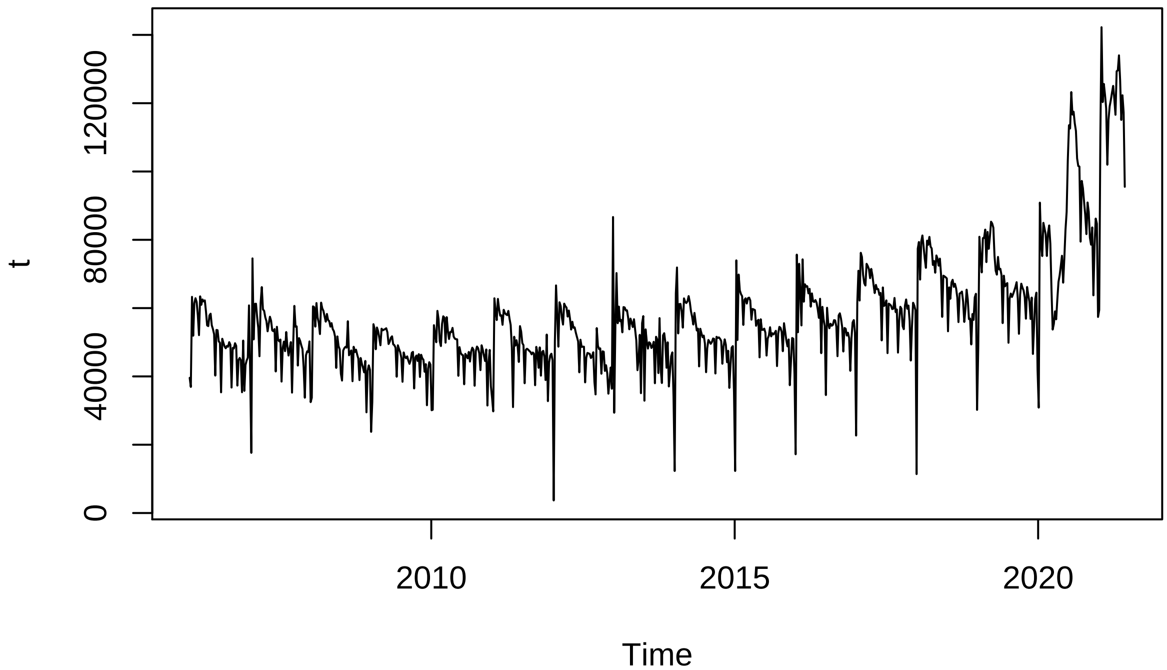
1.2 Główne cechy analizowanych danych

Tak prezentuje się wykres ilości wniosków w czasie:

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

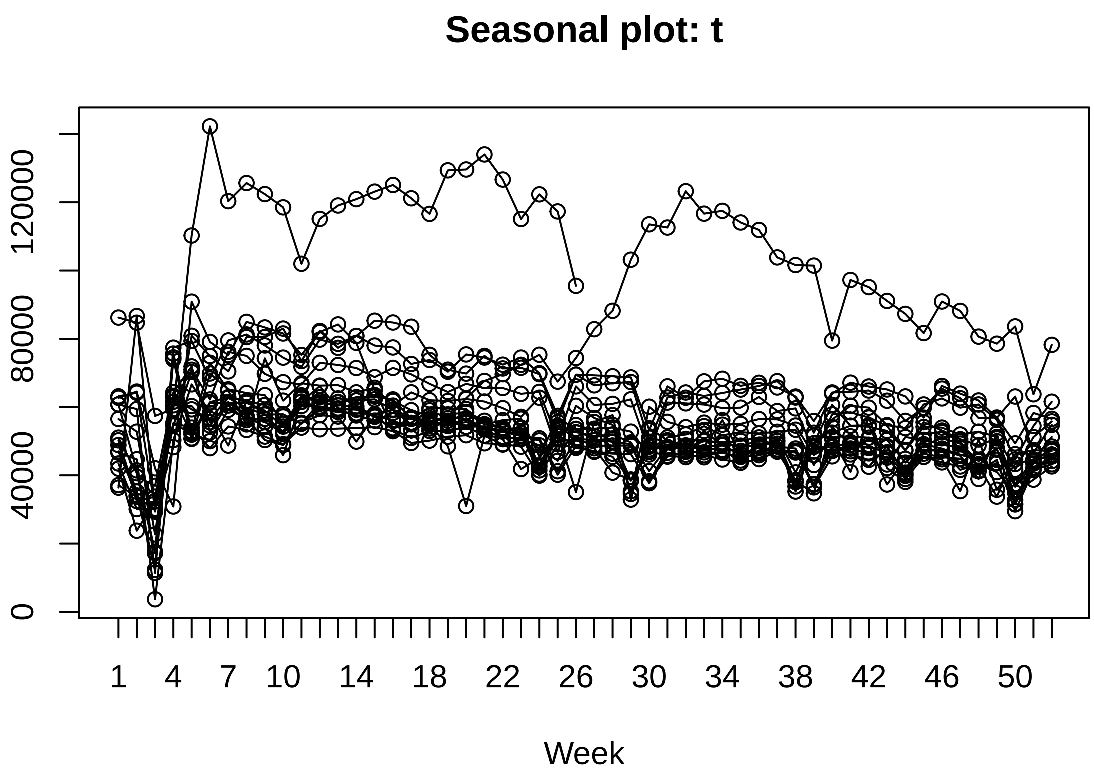
```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

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



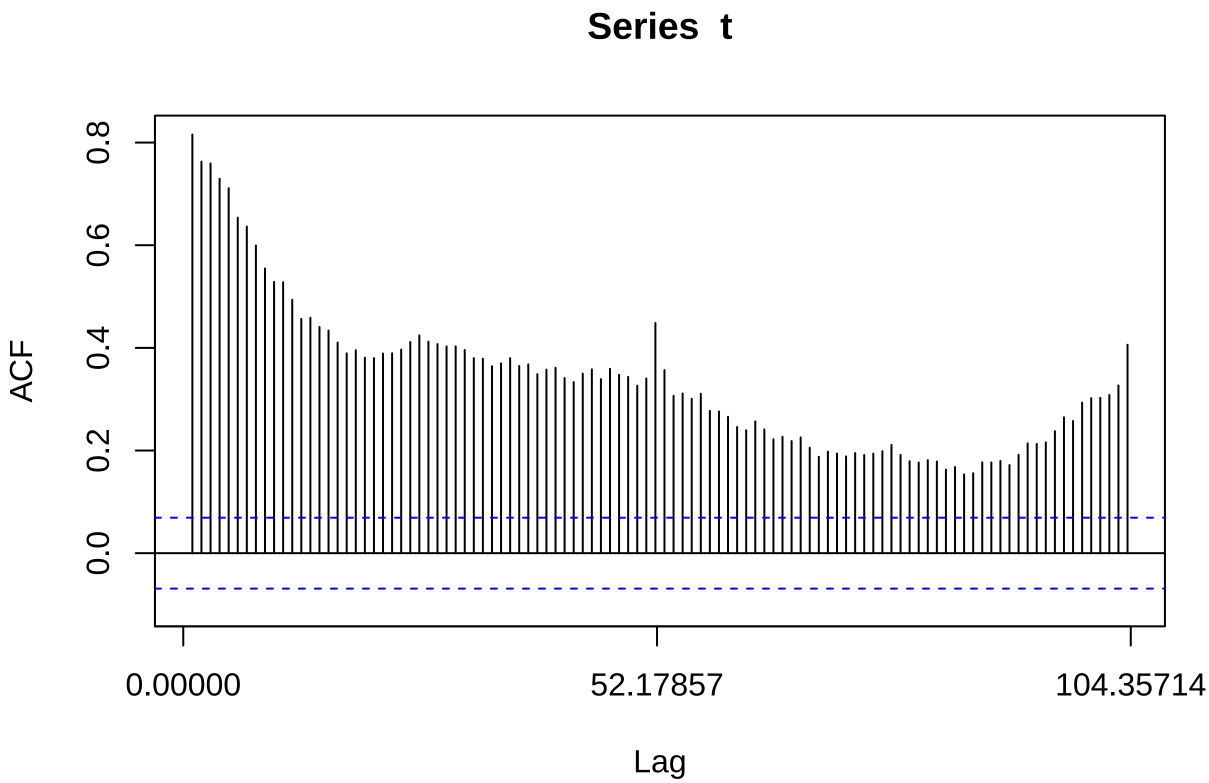
Z wykresu wywnioskować możemy że szereg ten posiada dużą sezonowość, pojawia się tu charakterystyczny wzorzec (odstające szpilki). Widać także niewielki dodatni trend, który gwałtownie rośnie na początku roku 2020.

```
seasonplot(t)
```



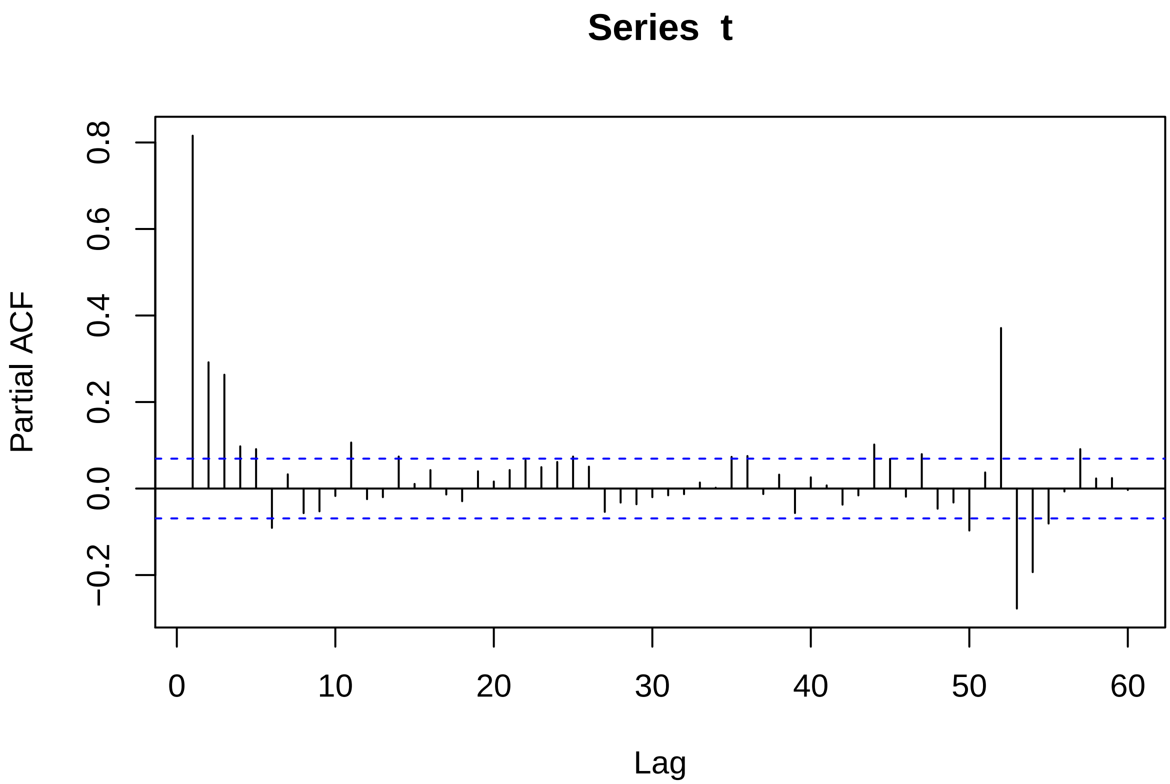
Porównując kolejne roczne sezony między sobą, sezonowość widać jeszcze dokładniej. Pojawia się też rok 2020, który znacznie odstaje wartościami, lecz kształtem nadal przypomina poprzednie sezony.

