Name:
Due: 2024/10/21

Exam 1

Be sure to submit **both** the .pdf and .qmd file to Canvas by Friday, October 25th at 11:59 pm.

> ❗ Important
>
> You may use any resource to complete this except **except** for other people (classmates, other faculty members, your parents, etc.) Please do not discuss this exam with anyone but me, and be sure to **cite all references and materials used to answer each question.** Please type your name below to acknowledge that you have not discussed this exam with anyone else.

Your name:

Question 1 [47 pt]

The Utah Frontier Observatory for Research in Geothermal Energy (FORGE) is a project designed to collect and disseminate data that enables research in the development of geothermal energy. In February of 2021, the UTAH-FORGE drilled a 9000 foot hole, labeled well 56-32, to enable future monitoring of seismic activity. The `utah_forge_56_32_2021_02_08_03_04.csv` file contains a subset of the measurements obtained from the drilling process at 10 second intervals between 3 and 4 hours into the drilling procedure. The entire dataset is publicly available on data.gov and a description of the drilling site is provided on the Utah FORGE website. For this analysis, we are interested in the total volume of mud lost to drilling at each time point, labeled `Total Mud Volume` in the data set.

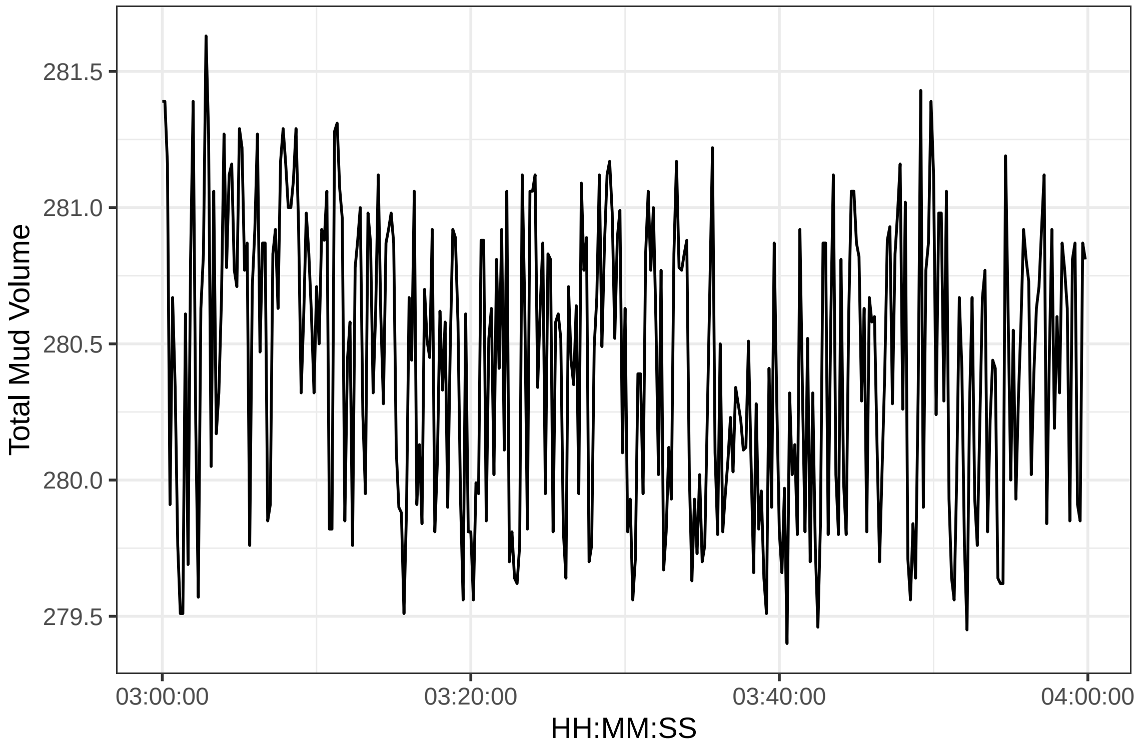
Drilling site, courtesy of Utah FORGE

1. [1 pt] Read in the data.

```r
utah <- readr::read_csv("utah_forge_56_32_2021_02_08_03_04.csv")
```

2. [3 pt] Plot the time series and describe the time plot in terms of the trend and seasonality.

```r
utah %>%
  ggplot() +
  geom_line(aes(x = `HH:MM:SS`, y = `Total Mud Volume`)) +
  theme_bw()
```



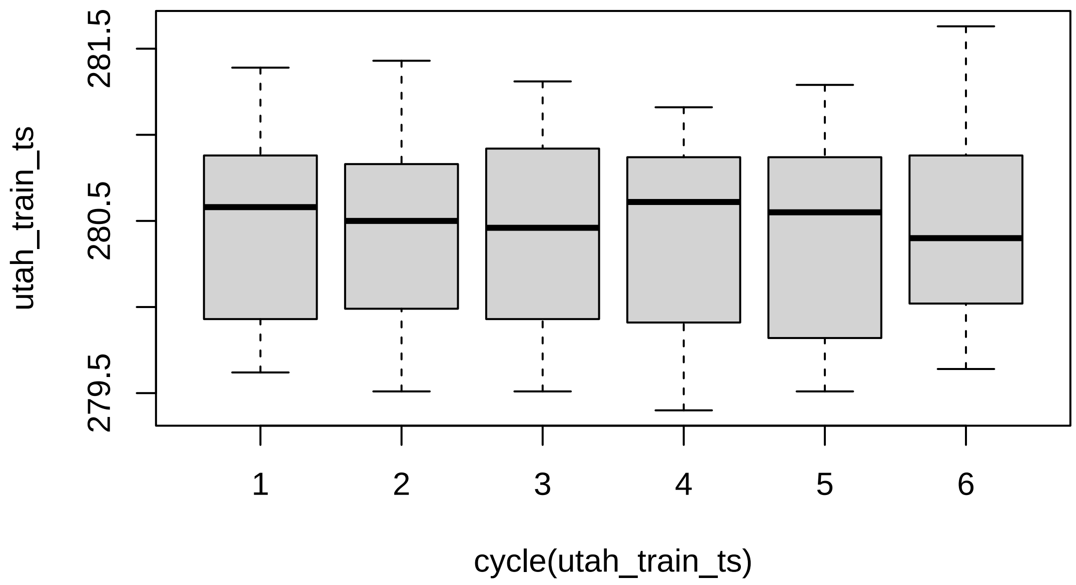
3. [1 pt] Hypothesize a reasonable seasonal frequency for this data set and justify your choice.

