Day 21 - ARIMA models

Introduction

We finally combine all all the stochastic error models we have considered this semester, culminating in the seasonal ARIMA model.

```
# packages
library(tidyverse)
library(lubridate)
library(forecast)
```

Review(ish)

Recall the random walk model, $x_t = x_{t-1} + w_t$, where w_t is a white noise series. Compute the *first-order differences* for the model: $\nabla x_t = x_t - x_{t-1}$.

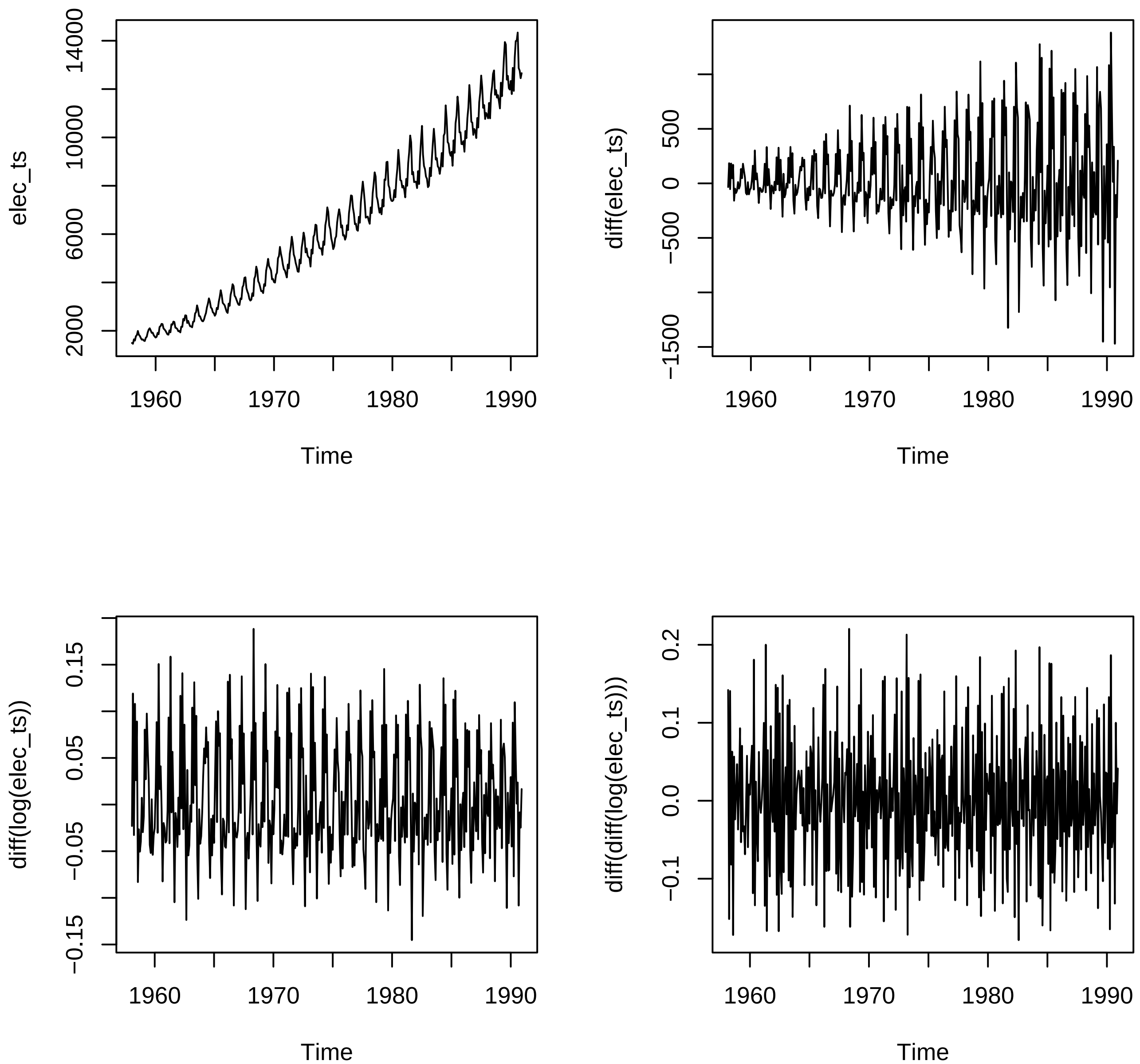
Integrated processes

Integrated model

A series $\{x_t\}$ is ______________________________, $I(d)$, if the d^{th} difference of $\{x_t\}$ is white noise, denoted $\nabla^d x_t = w_t$. It can be shown that $\nabla^d \equiv (1-B)^d$, where B is the backshift operator. Therefore, $\{x_t\}$ is $I(d)$ if