```
Acf(t)
```



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

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



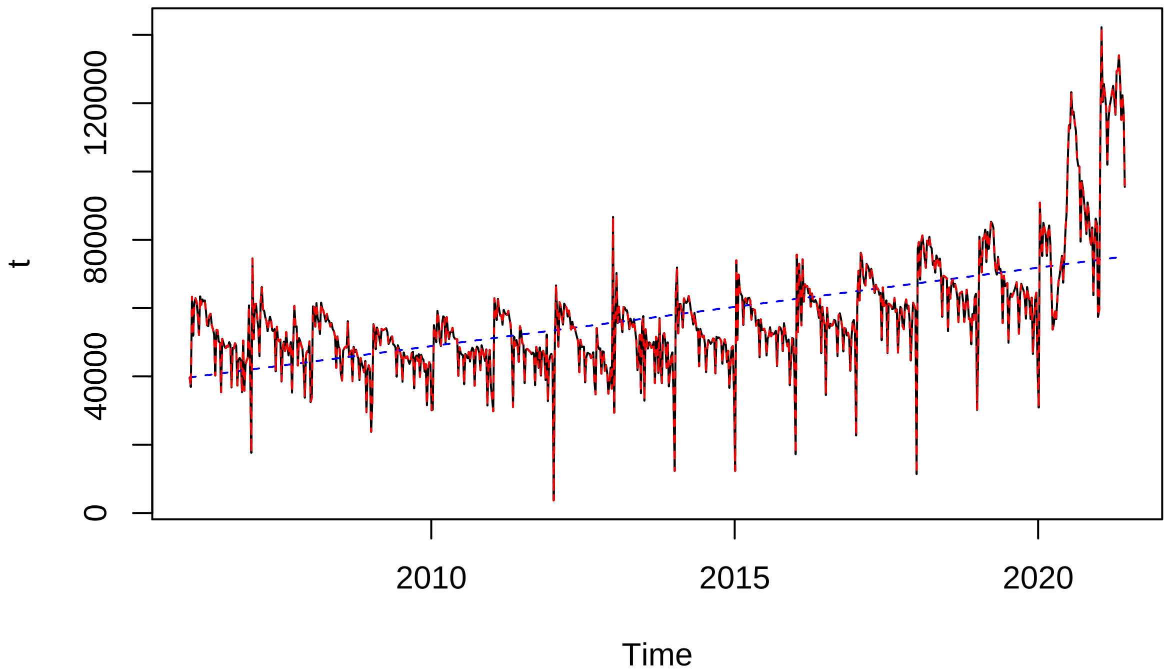
Na wykresie pojawia się wartość znacząca przy Lag=52, ponieważ dane są tygodniowe oznacza to korelację z danymi z poprzednich lat.

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)
```



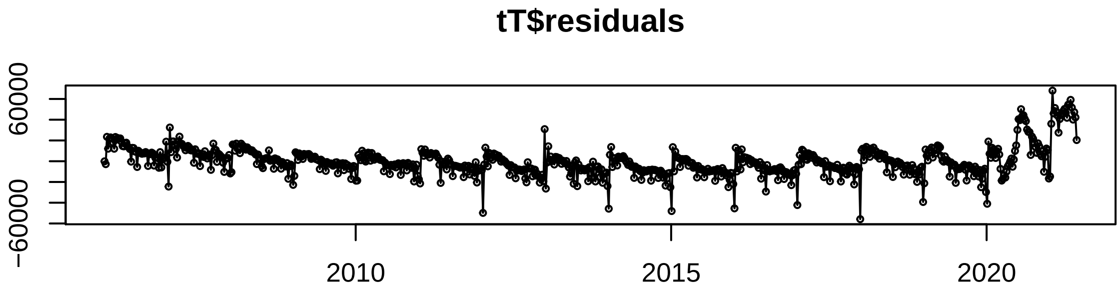
Model czerwony, uwzględniający sezonowość, został bardzo dobrze dopasowany do szeregu. Wręcz za dobrze (gdyż mogło dojść do przeuczenia), gdyż wektor reszt jest wektorem samych zer.

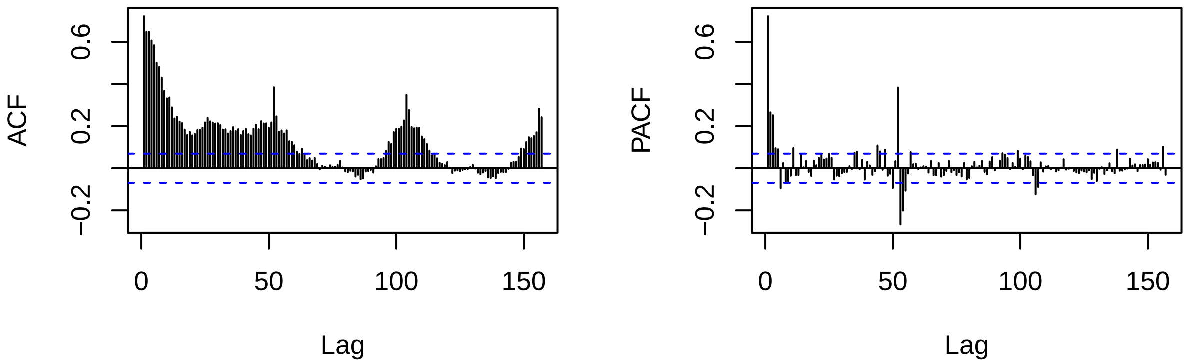
```
head(tTS$residuals)
```

```
## 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)
```

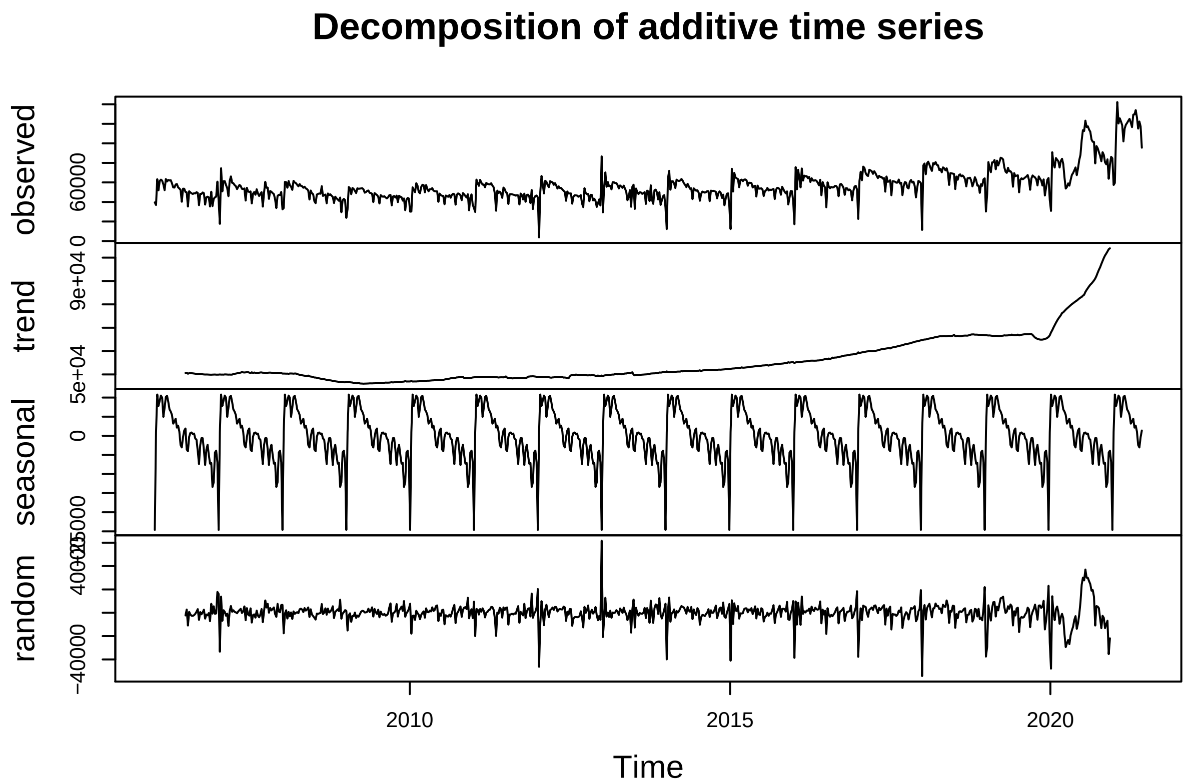




1.3.2 Model addytywny

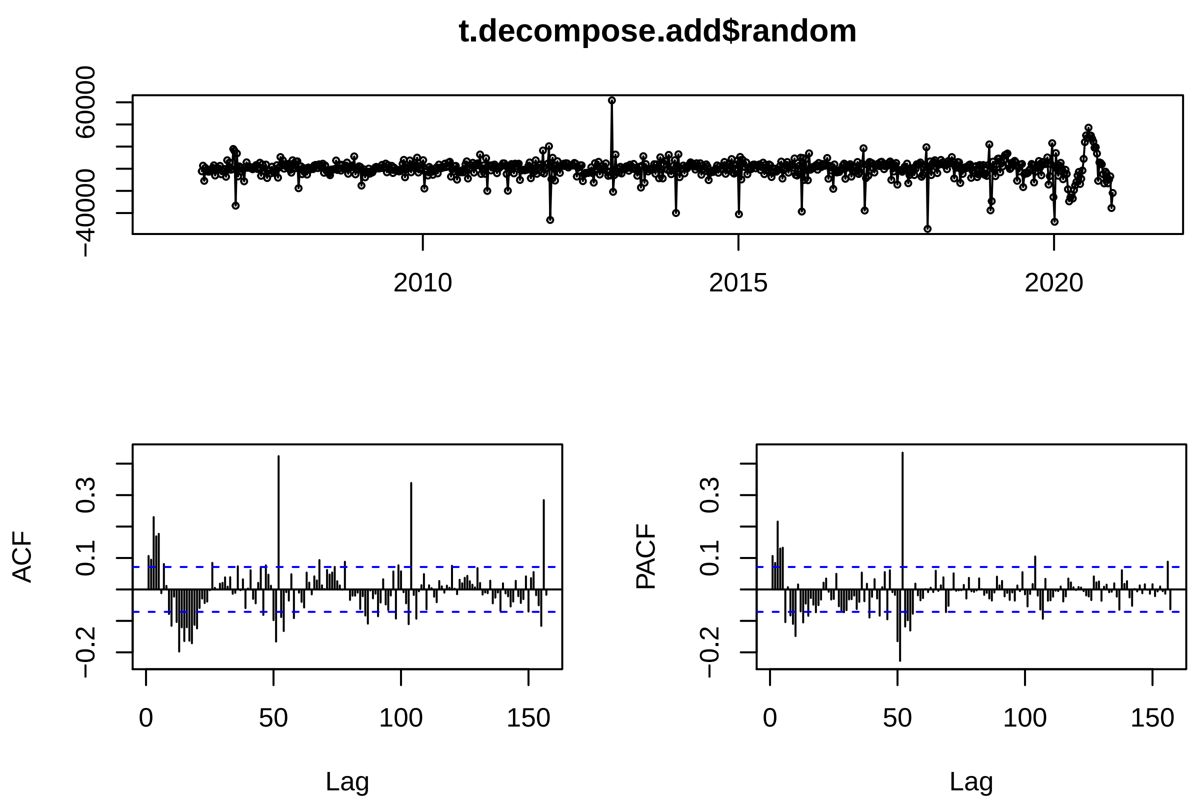
Ze względu na to , że wariancja sezonowa nie zmienia się w czasie (z wyjątkiem lat 2020 i w wzwyż), zastosowałem dekompozycję addytywną.

```
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)
```