4. [2 pt] Create two time series objects: 1) describing the total mud volume until (but not including) 03:50:00 (called `utah_train_ts`) and 2) describing the total mud volume from 03:50:00 onward (called `utah_test_ts`).

```
# filter first
utah_train <- utah %>%
  filter(`HH:MM:SS` < hms("03:50:00"))
utah_test <- anti_join(utah, utah_train)

# ts objects
utah_train_ts <- ts(
  utah_train$`Total Mud Volume`,
  start = c(1, 1),
  freq = 6
)
utah_test_ts <- ts(
  utah_test$`Total Mud Volume`,
  start = c(51, 1),
  freq = 6
)
```

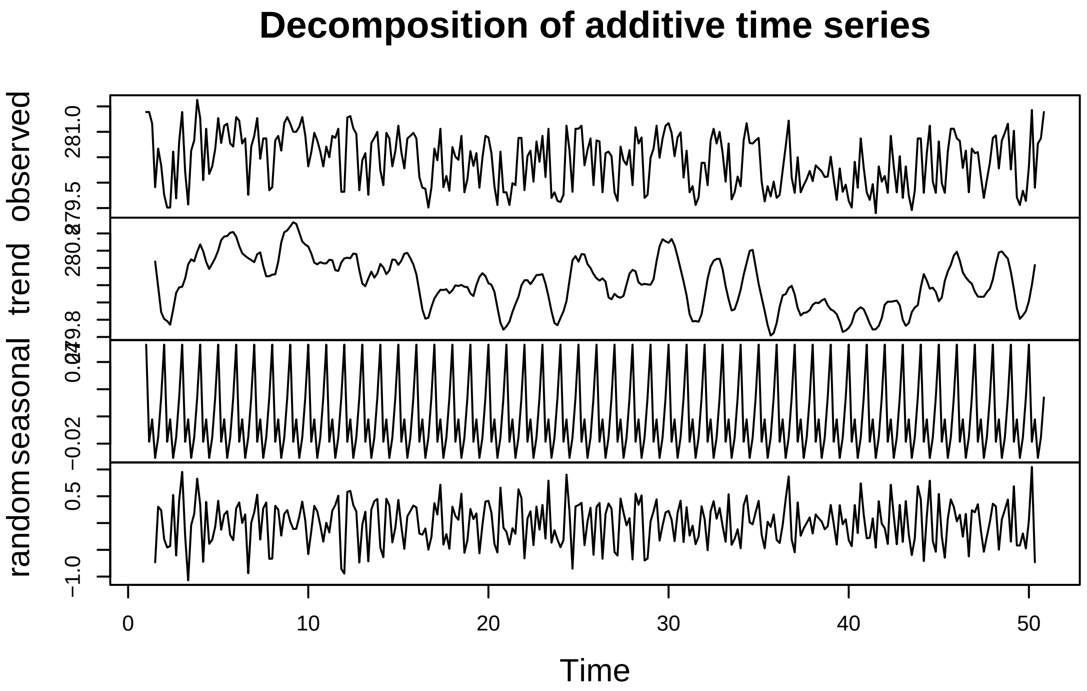
5. [1 pt] Create boxplots of the total volume of mud by each element of the seasonal cycle for the `utah_train_ts` time series.

```
boxplot(utah_train_ts ~ cycle(utah_train_ts))
```



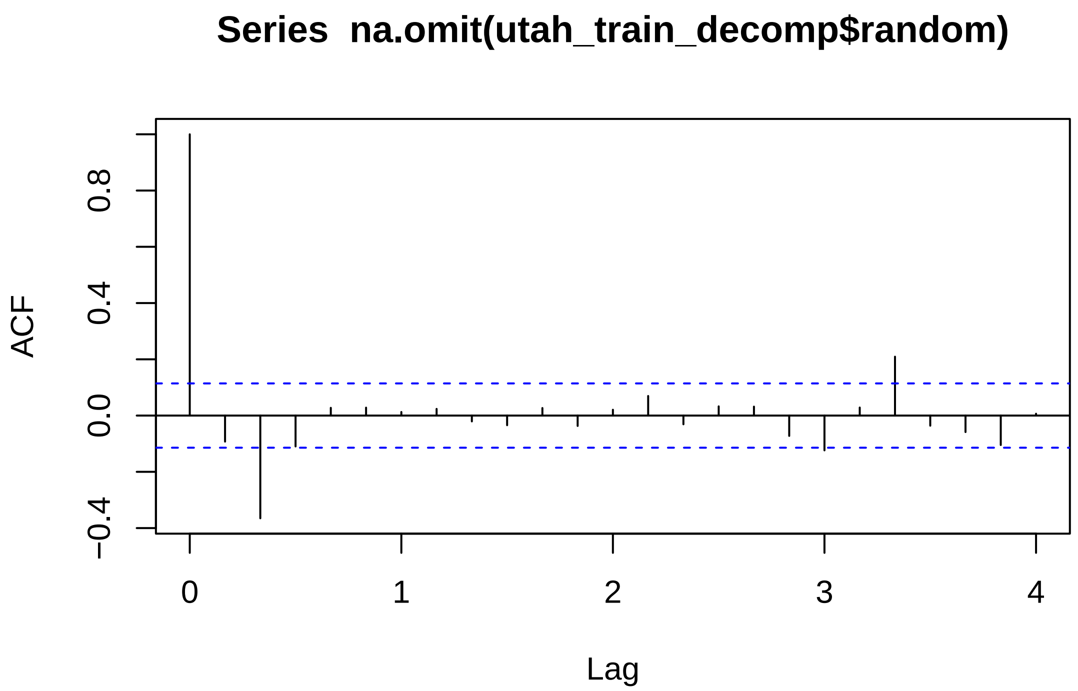
6. [2 pt] Plot a decomposition of the `utah_train_ts` time series and component on what you see in terms of trend and seasonality. Does the decomposed seasonal trend represent meaningful seasonal variation? Explain your answer.

```
utah_train_decomp <- decompose(utah_train_ts)
plot(utah_train_decomp)
```



7. [2 pt] Create an ACF plot of the residual error series from the decomposition and comment on what the plot suggests about serial correlation in the residual error series.

```
acf(na.omit(utah_train_decomp$random))
```

