Analyzing the Average Monthly Temperatures of Mississippi

Connor Robison

University of Southern Mississippi

Dr. Schroeder

MAT 482: Data Analysis

May 3, 2023

Abstract

Whether people realize it or not, the temperature outside affects their daily life. If this statement were not true, news companies would not hire meteorologists to inform the community of the weather every morning and evening. One growing concern in the media is climate change and increasing global temperatures. If average temperatures are on the rise, how might this affect the average temperature of Mississippi? Will the upward trend be detectable? Is it significant? And how accurately can a rudimentary model approximate future average temperatures in MS? Both nonlinear and linear regression on Mississippi average monthly temperatures beginning in 1950 substantiate the claim that average temperatures in Mississippi are rising. The nonlinear model can predict more moderate average temperatures well but will not always capture high and low average temperatures. Producing an AR or ARIMA process from the residuals of the nonlinear model compared to the input data and utilizing one of these processes in conjunction with the nonlinear model can improve predictions for future average temperatures, especially the high and low average temperatures. The overparameterized AR model performs marginally better than the ARIMA model, but the latter achieves nearly the same amount of accuracy, within 0.0012 percent, while using fewer parameters.

Analyzing the Average Monthly Temperatures of Mississippi

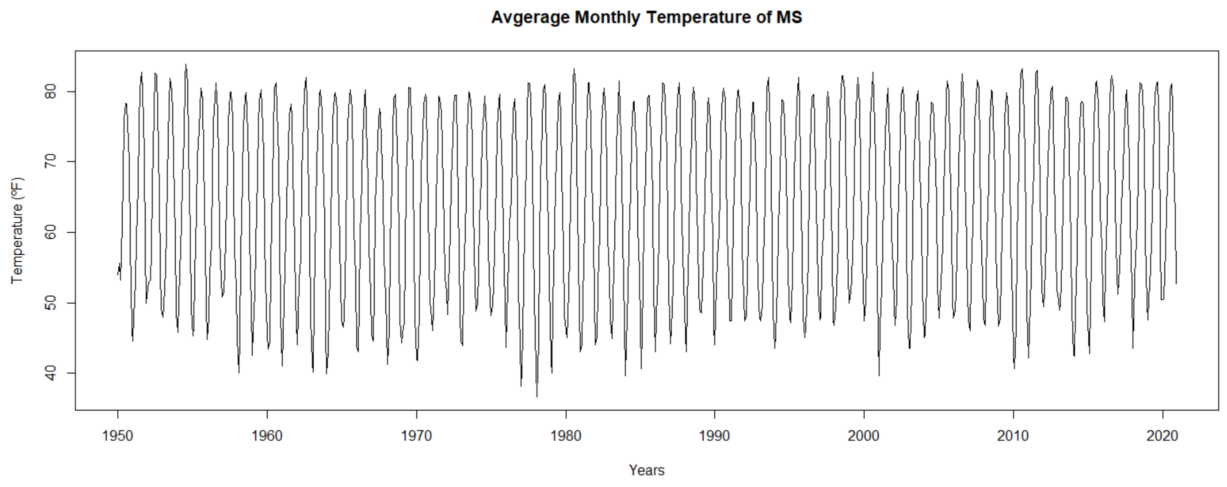
The data utilized for this analysis initially recorded the average monthly temperatures for all fifty states in the United States (Wong, 2022). From the set of over 41,000 entries, temperatures for Mississippi were filtered out and used for further scrutiny. Additionally, about a year's worth of data was removed from the end. The author will use these values for validation of the models used in this investigation. The subsequent data set spans about seventy years, from January 1950 to December 2020. Time is the independent variable, measured in years, and temperature is the dependent variable, measured in degrees Fahrenheit. There are multiple reasons why knowing the temperature outside is meaningful to the public. For example, people leaving their homes decide what to wear based on the temperature outside. A person planning a trip must know the temperature forecast to make appropriate accommodations during their outing. Farmers care about the temperature outside because the health of their crops is contingent on the climate. Weather models created by climate scientists would be incomplete without understanding the effects of temperature. Temperature matters to people. If this statement were false, news companies would not employ meteorologists, and materials such as almanacs would be obsolete.

What Does the Data Look Like?

The following plot (Figure 1) is a time series generated from the average monthly temperatures of Mississippi from January 1950 to December 2020. The first detail the author noticed about the figure was its seasonality. A crest and a trough are present every year on the plot. These peaks and valleys correspond to summer and winter in Mississippi, respectively. So, if, for example, a person wanted to represent this plot using a sine wave, the function would have a period of one year. The following detail the author noticed was that the graph exhibits an upward trend. This detail is not as apparent as the plot's seasonality and may be difficult for the observer to spot initially, but this trend will become more evident after regression analysis.

Figure 1:

Average Monthly Temperatures for Mississippi

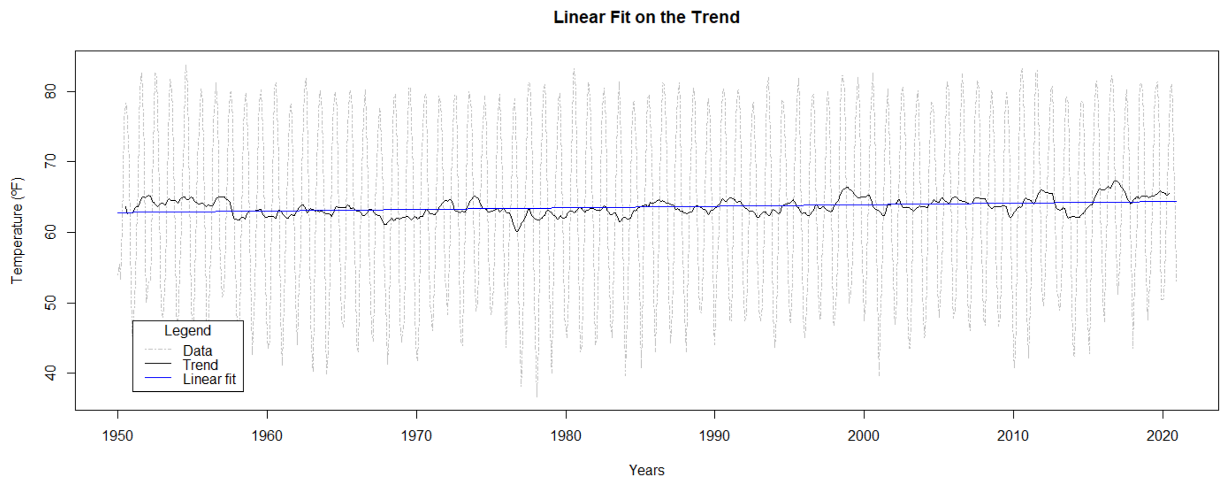


Regression Analysis

Firstly, to visualize that upward trend, the author performed a linear regression to get a line of best fit. However, since the temperature has a lot of variation, linear regression will result in a slope that is not significantly distinguishable from zero. For linear models, high variation leads to a wider confidence interval, and since the slope of the line is near zero, a wide confidence interval for slope will inevitably include zero. A narrower confidence interval is required to validate this upward tendency. One way to remove variation from the data is to decompose the time series and use the trend component. This decomposition will extract any upward or downward behavior over time from the seasonal and random parts, which do not have any upward or downward behavior over time. So the trend component from the decompose function will possess any positive or negative trend, only with less temperature variation. Decomposing the time series and fitting a linear model to the trend component produced the following plot (Figure 2). The line of best fit has a positive slope that is statically significant (p < 2e-16), clearly showing an upward trend in this time series.

Figure 2:

The Upward Trend of MS Average Temperatures



For clarity, the dotted gray line is the initial time series, identical to Figure 1, the solid black line is the trend component obtained via decomposition, and the solid blue line is the linear fit of that trend.

A nonlinear fit would also be suitable for the time series. In this case, the author utilized a nonlinear least squares fit to produce a regression fit of the data set. The following family of functions should feel familiar to anyone who has studied sine transformations:

$$at + bsin(ct + d) + e,$$

where t is time; a, b, and c are coefficients; and d and e are constants. Figure 3 shows the result of performing a regression fit with these five parameters, and Table 1 shows the estimated value and level of significance for each of the five parameters.

Figure 3:

The Input Data, Regression Fit, and Prediction

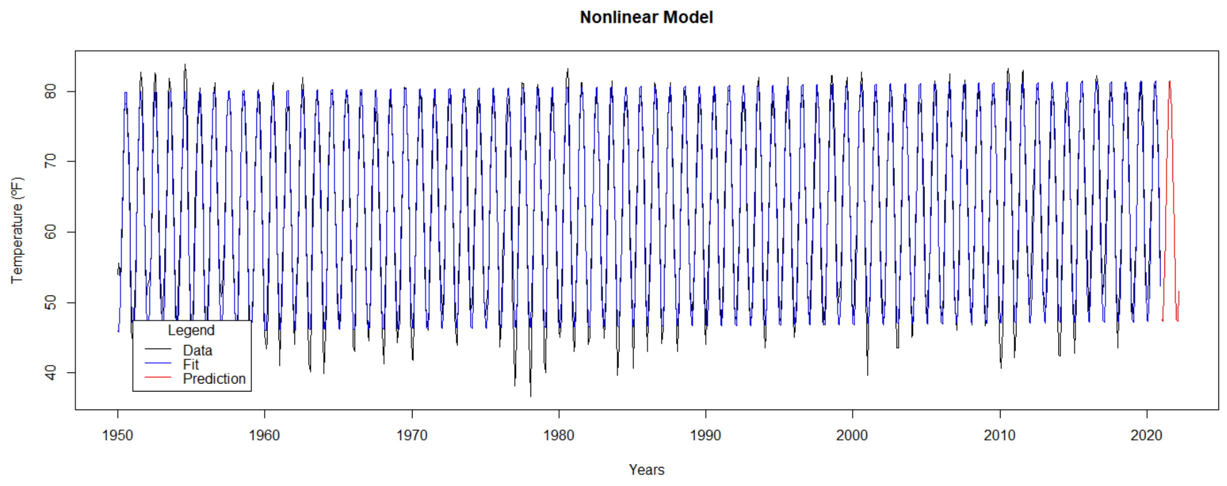


Table 1:

Parameter Estimates for the Nonlinear Fit

Parameter:	**Estimate:**	**Significance: [Pr(> | t |)]**
a	0.020099	3.39e-07
b	-17.54	2e-16
c	6.283	2e-16
d	-10.73	2e-16
e	21.93	0.00696

In Figure 3, the black line is still the initial time series, identical to Figure 1. The blue line is the nonlinear fit. This regression fit captures the seasonality and trend of the input data well. Notice how the blue line covers the black line well for temperatures between 45 and 80 degrees. However, the regression fit does not contain every temperature. The maximum and minimum temperatures for any given year often do not fall onto the regression fit. This model may be a reasonable estimator for more moderate average monthly temperatures in autumn and spring, but this model cannot reliably estimate average monthly temperatures in winter and summer. Despite this shortcoming, the forecast for 2021 is acceptable. The prediction captures the trend and seasonality of the initial time series well, akin to the nonlinear fit.

Residuals show the difference between an actual value and the estimated value. Figure 4 is a graph of residuals comparing the regression fit and the input data.

Figure 4:

Residuals from the Nonlinear Fit

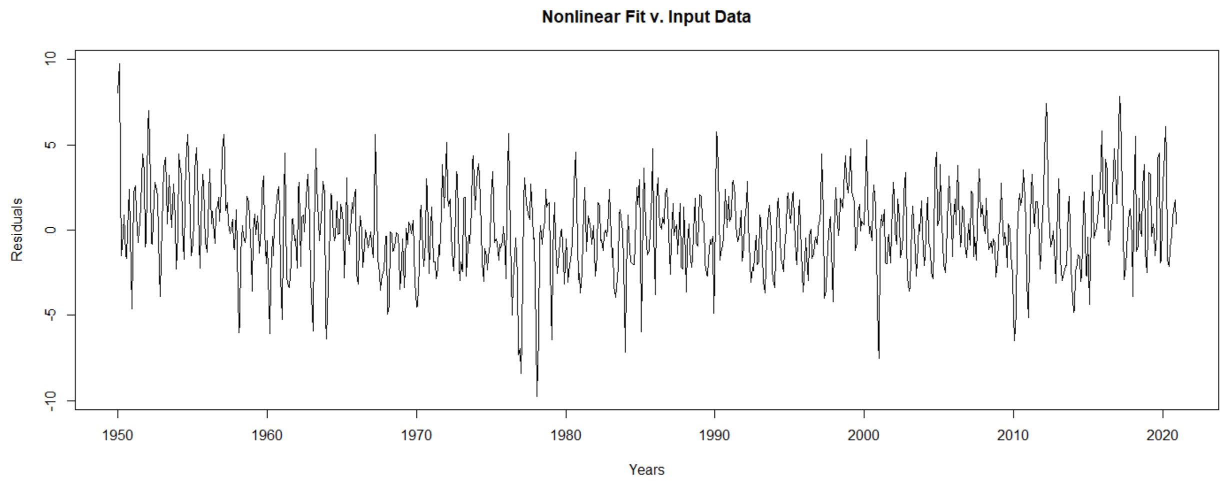
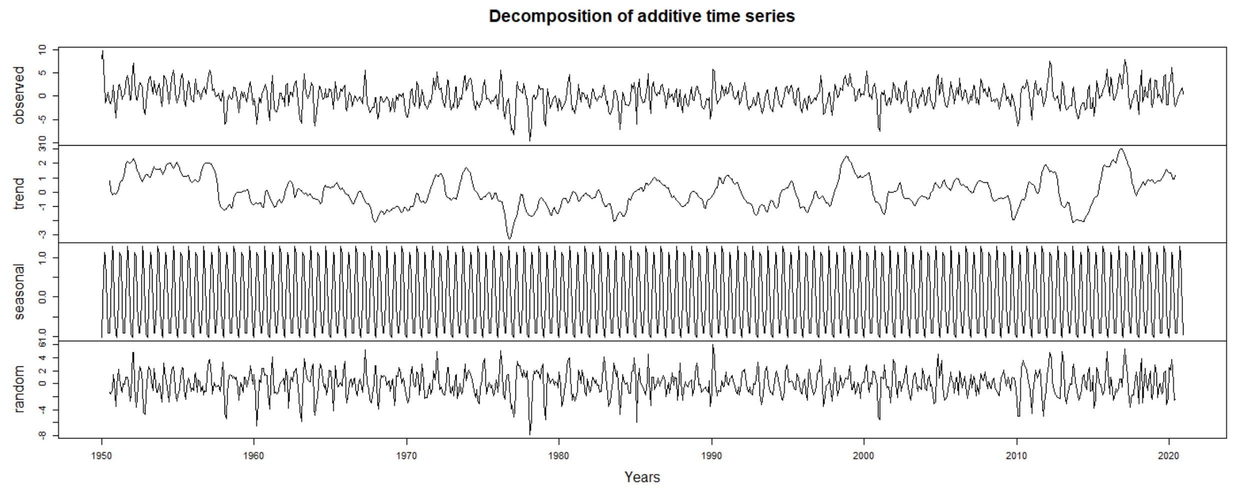


Figure 5:

Decomposition of Residuals from the Nonlinear Fit

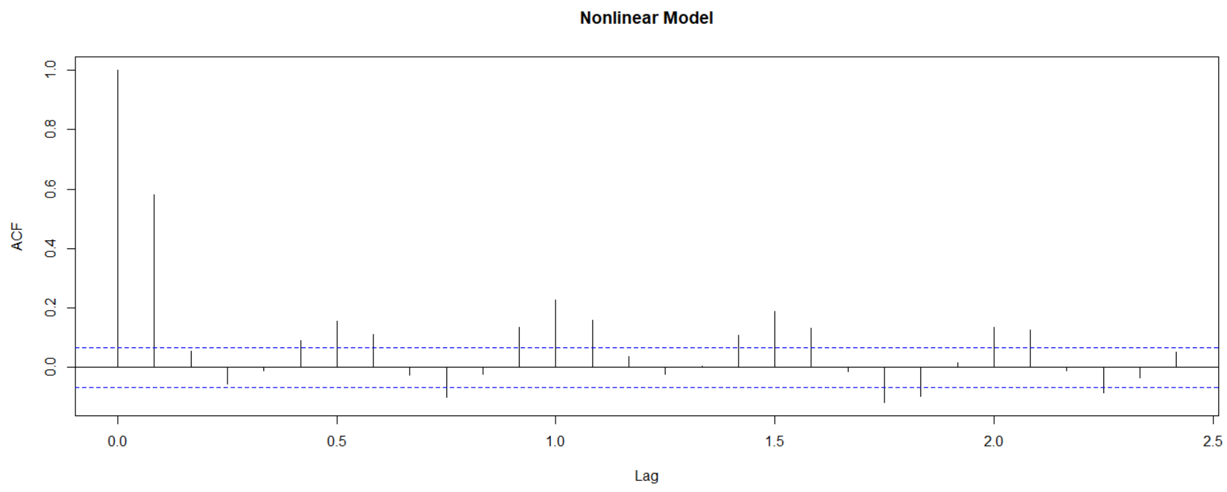


The plot in Figure 4 still has some seasonality which is not surprising since, each year, maximum and minimum temperatures usually did not fall onto the regression fit. Figure 5 confirms that there is some seasonality in Figure 4. Although, the scale of the seasonal component is less than the scale of the trend or random parts. An autocorrelogram should help to determine whether or not that seasonality is significant. Additionally, Figure 4 shows that the regression fit usually estimates the average monthly temperature of Mississippi within five degrees, plus or minus ten degrees at most. These residuals are centered around zero, and there is no trend upward or downward spanning all seventy years of the data. Despite the seasonality present in Figure 4, these residuals are predominantly random.

Looking at an autocorrelogram for the residuals of the nonlinear model will help to determine the quality of the model and assess whether there is room for improvement.

Figure 6

An Autocorrelogram of the Residuals from the Nonlinear Fit



Notice that there are many instances of significant autocorrelation. This correlation indicates that further analysis should lead to a better prediction. Furthermore, the spacing between the moments of the highest correlation is usually 0.25 lag, and the correlation alternates from negative to positive. This behavior suggests that that is some significant seasonality present in these residuals. A seasonal ARIMA model may be beneficial for approximating average monthly temperatures, but first, the author will generate an AR model.

