Data Cleaning and scraping

```
epl_2020_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/10728/2020-2021-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2021_team_results <- get_team_match_results(team_url = epl_2020_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2020<-subset(epl_2021_team_results,Comp=='Premier League')
```

```
epl_2019_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/3232/2019-2020-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2019_team_results <- get_team_match_results(team_url = epl_2019_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2019<-subset(epl_2019_team_results,Comp=='Premier League')
```

```
epl_2018_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/1889/2018-2019-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2018_team_results <- get_team_match_results(team_url = epl_2018_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2018<-subset(epl_2018_team_results,Comp=='Premier League')
```

```
epl_2017_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/1631/2017-2018-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2017_team_results <- get_team_match_results(team_url = epl_2017_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2017<-subset(epl_2017_team_results,Comp=='Premier League')
```

```
epl_2016_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/1526/2016-2017-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2016_team_results <- get_team_match_results(team_url = epl_2016_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2016<-subset(epl_2016_team_results,Comp=='Premier League')
```

```
epl_2015_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/1467/2015-2016-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2015_team_results <- get_team_match_results(team_url = epl_2015_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2015<-subset(epl_2015_team_results,Comp=='Premier League')
```

```
epl_2014_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/733/2014-2015-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2014_team_results <- get_team_match_results(team_url = epl_2014_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2014<-subset(epl_2014_team_results,Comp=='Premier League')
```

```
epl_2013_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/669/2013-2014-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2013_team_results <- get_team_match_results(team_url = epl_2013_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2013<-subset(epl_2013_team_results,Comp=='Premier League')
```

```
epl_2012_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/602/2012-2013-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2012_team_results <- get_team_match_results(team_url = epl_2012_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2012<-subset(epl_2012_team_results,Comp=='Premier League')

epl_2011_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/534/2011-2012-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2011_team_results <- get_team_match_results(team_url = epl_2011_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2011<-subset(epl_2011_team_results,Comp=='Premier League')

epl_2010_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/467/2010-2011-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2010_team_results <- get_team_match_results(team_url = epl_2010_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2010<-subset(epl_2010_team_results,Comp=='Premier League')

epl_2009_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/400/2009-2010-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2009_team_results <- get_team_match_results(team_url = epl_2009_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2009<-subset(epl_2009_team_results,Comp=='Premier League')

epl_2008_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/338/2008-2009-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2008_team_results <- get_team_match_results(team_url = epl_2008_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2008<-subset(epl_2008_team_results,Comp=='Premier League')
```

```
epl_2007_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/282/2007-2008-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2007_team_results <- get_team_match_results(team_url = epl_2007_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2007<-subset(epl_2007_team_results,Comp=='Premier League')
```

```
epl_2006_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/229/2006-2007-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2006_team_results <- get_team_match_results(team_url = epl_2006_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2006<-subset(epl_2006_team_results,Comp=='Premier League')
```

```
epl_2005_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/183/2005-2006-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2005_team_results <- get_team_match_results(team_url = epl_2005_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2005<-subset(epl_2005_team_results,Comp=='Premier League')
```

```
epl_2004_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/146/2004-2005-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2004_team_results <- get_team_match_results(team_url = epl_2004_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2004<-subset(epl_2004_team_results,Comp=='Premier League')
```

```
epl_2003_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/112/2003-2004-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2003_team_results <- get_team_match_results(team_url = epl_2003_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2003<-subset(epl_2003_team_results,Comp=='Premier League')
```

```
epl_2002_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/84/2002-2003-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2002_team_results <- get_team_match_results(team_url = epl_2002_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2002<-subset(epl_2002_team_results,Comp=='Premier League')
```

```
epl_2001_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/63/2001-2002-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2001_team_results <- get_team_match_results(team_url = epl_2001_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2001<-subset(epl_2001_team_results,Comp=='Premier League')
```

```
epl_2000_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/47/2000-2001-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_2000_team_results <- get_team_match_results(team_url = epl_2000_team_urls)
```

```
## [1] "Scraping team match logs..."
```

```
e2000<-subset(epl_2000_team_results,Comp=='Premier League')
```

```
epl_1999_team_urls <- fb_teams_urls('https://fbref.com/en/comps/9/38/1999-2000-Premier-League-Stats')
```

```
## [1] "Scraping team URLs"
```

```
epl_1999_team_results <- get_team_match_results(team_url = epl_1999_team_urls)
```