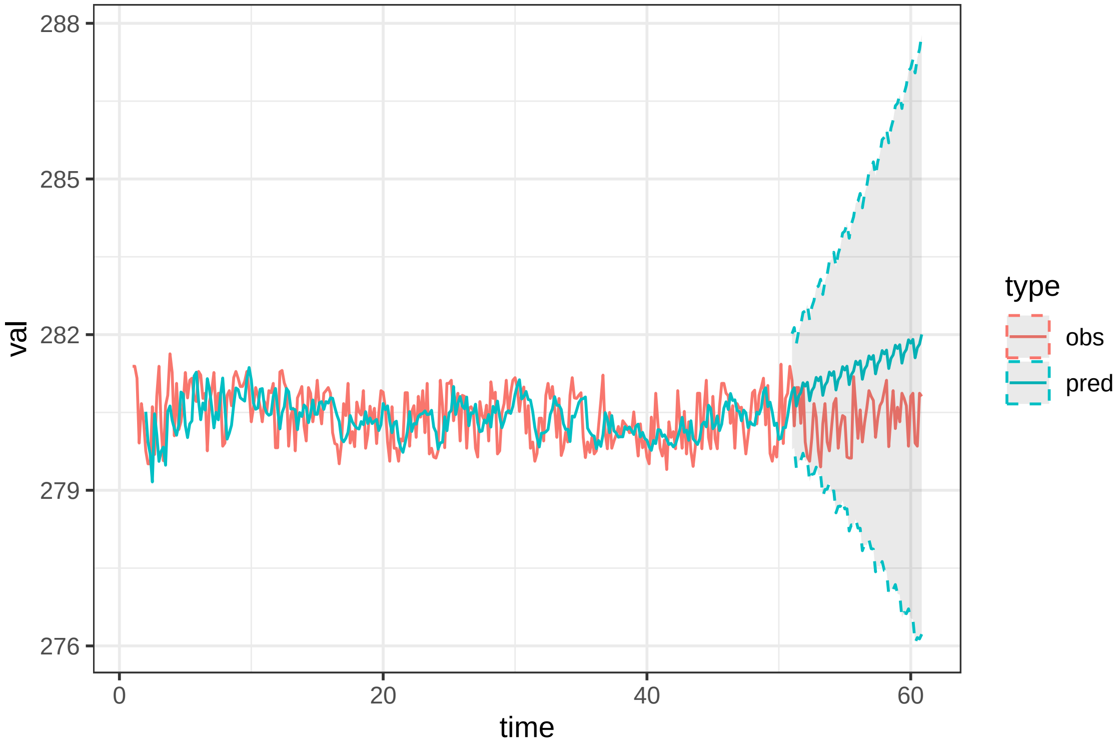


8. [2 pt] Could you use the decomposition model to forecast the total mud volume values in the `utah_test_ts` series? Why or why not?

> No.

9. [5 pt] Fit an additive Holt-Winters model to the `utah_train_ts` series, allowing R to estimate the model parameters. Use that model to forecast the total volume of mud in the `utah_test_ts` series, and plot the observed, estimated, and forecasted series on a single plot, including 95% prediction intervals for the forecast.

```r
# hw model
hw <- HoltWinters(utah_train_ts)

# forecast
hw_pred <- predict(hw, n.ahead = 60, prediction.interval = TRUE)

# plot
bind_rows(
  tibble(
    time = c(time(utah_train_ts), time(utah_test_ts)),
    val = c(utah_train_ts, utah_test_ts)
  ) %>% mutate(type = "obs"),
  bind_rows(
    tibble(
      time = time(hw$fitted[,1]),
      val = c(hw$fitted[,1])
    ),
    tibble(
      time = time(hw_pred),
      val = c(hw_pred[,1]),
      lwr = c(hw_pred[,3]),
      upr = c(hw_pred[,2])
    )
  )  %>% mutate(type = "pred")
) %>%
  ggplot(aes(x = time, y = val, col = type)) +
  geom_line() +
  geom_ribbon(
    aes(ymin = lwr, ymax = upr),
    linetype = 2,
    alpha = 0.1
  ) +
  theme_bw()
```



10. [2 pt] Comment on the quality of the forecast, discussing both the forecasted total mud volume, and the uncertainty in that forecast.

11. [2 pt] Propose a way to assess the quality of the forecast that incorporates **both** the forecasted volume (the point estimate) **and** the uncertainty in that forecasted volume (the interval estimate). **Note:** it is okay if your loss function results in two numbers.

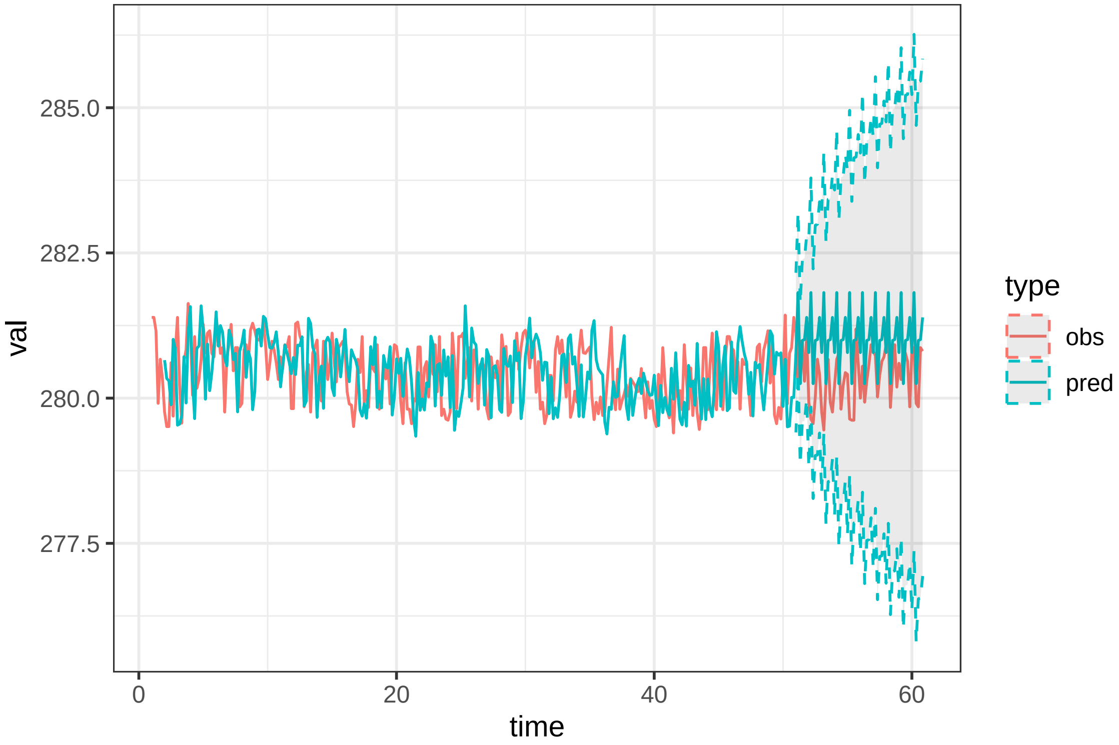
12. [1 pt] Implement your proposed loss function on the Holt-Winters model estimated by R and print out the value of that loss function.

```
# one option is to get SSE and sum of interval widths
SSE <- sum((utah_test_ts - hw_pred[,1])^2)
widths <- sum(hw_pred[,2] - hw_pred[,3])
c(SSE, widths)
```

```
[1]  70.5692 385.7780
```

13. [5 pt] Find values of the Holt-Winters model parameters that result in a better forecast, according to the criterion you proposed. Plot the observed, estimated, and forecasted volumes from this model as before, and comment on why the new fit is better.

```
# hw model
hw <- HoltWinters(utah_train_ts, alpha = .1, beta = F, gamma = T)

# forecast
hw_pred <- predict(hw, n.ahead = 60, prediction.interval = TRUE)

# plot
bind_rows(
  tibble(
    time = c(time(utah_train_ts), time(utah_test_ts)),
    val = c(utah_train_ts, utah_test_ts)
  ) %>% mutate(type = "obs"),
  bind_rows(
    tibble(
      time = time(hw$fitted[,1]),
      val = c(hw$fitted[,1])
    ),
    tibble(
      time = time(hw_pred),
      val = c(hw_pred[,1]),
      lwr = c(hw_pred[,3]),
      upr = c(hw_pred[,2])
    )
  )  %>% mutate(type = "pred")
) %>%
  ggplot(aes(x = time, y = val, col = type)) +
  geom_line() +
  geom_ribbon(
    aes(ymin = lwr, ymax = upr),
    linetype = 2,
    alpha = 0.1
  ) +
  theme_bw()
```



```
SSE <- sum((utah_test_ts - hw_pred[,1])^2)
widths <- sum(hw_pred[,2] - hw_pred[,3])
c(SSE, widths)
```

```
[1]  50.71786 378.60405
```

