Univariate Forecasts for min__temp__val and heat__days__val

In this part we need to evaluate univariate forecasts of the temperatures for the multiple linear regression. We will thus need to use the predicted values : - of min__temp__val and heat__days__val from February 2017 to July 2017 - of extreme__heat and extreme__cold from March 2017 to July 2017. Here, we will only predict min__temp__val and heat__days__val. In another notebook, we will predict extreme__heat and extreme__cold.

Load the data

```
library(readr)
temp <- na.omit(read_csv("data/temp.csv", col_types = cols(date = col_date(format = "%Y-%m-%d"))))
rmse <- function(l, r) {
  sqrt(sum((l - r)^2)/NROW(l))
}
exp_rmse <- function(l, r) {
  rmse(exp(l), exp(r))
}
```

min__temp__val

Stationarity and integration of the min__temp__val

```
library(forecast)
```

```
## Loading required package: zoo
```

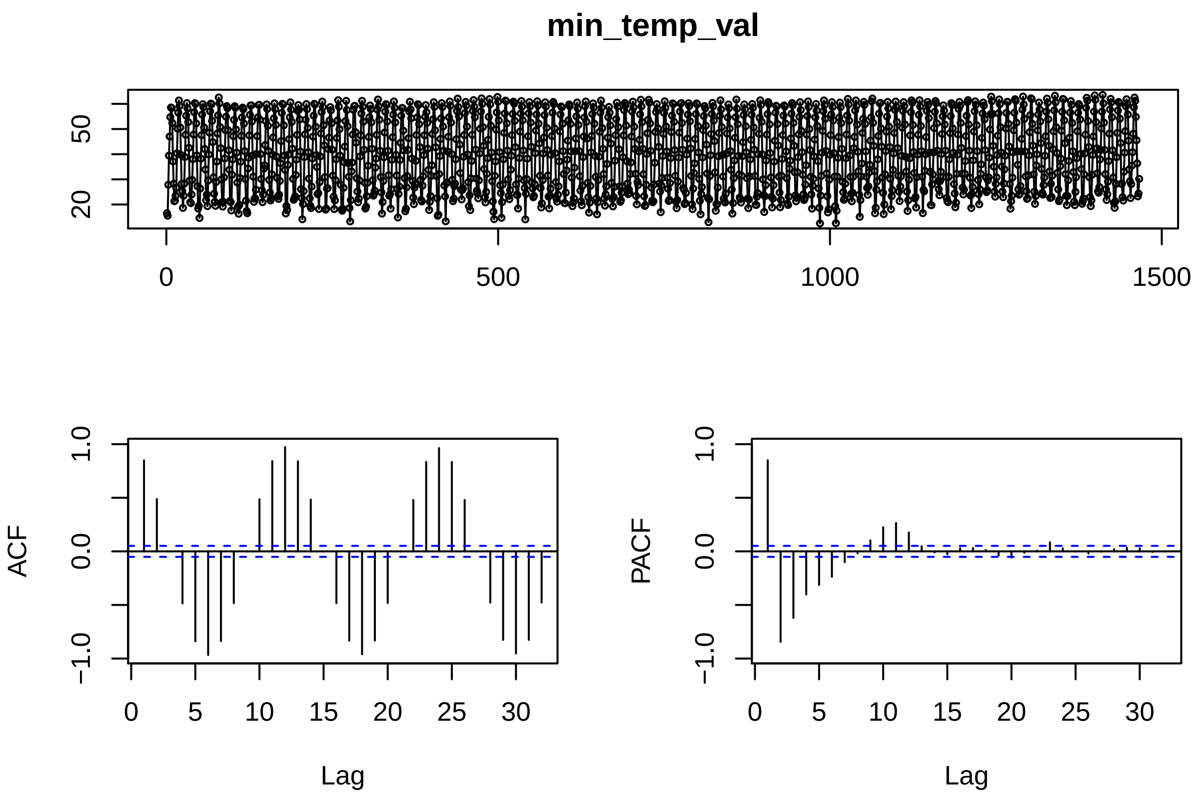
```
##
## Attaching package: 'zoo'
```

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

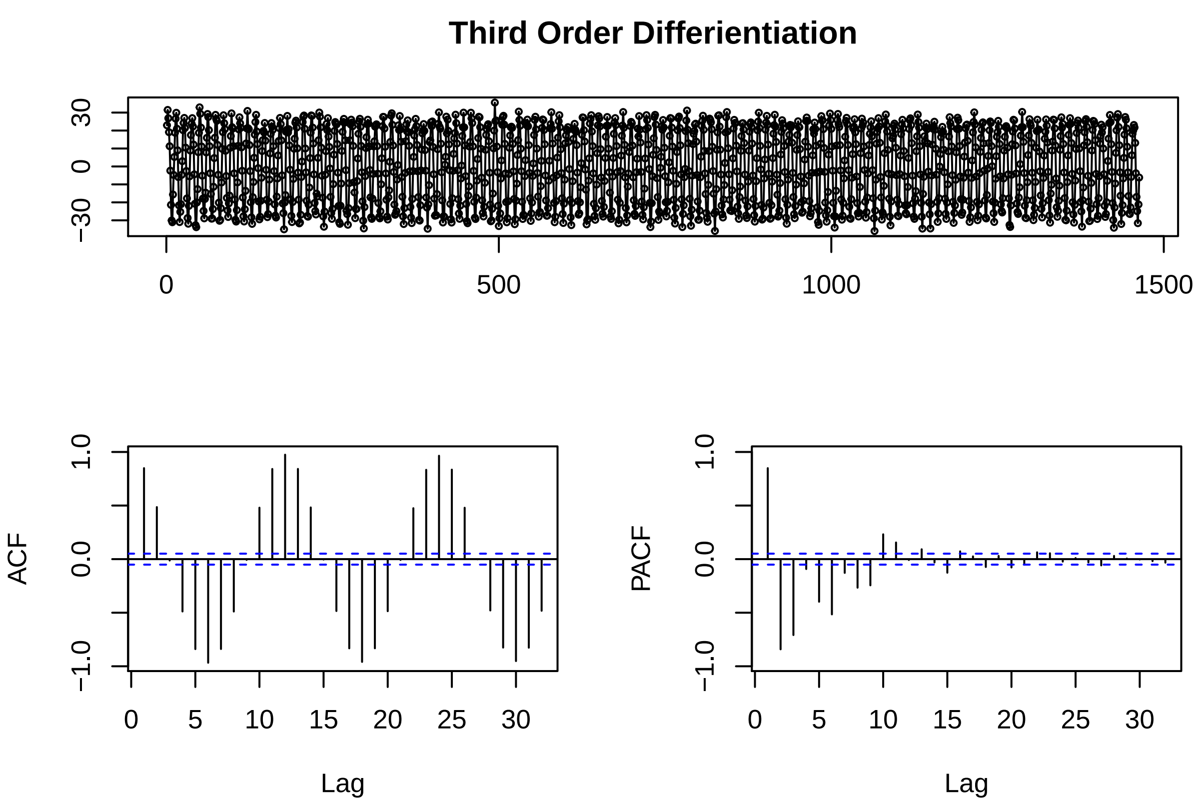
```
## Loading required package: timeDate
```

```
## This is forecast 7.3
```

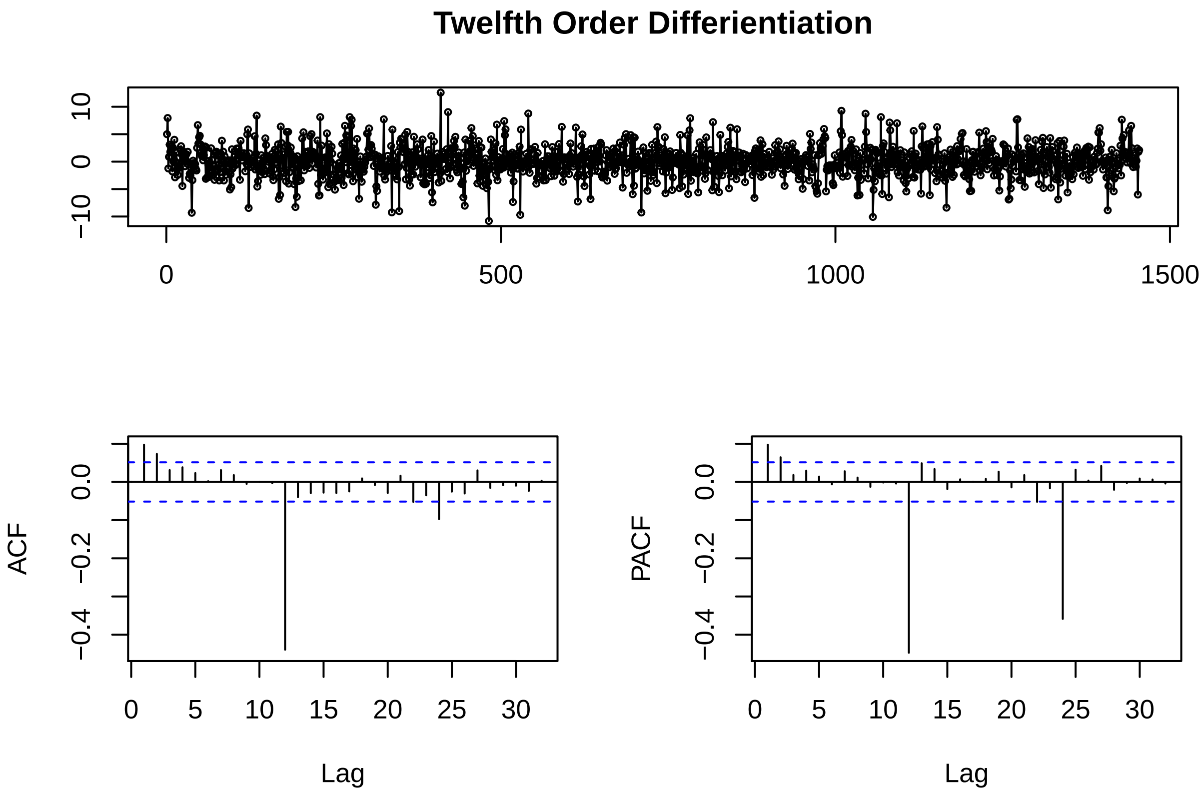
```r
tsdisplay(temp$min_temp_val,main="min_temp_val")
```




```r
tsdisplay(diff(temp$min_temp_val, 3), main="Third Order Differientiation")
```



```r
tsdisplay(diff(temp$min_temp_val, 12), main="Twelfth Order Differientiation")
```



As we can see, we need to do a 12th order differenciation in order to remove the seasonality.