$$(1-B)^d x_t = w_t$$

```
cbe <- read_delim("cbe.dat")
elec <- dplyr::select(cbe, elec)
elec_ts <- ts(elec$elec, start = 1958, freq = 12)
par(mfrow = c(2,2))
plot(elec_ts)
plot(diff(elec_ts))
plot(diff(log(elec_ts)))
plot(diff(diff(log(elec_ts))))
```


elec_ts
2000
6000
10000
14000
1960
1970
1980
1990
Time
diff(elec_ts)
−1500
−500
0
500
1960
1970
1980
1990
Time
diff(log(elec_ts))
−0.15
−0.05
0.05
0.15
1960
1970
1980
1990
Time
diff(diff(log(elec_ts)))
−0.1
0.0
0.1
0.2
1960
1970
1980
1990
Time

ARIMA models

ARIMA models

A time series $\{x_t\}$ follows and ARIMA(p, d, q) process if the d^{th} differences of the series are an ARMA(p, q) process. That is

$$\theta_p(B)(1 - B)^d x_t = \phi_q(B)w_t$$

You will not be expected to manipulate ARIMA models to determine stationarity or invertibility in this class, but it can be done.

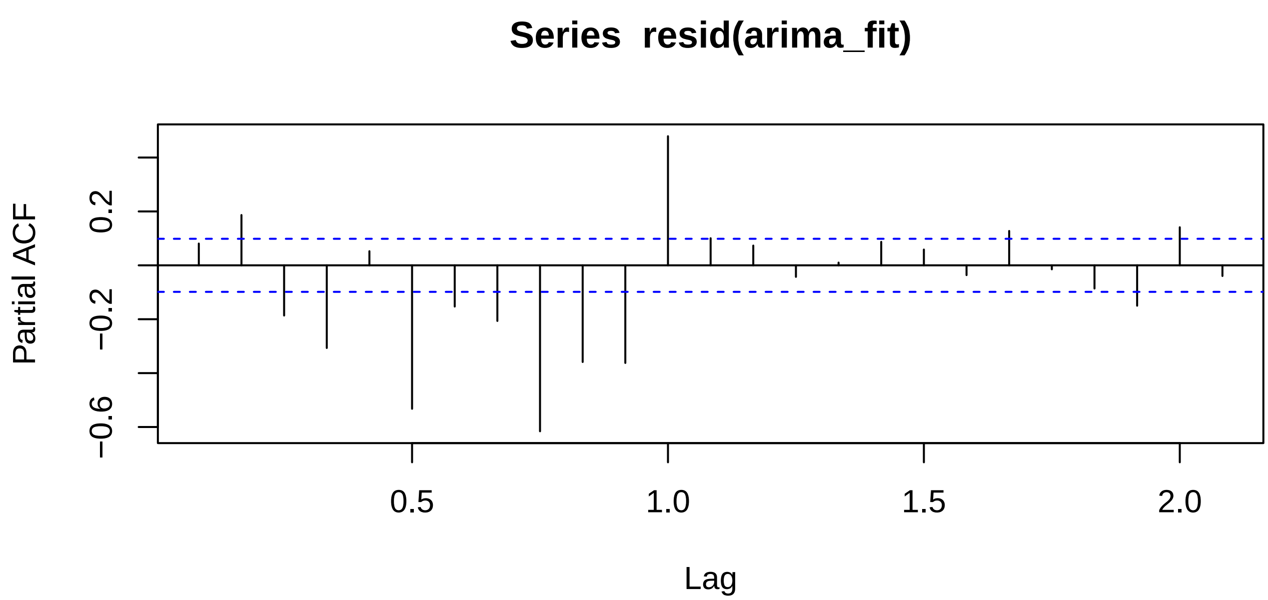
```
(arima_fit <- arima(elec_ts, order = c(1, 1, 1)))
```

```
Call:
arima(x = elec_ts, order = c(1, 1, 1))

Coefficients:
          ar1     ma1
      -0.5577  0.4502
s.e.   0.1177  0.1194

sigma^2 estimated as 183803:  log likelihood = -2954.51,  aic = 5915.02
```

```
pacf(resid(arima_fit)) # not good
```