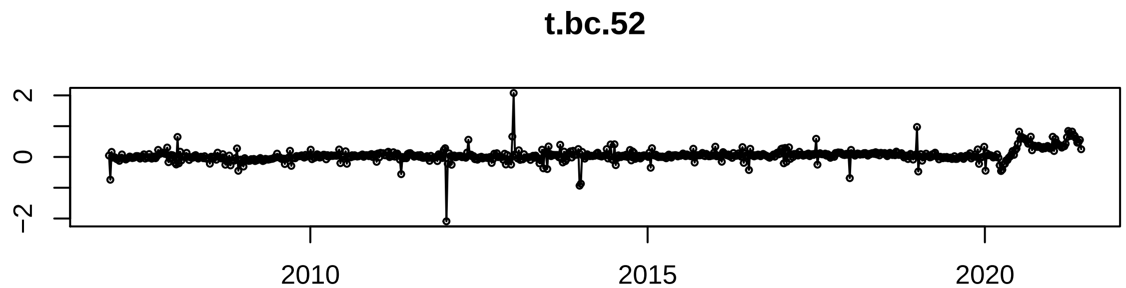


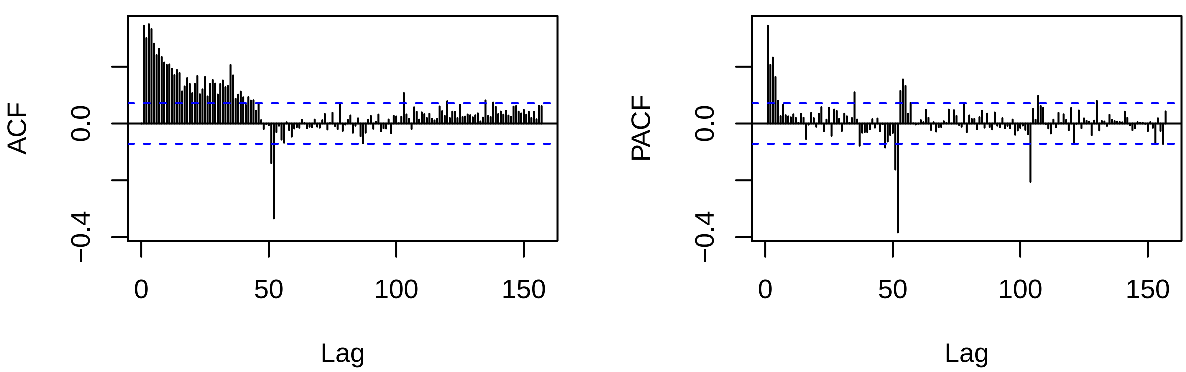
Z wykresów funkcji ACF i PACF odczytać możemy, że cała sezonowość nie została usunięta z szeregu(PACF posiada wartość odstającą ~52).

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.

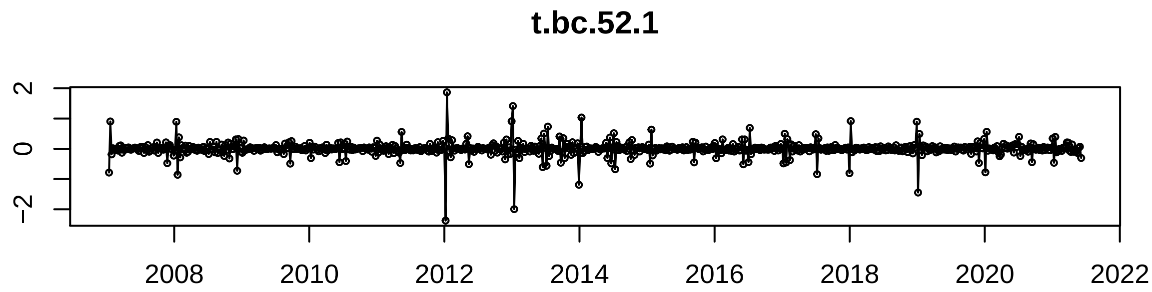
```
t.bc <- BoxCox(t, lambda = 0)
t.bc.52 <- diff(t.bc, lag = 52)
tsdisplay(t.bc.52)
```

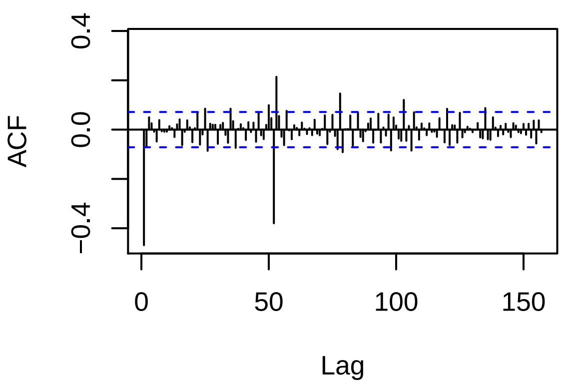


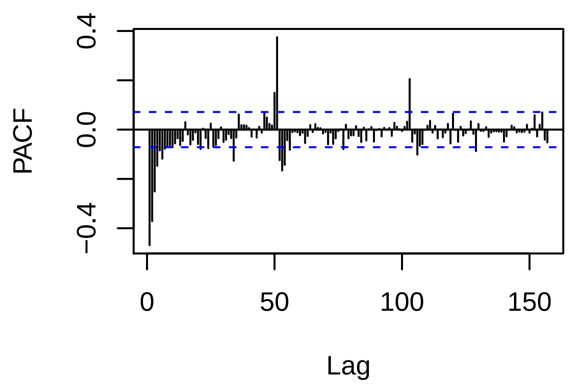


Po usunięciu sezonowości i zastosowaniu transformacji Boxa-Coxa, nadal pozostał silny trend - wykres funkcji ACF jest dodatni i stopniowo maleje.

```
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()
```

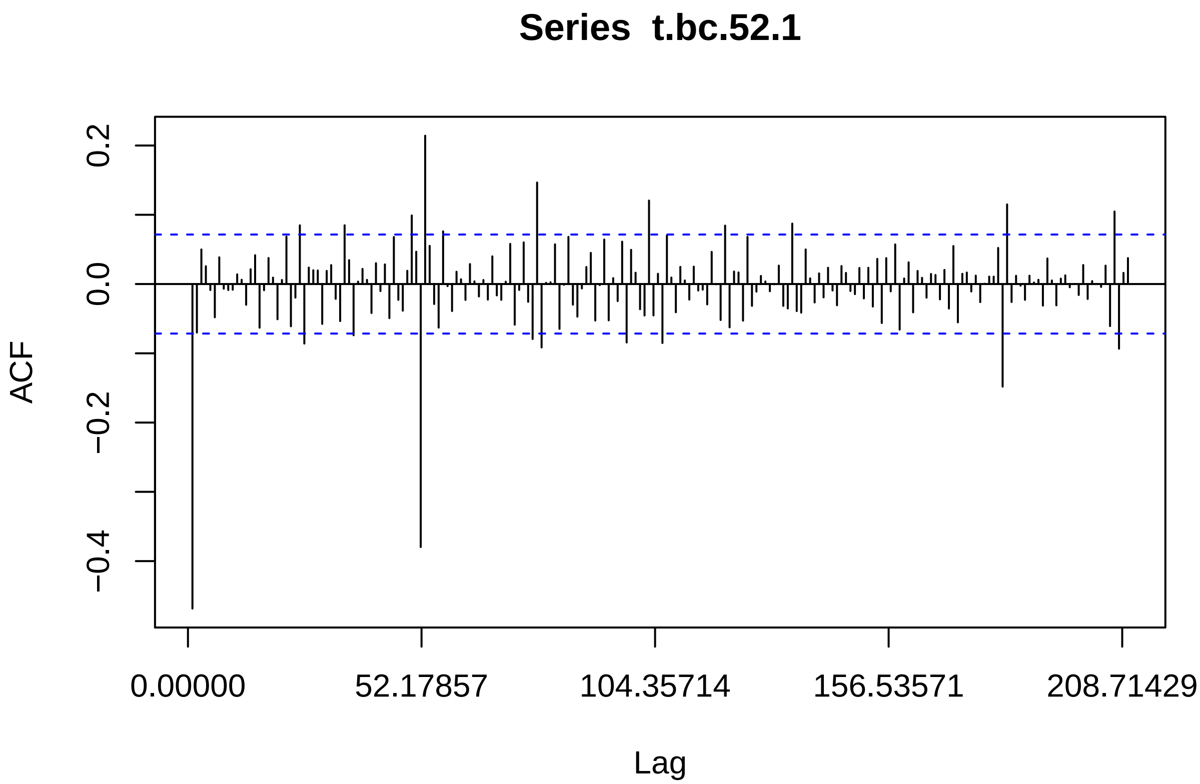
```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 6 lags.
##
## Value of test-statistic is: 0.0072
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739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

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))
```

Oto część obliczonych współczynników dla modeli:

```
c(st.ma26$aic, st.ma26$aicc, st.ma26$bic)
```

```
## [1] -379.6606 -377.4144 -250.2239
```

```
st.ma26$coef[1:5]
```

```
##          ma1          ma2          ma3          ma4          ma5
## -0.859902576 -0.049606422  0.075821570  0.005219935 -0.050431425
```

```
c(st.ma52$aic, st.ma52$aicc, st.ma52$bic)
```

```
## [1] -475.6156 -467.0934 -225.9879
```