Adapting the Nonlinear Model with an AR Process

Figure 7 shows an AR model: combining the nonlinear fit plus an AR(23) process produced from the residuals shown in Figure 4. This model is, admittedly, overparameterized, but with seventy years of data, sacrificing twenty-three months to generate a model that can make predictions is viable. The high order of the AR process explains why about two years of approximations are missing from the start of the AR model. This AR model has all the pros of the nonlinear model with fewer cons. Similarly to the nonlinear model, the AR model captures the seasonality and trend of the times series well. Although, now the model estimates crests and troughs of the initial time series more accurately. Looking at the new residuals, which compares the AR fit to the input data, can confirm this observation.

Figure 7:

The Input Data, AR Fit, and Prediction

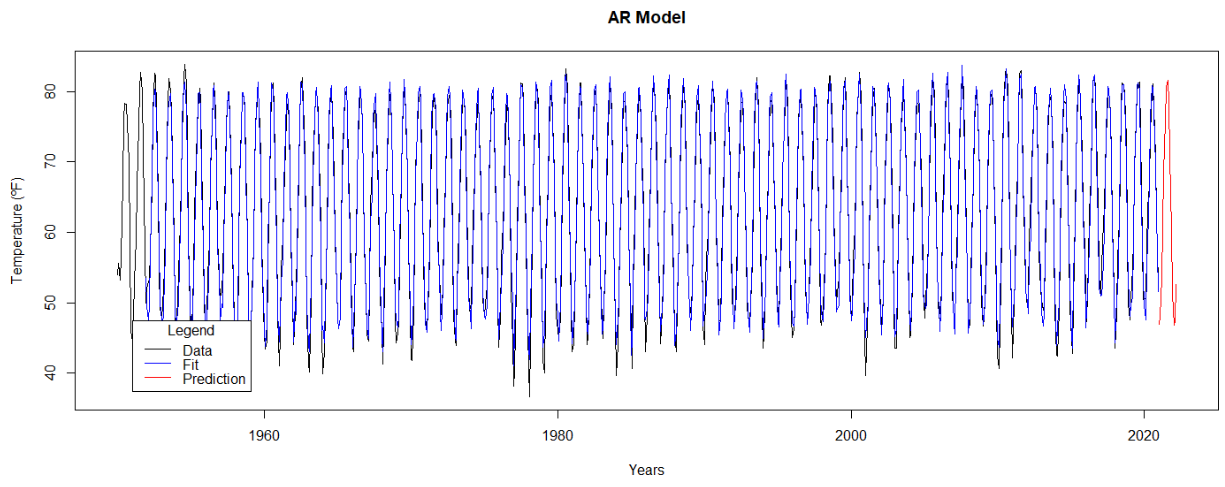


Figure 8:

Residuals from the AR Fit

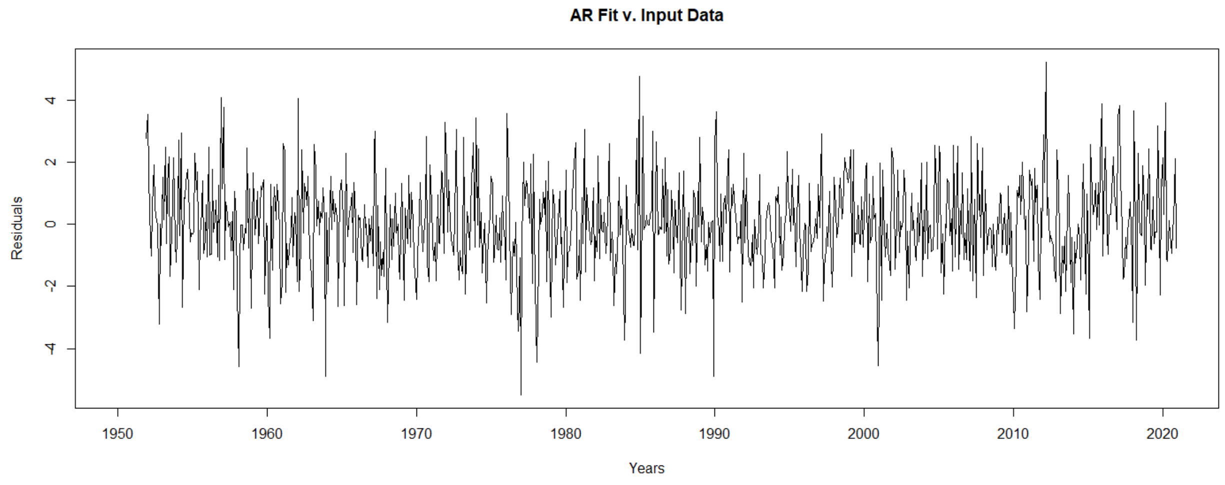
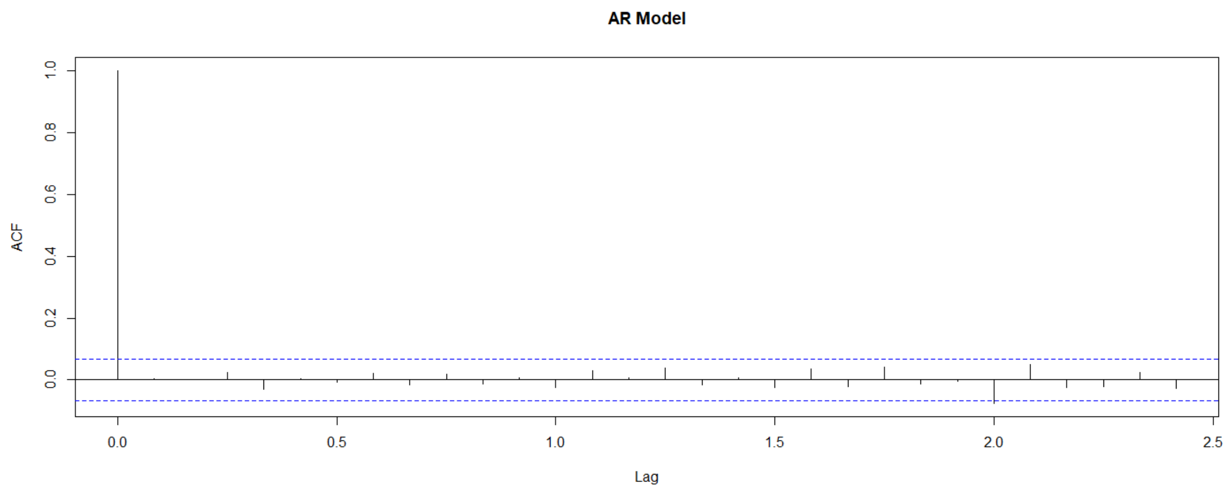


Figure 8 shows that the residuals from the AR model are less than the residuals from the nonlinear model (Figure 4). Rather than the estimate being within plus or minus ten degrees, usually five, the new approximation is generally within two degrees of the actual value, plus or minus five degrees at most. The prediction generated from the AR model is reasonable. It seems as accurate as it can be considering the strengths and weaknesses of this model. The new prediction looks very similar to the forecast in Figure 3, so only minor adjustments were made by the AR(23) process.