Models evaluation

Let us convert data to a time series

```r
smin_temp_val = ts(temp$min_temp_val, start = c(1895, 1), frequency=12)
train = window(smin_temp_val, start = c(1895, 1), end = c(2016, 1))
test = window(smin_temp_val, start = c(2016,2), end = c(2017, 1))
```

Simple Snaive

We first start with a simple seasonal naive process (ie. repetition of the last temporality).

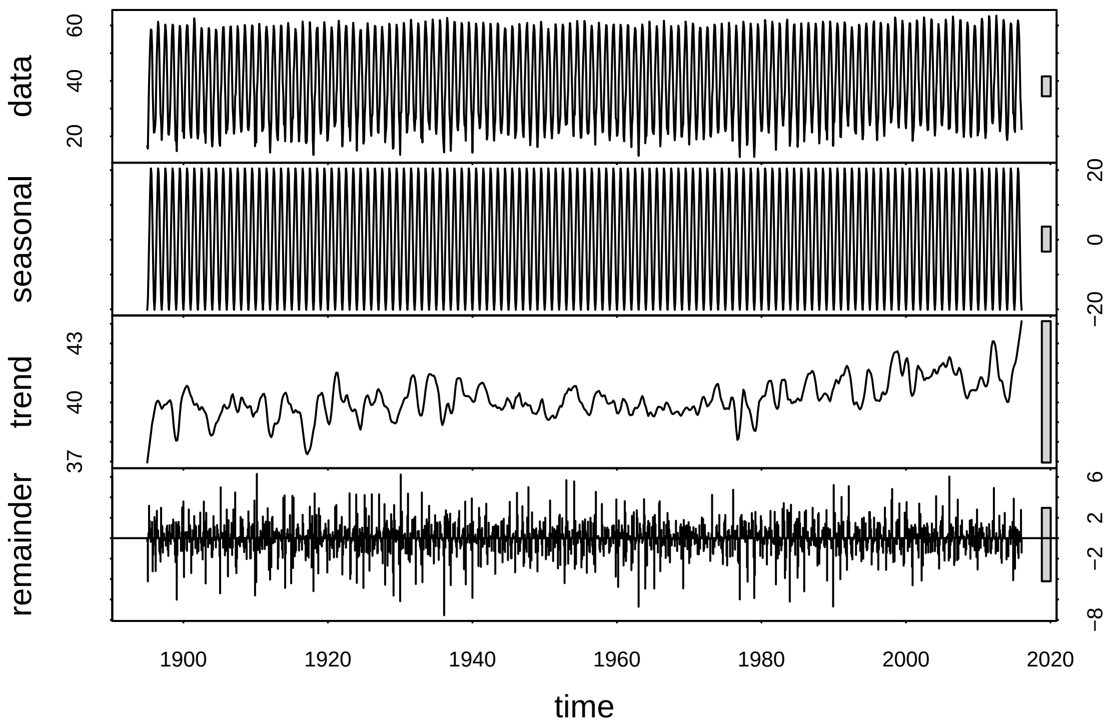
```r
snaive = snaive(train, h = 12)
accuracy(snaive, test)
```

```
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 0.03825815 2.741758 2.022949 -0.4847231 6.877580 1.0000000
## Test set     0.58000000 2.850453 1.976667  1.4572188 6.514494 0.9771212
##                     ACF1 Theil's U
## Training set  0.09912500        NA
## Test set     -0.08763248 0.3401596
```

Time Series Decomposition with min_temp_val

Now for Time Series decomposition.

```
decomp = stl(train, s.window="periodic")
plot(decomp)
```



As previously, our first attempt will just forecast the time series by removing seasonality, and then using the last observation to which we add back the seasonality as the next forecast value.

```
stlfk_naive = forecast(decomp, method="naive", h=12)
summary(stlfk_naive)
```

```
##
## Forecast method: STL +  Random walk
##
## Model Information:
## $drift
## [1] 0
##
## $drift.se
## [1] 0
##
## $sd
## [1] 2.611125
##
## $call
## rwf(y = x, h = h, drift = FALSE, level = level)
##
##
## Error measures:
##                         ME      RMSE       MAE         MPE      MAPE       MASE
```

```
## Training set 0.004180441 2.610229 1.899196 -0.5527495 6.405461 0.9388255
##                    ACF1
## Training set -0.4834527
##
## Forecasts:
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Feb 2016       25.32427 21.97912 28.66941 20.20831 30.44022
## Mar 2016       32.65522 29.31008 36.00036 27.53927 37.77118
## Apr 2016       41.31400 37.96886 44.65914 36.19804 46.42995
## May 2016       50.13459 46.78945 53.47974 45.01864 55.25055
## Jun 2016       58.35673 55.01158 61.70187 53.24077 63.47268
## Jul 2016       63.22406 59.87892 66.56921 58.10811 68.34002
## Aug 2016       61.74920 58.40406 65.09435 56.63325 66.86516
## Sep 2016       54.59187 51.24672 57.93701 49.47591 59.70782
## Oct 2016       44.04394 40.69880 47.38908 38.92799 49.15990
## Nov 2016       33.35246 30.00732 36.69761 28.23651 38.46842
## Dec 2016       25.30653 21.96139 28.65167 20.19058 30.42249
## Jan 2017       22.59000 19.24486 25.93514 17.47405 27.70595
```

```r
accuracy(stlfk_naive, test)
```

```
##                       ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 0.004180441 2.610229 1.899196 -0.5527495 6.405461 0.9388255
## Test set     0.548926772 1.817287 1.513659  1.7801928 4.484761 0.7482437
##                    ACF1 Theil's U
## Training set -0.4834527        NA
## Test set      0.0176467 0.2646893
```

Let's try with exponential smoothing to forecast the seasonally-adjusted series.

```r
stlfk_ets = forecast(decomp, method="ets", h=12)
summary(stlfk_ets)
```

```
##
## Forecast method: STL +  ETS(A,N,N)
##
## Model Information:
## ETS(A,N,N)
##
## Call:
##  ets(y = x, model = etsmodel, allow.multiplicative.trend = allow.multiplicative.trend)
##
##   Smoothing parameters:
##     alpha = 0.0208
##
##   Initial states:
##     l = 39.4037
##
##   sigma:  1.972
##
##      AIC     AICc      BIC
## 12559.19 12559.21 12575.03
##
## Error measures:
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 0.07265587 1.972015 1.458121 -0.4268779 5.006638 0.7207896
```

```
##                     ACF1
## Training set 0.1041581
##
## Forecasts:
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Feb 2016       24.20707 21.67983 26.73431 20.34200 28.07215
## Mar 2016       31.53803 29.01024 34.06581 27.67211 35.40394
## Apr 2016       40.19681 37.66847 42.72514 36.33005 44.06356
## May 2016       49.01740 46.48852 51.54628 45.14981 52.88499
## Jun 2016       57.23953 54.71010 59.76896 53.37110 61.10796
## Jul 2016       62.10687 59.57689 64.63685 58.23760 65.97614
## Aug 2016       60.63201 58.10148 63.16254 56.76190 64.50212
## Sep 2016       53.47467 50.94360 56.00575 49.60373 57.34562
## Oct 2016       42.92675 40.39513 45.45837 39.05497 46.79853
## Nov 2016       32.23527 29.70310 34.76744 28.36265 36.10789
## Dec 2016       24.18934 21.65662 26.72206 20.31588 28.06279
## Jan 2017       21.47281 18.93954 24.00607 17.59851 25.34710
```

```r
accuracy(stlfk_ets, test)
```

```
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 0.07265587 1.972015 1.458121 -0.4268779 5.006638 0.7207896
## Test set     1.66612019 2.403574 1.970577  4.6679047 5.648559 0.9741108
##                   ACF1 Theil's U
## Training set 0.1041581        NA
## Test set     0.0176467 0.3538235
```

As a last attempt, we can also use arima on the seasonally-adjusted data.

```r
stlfk_arima = forecast(decomp, method="arima", h=12)
summary(stlfk_arima)
```

```
##
## Forecast method: STL +  ARIMA(2,1,2) with drift
##
## Model Information:
## Series: x
## ARIMA(2,1,2) with drift
##
## Coefficients:
##           ar1     ar2      ma1      ma2   drift
##       -0.0162  0.0834  -0.8656  -0.1210  0.0015
## s.e.   0.4742  0.0554   0.4762   0.4699  0.0008
##
## sigma^2 estimated as 3.833:  log likelihood=-3034.98
## AIC=6081.96   AICc=6082.02   BIC=6113.65
##
## Error measures:
##                      ME     RMSE      MAE        MPE    MAPE      MASE
## Training set 0.01816957 1.953752 1.443236 -0.5625857 4.95502 0.7134317
##                       ACF1
## Training set -6.348877e-05
##
## Forecasts:
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Feb 2016       24.66053 22.15151 27.16955 20.82331 28.49774
```

```
## Mar 2016       31.61512 29.08863 34.14161 27.75118 35.47906
## Apr 2016       40.22605 37.68838 42.76372 36.34502 44.10708
## May 2016       49.01746 46.47920 51.55571 45.13553 52.89938
## Jun 2016       57.23750 54.69871 59.77630 53.35475 61.12025
## Jul 2016       62.10387 59.56481 64.64294 58.22071 65.98704
## Aug 2016       60.63029 58.09095 63.16963 56.74670 64.51387
## Sep 2016       53.47428 50.93469 56.01387 49.59031 57.35825
## Oct 2016       42.92787 40.38803 45.46772 39.04351 46.81223
## Nov 2016       32.23791 29.69781 34.77801 28.35317 36.12266
## Dec 2016       24.19352 21.65316 26.73387 20.30838 28.07865
## Jan 2017       21.47852 18.93791 24.01913 17.59300 25.36404
```

```r
accuracy(stlfk_arima, test)
```

```
##                      ME     RMSE      MAE        MPE    MAPE      MASE
## Training set 0.01816957 1.953752 1.443236 -0.5625857 4.95502 0.7134317
## Test set     1.61892314 2.332963 1.924086  4.5061107 5.48978 0.9511289
##                       ACF1 Theil's U
## Training set -6.348877e-05        NA
## Test set     -1.861718e-02 0.3507002
```

R picked an ARIMA(2,1,2).