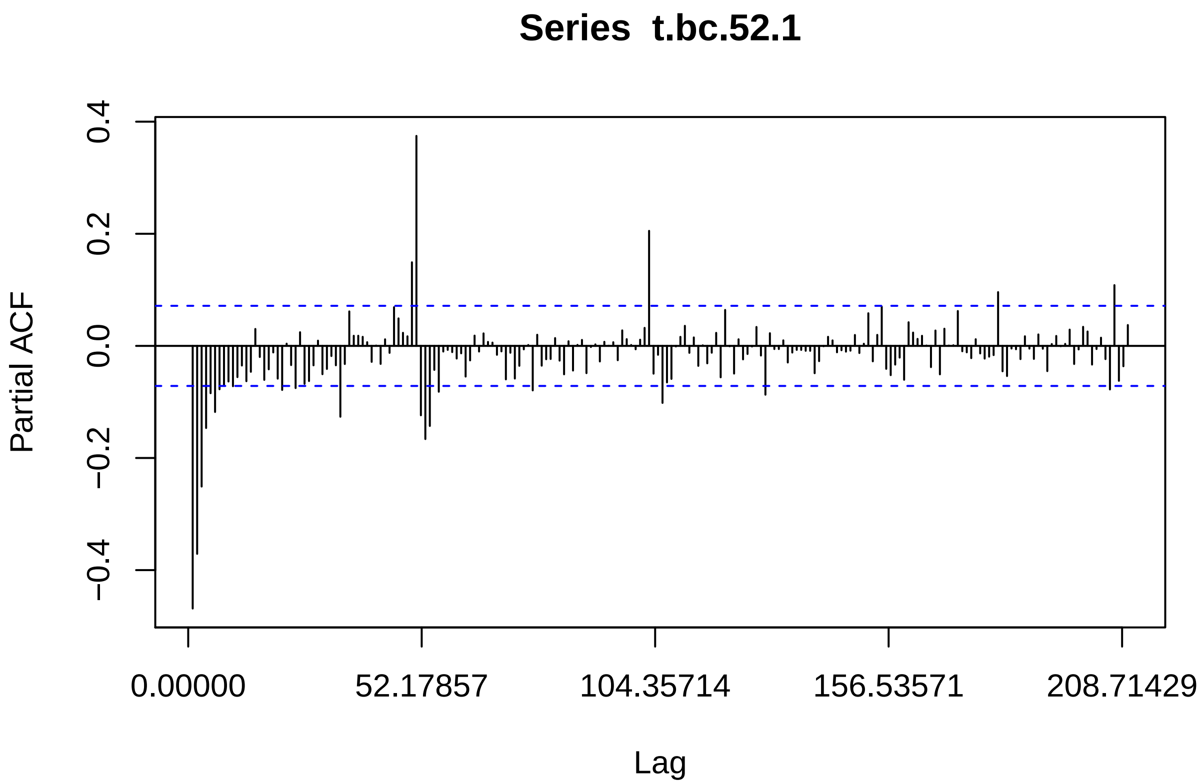
```
st.ma52$coef[1:5]
```

```
##         ma1         ma2         ma3         ma4         ma5
## -1.06816716  0.17039435  0.24707782 -0.08641077 -0.11456170
```

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
```

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)
```

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

Współczynniki, aic, aicc oraz bic:

```
c(st.ar56$aic, st.ar56$aicc, st.ar56$bic)
```

```
## [1] -540.7466 -530.8707 -272.6279
```

```
st.ar56$coef[1:10]
```

```
##         ar1         ar2         ar3         ar4         ar5         ar6         ar7
```

```
## -0.9487120 -0.8260767 -0.6259190 -0.4940899 -0.3920653 -0.3648660 -0.3333275
##        ar8        ar9       ar10
## -0.2994878 -0.2852825 -0.2868088
```

```
c(st.ar52$aic, st.ar52$aicc, st.ar52$bic)
```

```
## [1] NA NA NA
```

```
st.ar52$coef[1:10]
```

```
##        ar1        ar2        ar3        ar4        ar5        ar6        ar7
## -0.9007710 -0.8146844 -0.6827974 -0.5901726 -0.5369683 -0.5193530 -0.4808162
##        ar8        ar9       ar10
## -0.4372768 -0.4041186 -0.3910632
```

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

1.7 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
```

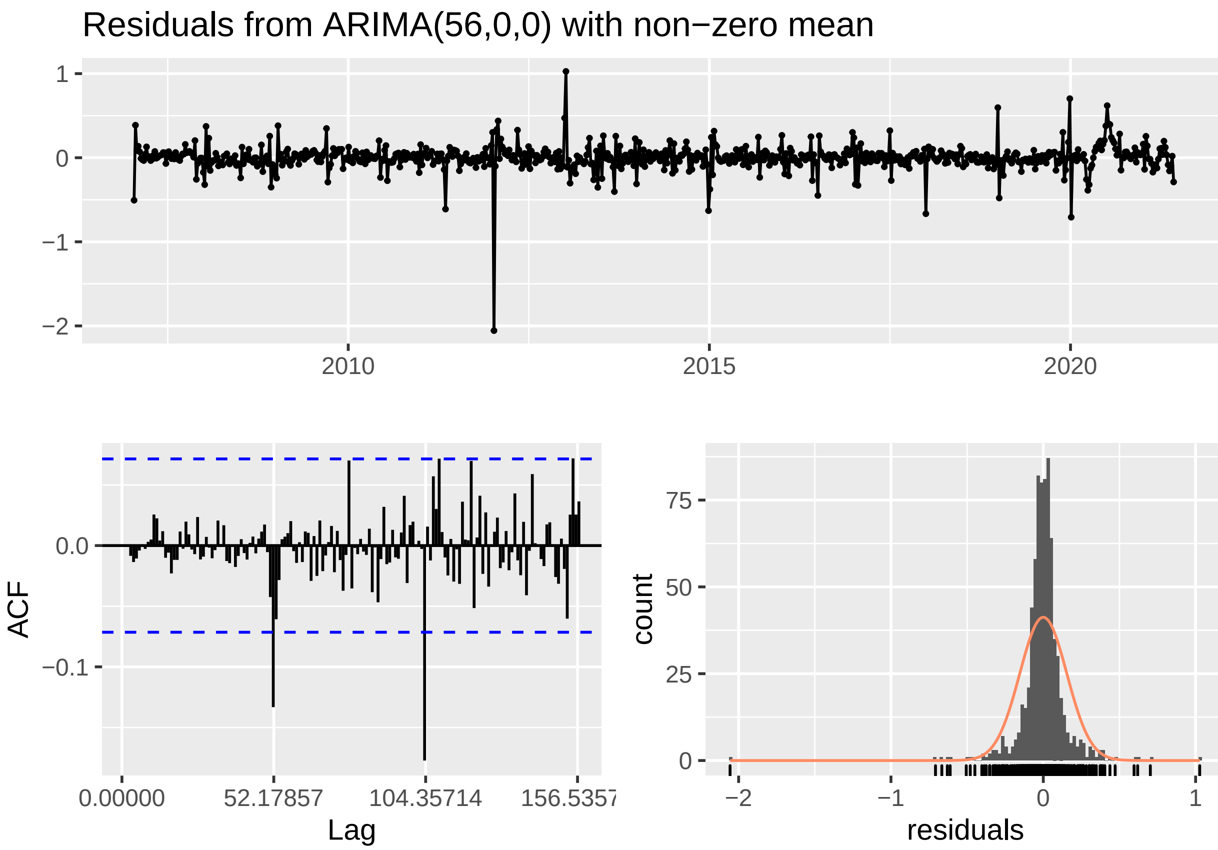
1.8 Porównanie analizowanych modeli

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)
```

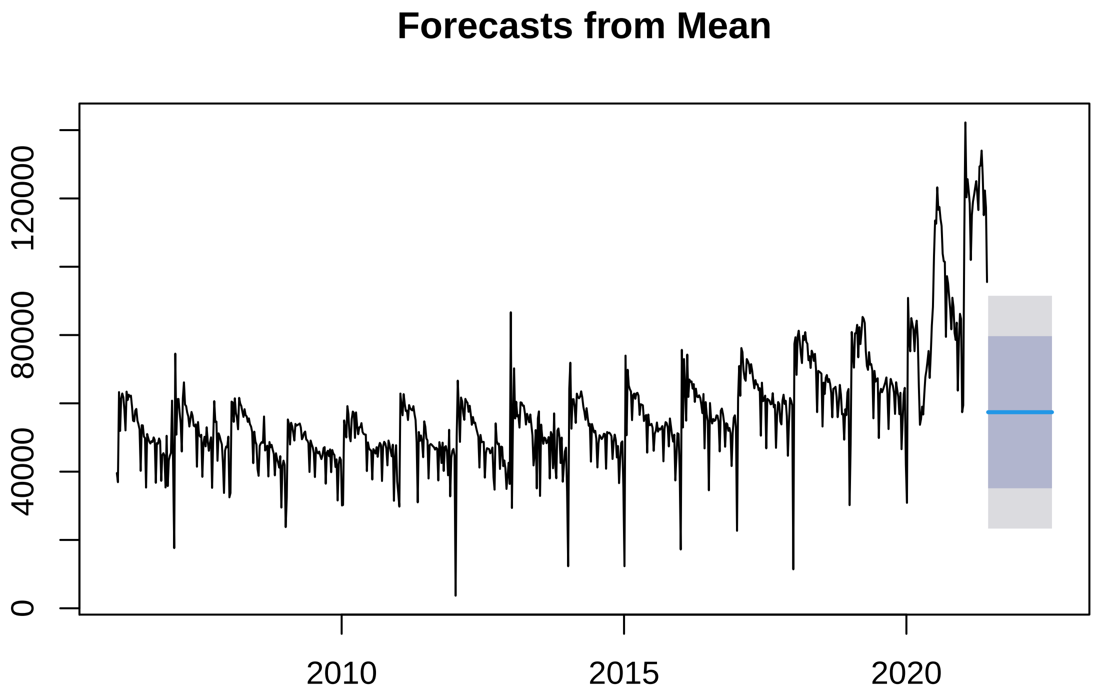



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(56,0,0) with non-zero mean
## Q* = 70.555, df = 47.357, p-value = 0.01601
##
## Model df: 57.   Total lags used: 104.357142857143
```

1.9 Prognozowanie

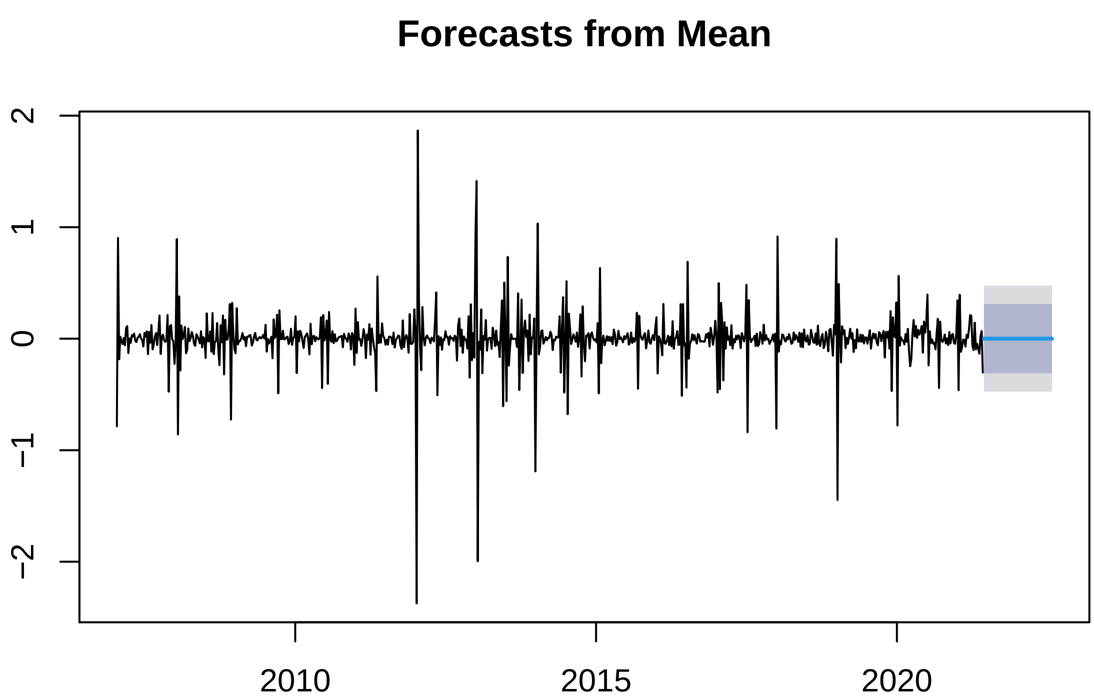
1.9.1 Prognozowanie naiwne metodą średniej