Once again, an autocorrelogram can assist in assessing the integrity of this model. Comparing Figure 9 to Figure 6 shows that the AR fit is an improvement from the nonlinear model.

Figure 9:

An Autocorrelogram of the Residuals from the AR Fit



With only one instance of significant correlation, it is not likely that further analysis will lead to a better model. However, twenty-three parameters are excessive, so it is time to see how an ARIMA model compares to this AR model. Maybe an ARIMA fit will have fewer parameters.

Adapting the Nonlinear Model with an ARIMA Process

Figure 10 shows the ARIMA model, which is a combination of the nonlinear model and a seasonal ARIMA$(1, 0, 1)(1, 1, 2)_{12}$ process, and Figure 11 shows the residuals from this model when compared to the original time series.

Figure 10:

The Input Data, ARIMA Fit, and Prediction

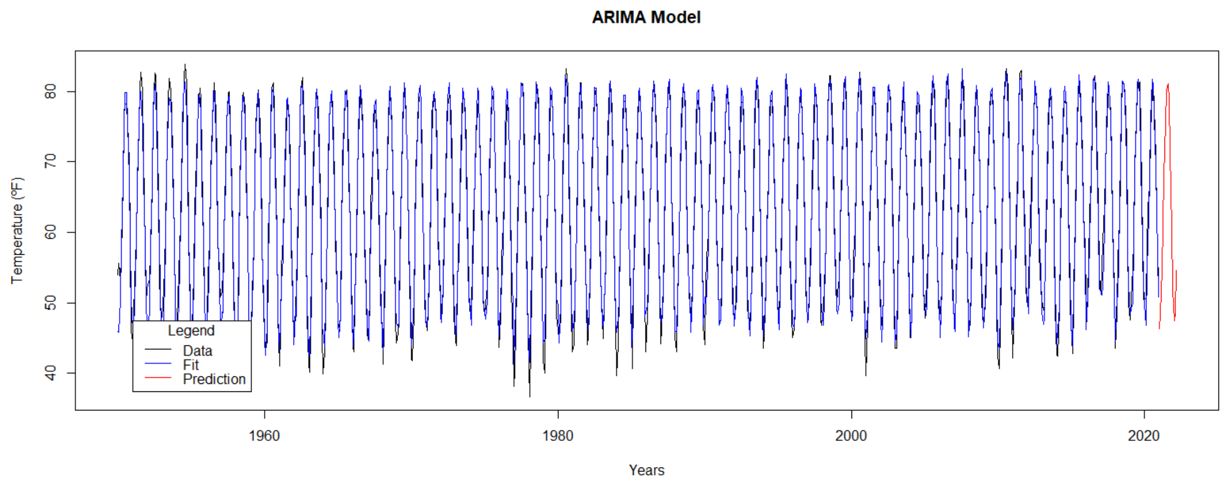
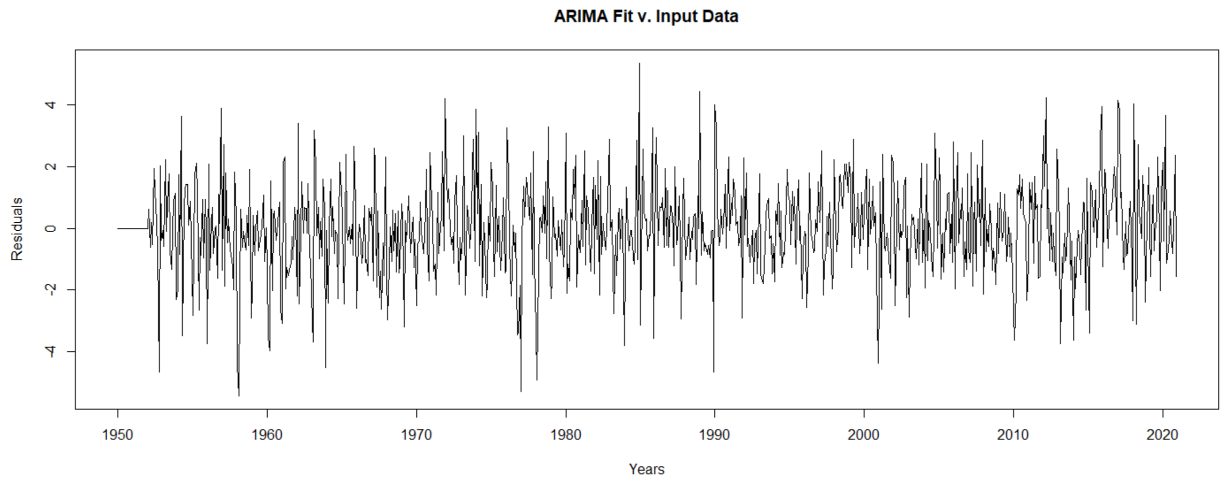


Figure 11:

Residuals from the ARIMA Model

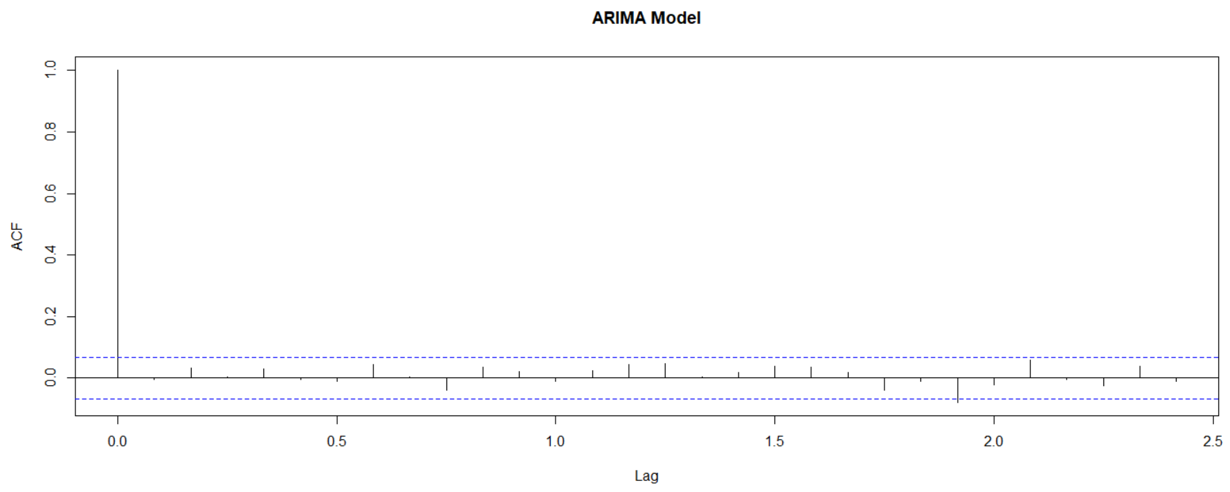


The author used the function get.best.arima() to arrive at his choice of orders. This function iterates through a range of orders, finding the best combination according to the Akaike information criterion (Cowpertwait & Metcalfe, 2009). At first glance, the ARIMA model is indistinguishable from the AR model. This model has the same strengths as the AR model: it captures the seasonality and trend of the data as well as high and low average monthly temperatures. Upon further inspection, the ARIMA model includes approximations for the first two years of data. Before, the AR model used these values to approximate the rest of the data, but that is because the AR process in that model is of order twenty-three. The ARIMA process in this model has far fewer parameters at five. The prediction for the ARIMA model seems as reasonable as the two predictions before. Furthermore, all three predictions look nearly identical. Looking at Figure 11 confirms the accuracy of this model and its resemblance to the AR model. Similar to the AR model, each approximation from the ARIMA model is usually within two degrees of the actual value, at most plus or minus five degrees.

Looking at an autocorrelogram for the ARIMA model residuals shows that this model is, once again, an improvement compared to the nonlinear model.

Figure 12:

Autocorrelogram of the Residuals from the ARIMA Fit



There is nearly no autocorrelation present in Figure 12, with only one instance of significant correlation around lag two. This autocorrelogram is similar to the autocorrelogram for the AR model residuals and shows that the ARIMA model does not have much room for improvement.

Ways to Validate or Invalidate These Models

Usually, when creating a model that can make predictions, there is no way to compare the estimate with the future without waiting a sufficient amount of time. In this case, there is real-life data for comparing the predictions from the nonlinear, AR, and ARIMA models with one another. Recall that the author removed some data from the end of a dataset before analyzing the average monthly temperatures of Mississippi from January 1950 to December 2020. The author will use this removed portion of data to judge the quality of each prediction. This method of reserving data somewhat as an "oracle" for comparison will be valuable but increases the risk of data snooping. The author tried to minimize his exposure to the oracle while getting the average monthly temperatures of Mississippi so that it would not influence his analysis.

The original dataset had the average temperature of Mississippi for every month from January 1950 to August 2022. The author truncated the data set after December 2020 and used the remaining data beginning in January 2021 to judge the predictions for each model. One period, from trough to trough, for the oracle was from February 2021 to February 2022, so the author used the models to make predictions from January 2021 to March 2023 to encapsulate this period of temperatures.

The author used the following expression as a tool for judging the accuracy of each prediction.

$$\left(1 - \frac{\Sigma\left|x_i - y_i\right|^2}{\Sigma\left|y_i\right|^2}\right) \times 100\%$$

Let x be a prediction, and y be the previously mentioned oracle. This computation will provide a scale-independent measure of prediction quality. This method is not the only way to judge each prediction. Other methods include mean absolute error, root mean squared error, error sum of squares, explained variation, etc. All these measures for error involve differencing corresponding values from the prediction and the oracle, but not every method will produce a scale-independent percentage for accuracy.

What were the Results?

The author produced the following plot and table after using the preceding expression to calculate prediction accuracy.

Figure 13:

Nonlinear, AR, and ARIMA Model with the Oracle Data

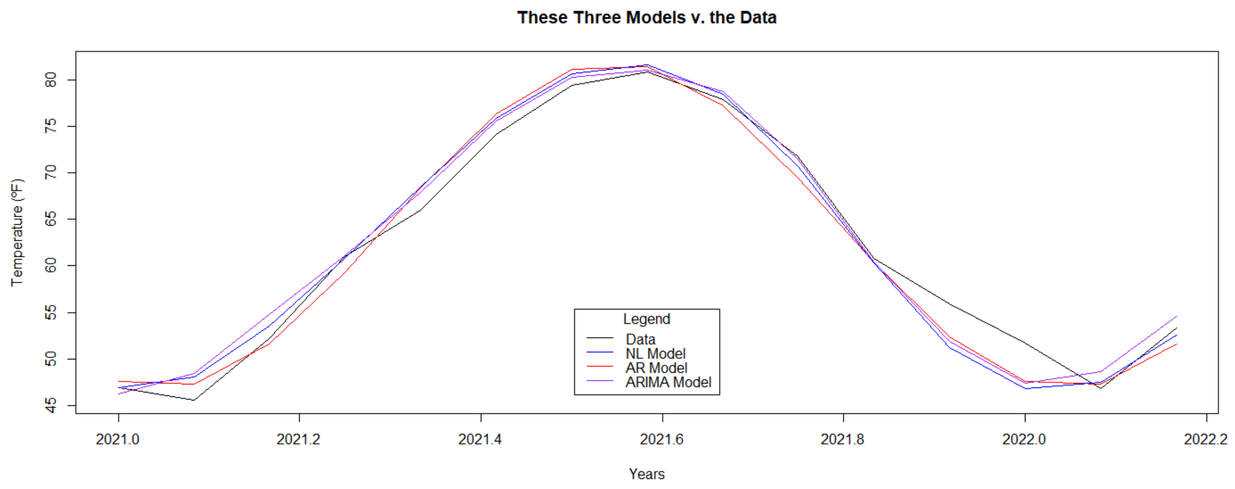


Table 2:

A Comparison of the Three Preceding Models

Model:	**Prediction Accuracy: (%)**	**Number of Parameters:**
Nonlinear (NL)	99.92598	5
NL + AR(23) process	99.93750	28 (5 + 23)
NL + $ARIMA(1, 0, 1)(1, 1, 2)_{12}$	99.93631	10 (5 + 5)

All three models performed well: each prediction accuracy was above 99.9 percent. Figure 13 illustrates precisely how close these three models were to each other and the data used for comparison. Using prediction accuracy alone, the nonlinear model would place third, the ARIMA model would come second, and the AR model would receive first place. However, remember that the AR process of the AR model is of order twenty-three. This AR process is exceptionally overparameterized compared to the five parameters of the seasonal ARIMA process in the ARIMA model. The ARIMA model will have ten parameters total, including the nonlinear part. Although ten parameters are greater than five from the nonlinear model alone, the ARIMA model provides more accurate predictions that capture high and low average temperatures more reliably (recall Figures 3 and 10). Because of its satisfactory predictions and low number of parameters, the ARIMA model would be the best of these three methods to estimate future average monthly temperatures of Mississippi

Reflection

Overall, my project went well. I would even say it was enjoyable, especially during the second half of the project. Personally, the main obstacle was starting the project, i.e., deciding on a topic, finding usable data, managing and cleaning the data, loading it into R, and performing regression analysis. The biggest hurdle I encountered was getting starting parameters for the nonlinear fit (thanks to Dr. Schroeder for helping me overcome my oversight regarding the period of the time series and the starting parameters). I felt comfortable using R after completing the first four stages of this project, and the remaining stages went much more smoothly. I successfully produced an AR model and various plots with assistance from the time series notes, the textbook, and online resources. The next hurdle I encountered was when I tried to make the ARIMA model before the class learned about the get.best.arima function. Luckily, I knew enough to stop there rather than trying to figure out the seasonal ARIMA part myself without the get.best.arima function. Another issue I encountered was working on my code rather than the report. I often wrote code in my free time, delaying my progress on the written portion because I currently enjoy writing code more than writing narratives. If I had to do this assignment again, I would likely begin the project as soon as possible and ask more questions. I have a habit of going to Google first if I have a query outside of class, when emailing the professor or asking a question in-person would likely be more valuable. Once again, I am happy with how my project went. I believe this assignment provided me with a gratifying introduction to data analysis and R, strengthening my understanding of statistics.

References

Cowpertwait, P. S. P., & Metcalfe, A. V. (2009). get.best.arima. In *Introductory Time Series with R* (pp. 144–145). essay, Springer.

Wong, J. (2022). *Average Monthly Temperature by US State*. Kaggle. Retrieved from https://www.kaggle.com/datasets/justinrwong/average-monthly-temperature-by-us-state