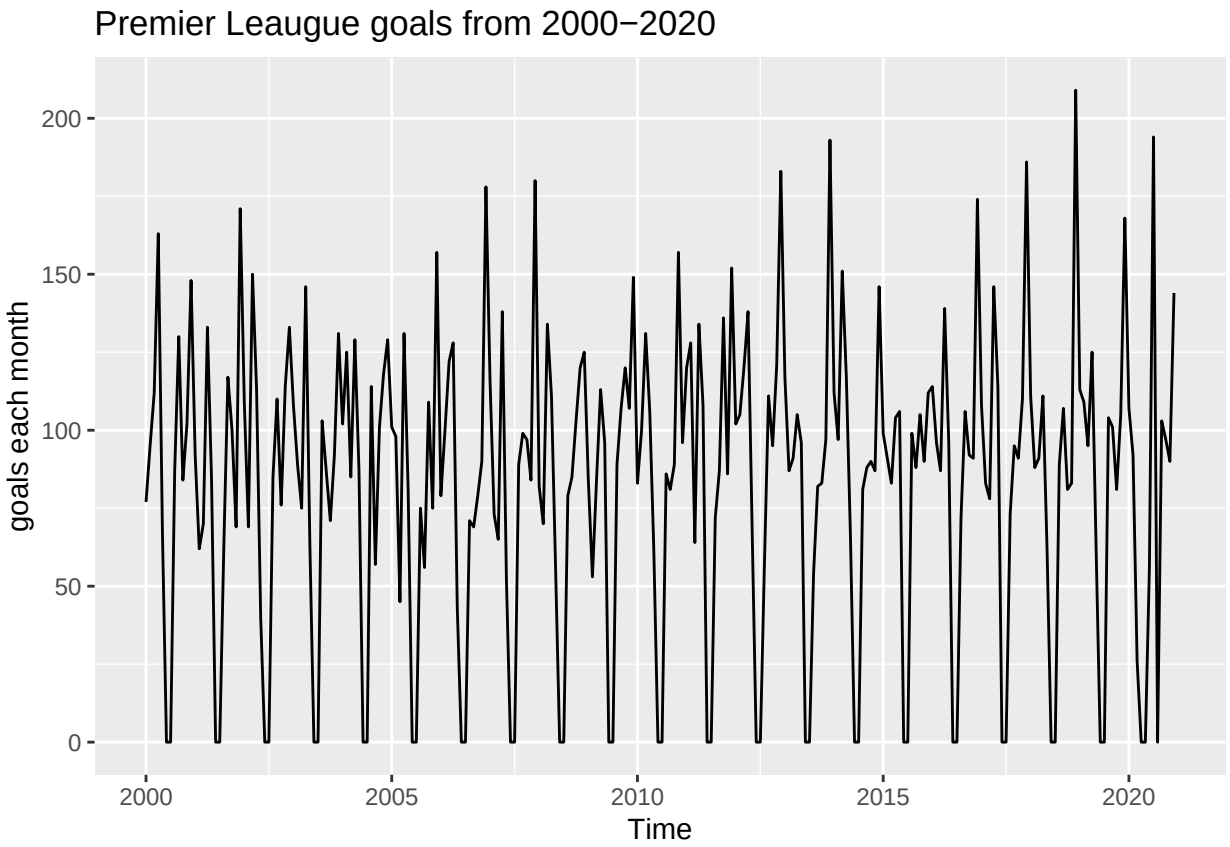
```
## [1] "Scraping team match logs..."
```

```
e1999<-subset(epl_1999_team_results,Comp=='Premier League')

# The above code is just for scraping the premier league tables
#from 1999/2000-2019/2020 season years



columns<-c('Date','GF')
#we just want the date and goals scored from our dataset
e2020<-e2020[,columns]
e2019<-e2019[,columns]
e2018<-e2018[,columns]
e2017<-e2017[,columns]
e2016<-e2016[,columns]
e2015<-e2015[,columns]
e2014<-e2014[,columns]
e2013<-e2013[,columns]
e2012<-e2012[,columns]
e2011<-e2011[,columns]
e2010<-e2010[,columns]
e2009<-e2009[,columns]
e2008<-e2008[,columns]
e2007<-e2007[,columns]
e2006<-e2006[,columns]
e2005<-e2005[,columns]
e2004<-e2004[,columns]
e2003<-e2003[,columns]
e2002<-e2002[,columns]
e2001<-e2001[,columns]
e2000<-e2000[,columns]
e1999<-e1999[,columns]
#The above code is just to apply the columns that we want in our datasets

prem<-rbind(e1999,e2000,e2001,e2002,e2003
            ,e2004,e2005,e2006,e2007,e2008,e2009
            ,e2010,e2011,e2012,e2013,e2014,e2015,
            e2016,e2017,e2018,e2019,e2020)
#I set the dataframe prem to combine all of our datasets
#from 1999/2000-2019/2020 season years

prem$Date <- ymd(prem$Date)#This is to set the Date to a date object

prem$Date<-floor_date(prem$Date, "month")
#This is to set all of the games in each month
#to the first day of the month
#for example 2000-10-4 would be converted
#to 2000-10-01 or 2004-05-25 would be converted to 2005-05-01,
#this will be useful for our time series
```

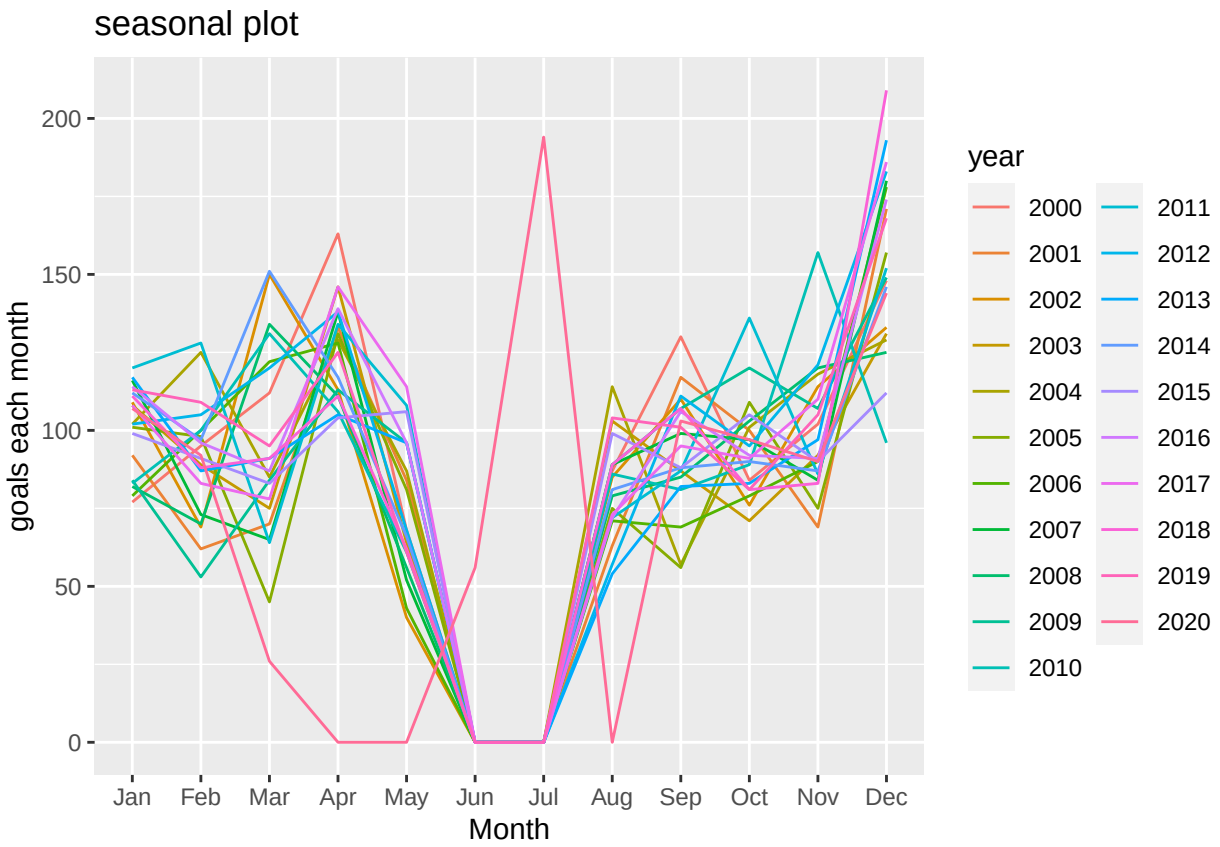
```r
prem$GF<-as.numeric(prem$GF)#converting goals columns to integers

prem<-prem %>%
  mutate(Date = as.Date(Date)) %>%
  complete(Date = seq.Date(min(Date), max(Date), by="month"))
#Since every season there is no game in the months june-july
#then we dont have any data in our dataframe for it
#therefore we need to add missing dates
#for each year to our dataframe and
#by deafualt the goals will ba null values and we can replace nulls to be 0

dodo<-prem %>%
  group_by(Date)%>%
  summarise(sum(GF))
#grouping by date and summing goals and saving it to dataframe dodo

dodo[is.na(dodo)] <- 0#setting null values to 0

dodo<-dodo[-c(1,2,3,4,5,258,259,260,261,262), ]
#These are the rows in our dataframe that are
#from 4 months of 1999 season and 4 months of 2021 season but since
#we are only focused on 2000-2020 we drop them.