```
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)
```



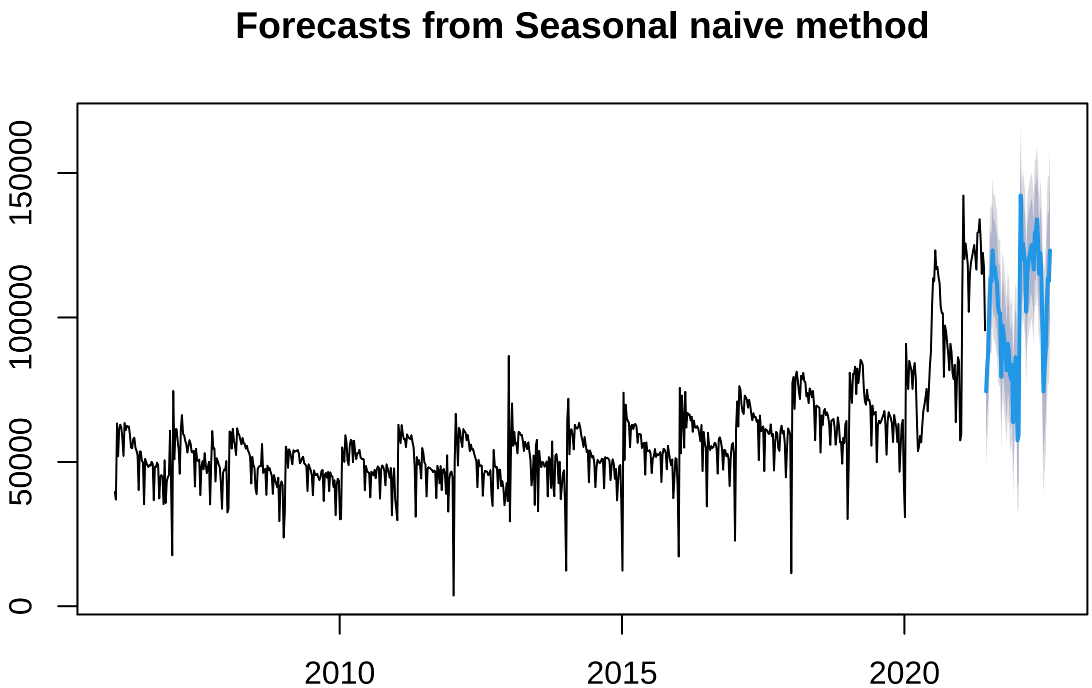
Prognoza ta jest dużo lepsza. Dodając trend i sezonowość moglibyśmy uzyskać nią lepsze przewidywania, niż za pierwszym razem.

1.9.2 Prognozowanie naiwne sezonowe

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

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

```r
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

Szereg ten pochodzi ze strony FRED. Szereg obliczany jest na podstawie danych z obrotów nieruchomościami i wygładzany jest z pomocą 3-miesięcznej średniej ruchomej. Głównie brane pod uwagę są domy jednorodzinne. Szereg rozstał unormowany tak aby cena ze stycznia 2000 roku była równa 100 i każda następna jest określona wobec niej.

Korzyści jakie może przynieść analiza tego szeregu to przewidywanie cen nieruchomości na rynku czy przewidywanie kolejnej bańki finansowej.

2.1 Wczytanie danych

Dane pobrane zostały ze strony https://fred.stlouisfed.org/series/CSUSHPINSA w formacie csv.

```r
ind <- read.csv2("Datasets/CSUSHPINSA.csv", sep = ",")
ind$CSUSHPINSA <- as.numeric(ind$CSUSHPINSA)
head(ind)
```

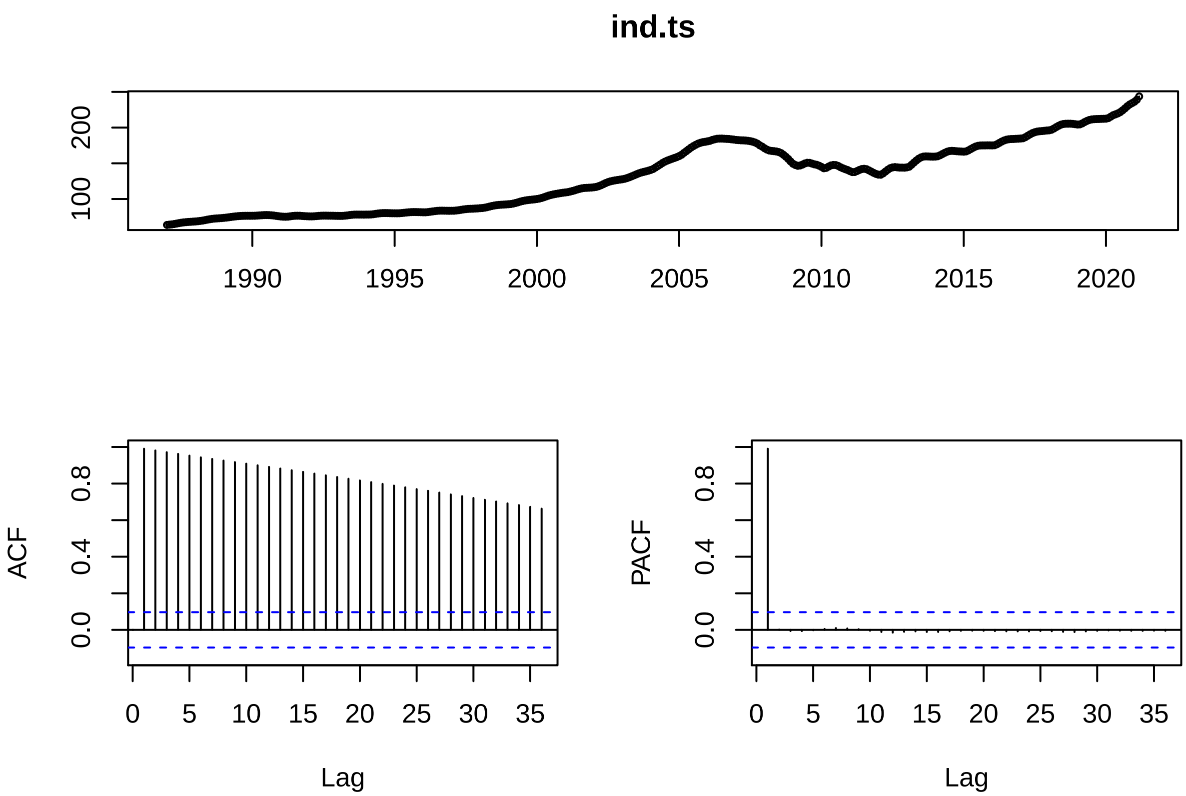
```
##         DATE CSUSHPINSA
## 1 1987-01-01     63.735
## 2 1987-02-01     64.135
## 3 1987-03-01     64.471
```

```
## 4 1987-04-01     64.977
## 5 1987-05-01     65.552
## 6 1987-06-01     66.221
```

2.2 Główne cechy analizowanych danych

Zacznę od zamiany szeregu na szereg czasowy oraz analizy funkcji ACF i PACF.

```
ind.ts <- ts(ind$CSUSHPINSA, start = c(1987, 01), frequency = 12)
tsdisplay(ind.ts)
```



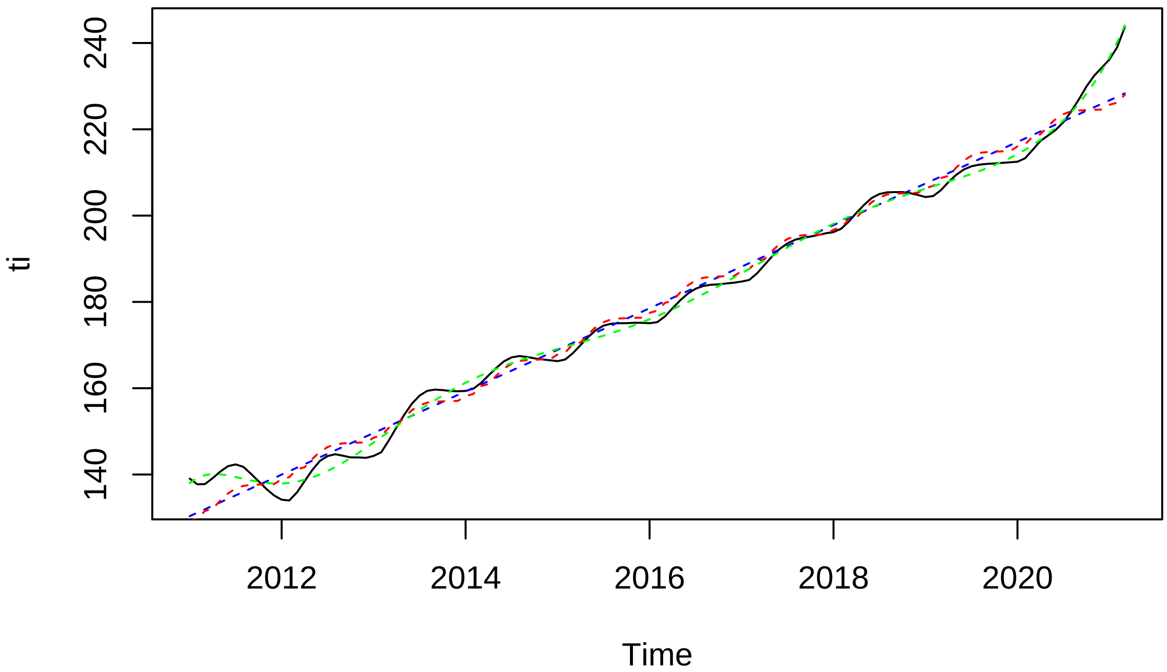
Szereg charakteryzuje się dodatnim trendem (dodatnia, powoli opadająca funkcja ACF). Na pierwszy rzut oka nie widać sezonowości, także funkcja PACF na nią nie wskazuje. Problemem natomiast mogą być dane z roku 2010. Zatem w celu dopasowania szeregu do analizy dane sprzed roku 2011 zostaną pominięte.

```
ind.ts <- window(ind.ts, start = c(2011,01))
```