The best model for now when we consider RMSE on the Test set is the STL + Random walk.

Seasonal ARIMA

Auto Arima on the log data

We first look at the auto.arima output.

```r
autofit = auto.arima(train, seasonal=TRUE)
print(autofit)
```

```
## Series: train
## ARIMA(1,0,5)(0,0,2)[12] with non-zero mean
##
## Coefficients:
##          ar1     ma1     ma2      ma3      ma4      ma5    sma1    sma2
##       0.6689  0.1510  0.1220  -0.2583  -0.4240  -0.3827  0.5963  0.3398
## s.e.  0.0293  0.0344  0.0248   0.0202   0.0248   0.0246  0.0290  0.0238
##       intercept
##         40.1787
## s.e.     0.1199
##
## sigma^2 estimated as 14.02:  log likelihood=-3978.89
## AIC=7977.78   AICc=7977.93   BIC=8030.59
```

```r
autofk = forecast(autofit,h=12)
accuracy(autofk,test)
```

```
##                        ME     RMSE      MAE       MPE      MAPE     MASE
## Training set -0.004595546 3.732583 2.947751 -2.906153  9.412789 1.457155
## Test set      3.196560060 7.831015 6.961306  2.639973 18.186374 3.441167
##                    ACF1 Theil's U
## Training set 0.04430466        NA
```

```
## Test set     0.69141446  1.048407
```

Let us now try with our own seasonal ARIMA. Based on the ACF PACF of the annual differentiation, we try with an ARIMA(1,0,5)(0,0,2).

Custom Seasonal Arima on the data

```r
custfit = Arima(train, order=c(1,0,5), seasonal=c(0,0,2))
print(custfit)
```

```
## Series: train
## ARIMA(1,0,5)(0,0,2)[12] with non-zero mean
##
## Coefficients:
##          ar1     ma1     ma2      ma3      ma4      ma5    sma1    sma2
##       0.6689  0.1510  0.1220  -0.2583  -0.4240  -0.3827  0.5963  0.3398
## s.e.  0.0293  0.0344  0.0248   0.0202   0.0248   0.0246  0.0290  0.0238
##       intercept
##         40.1787
## s.e.     0.1199
##
## sigma^2 estimated as 14.02:  log likelihood=-3978.89
## AIC=7977.78   AICc=7977.93   BIC=8030.59
```

```r
custfk = forecast(custfit,h=12)
accuracy(custfk,test)
```

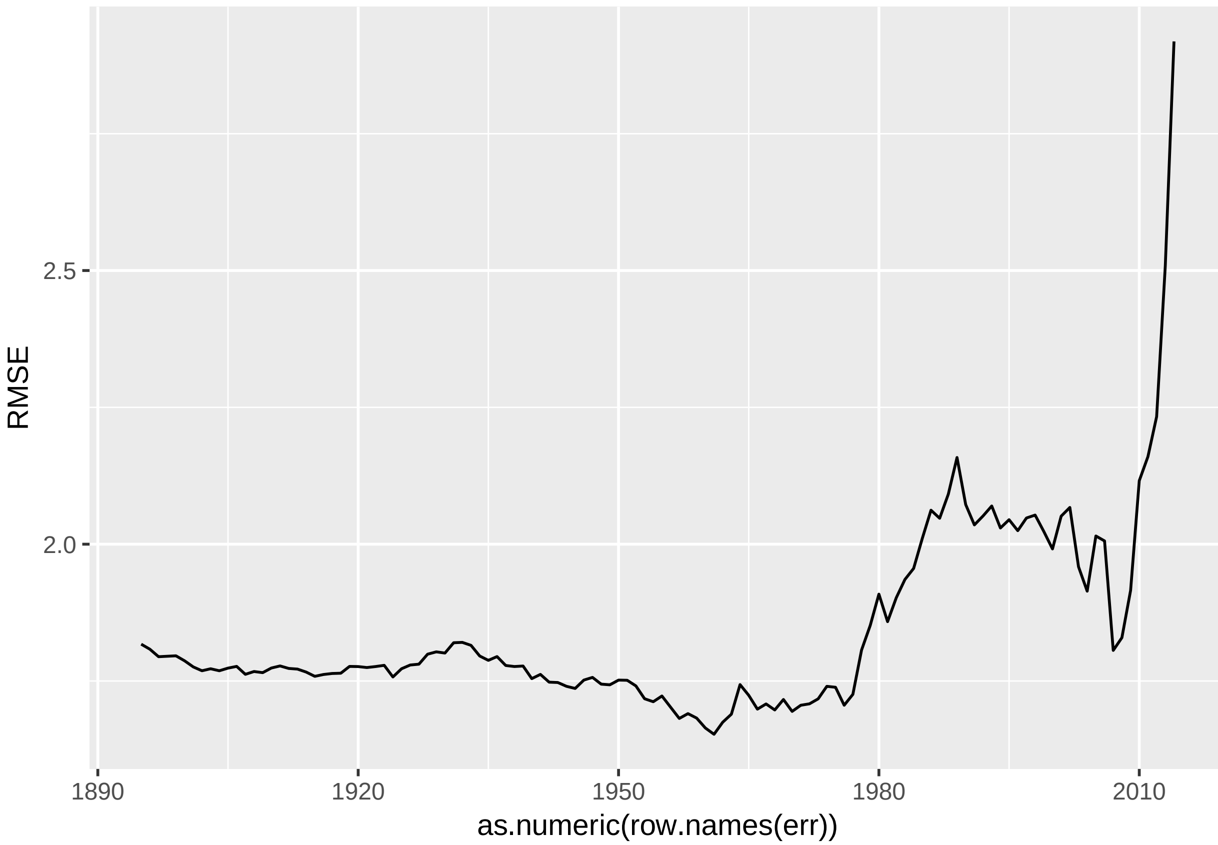
```
##                        ME     RMSE      MAE       MPE      MAPE     MASE
## Training set -0.004595546 3.732583 2.947751 -2.906153  9.412789 1.457155
## Test set      3.196560060 7.831015 6.961306  2.639973 18.186374 3.441167
##                    ACF1 Theil's U
## Training set 0.04430466        NA
## Test set     0.69141446  1.048407
```

Although the statistical tests and criterion seem better here, the error on train and test sets are higher, so we keep the STL + Random walk.

Rolling Window Evaluation

To try more model, we would like to setup a rolling window. Let us find the optimal size of that window using the STL + Random walk we just fitted.

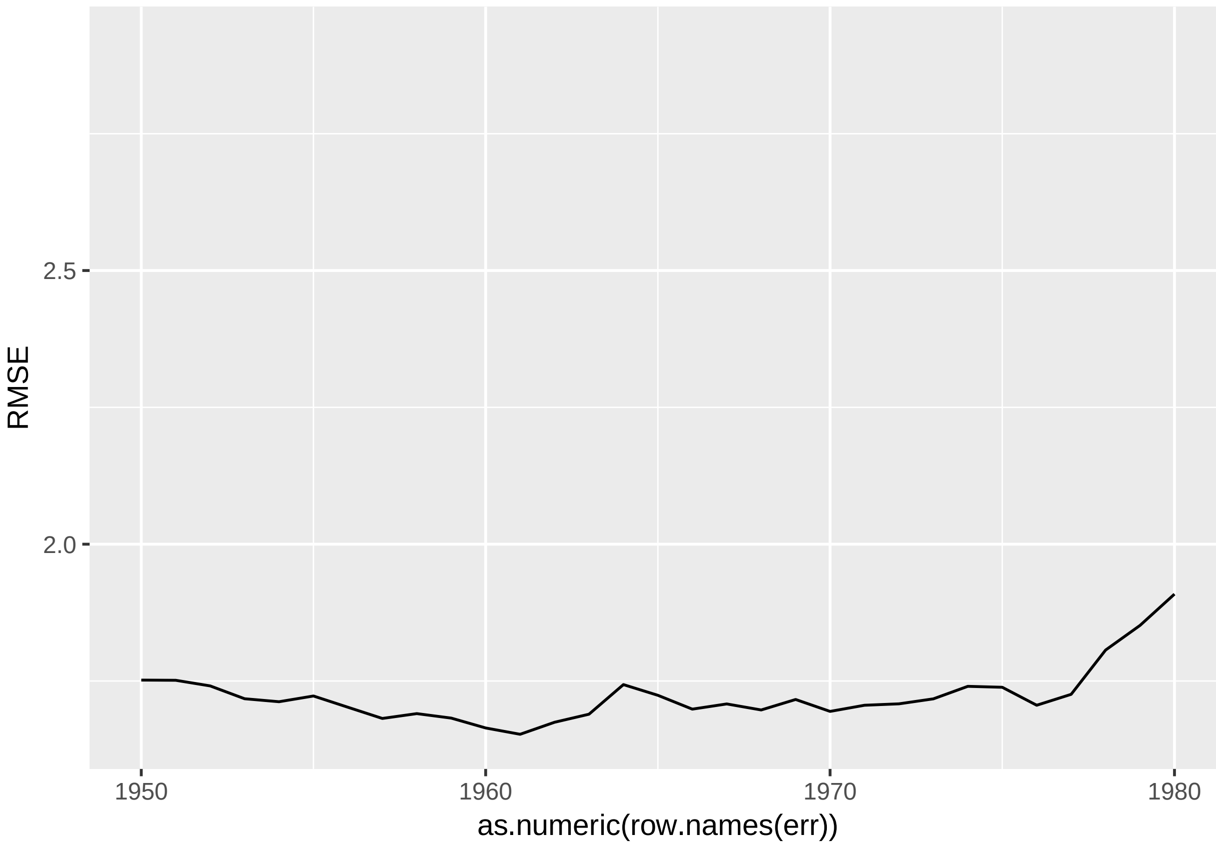
```r
library(ggplot2)
err = data.frame("RMSE" = rep(0, 2014-1895+1), row.names = seq(1895, 2014))
for (y in 1895:2014){
    model = stl(window(train, start=c(y,1)), s.window="periodic")
    rmse = accuracy(forecast(model, method="naive", h=12),test)[2,"RMSE"]
    err[as.character(y),] = rmse
}
ggplot(err)+geom_line(aes(x=as.numeric(row.names(err)), y=RMSE))
```



Let us take a closer look at the years between 1950 and 1980.