write.csv(dodo,'good.csv',row.names=F)#write our final csv named good.csv
```

Time Series Analysis

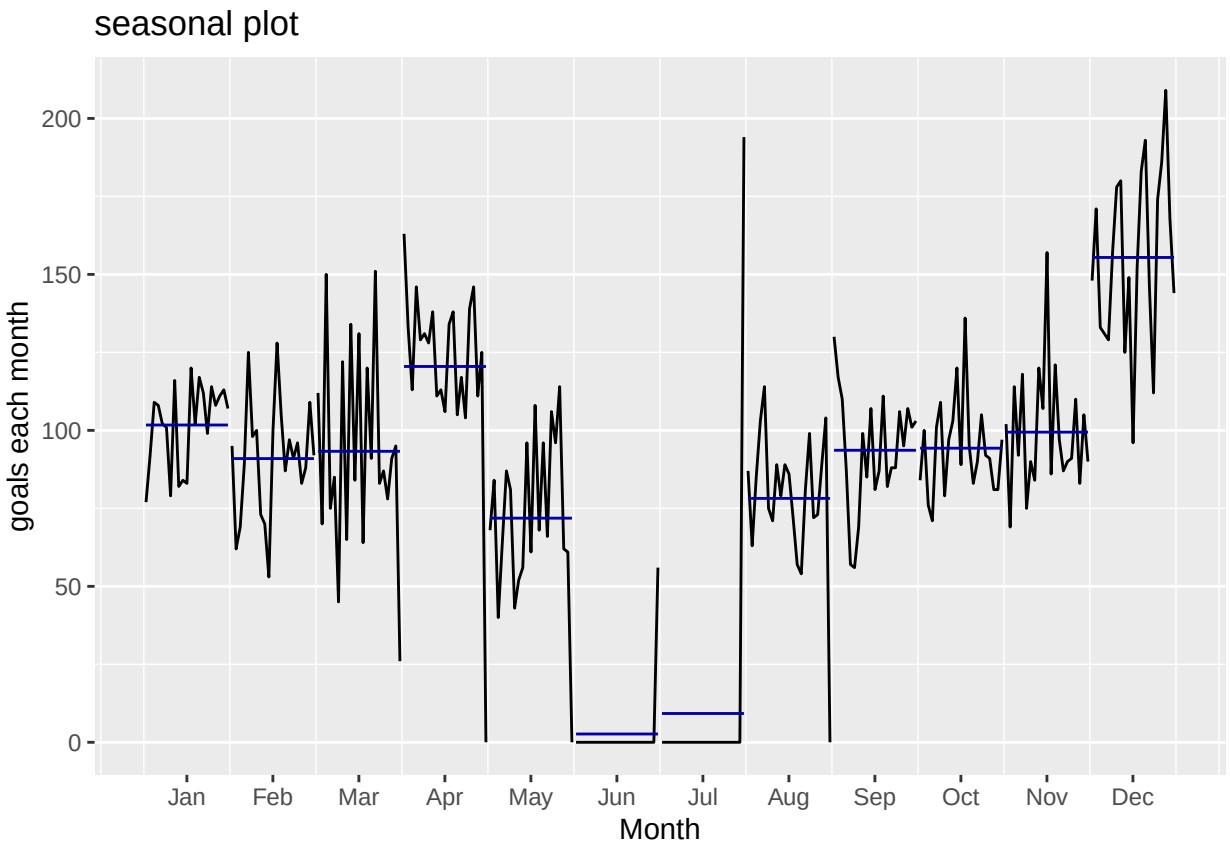
```
dat<-read.csv('good.csv',header = T)
dat$goals<-dat$sum.GF.#renaming the column name to goals

dat$goals<-as.numeric(dat$goals)#converting it to integer
dat$Date<-as.Date(dat$Date)


X<-ts(dat[,2],start=c(2000,1),frequency = 12)
#Time series object for our goals column
#frequency=12 beacuse we have a monthly data

autoplot(X)+ggtitle('Premier Leaugue goals from 2000-2020')+ylab('goals each month')
```




```
#ploting the timeseries plot

ggseasonplot(X)+ggtitle('seasonal plot')+ylab('goals each month')#plot seasonal
```




```
ggsubseriesplot(X)+ylab('goals each month')+ggtitle('seasonal plot')#plot seasonal
```

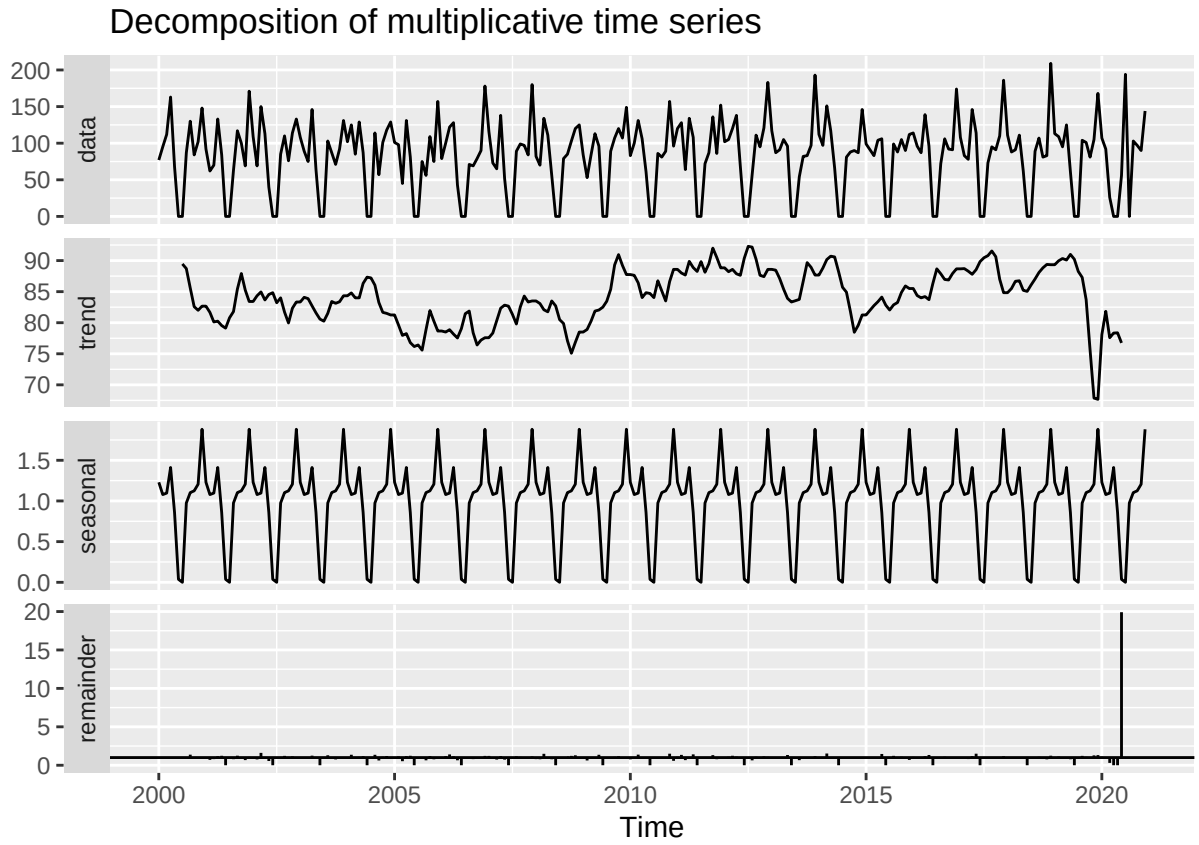



```