2.3 Dekompozycje szeregu

2.3.1 Modele z trendem liniowym, wielomianowym i sezonowością

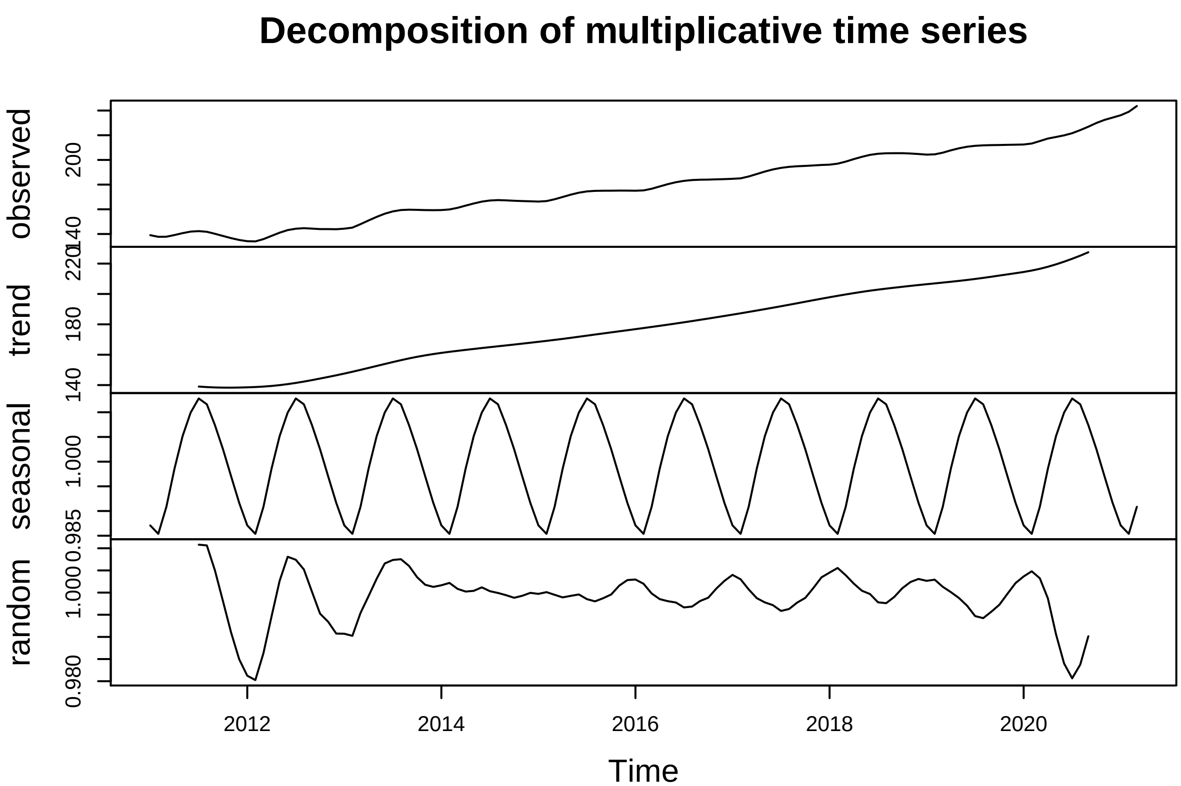
```
ti <- ind.ts
tT <- tslm(ti ~ trend) # Model regrasji z trendem liniowym
tTS <- tslm(ti ~  trend + season) # Model regresji z trendem liniowym i sezonowością
tPS <- tslm(ti ~  poly(trend, raw=TRUE, degree = 9)) # Model regresji z trendem liniowym i
plot(ti)
lines(fitted(tT), col = "blue", lty = 2)
lines(fitted(tTS), col = "red", lty = 2)
lines(fitted(tPS), col = "green", lty = 2)
```



Dekompozycja wskazuje na to, że nie jest to trend liniowy. Sezonowość również nie jest wyraźnie widoczna i nie wpływa na dopasowanie modelu do szeregu.

2.3.2 Model multiplikatywny

```
ind.decompose <- decompose(ind.ts, type ="multiplicative")
plot(ind.decompose)
```



Dekompozycja multiplikacyjna potwierdza wcześniejsze wyniki. Wyraźny jest trend, patrząc na rząd uzyskanej sezonowości, jest on dwukrotnie mniejszy od trendu.

2.4 Usunięcie trendu i sezonowości

Tym razem skorzystam z pomocy funkcji `ndiffs` i `nsdiffs`, wskazują one ile razy należy różnicować, aby usunąć trend i sezonowość.

```
ndiffs(ind.ts)
```

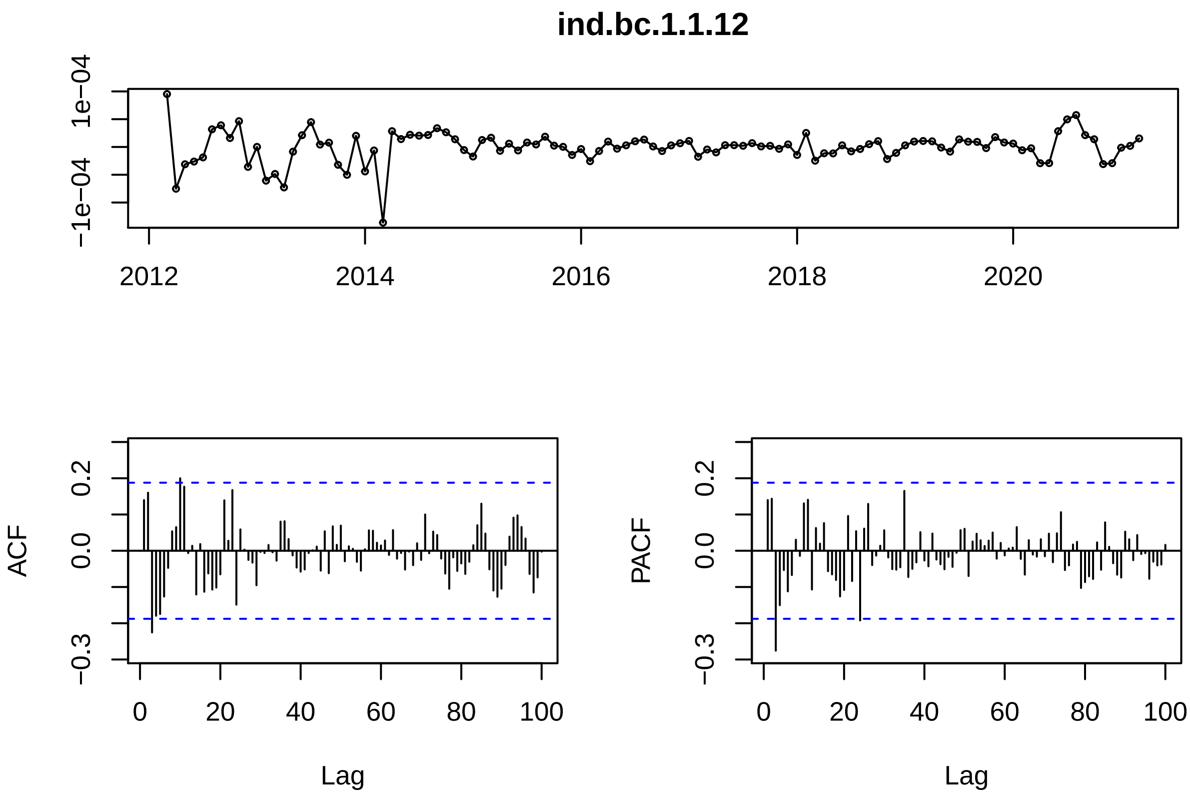
```
## [1] 1
```

```
nsdiffs(ind.ts)
```

```
## [1] 1
```

Funkcje wskazują na to, że aby uzyskać szereg stacjonarny należy zróżnicować co najmniej jednokrotnie z lag=1 oraz jednokrotnie z lag=12.

```
ind.lambda <- BoxCox.lambda(ind.ts)
ind.bc <- BoxCox(ind.ts, ind.lambda)
ind.bc.1.1 <- diff(diff(ind.bc, lag = 1), lag = 1)
ind.bc.1.1.12 <- diff(ind.bc.1.1, lag = 12)
tsdisplay(ind.bc.1.1.12, lag.max = 100)
```



Jak widać po zróżnicowaniu reszty przypominają już szereg stacjonarny. Zostanie to jeszcze potwierdzone testem.

```
shapiro.test(ind.bc.1.1.12)
```

```
##
##  Shapiro-Wilk normality test
##
```

```
## data:  ind.bc.1.1.12
## W = 0.90079, p-value = 6.23e-07
```

Szereg reszt nie jest realizacją szumu białego.

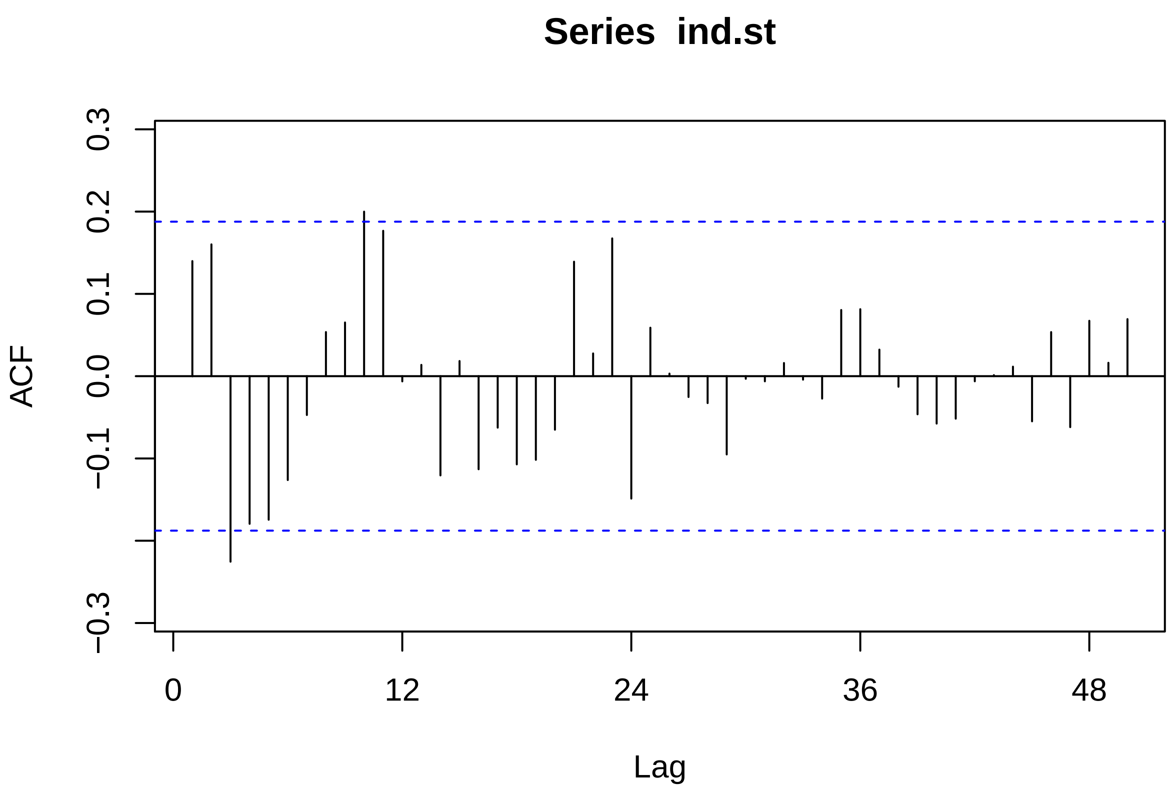
```r
ind.bc.1.1.12 %>% ur.kpss(use.lag = 12) %>% summary()
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 12 lags.
##
## Value of test-statistic is: 0.2254
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

Test wskazuje na to, że nie ma podstaw do odrzucenia hipotezy o tym, że szereg jest stacjonarny.

2.5 Wyznaczenie rzędu MA