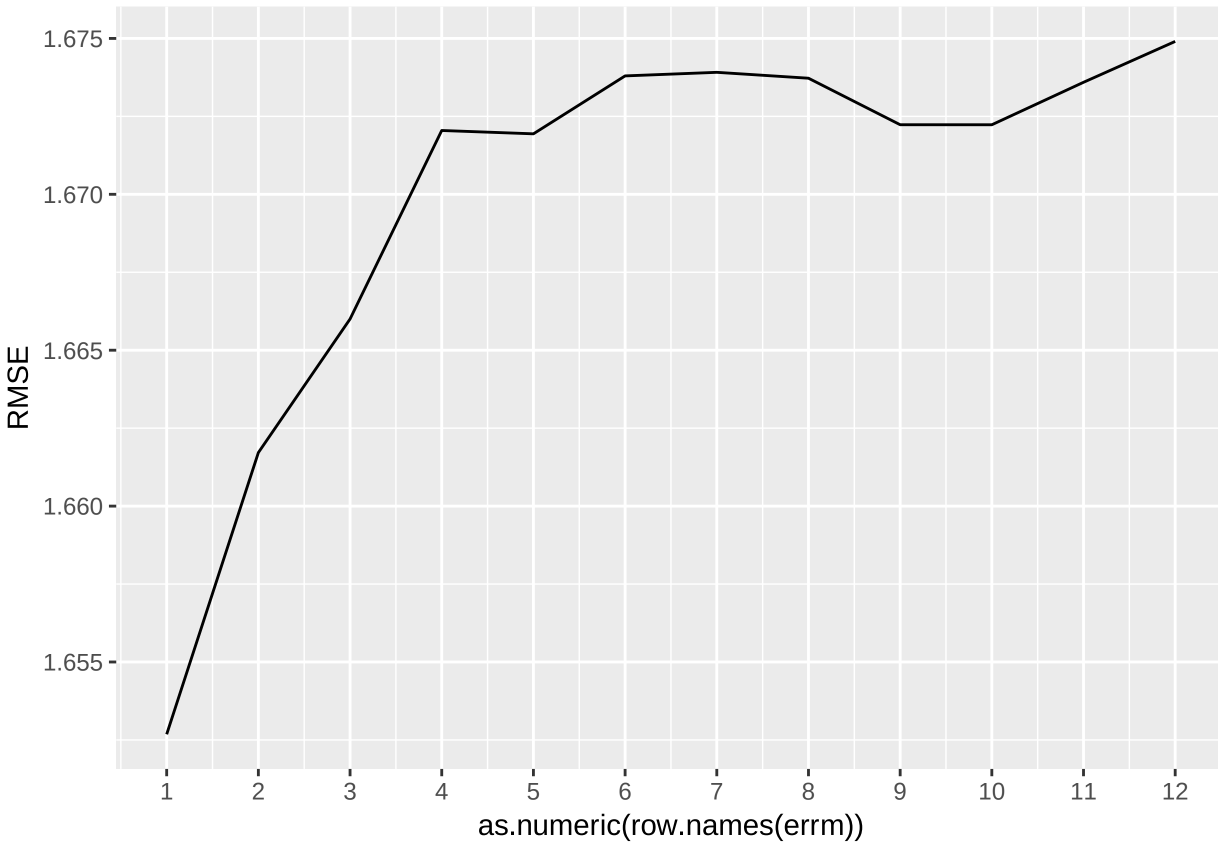
```
ggplot(err)+geom_line(aes(x=as.numeric(row.names(err)), y=RMSE))+xlim(1950, 1980)
```

```
## Warning: Removed 89 rows containing missing values (geom_path).
```



Let's take 1961, that is to say 55 years !

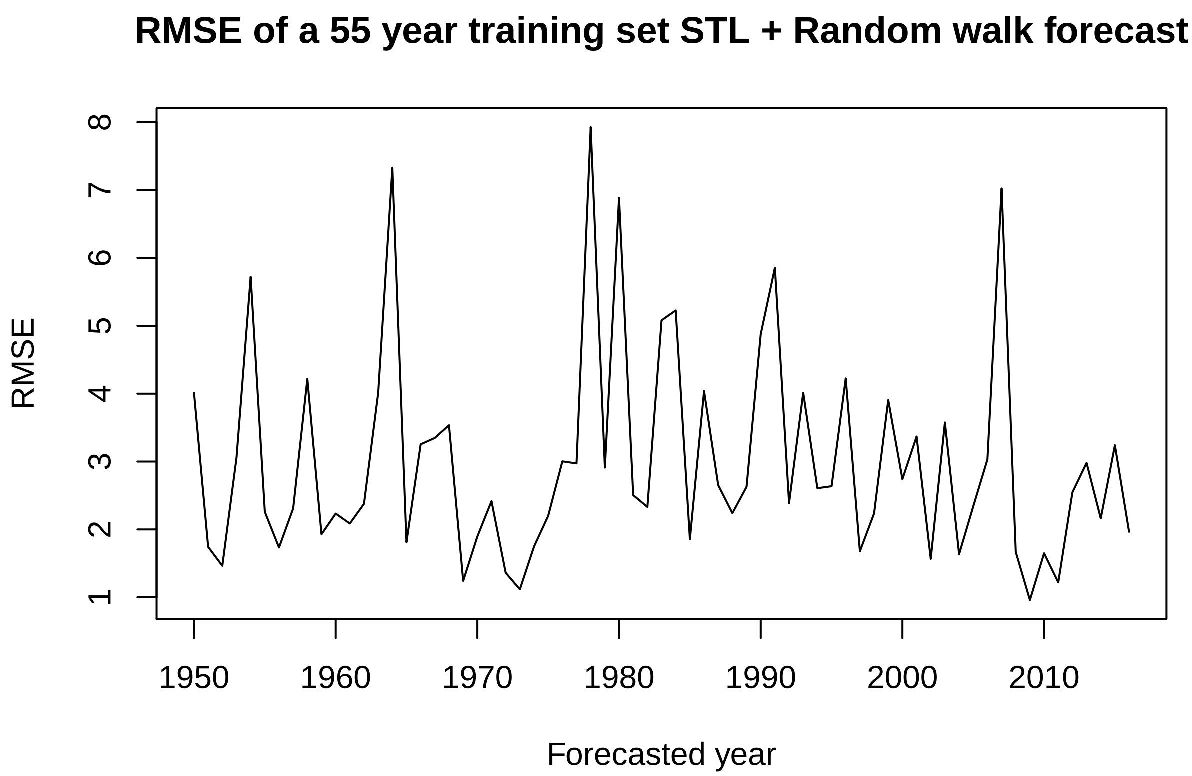
```
errm = data.frame("RMSE" = rep(0, 12), row.names = seq(1, 12))
for (m in 1:12){
    model = stl(window(train, start=c(1961,m)), s.window="periodic")
    rmse = accuracy(forecast(model, method="naive", h=12),test)[2,"RMSE"]
    errm[as.character(m),] = rmse
}
ggplot(errm)+geom_line(aes(x=as.numeric(row.names(errm)), y=RMSE))+scale_x_continuous(breaks=seq(1,12))
```



We will thus use a rolling window of 7 full years.

```
rollwitme = function(lgas){
    RMSE = rep(0,66)
    for (y in 1895:1961){
        tr = window(lgas, start=c(y,2), end=c(y+54,1))
        te = window(lgas, start=c(y+54,2), end=c(y+55,1))
        model = stl(tr, s.window="periodic")
        RMSE[y+1-1895] = accuracy(forecast(model, method="naive", h=12),te)[2,"RMSE"]
    }
    return(RMSE)
}
```

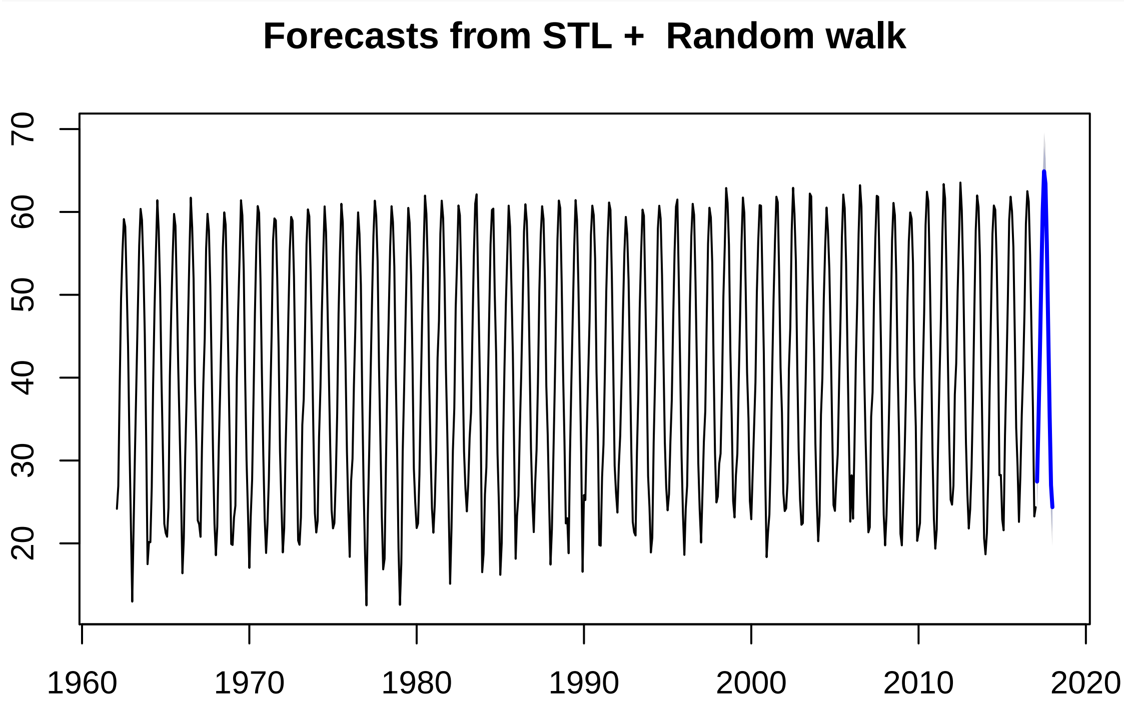
```
plot(1950:2016, rollwitme(smin_temp_val), type = 'l', main = "RMSE of a 55 year training set STL + Rand
```



We notice a huge variability in the forecast performances over time !