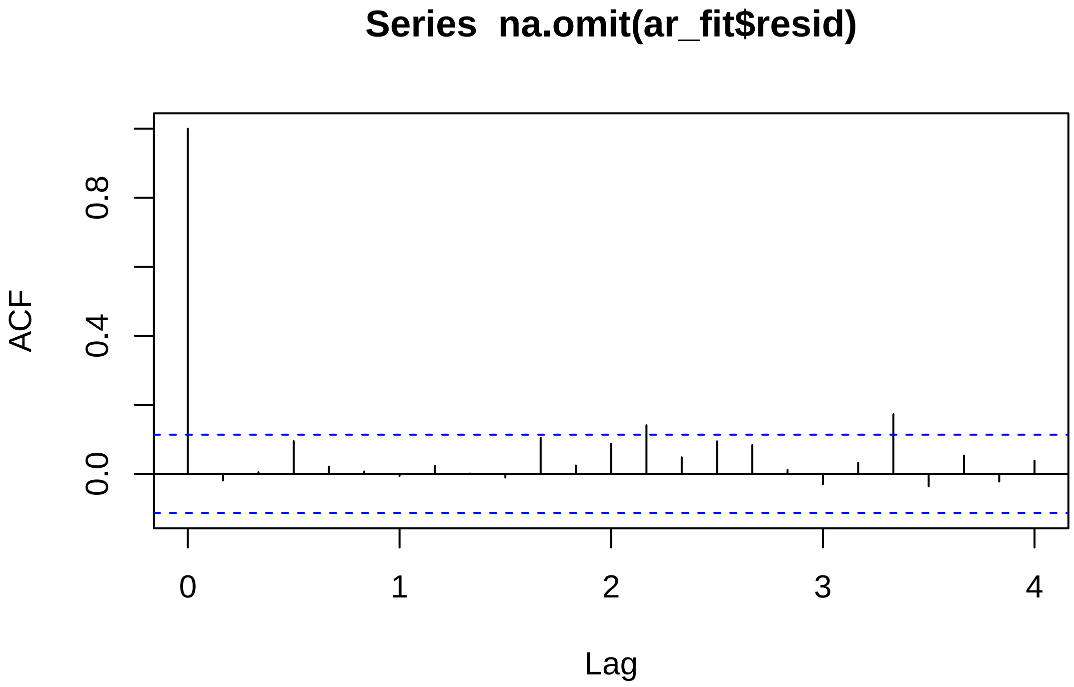
14. [2 pt] Fit an AR(1) model to the `utah_train_ts` series and allow R to estimate the smoothing parameter. Write out the estimated AR model.

```
# ar model
ar_fit <- ar(utah_train_ts)
```

15. [2 pt] Does the estimated AR model represent a stationary process? Why or why not?

16. [2 pt] Create an acf plot of the residuals from the AR model and comment on what this plot means in terms of residual autocorrelation.

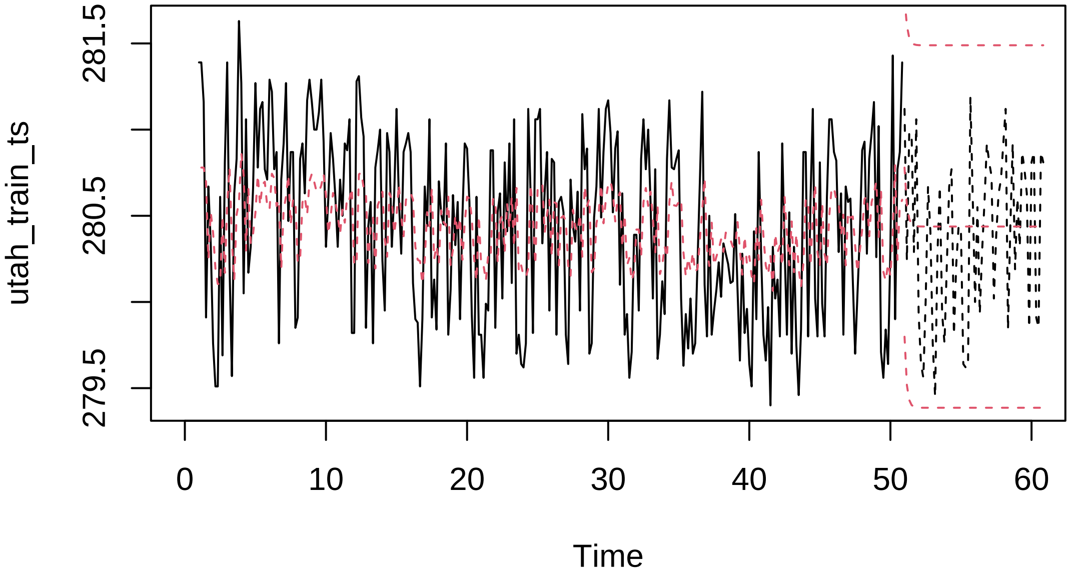
```
# ar model
acf(na.omit(ar_fit$resid))
```



17. [5 pt] Use the AR model to forecast the `utah_test_ts` series and plot the observed, estimated, and forecasted series on a single plot. Comment on the quality of the forecast.

```
# obtain predictions 2:N for AR(1) model
fitted <- ar_fit$x.mean + ar_fit$ar * (utah_train_ts[1:(length(utah_train_ts)-1)] - a

# what about forecasts?
```

```
pred <- predict(ar_fit, n.ahead = 60)
plot(utah_train_ts, xlim = c(0, 60))
lines(utah_test_ts, lty = 2)
lines(fitted ~ time(utah_train_ts)[c(-1)], lty = 2, col = 2)
lines(pred$pred, lty = 2, col = 2)
lines(pred$pred - 2*pred$se, lty = 2, col = 2)
lines(pred$pred + 2*pred$se, lty = 2, col = 2)
```



18. [2 pt] You should have come to the conclusion that neither the AR(1) model nor the Holt-Winters model provided a very good forecast for these data. Comment on why that might be the case, referencing both the structure of the observed series, and the length of the period over which we are attempting to forecast.

Question 2 [12 pt]

For each of the following AR models, write the model in terms of the backshift operator and determine whether the model is stationary.