adf.test(X)
```

```
## Warning in adf.test(X): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  X
## Dickey-Fuller = -12.127, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```
ddata<-decompose(X,'multiplicative')
autoplot(ddata) #plotting the decomposed timeseries
```

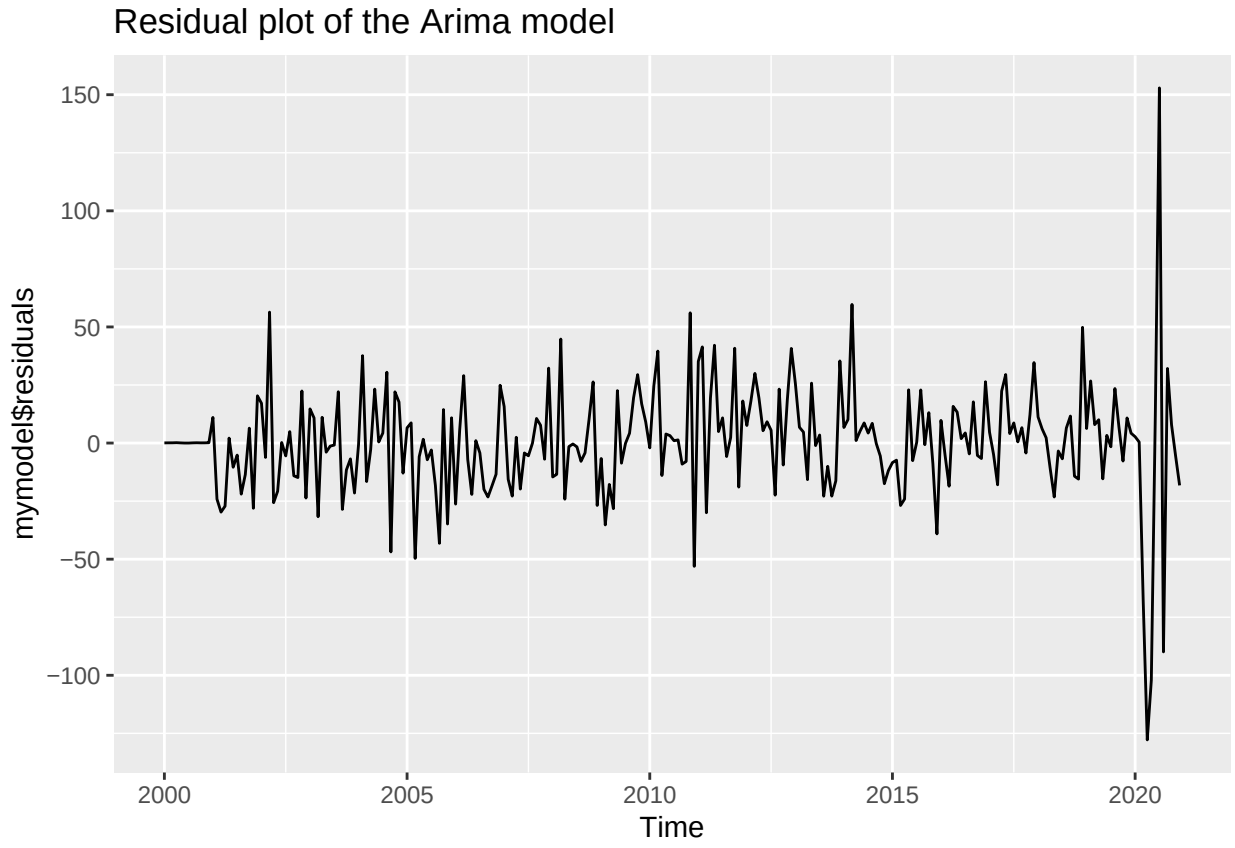



```
mymodel<-auto.arima(X,ic='aic',trace=T)
```

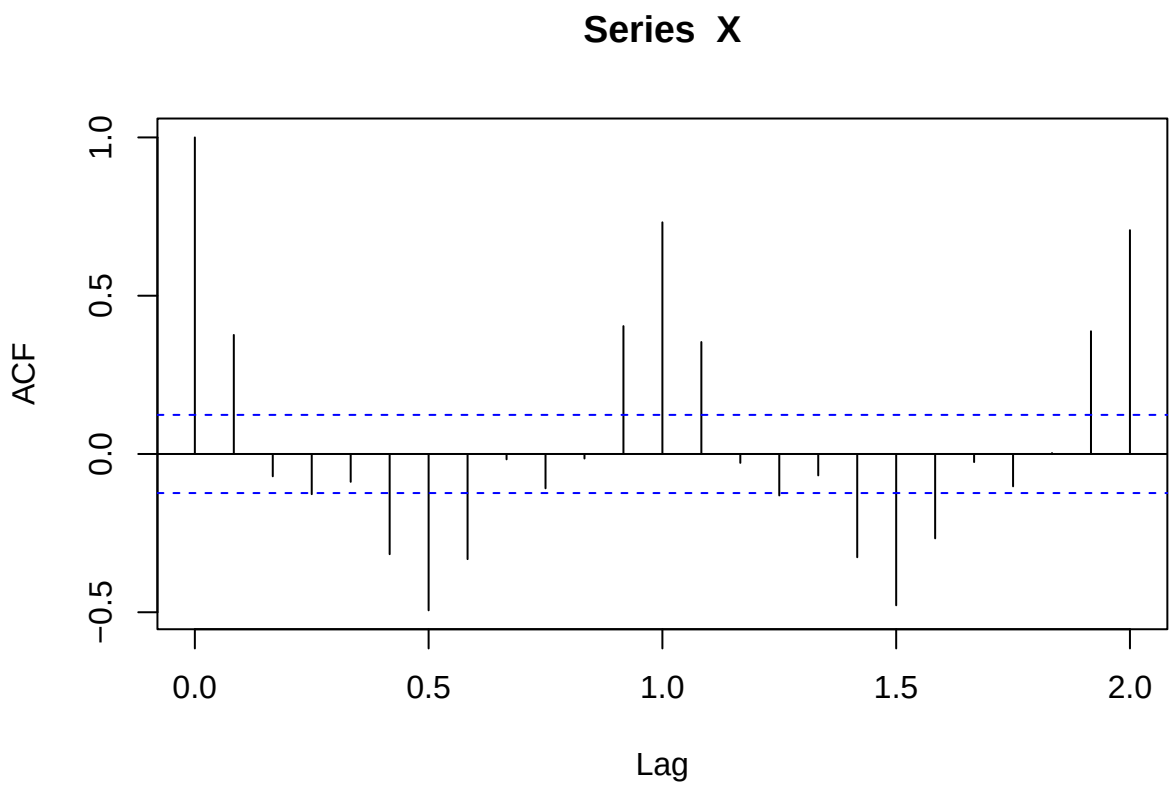
```
##
##  Fitting models using approximations to speed things up...
##
##  ARIMA(2,0,2)(1,1,1)[12] with drift         : 2183.513
##  ARIMA(0,0,0)(0,1,0)[12] with drift         : 2253.584
##  ARIMA(1,0,0)(1,1,0)[12] with drift         : 2204.194
##  ARIMA(0,0,1)(0,1,1)[12] with drift         : 2186.86
##  ARIMA(0,0,0)(0,1,0)[12]                    : 2251.69
##  ARIMA(2,0,2)(0,1,1)[12] with drift         : 2173.171
##  ARIMA(2,0,2)(0,1,0)[12] with drift         : Inf
##  ARIMA(2,0,2)(0,1,2)[12] with drift         : 2175.001
##  ARIMA(2,0,2)(1,1,0)[12] with drift         : 2191.81
##  ARIMA(2,0,2)(1,1,2)[12] with drift         : Inf
##  ARIMA(1,0,2)(0,1,1)[12] with drift         : 2176.698
##  ARIMA(2,0,1)(0,1,1)[12] with drift         : 2171.171
##  ARIMA(2,0,1)(0,1,0)[12] with drift         : 2240.755
##  ARIMA(2,0,1)(1,1,1)[12] with drift         : 2184.485
##  ARIMA(2,0,1)(0,1,2)[12] with drift         : 2173.001
##  ARIMA(2,0,1)(1,1,0)[12] with drift         : 2197.003
##  ARIMA(2,0,1)(1,1,2)[12] with drift         : Inf
##  ARIMA(1,0,1)(0,1,1)[12] with drift         : 2185.222
##  ARIMA(2,0,0)(0,1,1)[12] with drift         : 2180.501
##  ARIMA(3,0,1)(0,1,1)[12] with drift         : 2174.683
```