Appendix

```
# MAT482 Data Analysis: Times Series Project
# This is an analysis of Mississippi average temperatures from 1950
through 2020.
# A prediction will be made for 2021 temperatures.
# Then, the prediction will be compared to real values for 2021.

rm(list = ls())

# Get the data [Stage 1]
setwd("/Users/razor/Desktop/MAT 482/TS Project")

# msat = MS Average Temperatures (recorded monthly)
data <- ts(read.csv("MS_avg_temp_1950-2022.csv")[,4], st=1950, fr=12)
msat.ts <- window(data, st=1950, end=2020+(11/12))
# saving the last period for later
#plot(decompose(msat.ts))


# Labeled time series plot of the data [Stage 2]
plot(msat.ts, main='Avgerage Monthly Temperature of MS',
ylab='Temperature (ºF)', xlab='Years')
plot(decompose(msat.ts)$trend, main='Trend from Decomposition',
ylab='Temperature (ºF)', xlab='Years')


# Linear regression [Stage 3]
trend <- decompose(msat.ts)$trend
ms.t <- time(msat.ts) # time variable for msat.ts [1950,2021)
linfit <- lm(trend ~ ms.t)
I <- linfit$coef[1]   # intercept
S <- linfit$coef[2]   # slope
linmodel <- S*ms.t + I
summary(linfit)

# Checking some confidence intervals
#confint(linfit, 'ms.t')
#confint(linfit, '(Intercept)')
```

```
# Plot of the data with linear regression
ts.plot(cbind(msat.ts,trend), lty=c(4,1),col=c('gray','black'),
main='Linear Fit on the Trend', ylab='Temperature (ºF)',
xlab='Years')
lines(linmodel, col='blue')
legend('bottomleft', inset=0.05, c('Data', 'Trend', 'Linear fit'),
col=c('gray','black', 'blue'), lty=c(4,1,1), title='Legend')

# Nonlinear regression
a_s <- 0.02   # a start
b_s <- -18    # b start
c_s <- 6.28   # c start
d_s <- -5     # d start
e_s <- 20     # e start
nlfit <- nls(msat.ts ~ a*ms.t + b*sin(c*ms.t + d) + e,
start=list(a=a_s, b=b_s, c=c_s, d=d_s, e=e_s))
A <- coef(nlfit)[1]
B <- coef(nlfit)[2]
C <- coef(nlfit)[3]
D <- coef(nlfit)[4]
E <- coef(nlfit)[5]
fit <- A*ms.t + B*sin(C*ms.t + D) + E
summary(nlfit)

# Checking some confidence intervals
#confint(nlfit, 'a')
#confint(nlfit, 'b')
#confint(nlfit, 'c')
#confint(nlfit, 'd')
#confint(nlfit, 'e')

# Messing with time
future.t <- time(ts(fit, st=2021, end=2022+(1/6), fr=12))  # time
variable for the future fit [2021,2022)
past <- A*ms.t + B*sin(C*ms.t + D) + E
future <- A*future.t + B*sin(C*future.t + D) + E

# Combined plot of msat.ts, fit.ts, and future.ts (with legend)
```

```
ts.plot(cbind(msat.ts, past, future), col=c('black','blue', 'red'),
xlim=c(1950,2022), main='Nonlinear Model', ylab='Temperature (ºF)',
xlab='Years')
legend('bottomleft', inset=0.05, c('Data', 'Fit', 'Prediction'),
col=c('black','blue', 'red'), lty=1, title='Legend')


# Residuals (nlfit v. data) [Stage 4]
res <- residuals(nlfit)
#plot(past + res.ts - msat.ts)
res.ts <- ts(res, st=1950, fr=12)
plot(res.ts, main='Nonlinear Fit v. Input Data', xlab='Years',
ylab='Residuals')
plot(decompose(res.ts), xlab='Years')


# Autocorrelation function [Stage 5]
acf(res.ts, main='Nonlinear Model')
# acf(decompose(res.ts)$seas, main='Seasonal of NL Model')
# comparing these correlelograms confirms that there is significant
seasonal correlation in the residuals


# AR model [Stage 6]
res.ar <- ar(res.ts)                                # ar of the
residuals time series
ar_fit <- past + res.ar$resid                       # nlfit of the
input data + ar residuals
ar_fut_res <- predict(res.ar, n.ahead=15)$pred     # prediction for
future residuals
ar_future <- future + ar_fut_res                    # future fit + ar
residuals

# Generate a plot of all three parts
ts.plot(cbind(msat.ts,ar_fit,ar_future), col=c('black','blue','red'),
main='AR Model', ylab='Temperature (ºF)', xlab='Years')
legend('bottomleft', inset=0.05, c('Data', 'Fit', 'Prediction'),
col=c('black','blue', 'red'), lty=1, title='Legend')
plot(res.ar$resid,main='AR Fit v. Input Data', xlab='Years',
ylab='Residuals')
summary(res.ar$resid)
summary(res.ts)

# AR residual [Stage 7]
acf(na.omit((res.ar$resid)), main='AR Model')

# The get.best.arima function
get.best.arima <- function(x.ts, maxord = c(1,1,1,1,1,1))
{
  best.aic <- 1e8
  n <- length(x.ts)

  for(p in 0:maxord[1])
    for (d in 0:maxord[2])
      for(q in 0:maxord[3])
        for(P in 0:maxord[4])
          for(D in 0:maxord[5])
            for(Q in 0:maxord[6])
            {
              try(
                {
                  fit <- arima(x.ts, order=c(p,d,q), seas =
list(order=c(P,D,Q), frequency(x.ts) ), method = "CSS")
                  fit.aic <- -2*fit$loglik + (log(n)+1) *
length(fit$coef) #suppressing code after the + gives better fits
(using loglik as the only criterion that way)
                  if (fit.aic < best.aic)
                  {
                    best.aic <- fit.aic
                    best.fit <- fit
                    best.model <- c(p,d,q,P,D,Q)
                  } #end if
                } # end first argument of try
                , FALSE) #end try

              print(c(p,d,q,P,D,Q))
              flush.console()
            } # end for
  dev.new()


  over <- paste("Process Fit with ARIMA(", toString(best.model[1]),
",", toString(best.model[2]), ",", toString(best.model[3]), ")
Process. \n Coefficients:", toString(round(best.fit$coef, digits=3)))
  under <- paste("Periodic Coefficients:", toString(best.model[4]),
",", toString(best.model[5]), ",", toString(best.model[6]))

  acf(na.omit(best.fit$resid), lag.max=100, main=over, xlab=under)



  list(akaike=best.aic, data=best.fit, ordersbest_arima=best.model)
} # end get.best.arima

#get.best.arima(res.ts, c(24,1,1,1,1,2))
best_arima <- arima(res.ts, order=c(1,0,1), seas=list(order=c(1,1,2),
12), method = "CSS")
summary(best_arima)


# ARIMA model [Stage 8]
arima_res <- best_arima$res
arima_fut_res <- predict(best_arima, n.ahead=15)$pred
arima_fit <- past + arima_res
arima_future <- future + arima_fut_res
#plot(best_arima$data)

# Generate a plot of all three parts
ts.plot(cbind(msat.ts,arima_fit,arima_future),
col=c('black','blue','red'), main='ARIMA Model', ylab='Temperature
(ºF)', xlab='Years')
legend('bottomleft', inset=0.05, c('Data', 'Fit', 'Prediction'),
col=c('black','blue', 'red'), lty=1, title='Legend')
plot(arima_res, main='ARIMA Fit v. Input Data', xlab='Years',
```

```r
ylab='Residuals')
summary(arima_res)
summary(res.ar$resid)


# ARIMA residual [Stage 9]
acf(arima_res, main='ARIMA Model')


# Reload the original data set [Stage 10]
plot(data)


# Extract the last period [Stage 11]
vault <- window(data, st=2021, end=2022+(1/6))
plot(vault)


# Write a function to compute differences [Stage 12]
difference <- function(x, y){
  length.x <- length(x)
  length.y <- length(y)

  if(length.x == length.y){
    numerator <- sum(abs(x-y)[1:length.x])^2
    denominator <- sum(abs(y)[1:length.y])^2

    return((1-(numerator/denominator))*100)
  }
  else{
    stop('The length of vectors x and y must be equal')
  }
}


# Compute differences for every prediction [Stage 13]
difference(vault, future)
difference(vault, ar_future)
difference(vault, arima_future)
```

```
# Plot showing all the predictions v. the input data
ts.plot(cbind(vault, future, ar_future, arima_future),
col=c('black','red','blue','purple'), main='These Three Models v. the
Data', xlab='Years', ylab='Temperature (ºF)')
legend('bottom', inset=0.05, c('Data','NL Model', 'AR Model', 'ARIMA
Model'), col=c('black','blue', 'red','purple'), lty=1,
title='Legend')

# Calculate explained variance for all three models
exp_var <- function(data, estimator){
  length.data <- length(data)
  length.est <- length(estimator)
  data.avg <- mean(data)

  if(length.data == length.est){
    numerator <- sum(((data - estimator)^2)[1:length.est])
    #print('numerator:'); print(numerator)
    denominator <- sum(((data - data.avg)^2)[1:length.data])
    #print('denominator:'); print(denominator)

    return((1-(numerator/denominator))*100)
  }
  else{
    stop('The length of the two vectors must be equal')
  }
}

exp_var(msat.ts, past)
exp_var(window(msat.ts, st=1951+(11/12)), window(ar_fit,
st=1951+(11/12)))
exp_var(msat.ts, arima_fit)
```