1. [3 pt] $x_t = .25x_{t-1} + w_t$

```r
polyroot(c(1, -.25))
```

```
[1] 4+0i
```

2. [3 pt] $x_t = -1.1x_{t-1} + w_t$

```r
polyroot(c(1, 1.1))
```

```
[1] -0.9090909+0i
```

3. [3 pt] $x_t = -1.1x_{t-1} + .25x_{t-2} + w_t$

```r
polyroot(c(1, 1.1, -.25))
```

```
[1] -0.7732137+0i  5.1732137-0i
```

4. [3 pt] $x_t = 1.2x_{t-1} - .3x_{t-2} + w_t$

```r
polyroot(c(1, -1.2, .3))
```

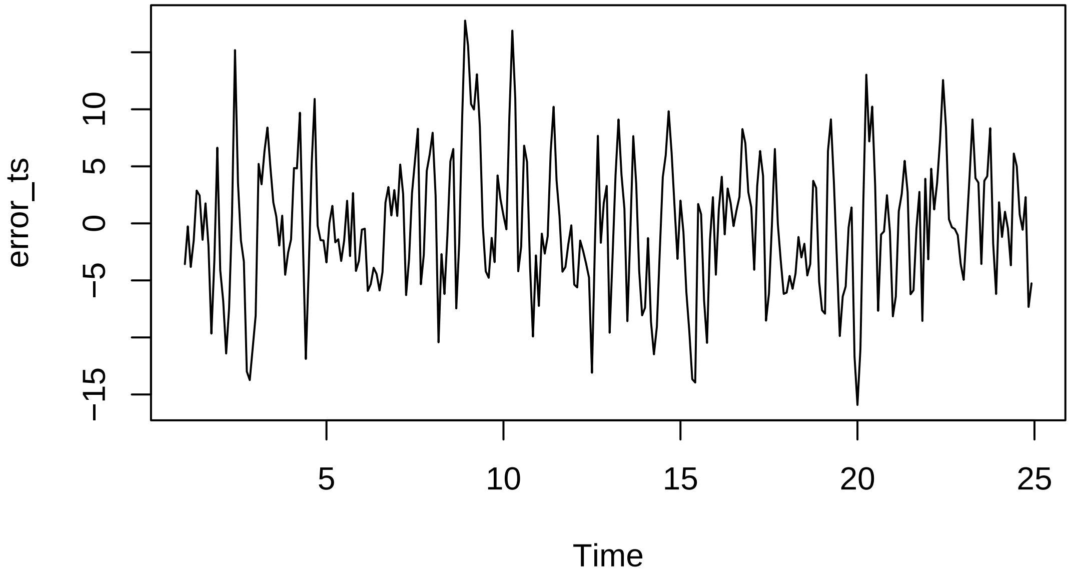
```
[1] 1.183503+0i 2.816497-0i
```

Question 3

We discussed in class how useful simulation can be for understanding time series concepts. In this question, we will use simulation to better understand how to construct time series using AR processes.

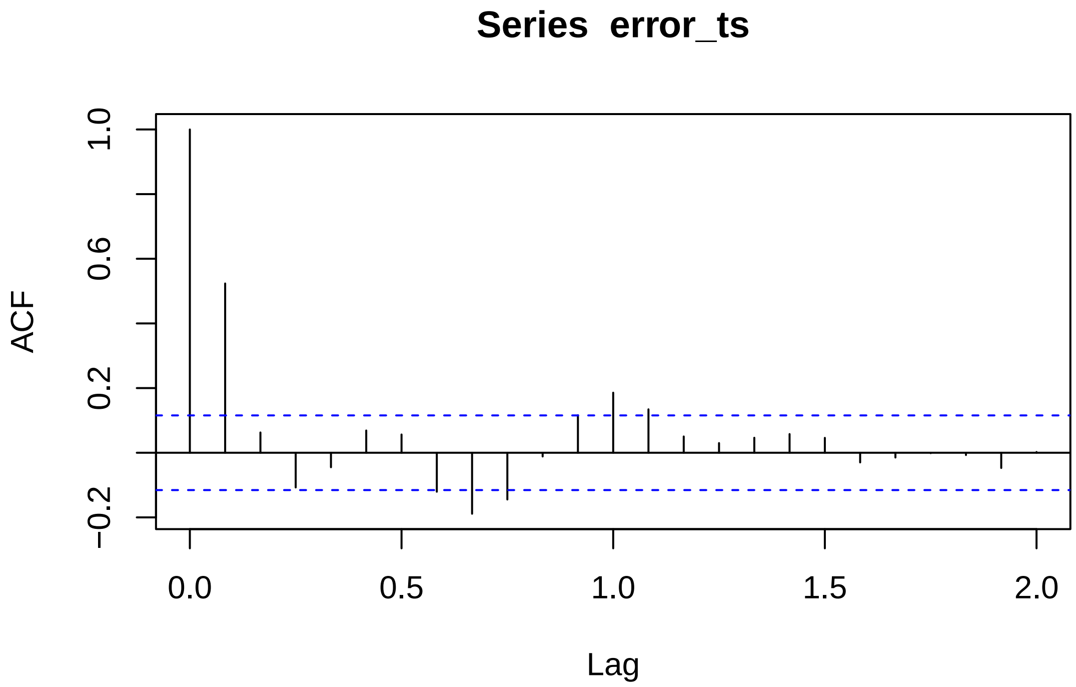
1. [3 pt] Use simulation to generate 24 years of monthly data from an AR(2) process with $\alpha_1 = .7$, $\alpha_2 = -.3$, and $\sigma = 5$. Create a time series object called `error_ts` and plot that series.

```
# simulation
x <- w <- rnorm(24*12, 0, 5)
for(t in 3:(24*12)) x[t] <- .7*x[t-1] - .3*x[t-2] + w[t]

# createt ts
error_ts <- ts(
  x,
  start = c(1,1),
  freq = 12
)

plot(error_ts)
```



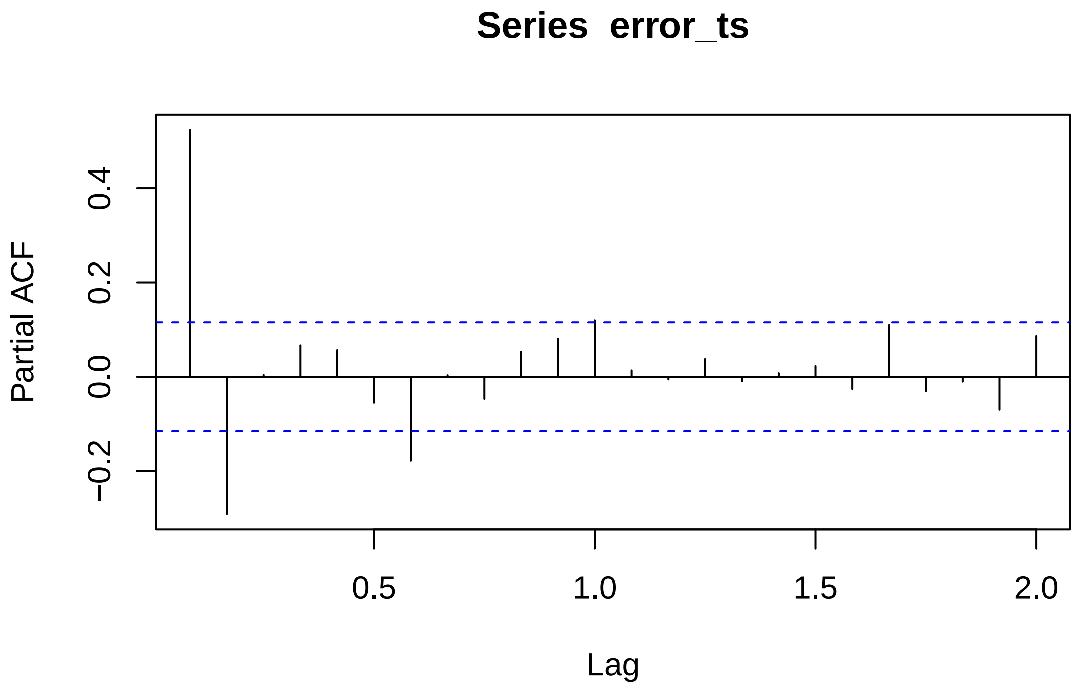
2. [2 pt] Create an autocorrelation function plot of `error_ts` and comment on what the plot suggests about the residual correlation in the series.

```
# simulation
acf(error_ts)
```



3. [2 pt] Create a partial autocorrelation function plot of `error_ts` and comment on what the plot suggests about the residual correlation in the series.