Creation of the needed forecast : from Feb. 2017 to Feb. 2018 (we will only use from Feb. to July 2017).

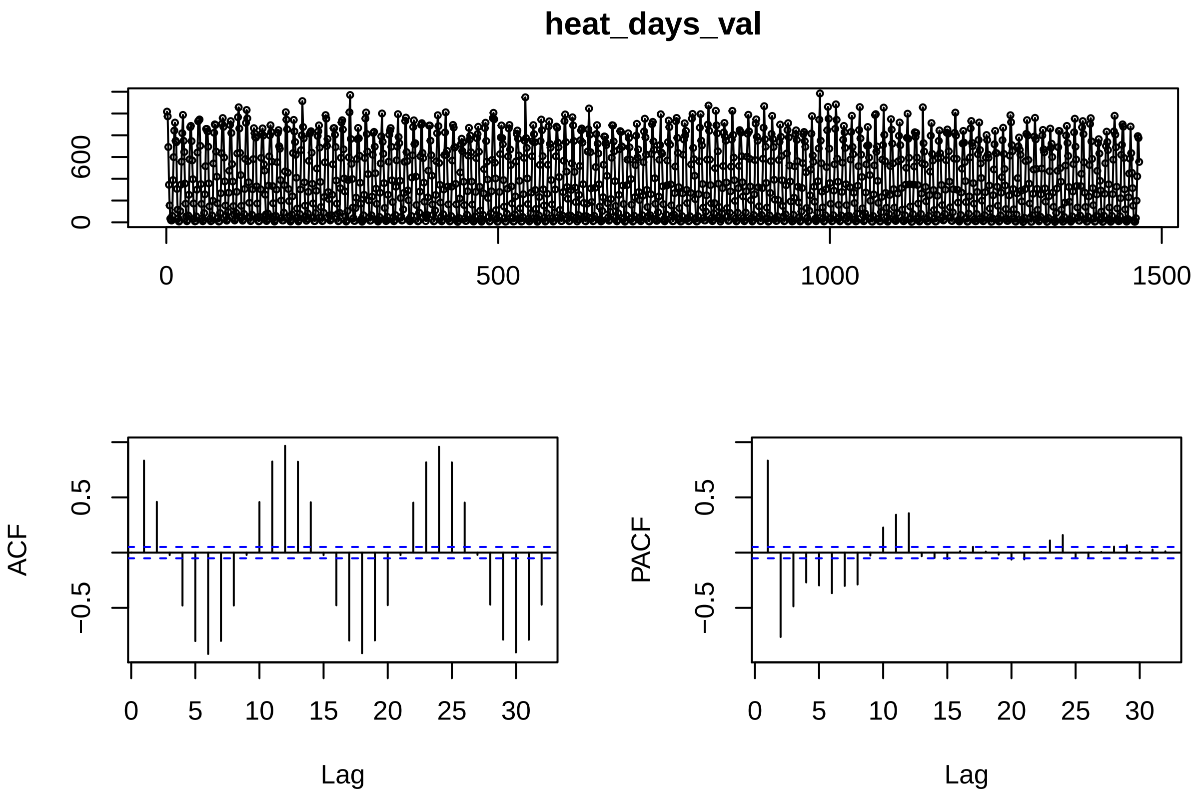
```
train_final = window(smin_temp_val, start=c(1962,2), end=c(1962+55,1))
model = stl(train_final, s.window="periodic")
yearly_forecast = forecast(model, method="naive", h=12)
plot(yearly_forecast)
```



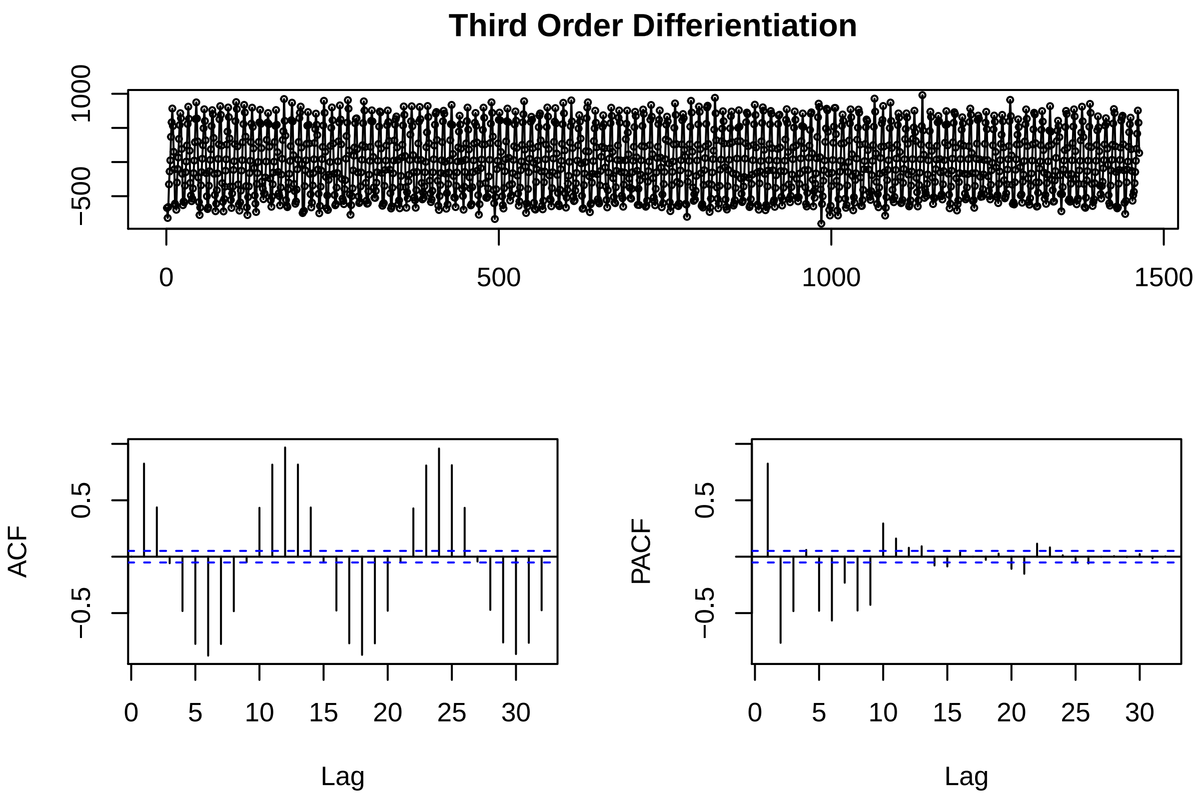
heat__days__val

Stationarity and integration of the heat__days__val

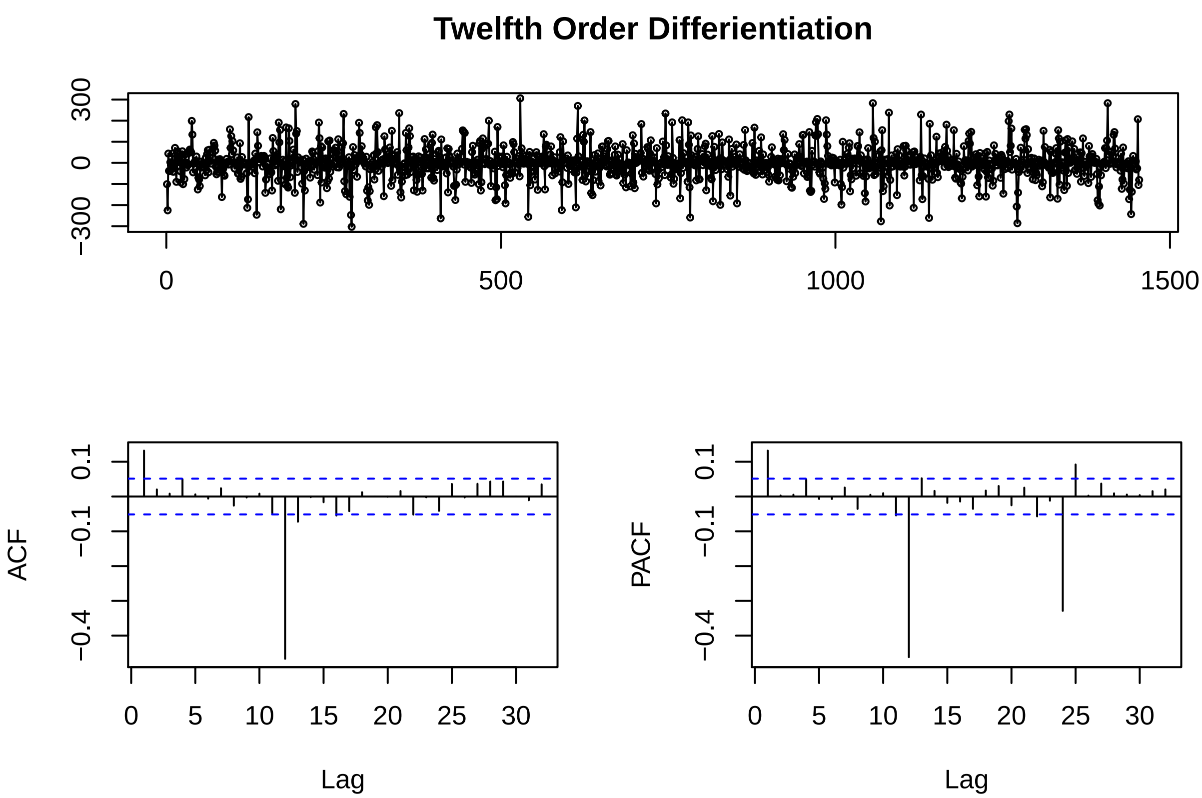
```r
library(forecast)
tsdisplay(temp$heat_days_val,main="heat_days_val")
```




```r
tsdisplay(diff(temp$heat_days_val, 3), main="Third Order Differientiation")
```




```r
tsdisplay(diff(temp$heat_days_val, 12), main="Twelfth Order Differientiation")
```



As previously, we need to do a 12th order differenciation in order to remove the seasonality.

Models evaluation

Let us convert data to a time series

```
sheat_days_val = ts(temp$heat_days_val, start = c(1895, 1), frequency=12)
train = window(sheat_days_val, start = c(1895, 1), end = c(2016, 1))
test = window(sheat_days_val, start = c(2016,2), end = c(2017, 1))
```

Simple Snaive

We first start with a simple seasonal naive process (ie. repetition of the last temporality).