```r
ind.st <- ind.bc.1.1.12 # Szereg stacjonarny
Acf(ind.st, lag.max = 50)
```



Do rozważenia mamy następujące modele MA:

```r
ind.st.acf <- Acf(ind.st, plot = FALSE, lag.max = 100)
ind.st.acf$lag[which(abs(ind.st.acf$acf)>1.96/sqrt(ind.st.acf$n.used))] # Wszystkie lag poza przedziałem
```

```
## [1]  0  3 10
```

Wyznaczę modele MA(12), MA(9) oraz MA(3):

```
ind.st.ma10 <- Arima(st, order = c(0,0,10))
ind.st.ma3 <- Arima(st, order = c(0,0,3))
```

Część współczynników oraz metryki modeli:

```
c(ind.st.ma10$aic, ind.st.ma10$aicc, ind.st.ma10$bic)
```

```
## [1] -394.3058 -393.8837 -338.8330
```

```
ind.st.ma10$coef[1:5]
```

```
##          ma1          ma2          ma3          ma4          ma5
## -0.858563469 -0.043624786  0.084099016 -0.008189441 -0.050676046
```

```
c(ind.st.ma3$aic, ind.st.ma3$aicc, ind.st.ma3$bic)
```

```
## [1] -397.3129 -397.2324 -374.1992
```

```
ind.st.ma3$coef
```

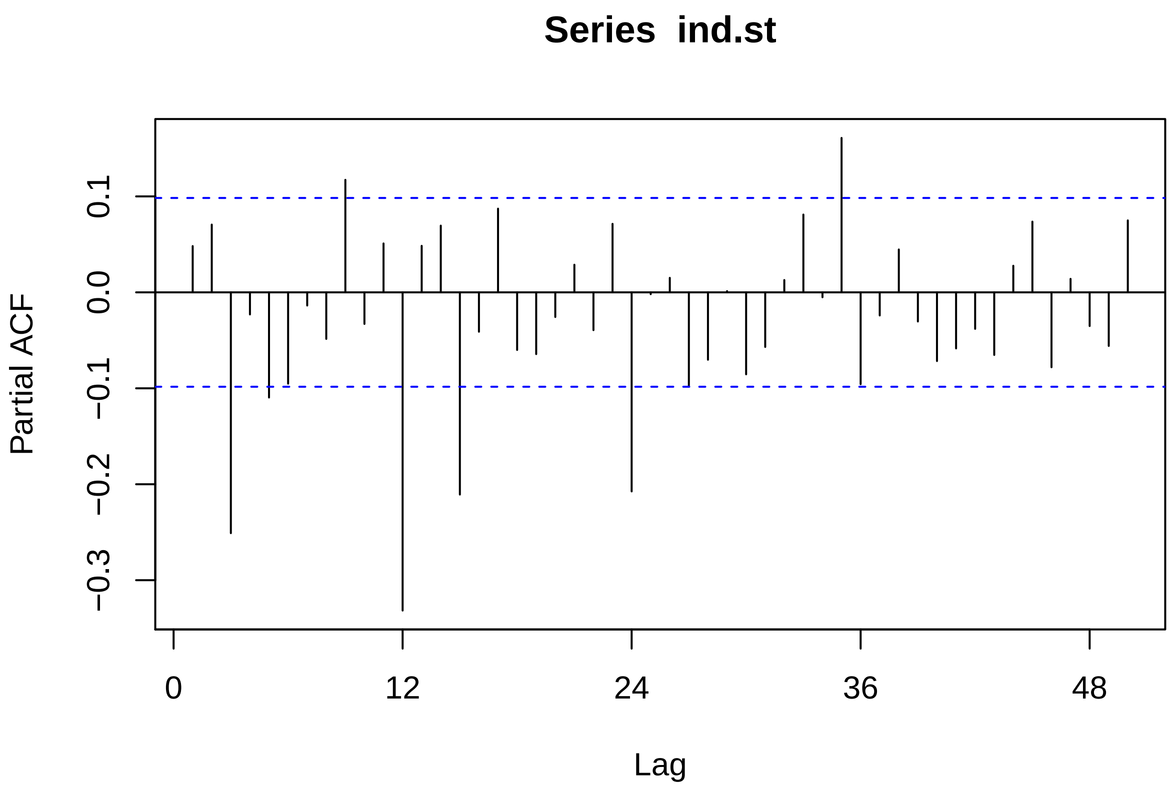
```
##           ma1           ma2           ma3     intercept
## -0.8500634318 -0.0501600018  0.0141696263  0.0008092502
```

Współczynniki MA(10) i MA(3) są podobne. Wszystkie modele mają podobne wartości AIC, AICc oraz BIC.

2.6 Wyznaczenie rzędu AR

Rzędy modelu AR można odczytać z wykresu Pacf.

```
Pacf(ind.st, lag.max = 50)
```



Skorzystam z pomocniczej funkcji.

```
ind.st.pacf <- Pacf(ind.st, plot =  FALSE, lag.max = 100)
ind.st.pacf$lag[which(abs(ind.st.pacf$acf)>1.96/sqrt(ind.st.pacf$n.used))] # Wszystkie lag poza przedzi
```

```
## [1]  3  5  9 12 15 24 35 60 82
```

Obliczę współczynniki dla AR(3) i AR(24):

```
ind.st.ar3 <- Arima(st, order = c(3,0,0))
ind.st.ar24 <- Arima(st, order = c(24,0,0))
```

Część współczynników oraz metryki modeli:

```
c(ind.st.ar3$aic, ind.st.ar3$aicc, ind.st.ar3$bic)
```

```
## [1] -338.0322 -337.9517 -314.9185
```

```
ind.st.ar3$coef[1:3]
```

```
##        ar1        ar2        ar3
## -0.7416738 -0.5372127 -0.2538657
```

```
c(ind.st.ar24$aic, ind.st.ar24$aicc, ind.st.ar24$bic)
```

```
## [1] -374.3548 -372.4183 -254.1637
```

```
ind.st.ar24$coef[1:5]
```

```
##        ar1        ar2        ar3        ar4        ar5
## -0.8535014 -0.7637317 -0.6098465 -0.4895556 -0.4252854
```

Metryki ACC i ACCc tych modeli są podobne, ale model AR(3) ma lepszą (mniejszą) metrykę BIC.

2.7 auto.arima

```
ind.auto <- auto.arima(ind.st, ic="aicc")
summary(ind.auto)
```

```
## Series: ind.st
## ARIMA(0,0,0) with zero mean
##
## sigma^2 estimated as 7.693e-10:  log likelihood=989.05
## AIC=-1976.09   AICc=-1976.06   BIC=-1973.4
##
## Training set error measures:
##                         ME         RMSE          MAE MPE MAPE      MASE
## Training set -3.583462e-07 2.773631e-05 1.843991e-05 100  100 0.7323253
##                   ACF1
## Training set 0.139861
```

2.8 Porównanie analizowanych modeli

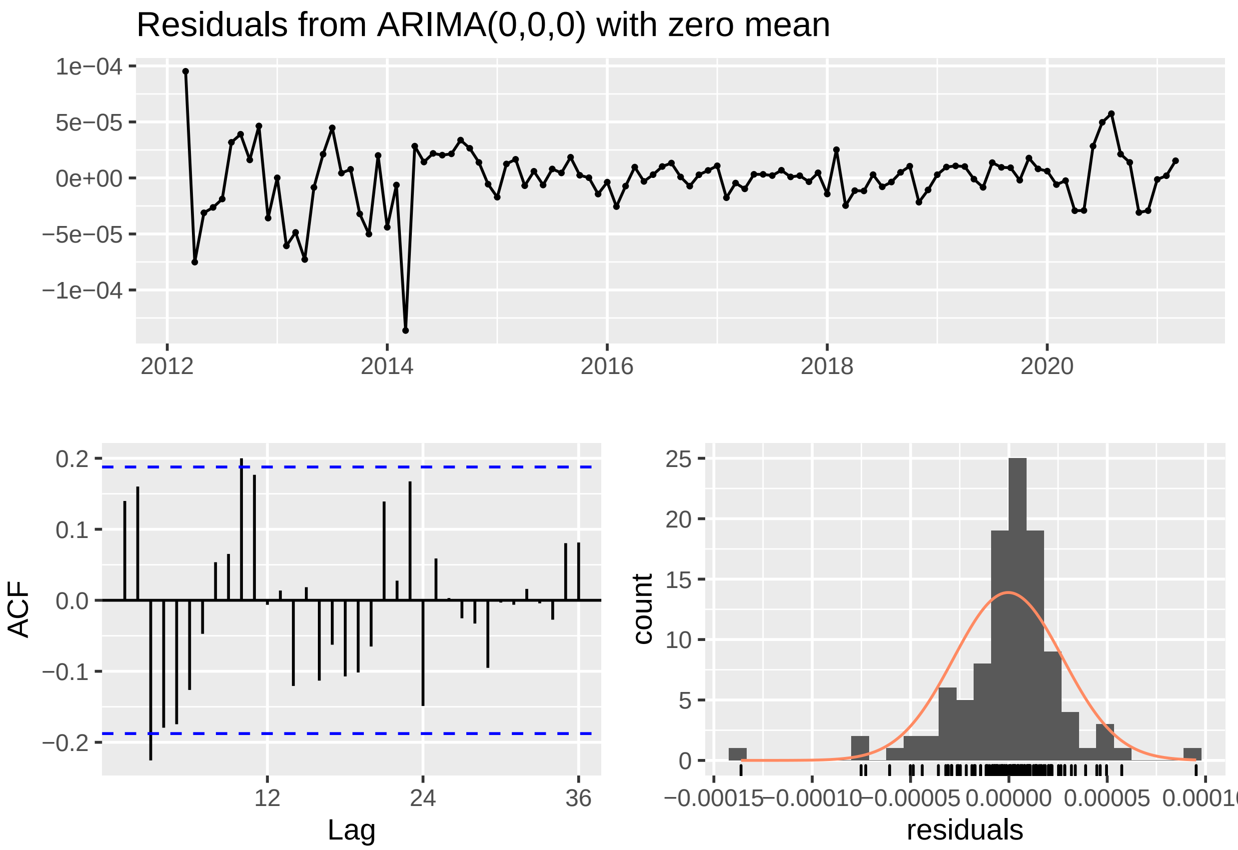
Zebrałem parametry AIC, AICc oraz BIC dla tego szeregu w tabeli poniżej. Wybrany zostanie model, który ma te współczynniki najmniejsze.