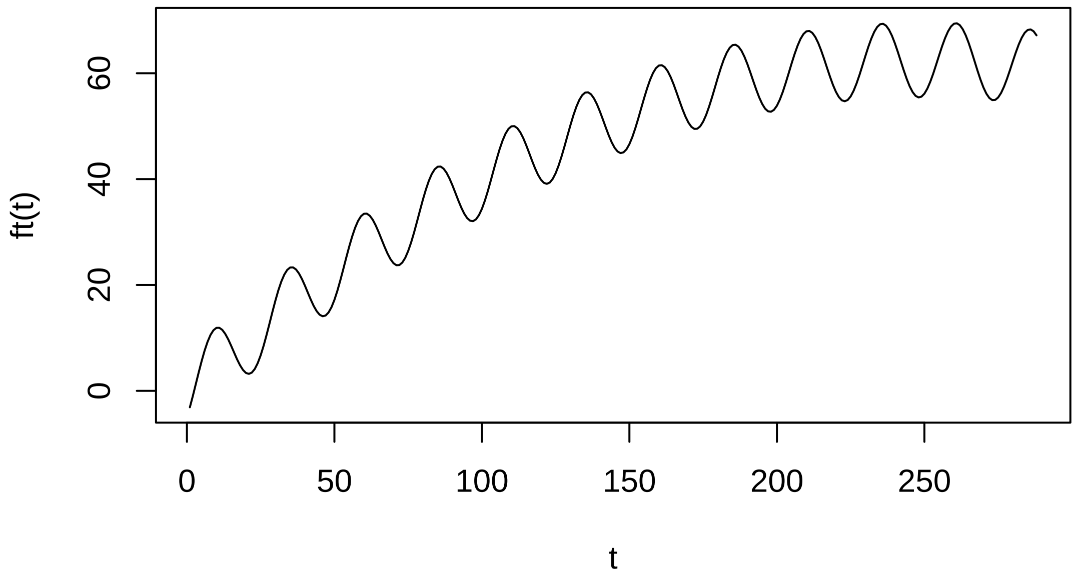
```
# simulation
pacf(error_ts)
```



4. [1 pt] Is the series constructed in question 1 capable of modeling time series with trends or seasonal components? Why or why not?

5. [2 pt] A time series may be constructed by adding a serially correlated error series (such as `error_ts`) to a model that describes the mean function over time. Regression is often used to model the mean function in times. For each time point in the 24 years simulated in the `error_ts` series ($t = 1, 2, \dots, 288$), compute $.5t - .001t^2 + 5\sin\left(\frac{t}{12}\right) - 5\cos\left(\frac{t}{12}\right)$ and plot the resulting function with respect to time.

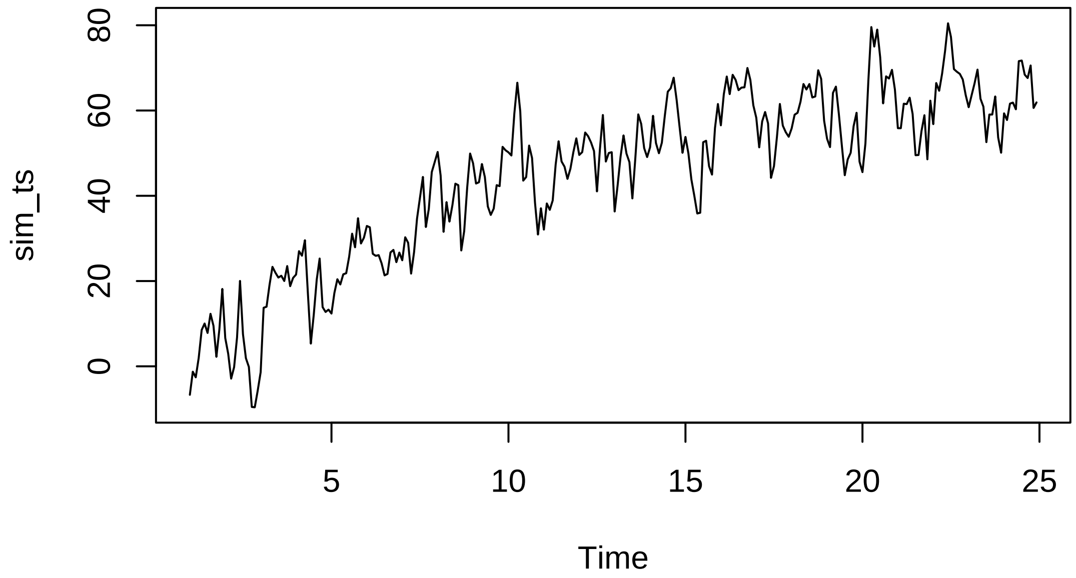
```
# simulation
t <- 1:288
ft <- function(t) .5*t - .001*t^2 + 5*sin(t/4) - 5*cos(t/4)
plot(ft(t) ~ t, type = "l")
```



6. [1 pt] The function plotted in the previous question is a smooth curve with no noise, rather than the noisy series we see with observed data. Why is that?

7. [2 pt] Add the function created in question 5 to `error_ts`. **Note:** You will likely have to convert `error_ts` to a vector using `c()` before adding the two series, then recreate a time series object after summing. Call this new series `sim_ts`. Plot the resulting series.

```
# simulation
series <- ft(t) + c(error_ts)
sim_ts <- ts(
  series,
  start  = c(1,1),
  freq = 12
)

plot(sim_ts)
```



8. [1 pt] Comment on the plot in the previous question. How does it compare to some of the series we have seen in this class so far?