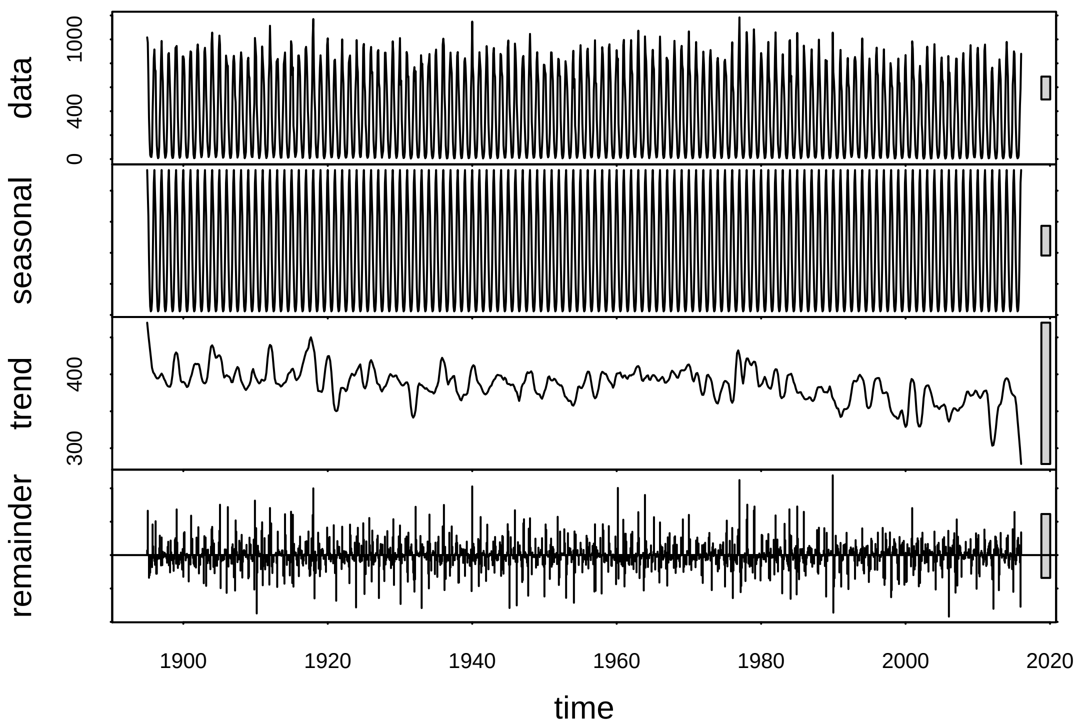
```
snaive = snaive(train, h = 12)
accuracy(snaive, test)
```

```
##                      ME      RMSE      MAE        MPE     MAPE     MASE
## Training set  -0.7175573  73.76217 48.51492  -4.291583 20.45033 1.000000
## Test set     -24.5000000 105.92057 67.50000 -13.097255 24.72578 1.391325
##                    ACF1 Theil's U
## Training set 0.13200394        NA
## Test set     0.04852778 0.2587805
```

Time Series Decomposition with min__temp__val

Now for Time Series decomposition.

```
decomp = stl(train, s.window="periodic")
plot(decomp)
```



As previously, our first attempt will just forecast the time series by removing seasonality, and then using the last observation to which we add back the seasonality as the next forecast value.

```r
stlfk_naive = forecast(decomp, method="naive", h=12)
summary(stlfk_naive)
```

```
##
## Forecast method: STL +  Random walk
##
## Model Information:
## $drift
## [1] 0
##
## $drift.se
## [1] 0
##
## $sd
## [1] 69.38436
##
## $call
## rwf(y = x, h = h, drift = FALSE, level = level)
##
##
## Error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -0.09435262 69.36053 46.56554 -2.124803 26.73723 0.9598189
##                    ACF1
## Training set -0.4405828
##
## Forecasts:
##          Point Forecast      Lo 80     Hi 80      Lo 95     Hi 95
## Feb 2016     726.391283  637.50219 815.28038  590.44715  862.3354
## Mar 2016     584.350500  495.46141 673.23959  448.40636  720.2946
## Apr 2016     313.854990  224.96590 402.74408  177.91085  449.7991
## May 2016     123.921473   35.03238 212.81057  -12.02266  259.8656
## Jun 2016       2.086111  -86.80298  90.97520 -133.85803  138.0302
## Jul 2016     -29.988918 -118.87801  58.90017 -165.93305  105.9552
## Aug 2016     -23.035134 -111.92423  65.85396 -158.97927  112.9090
## Sep 2016      33.083943  -55.80515 121.97304 -102.86019  169.0281
## Oct 2016     237.168855  148.27976 326.05795  101.22472  373.1130
## Nov 2016     516.824034  427.93494 605.71313  380.87990  652.7682
## Dec 2016     797.026640  708.13755 885.91573  661.08250  932.9708
## Jan 2017     880.000000  791.11091 968.88909  744.05586 1015.9441
```

```r
accuracy(stlfk_naive, test)
```

```
##                        ME     RMSE      MAE        MPE      MAPE      MASE
## Training set  -0.09435262 69.36053 46.56554  -2.124803  26.73723 0.9598189
## Test set     -28.97364824 64.29206 48.15324 100.520658 116.93311 0.9925449
##                    ACF1 Theil's U
## Training set -0.4405828        NA
## Test set      0.3326782 0.8786406
```

Let's try with exponential smoothing to forecast the seasonally-adjusted series.

```r
stlfk_ets = forecast(decomp, method="ets", h=12)
summary(stlfk_ets)
```

```
##
```

```
## Forecast method: STL +  ETS(A,N,N)
##
## Model Information:
## ETS(A,N,N)
##
## Call:
##  ets(y = x, model = etsmodel, allow.multiplicative.trend = allow.multiplicative.trend)
##
##   Smoothing parameters:
##     alpha = 0.0151
##
##   Initial states:
##     l = 404.6536
##
##   sigma:  53.3483
##
##      AIC     AICc      BIC
## 22142.56 22142.58 22158.40
##
## Error measures:
##                     ME     RMSE      MAE       MPE    MAPE      MASE
## Training set -2.081331 53.34834 36.42659 -4.748322 28.6741 0.7508327
##                   ACF1
## Training set 0.1404148
##
## Forecasts:
##          Point Forecast     Lo 80     Hi 80      Lo 95     Hi 95
## Feb 2016      738.51616 670.14751 806.88481  633.95533 843.07698
## Mar 2016      596.47537 528.09891 664.85184  491.90260 701.04815
## Apr 2016      325.97986 257.59559 394.36414  221.39514 430.56459
## May 2016      136.04635  67.65426 204.43843   31.44968 240.64302
## Jun 2016       14.21098 -54.18891  82.61088  -90.39763 118.81960
## Jul 2016      -17.86404 -86.27175  50.54366 -122.48460  86.75651
## Aug 2016      -10.91026 -79.32578  57.50525 -115.54276  93.72224
## Sep 2016       45.20882 -23.21451 113.63214  -59.43563 149.85326
## Oct 2016      249.29373 180.86260 317.72486  144.63735 353.95011
## Nov 2016      528.94891 460.50997 597.38784  424.28059 633.61723
## Dec 2016      809.15151 740.70477 877.59825  704.47126 913.83177
## Jan 2017      892.12487 823.67033 960.57942  787.43268 996.81707
```

```r
accuracy(stlfk_ets, test)
```

```
##                      ME     RMSE      MAE       MPE     MAPE      MASE
## Training set  -2.081331 53.34834 36.42659 -4.748322 28.67410 0.7508327
## Test set     -41.098521 70.59097 51.35135 53.388853 76.37106 1.0584651
##                   ACF1 Theil's U
## Training set 0.1404148        NA
## Test set     0.3326782 0.5698336
```

As a last attempt, we can also use arima on the seasonally-adjusted data.

```r
stlfk_arima = forecast(decomp, method="arima", h=12)
summary(stlfk_arima)
```

```
##
## Forecast method: STL +  ARIMA(0,1,3)
```

```
##
## Model Information:
## Series: x
## ARIMA(0,1,3)
##
## Coefficients:
##           ma1      ma2      ma3
##       -0.8437  -0.0968  -0.0450
## s.e.   0.0263   0.0334   0.0259
##
## sigma^2 estimated as 2793:  log likelihood=-7821.26
## AIC=15650.51   AICc=15650.54   BIC=15671.63
##
## Error measures:
##                     ME    RMSE      MAE       MPE     MAPE      MASE
## Training set -2.611127 52.7776 35.83986 -5.827953 26.18488 0.7387389
##                      ACF1
## Training set -0.002999695
##
## Forecasts:
##          Point Forecast     Lo 80     Hi 80      Lo 95     Hi 95
## Feb 2016      732.65773 664.92723 800.38824  629.07286 836.24260
## Mar 2016      598.19648 529.64324 666.74972  493.35334 703.03962
## Apr 2016      326.72897 258.05740 395.40053  221.70487 431.75306
## May 2016      136.79545  68.11683 205.47406   31.76057 241.83033
## Jun 2016       14.96009 -53.72559  83.64576  -90.08558 120.00575
## Jul 2016      -17.11494 -85.80767  51.57778 -122.17140  87.94151
## Aug 2016      -10.16116 -78.86094  58.53862 -115.22840  94.90608
## Sep 2016       45.95792 -22.74891 114.66475  -59.12011 151.03594
## Oct 2016      250.04283 181.32895 318.75671  144.95402 355.13164
## Nov 2016      529.69801 460.97708 598.41894  424.59842 634.79760
## Dec 2016      809.90062 741.17264 878.62859  704.79024 915.01099
## Jan 2017      892.87398 824.13895 961.60900  787.75282 997.99513
```

```r
accuracy(stlfk_arima, test)
```

```
##                      ME     RMSE      MAE       MPE     MAPE      MASE
## Training set  -2.611127 52.77760 35.83986 -5.827953 26.18488 0.7387389
## Test set     -41.377997 70.41489 51.13142 50.545718 73.92996 1.0539320
##                      ACF1 Theil's U
## Training set -0.002999695        NA
## Test set      0.319548556 0.5574365
```

R picked an ARIMA(0,1,3).

The best model for now when we consider RMSE on the Test set is still the STL + Random walk.