```
##  ARIMA(1,0,0)(0,1,1)[12] with drift         : 2186.053
##  ARIMA(3,0,0)(0,1,1)[12] with drift         : 2184.631
##  ARIMA(3,0,2)(0,1,1)[12] with drift         : 2175.878
##  ARIMA(2,0,1)(0,1,1)[12]                    : 2169.708
##  ARIMA(2,0,1)(0,1,0)[12]                    : 2238.785
##  ARIMA(2,0,1)(1,1,1)[12]                    : 2183.419
##  ARIMA(2,0,1)(0,1,2)[12]                    : 2171.565
##  ARIMA(2,0,1)(1,1,0)[12]                    : 2195.017
##  ARIMA(2,0,1)(1,1,2)[12]                    : Inf
##  ARIMA(1,0,1)(0,1,1)[12]                    : 2183.511
##  ARIMA(2,0,0)(0,1,1)[12]                    : 2178.529
##  ARIMA(3,0,1)(0,1,1)[12]                    : 2173.83
##  ARIMA(2,0,2)(0,1,1)[12]                    : 2171.572
##  ARIMA(1,0,0)(0,1,1)[12]                    : 2184.183
##  ARIMA(1,0,2)(0,1,1)[12]                    : 2174.918
##  ARIMA(3,0,0)(0,1,1)[12]                    : 2182.635
##  ARIMA(3,0,2)(0,1,1)[12]                    : 2174.857
##
##  Now re-fitting the best model(s) without approximations...
##
##  ARIMA(2,0,1)(0,1,1)[12]                    : 2257.51
##
##  Best model: ARIMA(2,0,1)(0,1,1)[12]
```

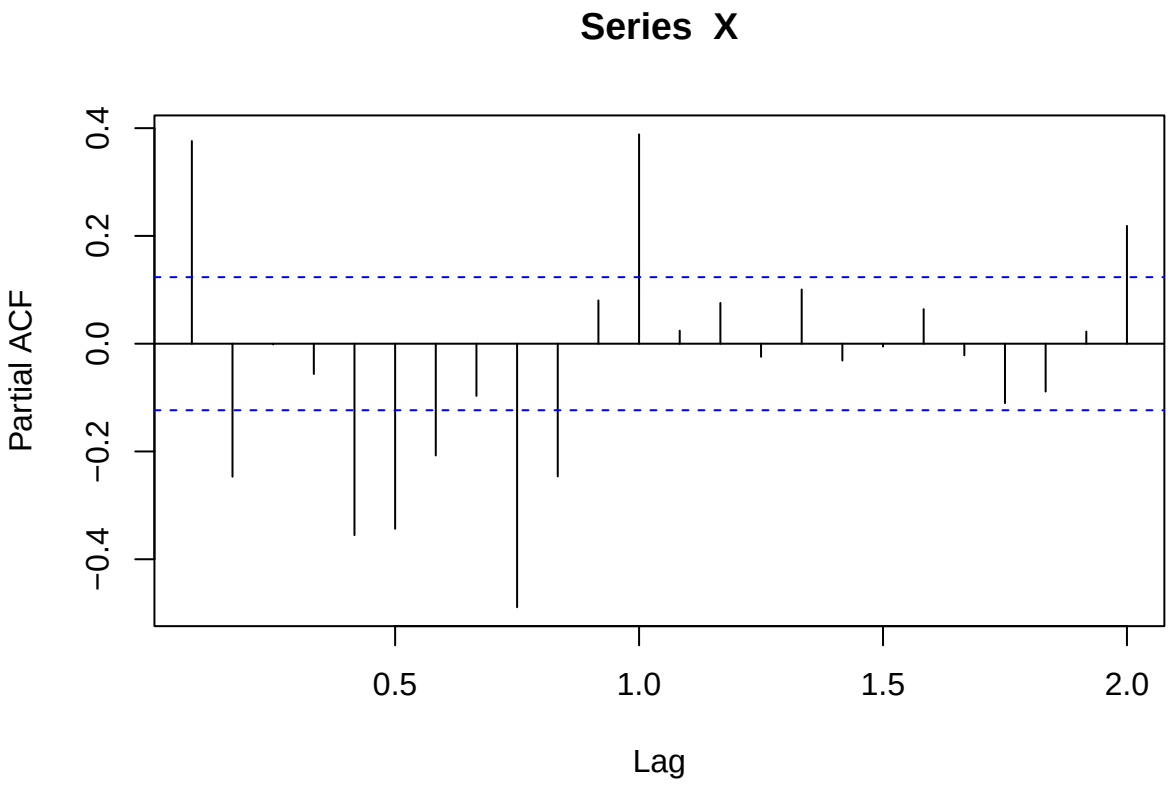
```
autoplot(mymodel$residuals)+ggtitle('Residual plot of the Arima model')
```



```
acf(X)
```




```
pacf(X)
```




```
summary(mymodel)
```

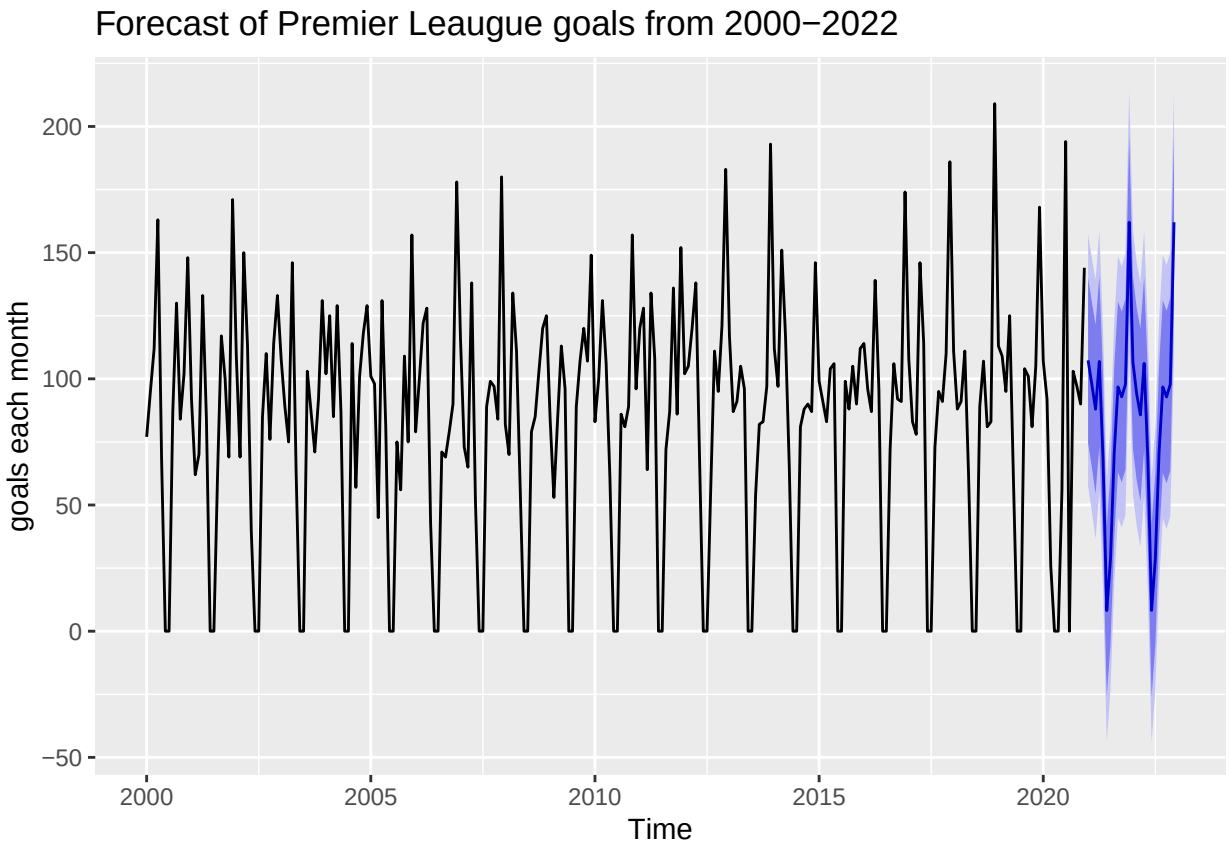
```
## Series: X
## ARIMA(2,0,1)(0,1,1)[12]
##
## Coefficients:
##          ar1      ar2      ma1     sma1
##       0.6408  -0.1793  -0.7021  -0.8527
## s.e.  0.1314   0.0702   0.1233   0.0677
##
## sigma^2 estimated as 650.5:  log likelihood=-1123.75
## AIC=2257.51   AICc=2257.77   BIC=2274.91
##
## Training set error measures:
##                     ME   RMSE      MAE MPE MAPE      MASE        ACF1
## Training set 0.2754297 24.682 16.29042 NaN  Inf 0.7877699 0.004637159
```

```
fcast<-forecast(mymodel,h=24)
summary(fcast)
```

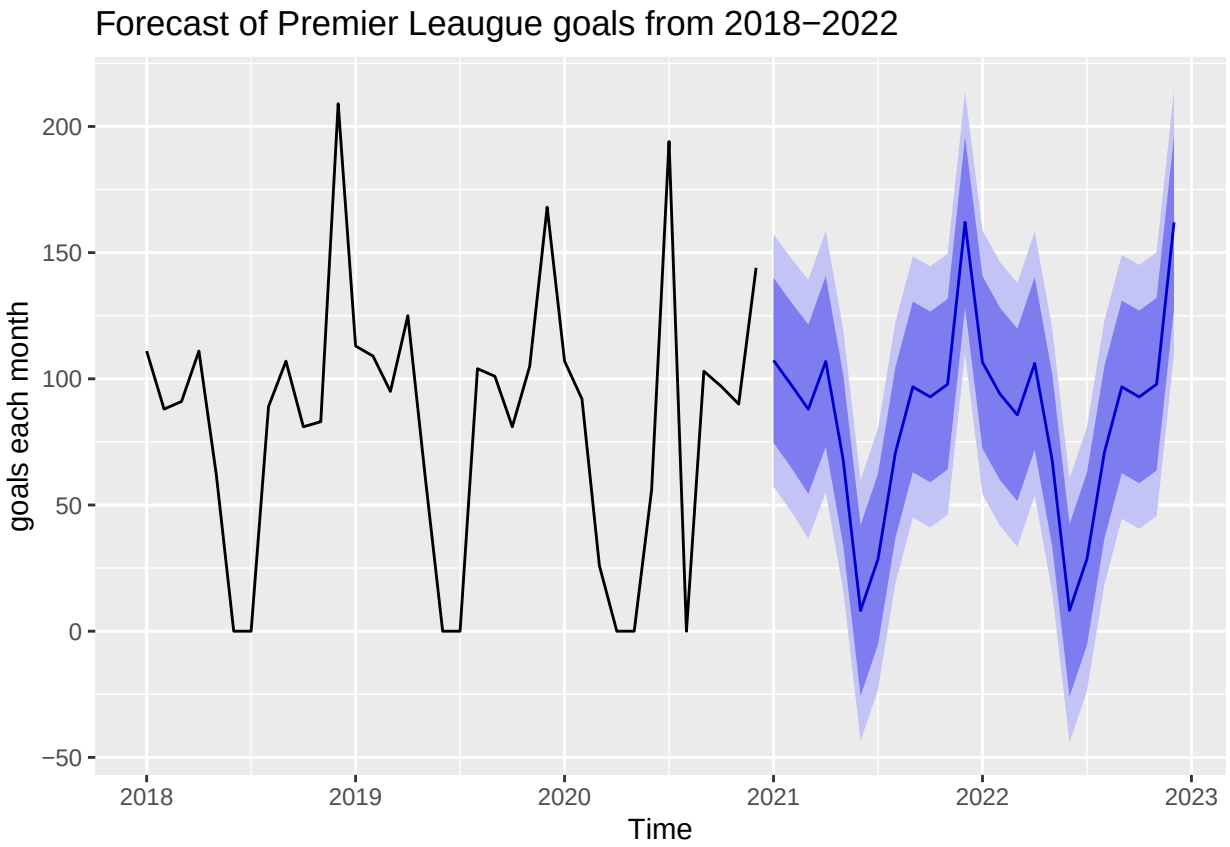
```
##
## Forecast method: ARIMA(2,0,1)(0,1,1)[12]
##
## Model Information:
```

```