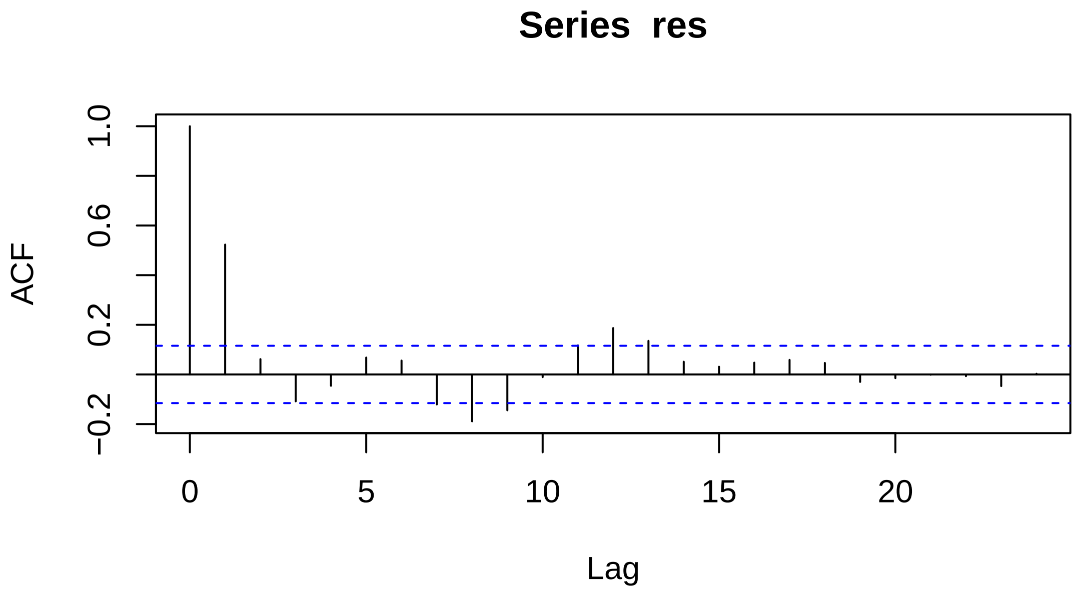
9. [1 pt] The code below fits a regression model with t, t^2, $\sin(t/4)$, and $\cos(t/4)$ as explanatory variables for the `sim_ts` time series and prints off the coefficient estimates. How do these values compare to the values used to generate the time series?

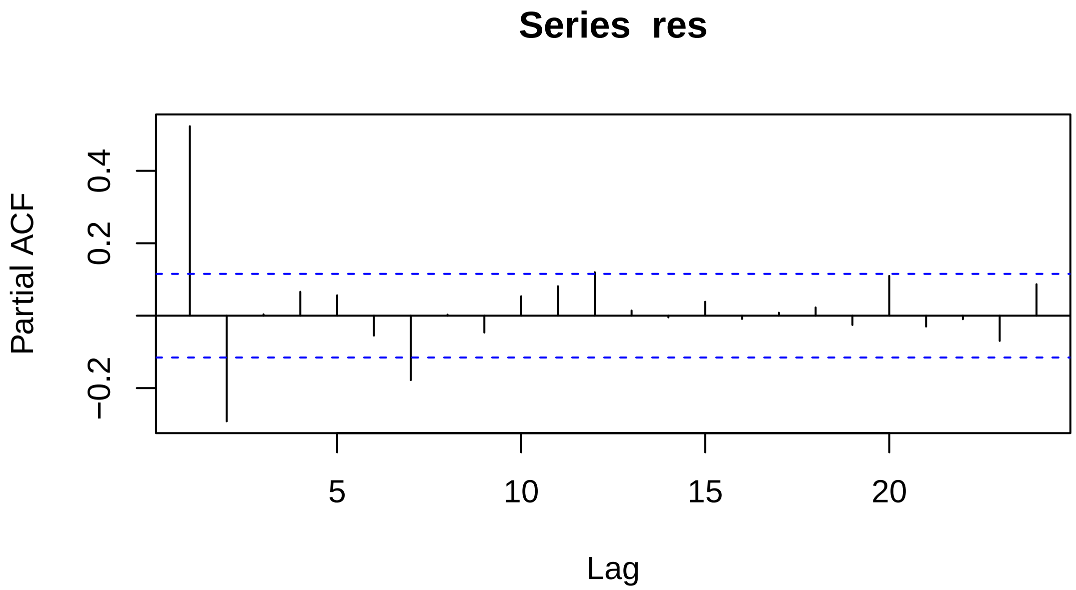
```
reg_tbl <- tibble(
  y = c(sim_ts)
) %>%
  mutate(
    t = 1:n(),
    sint = sin(t/4),
    cost = cos(t/4)
  )

sim_reg <- lm(
  y ~ t + I(t^2) + sint + cost,
  data = reg_tbl
)
sim_reg$coefficients[-1]
```

```
          t       I(t^2)         sint         cost 
0.501460485 -0.001002366  4.780028875 -5.083555605 
```

10. [2 pt] The code below calculates the residuals from the fitted regression model, which represents our residual error series. Create acf and pacf plots of the residual error series, and compare these plots to the acf and pacf plots created for the `error_ts` series.

```r
res <- resid(sim_reg)
par(mfrow = c(2, 1))
acf(res)
pacf(res)
```





```r
par(mfrow = c(1,1))
```

11. [2 pt] Fit an AR(2) model to the residual error series, called `ar_fit`. Print the estimated of α_1 and α_2 and comment on how these estimates relate to the values used to generate the error series in question 1.

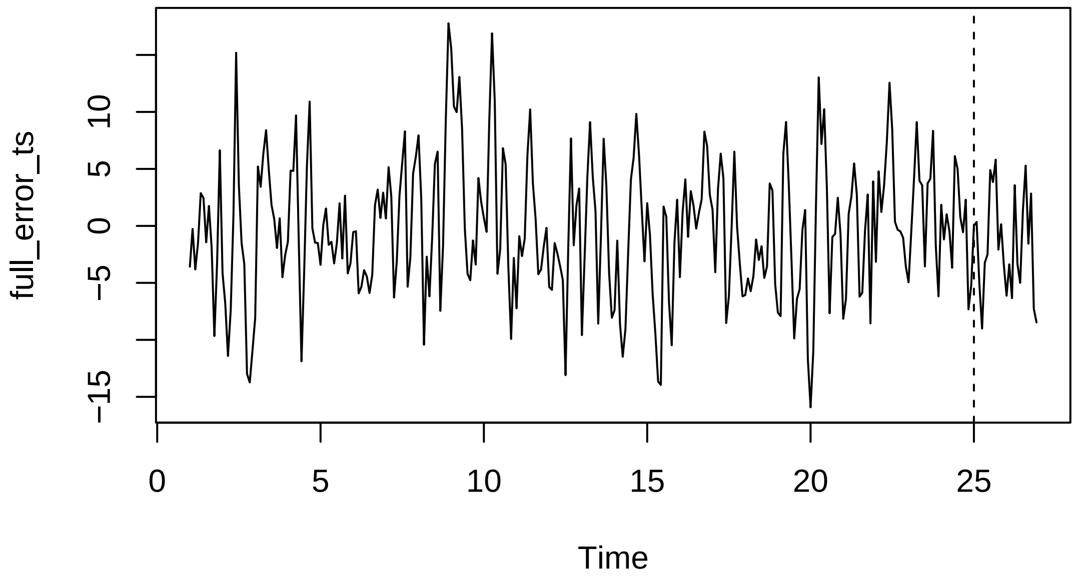
```r
ar_fit <- ar(res, order.max = 2)
ar_fit
```

```
Call:
ar(x = res, order.max = 2)

Coefficients:
      1        2  
 0.6758  -0.2918  

Order selected 2  sigma^2 estimated as  23.58
```

For the remainder of this question, assume we are trying to forecast two more years of the simulated series. To do so, we must first generate two more years of the series, corresponding to $t = 289, \dots, 312$.