Seasonal ARIMA

Auto Arima on the log data

We first look at the auto.arima output.

```r
autofit = auto.arima(train, seasonal=TRUE)
print(autofit)
```

```
## Series: train
## ARIMA(1,0,5)(0,0,2)[12] with non-zero mean
##
## Coefficients:
##          ar1     ma1     ma2      ma3      ma4      ma5    sma1    sma2
##       0.6436  0.1028  0.0463  -0.2717  -0.3473  -0.3371  0.5711  0.3630
## s.e.  0.0301  0.0345  0.0246   0.0224   0.0248   0.0237  0.0300  0.0239
##       intercept
##        386.1738
## s.e.     2.6853
##
## sigma^2 estimated as 9492:  log likelihood=-8714
## AIC=17448   AICc=17448.15   BIC=17500.81
```

```
autofk = forecast(autofit,h=12)
accuracy(autofk,test)
```

```
##                      ME      RMSE       MAE       MPE     MAPE     MASE
## Training set   0.1617336  97.12316  76.08718 -125.7192 138.8361 1.568325
## Test set     -71.6320733 182.19942 166.42356 -660.6438 673.1720 3.430358
##                    ACF1 Theil's U
## Training set 0.04203708        NA
## Test set     0.62740368  6.604064
```

Let us now try with our own seasonal ARIMA. Based on the ACF PACF of the annual differentiation, we try with an ARIMA(1,0,5)(0,0,2).

Custom Seasonal Arima on the data

```
custfit = Arima(train, order=c(1,0,5), seasonal=c(0,0,2))
print(custfit)
```

```
## Series: train
## ARIMA(1,0,5)(0,0,2)[12] with non-zero mean
##
## Coefficients:
##          ar1     ma1     ma2      ma3      ma4      ma5    sma1    sma2
##       0.6436  0.1028  0.0463  -0.2717  -0.3473  -0.3371  0.5711  0.3630
## s.e.  0.0301  0.0345  0.0246   0.0224   0.0248   0.0237  0.0300  0.0239
##       intercept
##        386.1738
## s.e.     2.6853
##
## sigma^2 estimated as 9492:  log likelihood=-8714
## AIC=17448   AICc=17448.15   BIC=17500.81
```

```
custfk = forecast(custfit,h=12)
accuracy(custfk,test)
```

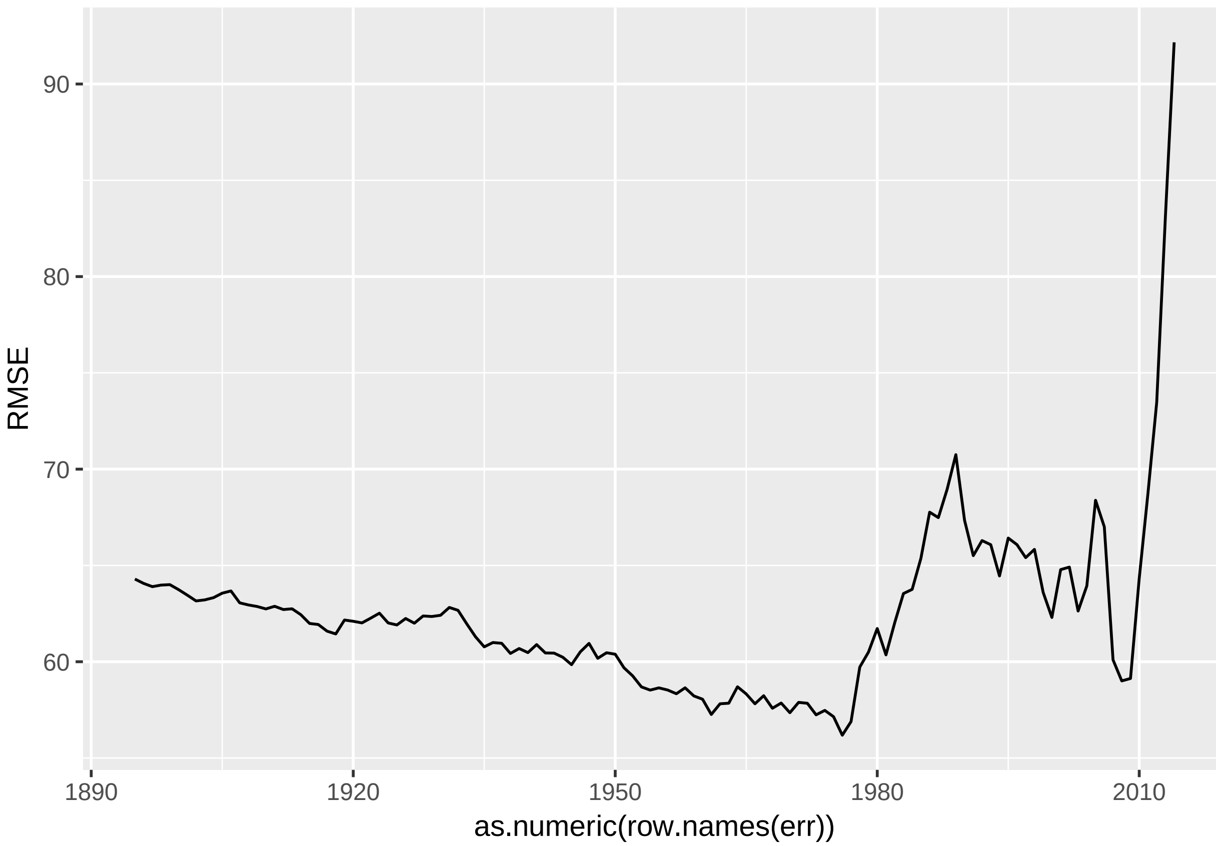
```
##                      ME      RMSE       MAE       MPE     MAPE     MASE
## Training set   0.1617336  97.12316  76.08718 -125.7192 138.8361 1.568325
## Test set     -71.6320733 182.19942 166.42356 -660.6438 673.1720 3.430358
##                    ACF1 Theil's U
## Training set 0.04203708        NA
## Test set     0.62740368  6.604064
```

Here again, we keep the STL + Random walk.

Rolling Window Evaluation

To try more model, we would like to setup a rolling window. Let us find the optimal size of that window using the STL + Random walk we just fitted.

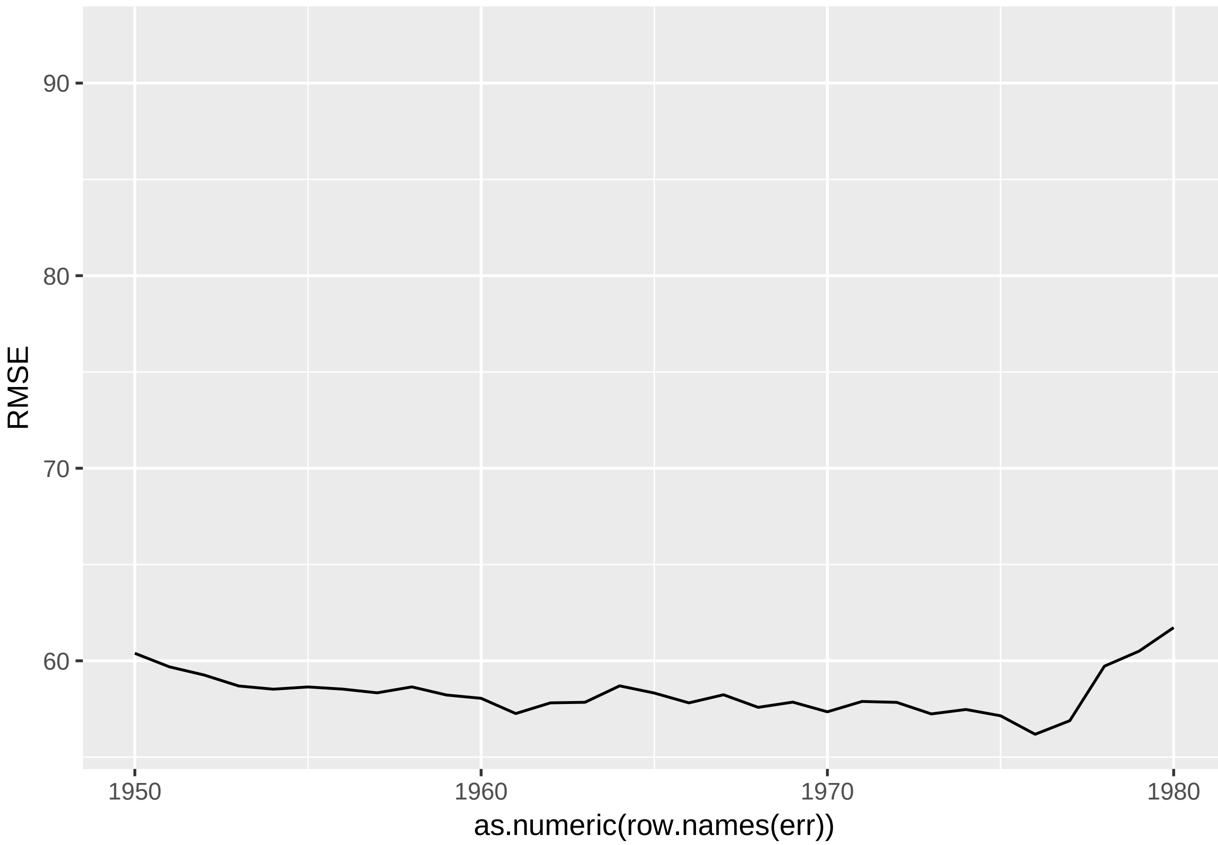
```
library(ggplot2)
err = data.frame("RMSE" = rep(0, 2014-1895+1), row.names = seq(1895, 2014))
for (y in 1895:2014){
    model = stl(window(train, start=c(y,1)), s.window="periodic")
    rmse = accuracy(forecast(model, method="naive", h=12),test)[2,"RMSE"]
    err[as.character(y),] = rmse
}
ggplot(err)+geom_line(aes(x=as.numeric(row.names(err)), y=RMSE))
```



Let us take a closer look at the years between 1950 and 1980.