## Series: X
## ARIMA(2,0,1)(0,1,1)[12]
##
## Coefficients:
##          ar1      ar2      ma1     sma1
##       0.6408  -0.1793  -0.7021  -0.8527
## s.e.  0.1314   0.0702   0.1233   0.0677
##
## sigma^2 estimated as 650.5:  log likelihood=-1123.75
## AIC=2257.51   AICc=2257.77   BIC=2274.91
##
## Error measures:
##                      ME   RMSE      MAE MPE MAPE      MASE        ACF1
## Training set 0.2754297 24.682 16.29042 NaN  Inf 0.7877699 0.004637159
##
## Forecasts:
##          Point Forecast      Lo 80     Hi 80     Lo 95     Hi 95
## Jan 2021     107.321962  74.630361 140.01356  57.32447 157.31945
## Feb 2021      97.766871  65.013949 130.51979  47.67560 147.85814
## Mar 2021      87.947988  54.424762 121.47121  36.67864 139.21734
## Apr 2021     106.878848  73.091123 140.66657  55.20498 158.55271
## May 2021      67.954184  34.136519 101.77185  16.23453 119.67384
## Jun 2021       8.175152 -25.642869  41.99317 -43.54505  59.89535
## Jul 2021      28.565834  -5.252545  62.38421 -23.15491  80.28658
## Aug 2021      70.755067  36.936447 104.57369  19.03395 122.47618
## Sep 2021      96.804117  62.985457 130.62278  45.08294 148.52529
## Oct 2021      92.829567  59.010929 126.64820  41.10842 144.55071
## Nov 2021      97.856909  64.038341 131.67548  46.13587 149.57794
## Dec 2021     162.027872 128.209310 195.84643 110.30684 213.74890
## Jan 2022     106.556935  72.397843 140.71603  54.31511 158.79876
## Feb 2022      94.044177  59.883801 128.20455  41.80039 146.28796
## Mar 2022      85.699626  51.522987 119.87627  33.43097 137.96829
## Apr 2022     106.105570  71.923263 140.28788  53.82824 158.38290
## May 2022      67.861800  33.678849 102.04475  15.58349 120.14011
## Jun 2022       8.254603 -25.928356  42.43756 -44.02372  60.53293
## Jul 2022      28.633312  -5.549655  62.81628 -23.64502  80.91165
## Aug 2022      70.784061  36.601089 104.96703  18.50572 123.06241
## Sep 2022      96.810598  62.627628 130.99357  44.53226 149.08894
## Oct 2022      92.828521  58.645575 127.01147  40.55022 145.10683
## Nov 2022      97.855077  63.672200 132.03795  45.57688 150.13328
## Dec 2022     162.026886 127.844015 196.20976 109.74870 214.30508
```

```
autoplot(fcast)+ggtitle('Forecast of Premier Leaugue goals from 2000-2022')+ylab('goals each month')
```




```
autoplot(fcast,includ=36)+
  ggtitle('Forecast of Premier Leaugue goals from 2018-2022')+
  ylab('goals each month')
```




```
summary(fcast)
```

```