12. [2 pt] The code below generates two more years of the `error_ts` series and creates a new series, `full_error_ts`, that includes all 26 years of data. Describe what the first four lines of code are doing and why they are necessary to maintain the AR(2) process.

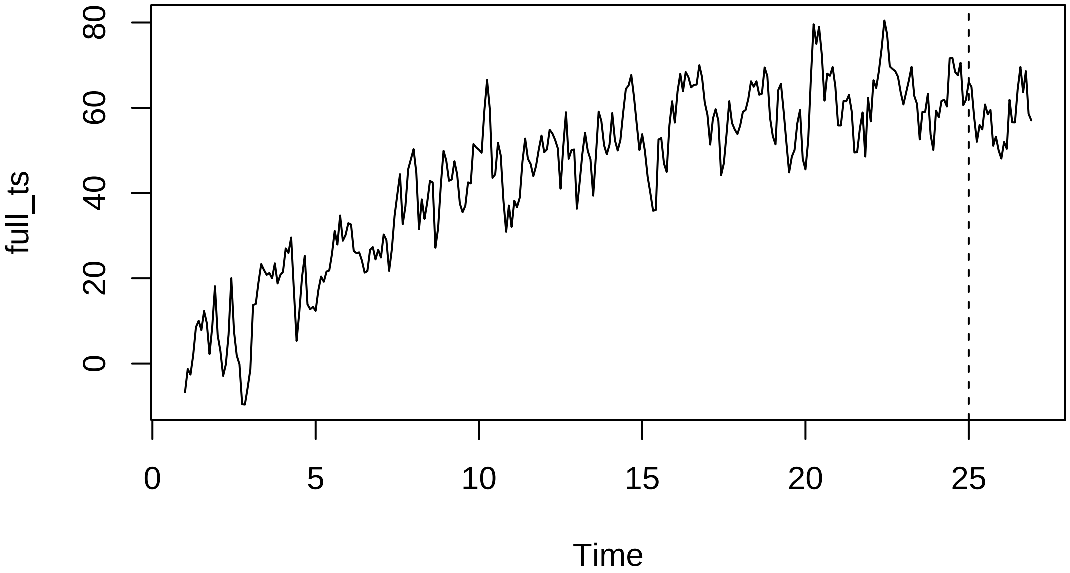
```r
# simulate two more years
x <- w <- rnorm(24, 0, 5)
x[1] <- .7*error_ts[278] - .3*error_ts[277] + rnorm(1, 0, 5)
x[2] <- .7*x[1] - .3*error_ts[278] + rnorm(1, 0, 5)
for(t in 3:24) x[t] <- .7*x[t-1] - .3*x[t-2] + w[t]

# combine series and plot
full_error_ts <- ts(
  c(error_ts, x),
  start = c(1, 1),
  freq = 12
)
plot(full_error_ts)
abline(v = 25, lty = 2)
```



13. [2 pt] Generate two more years ($t = 289, 290, \ldots, 312$) from the mean function defined in question 5. Create a new series, `full_ts`, that combines the mean function from $t = 1$ to $t = 312$ with the `full_error_ts` series, and plot that series.

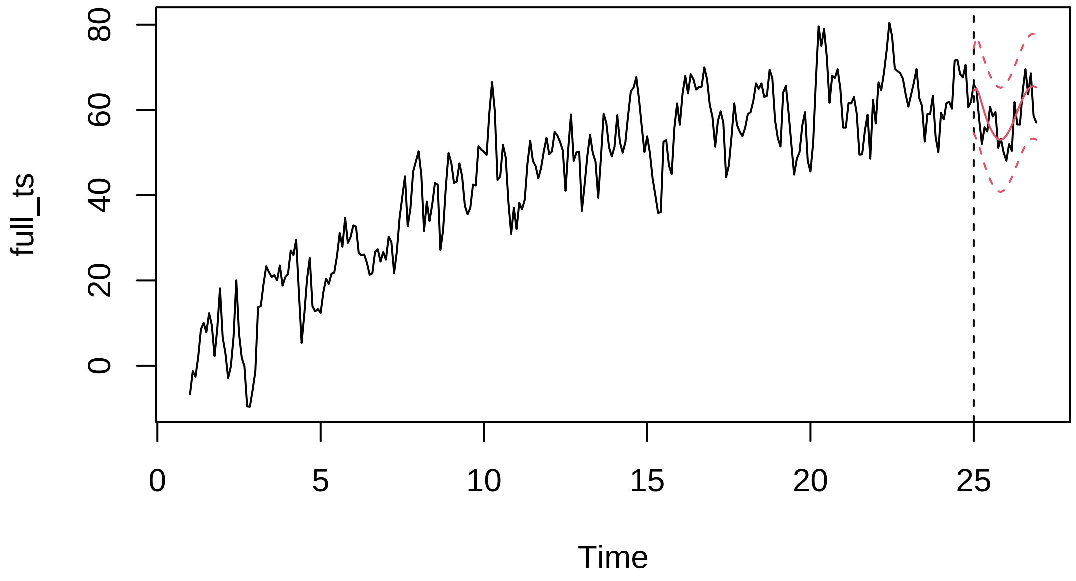
    Note that `full_ts` represents the time series between $t = 1, \ldots, 312$, but we are supposing that we only observed between $t = 1, \ldots, 288$. The purpose of this part of the question is to generate two more years so that we have something to forecast.

```
full_ts <- ts(
  c(full_error_ts) + ft(1:312),
  start = c(1,1),
  freq = 12
)
plot(full_ts)
abline(v = 25, lty = 2)
```



14. [2 pt] The code below obtains predictions for $t = 289, \ldots, 312$ from the fitted regression model and the AR process fit to the residual error series, and combines them to forecast the series for $t = 289, \ldots, 312$. Comment on the quality of the forecast.

```
# obtain forecast of mean function from the regression model
pred_tbl <- tibble(
  t = 289:312
) %>%
  mutate(sint = sin(t/4), cost = cos(t/4))
pred <- predict(sim_reg, newdata = pred_tbl, se.fit = T)

# predict from ar_fit
pred_error <- predict(ar_fit, n.ahead = 24, prediction.interval = T)

# combine them
forecast_series <- ts(
   c(pred$fit + pred_error$pred),
   start = c(25, 1),
   freq = 12
)
forecast_series_lwr <- ts(
  c(forecast_series) - 2*sqrt(pred$se.fit^2 + c(pred_error$se^2)),
   start = c(25, 1),
    freq = 12
)
forecast_series_upr <- ts(
  c(forecast_series) + 2*sqrt(pred$se.fit^2 + c(pred_error$se^2)),
   start = c(25, 1),
   freq = 12
)


# plot
plot(full_ts)
abline(v = 25, lty = 2)
lines(forecast_series, lty = 1, col = 2)
lines(forecast_series_lwr, lty = 2, col = 2)
lines(forecast_series_upr, lty = 2, col = 2)
```


full_ts
Time