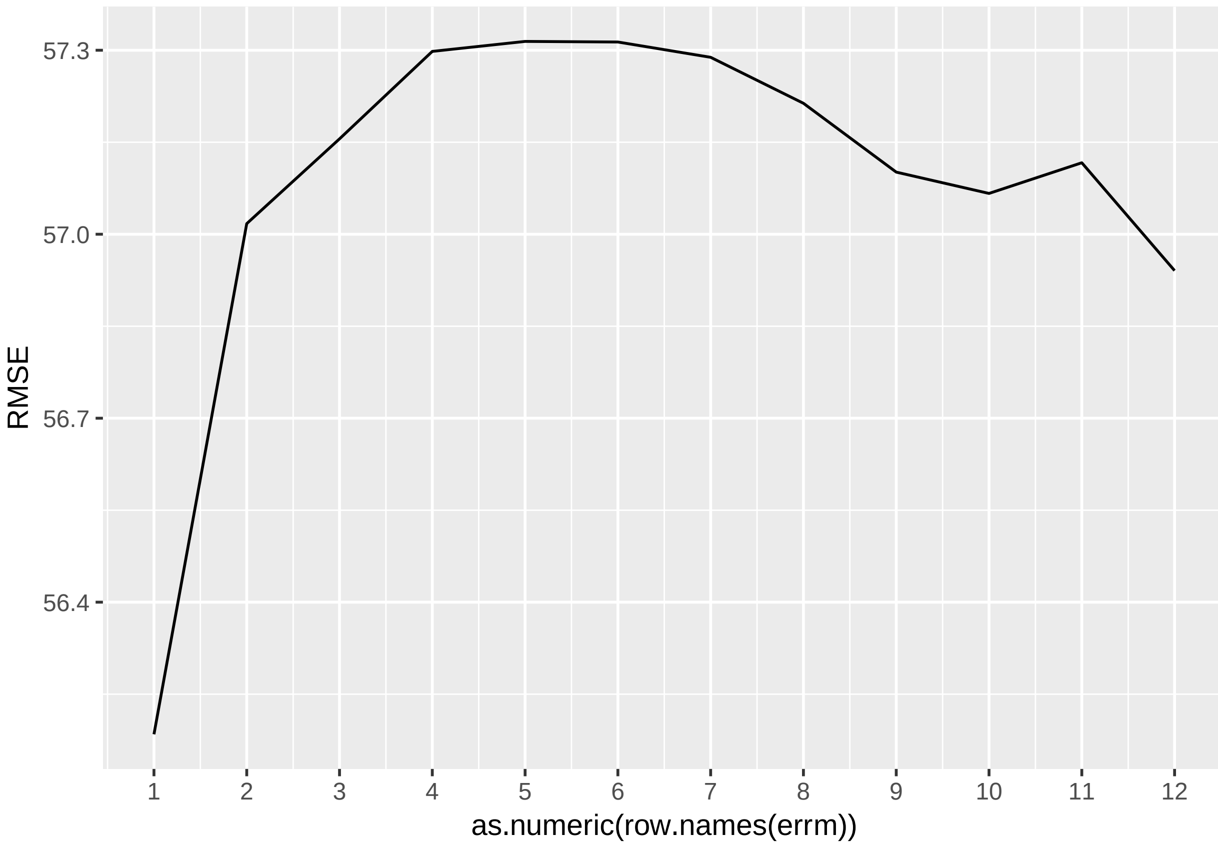
```
ggplot(err)+geom_line(aes(x=as.numeric(row.names(err)), y=RMSE))+xlim(1950, 1980)
```

```
## Warning: Removed 89 rows containing missing values (geom_path).
```



Let us take a closer look at the year 1976.

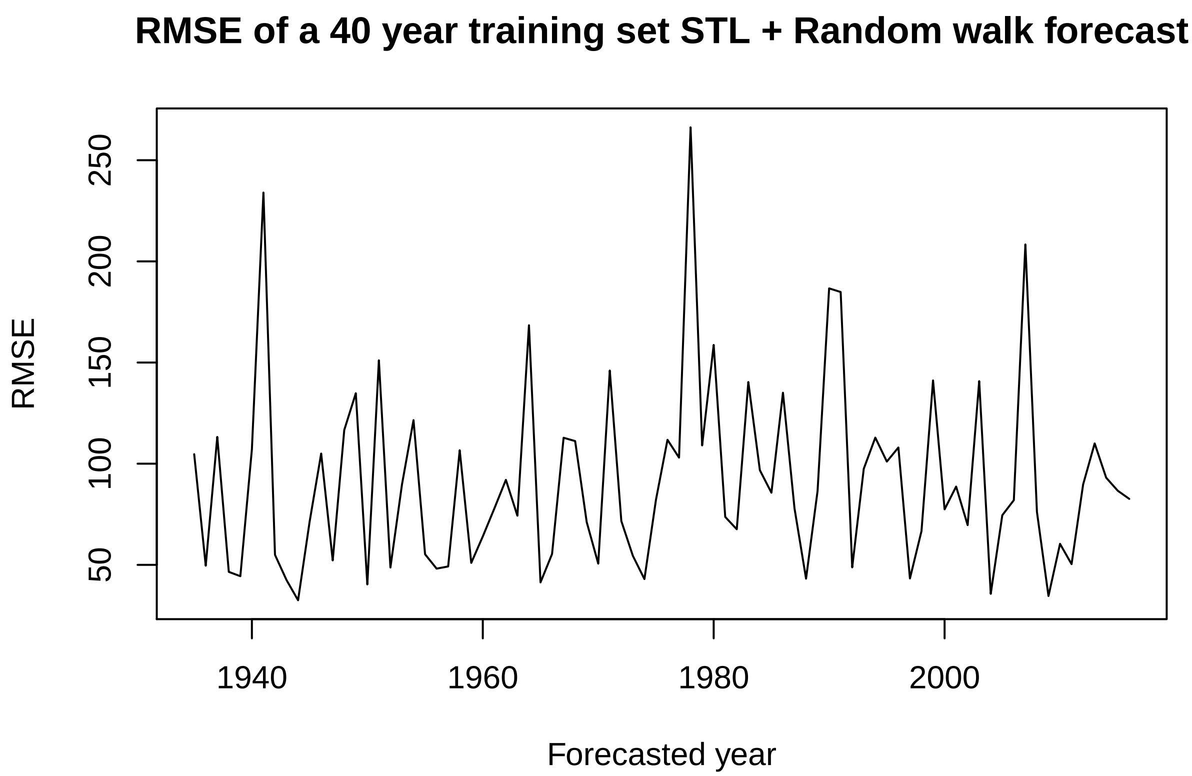
```
errm = data.frame("RMSE" = rep(0, 12), row.names = seq(1, 12))
for (m in 1:12){
    model = stl(window(train, start=c(1976,m)), s.window="periodic")
    rmse = accuracy(forecast(model, method="naive", h=12),test)[2,"RMSE"]
    errm[as.character(m),] = rmse
}
ggplot(errm)+geom_line(aes(x=as.numeric(row.names(errm)), y=RMSE))+scale_x_continuous(breaks=seq(1,12))
```



We will still use a rolling window of 40 full years !

```
rollwitme = function(lgas){
    RMSE = rep(0,81)
    for (y in 1895:1976){
        tr = window(lgas, start=c(y,2), end=c(y+39,1))
        te = window(lgas, start=c(y+39,2), end=c(y+40,1))
        model = stl(tr, s.window="periodic")
        RMSE[y+1-1895] = accuracy(forecast(model, method="naive", h=12),te)[2,"RMSE"]
    }
    return(RMSE)
}
```

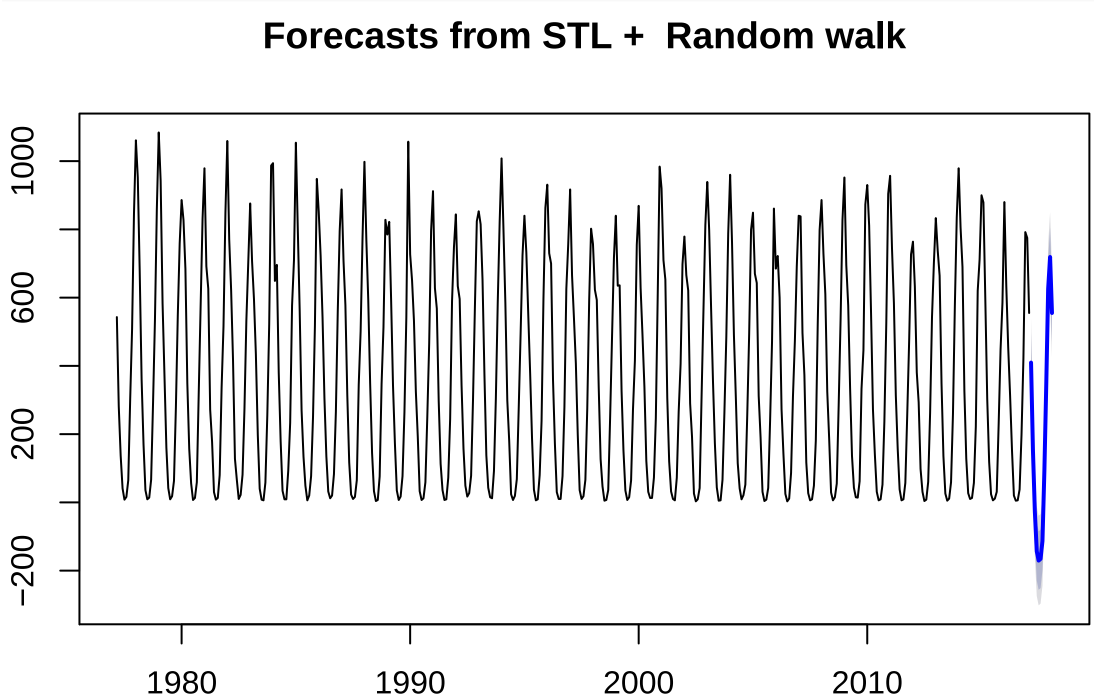
```
plot(1935:2016, rollwitme(sheat_days_val), type = 'l', main = "RMSE of a 40 year training set STL + Ran
```



Here again we notice a huge variability in the forecast performances !

Creation of the needed forecast : from Feb. 2017 to Feb. 2018 (we will only use from Feb. to July 2017).

```
train_final = window(sheat_days_val, start=c(1977,3), end=c(1977+40,2))
model = stl(train_final, s.window="periodic")
yearly_forecast = forecast(model, method="naive", h=12)
plot(yearly_forecast)
```



Warning :

Since here there are negative predictions, we will replace all negative predictions with 0 !