Seasonal ARIMA models

Seasonal ARIMA models

Finally, we introduce the seasonal ARIMA model, in which we allow we allow each of the autoregressive, integrated, and moving average components to depend on the value from the previous season. The seasonal ARIMA model, denoted ARIMA$(p, d, q)(P, D, Q)_s$, can be written in terms of the backshift operator:

$$\Theta_P(B^s)\theta_p(B)(1 - B^s)^D(1 - B)^d x_t = \Phi_Q(B^s)\phi_q(B)w_t$$

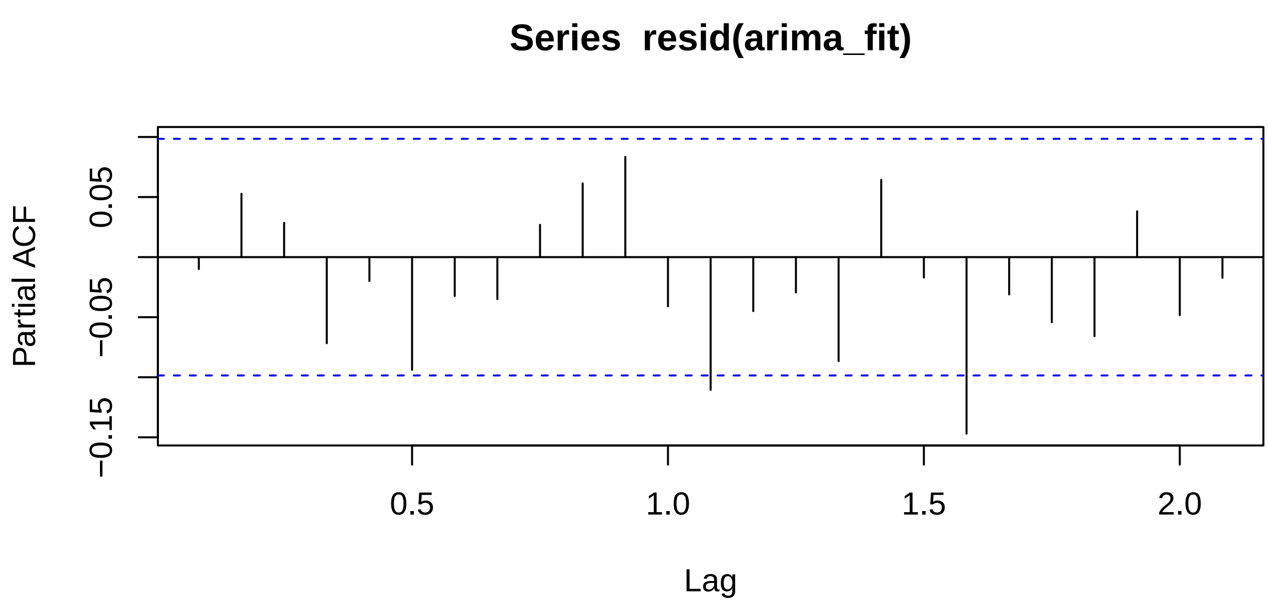
```
(arima_fit <- arima(elec_ts, order = c(1, 1, 1), seasonal = c(1, 1, 1)))
```

```
Call:
arima(x = elec_ts, order = c(1, 1, 1), seasonal = c(1, 1, 1))

Coefficients:
         ar1      ma1     sar1     sma1
      0.0908  -0.7409  -0.0224  -0.5318
s.e.  0.0856   0.0641   0.0813   0.0644

sigma^2 estimated as 22433:  log likelihood = -2464.45,  aic = 4938.89
```

```
pacf(resid(arima_fit)) # better, still not good
```



Fitting ARIMA

The `auto.arima` function in the `forecast` package uses AIC to select the "best" seasonal ARIMA model. You may restrict to a subset of seasonal ARIMA models by setting `max.p, max.d, max.q, max.P, max.D, max.Q` equal to a non-negative integer. Additionally, the `auto.arima` function accepts covariates for regression via the `xreg` argument.