##
## Forecast method: ARIMA(2,0,1)(0,1,1)[12]
##
## Model Information:
## Series: X
## ARIMA(2,0,1)(0,1,1)[12]
##
## Coefficients:
##          ar1      ar2      ma1     sma1
##       0.6408  -0.1793  -0.7021  -0.8527
## s.e.  0.1314   0.0702   0.1233   0.0677
##
## sigma^2 estimated as 650.5:  log likelihood=-1123.75
## AIC=2257.51   AICc=2257.77   BIC=2274.91
##
## Error measures:
##                      ME   RMSE      MAE MPE MAPE      MASE        ACF1
## Training set 0.2754297 24.682 16.29042 NaN  Inf 0.7877699 0.004637159
##
## Forecasts:
##          Point Forecast      Lo 80     Hi 80     Lo 95     Hi 95
## Jan 2021     107.321962  74.630361 140.01356  57.32447 157.31945
## Feb 2021      97.766871  65.013949 130.51979  47.67560 147.85814
```

```
## Mar 2021      87.947988  54.424762 121.47121  36.67864 139.21734
## Apr 2021     106.878848  73.091123 140.66657  55.20498 158.55271
## May 2021      67.954184  34.136519 101.77185  16.23453 119.67384
## Jun 2021       8.175152 -25.642869  41.99317 -43.54505  59.89535
## Jul 2021      28.565834  -5.252545  62.38421 -23.15491  80.28658
## Aug 2021      70.755067  36.936447 104.57369  19.03395 122.47618
## Sep 2021      96.804117  62.985457 130.62278  45.08294 148.52529
## Oct 2021      92.829567  59.010929 126.64820  41.10842 144.55071
## Nov 2021      97.856909  64.038341 131.67548  46.13587 149.57794
## Dec 2021     162.027872 128.209310 195.84643 110.30684 213.74890
## Jan 2022     106.556935  72.397843 140.71603  54.31511 158.79876
## Feb 2022      94.044177  59.883801 128.20455  41.80039 146.28796
## Mar 2022      85.699626  51.522987 119.87627  33.43097 137.96829
## Apr 2022     106.105570  71.923263 140.28788  53.82824 158.38290
## May 2022      67.861800  33.678849 102.04475  15.58349 120.14011
## Jun 2022       8.254603 -25.928356  42.43756 -44.02372  60.53293
## Jul 2022      28.633312  -5.549655  62.81628 -23.64502  80.91165
## Aug 2022      70.784061  36.601089 104.96703  18.50572 123.06241
## Sep 2022      96.810598  62.627628 130.99357  44.53226 149.08894
## Oct 2022      92.828521  58.645575 127.01147  40.55022 145.10683
## Nov 2022      97.855077  63.672200 132.03795  45.57688 150.13328
## Dec 2022     162.026886 127.844015 196.20976 109.74870 214.30508
```