```
# ARIMA(0,0,10)     AIC=-394.31     AICc=-393.88    BIC=-338.83
# ARIMA(0,0,3)      AIC=-397.31     AICc=-397.23    BIC=-374.2
# ARIMA(3,0,0)      AIC=-338.03     AICc=-337.95    BIC=-314.9
# ARIMA(24,0,0)     AIC=-374.35     AICc=-372.42    BIC=-254.16
```

```
# ARIMA(0,0,0)       AIC=-1976.09 +  AICc=-1976.06 + BIC=-1973.4 +
```

Najlepszym wydaje się model dobrany automatycznie ARIMA(0,0,0). Poprawność modelu, sprawdzę z pomocą funkcji `checkresiduals` .

```
checkresiduals(ind.auto)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,0,0) with zero mean
## Q* = 40.256, df = 22, p-value = 0.01009
##
## Model df: 0.   Total lags used: 22
```

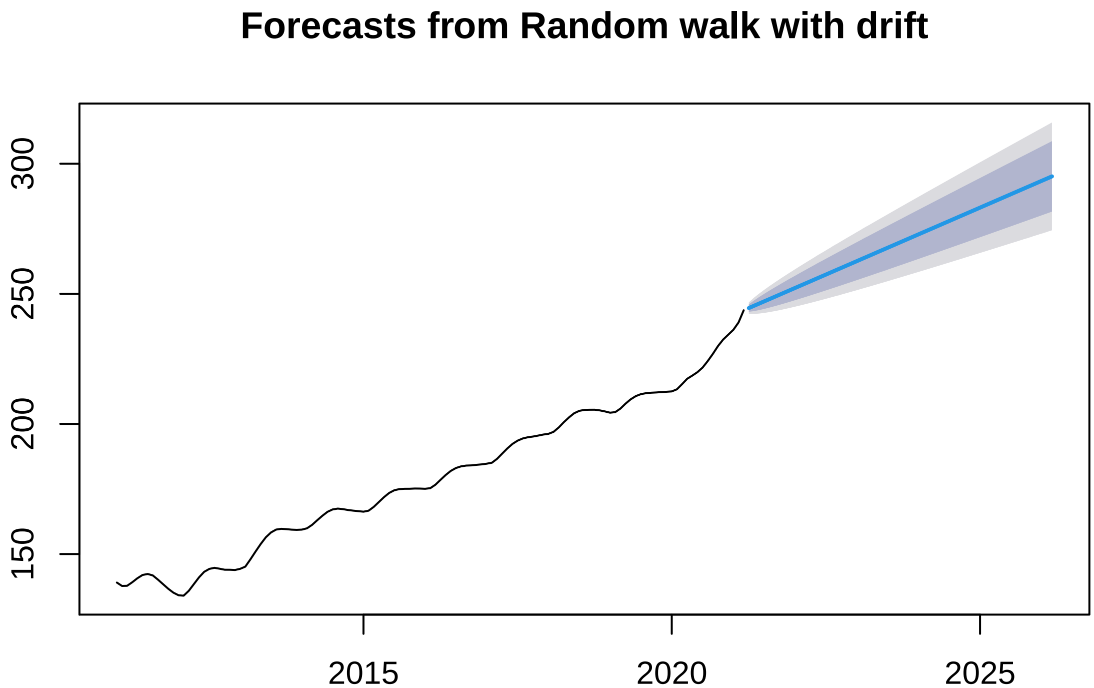
Reszty przechodzą test.

2.9 Prognozowanie

Do prognozowania zostaną wykorzystane modele naiwne.

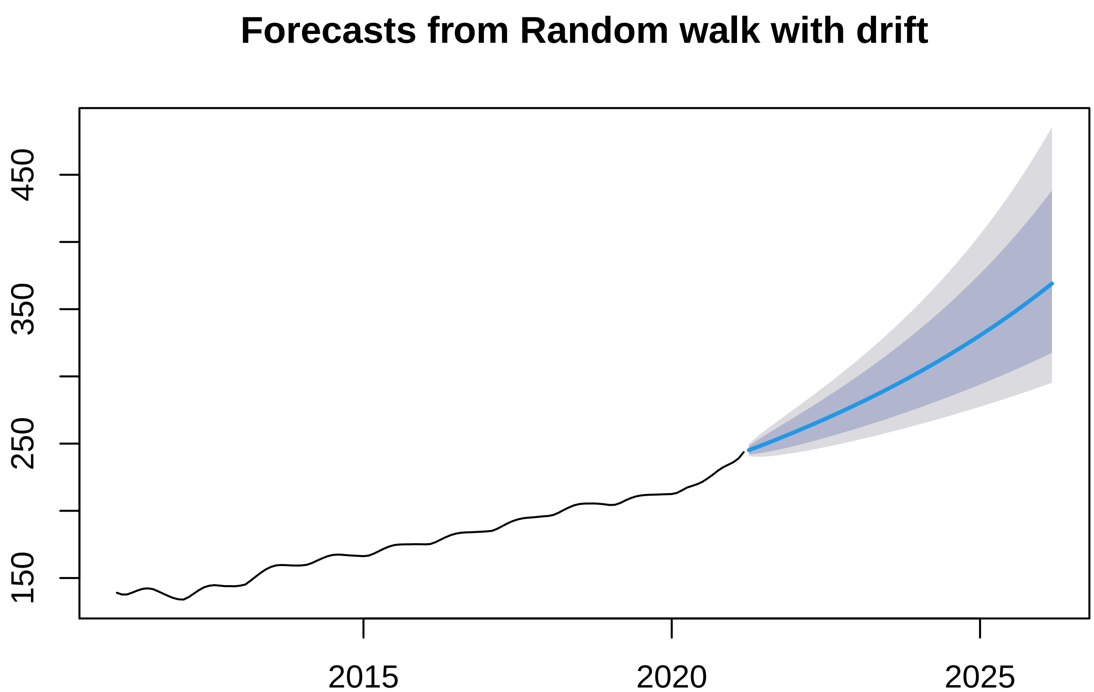
2.9.1 Błądzenie losowe z dryfem

```
plot(rwf(ind.ts, h = 60, drift = TRUE))
```



Model błądzenia losowego z dryfem dobrze sprawdza się dla tego modelu, ponieważ uwzględnia on jego rosnący trend. Predykcje może poprawić ustawienie parametru `lambda`.

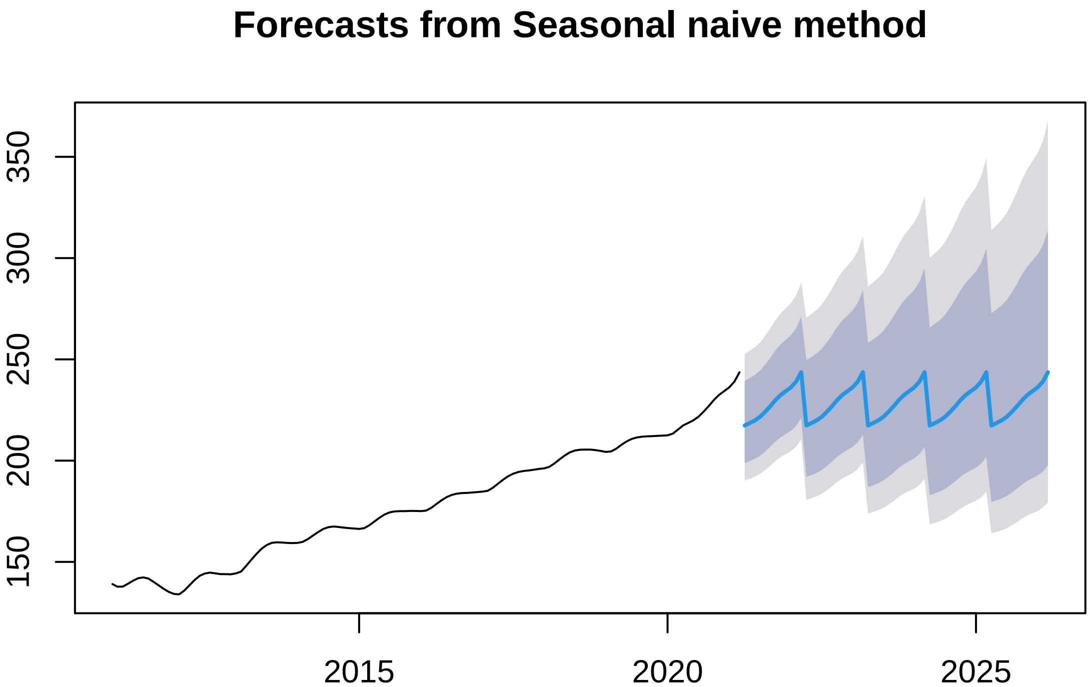
```
plot(rwf(ind.ts, h = 60, drift = TRUE, lambda = ind.lambda))
```



Predykcja na pierwszy rzut oka znacznie się poprawiła, przynajmniej dla najbliższych kilku miesięcy, raczej nie można się spodziewać ciągłego wzrostu tego indeksu.

2.9.2 Prognozowanie naiwne sezonowe

```
plot(snaive(ind.ts, h = 60, lambda = ind.lambda))
```



Prognoza ta nie wydaje się odpowiednia. Nie uwzględnia tego, że indeks może rosnąć.

3 Wnioski

Modelowanie szeregów z pomocą R jest bardzo proste. Mamy szereg gotowych funkcji, które przeprowadzą nas przez cały proces od dekompozycji szeregu, eliminacji trendu i sezonowości po automatyczne dopasowanie modelu do szeregu. Nie wszystkie szeregi jednak będzie się dało tak modelować, w przypadku modeli ARIMA generowanych funkcją `auto.arima` nadal należy sprawdzić czy szereg reszt zachowuje się jak biały szum (gdyż nie zawsze tak jest), można to zrobić chociażby funkcją `checkresiduals`. Prognozy naiwne dają proste predykcje na temat najbliższej przyszłości, które mogą być w miarę dokładne o ile wybraliśmy odpowiednią funkcję do predykcji oraz przedział predykcji nie jest zbyt duży.