To forecast with an `auto.arima` fit, we use the `forecast` function. See `?forecast.Arima` for more details.

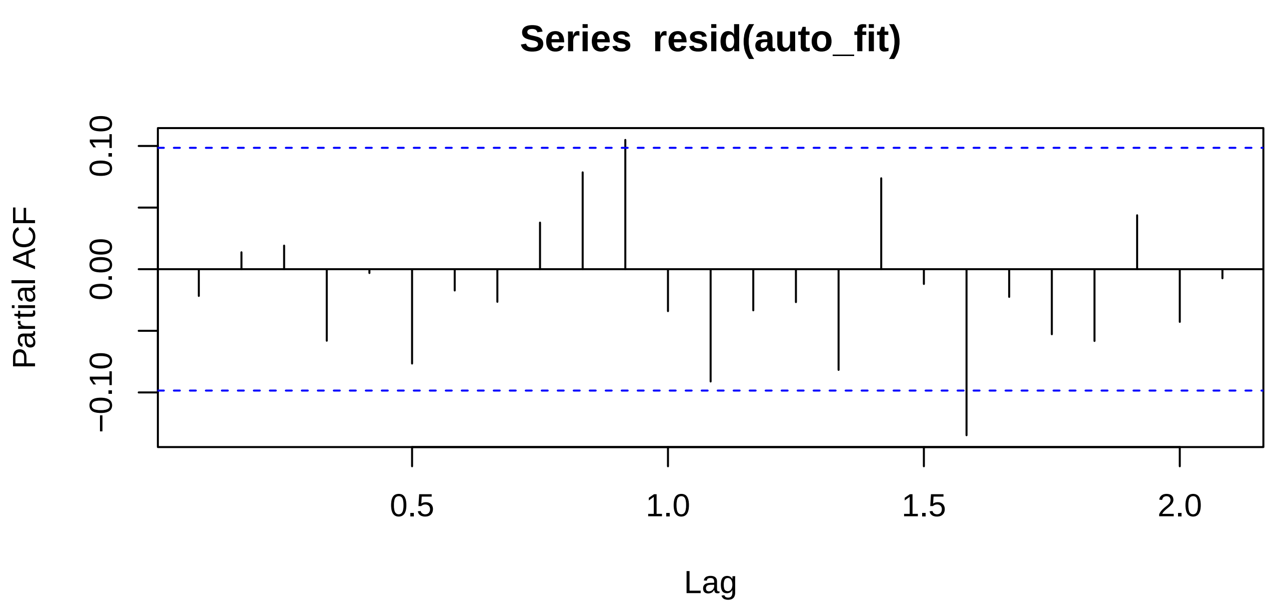
```
(auto_fit <- auto.arima(elec_ts))
```

```
Series: elec_ts
ARIMA(1,1,2)(0,1,1)[12]

Coefficients:
         ar1      ma1     ma2     sma1
      0.8294  -1.4833  0.5125  -0.5368
s.e.  0.1063   0.1301  0.1096   0.0439

sigma^2 = 22390:  log likelihood = -2462.25
AIC=4934.51   AICc=4934.67   BIC=4954.25
```

```
pacf(resid(auto_fit))
```



```
# fitted is defined for auto.arima fits!
fitted(auto_fit) |> head()
```

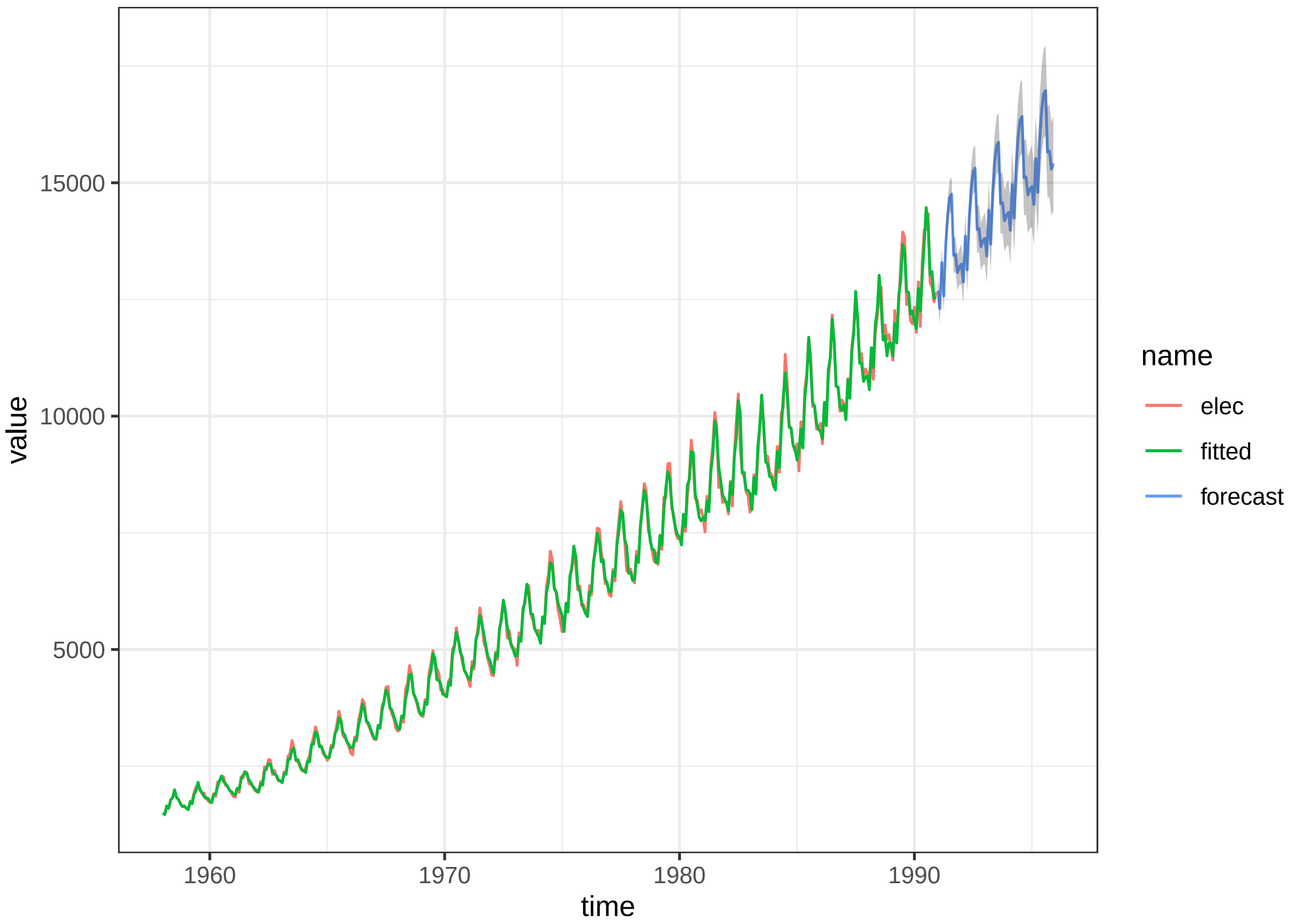
```
          Jan      Feb      Mar      Apr      May      Jun
1958 1496.136 1462.640 1647.608 1594.754 1776.639 1823.657
```

```
# use the forecast function to forecast
auto_forecast <- forecast(auto_fit, h = 5*12, level = 95)

# putting it together
obs_df <- elec %>%
  mutate(time = c(time(elec_ts)), fitted = c(fitted(auto_fit))) %>%
  pivot_longer(-time)

forecast_df <- tibble(
  time = c(time(auto_forecast$mean)),
  value = c(auto_forecast$mean),
  lwr = c(auto_forecast$lower),
  upr = c(auto_forecast$upper)
) %>% mutate(name = "forecast")

ggplot() +
  geom_line(data = obs_df, aes(x = time, y = value, col = name)) +
  geom_line(data = forecast_df, aes(x = time, y = value, col = name)) +
  geom_ribbon(
    data = forecast_df, aes(x = time, ymin = lwr, ymax = upr),
    alpha = .30
  ) +
  theme_bw()
```


15000
10000
5000
value
1960
1970
1980
1990
time
name
elec
fitted
forecast