1) For ARMA(2,1) process

$$X_t = 0.2X_{t-1} + 0.08X_{t-2} + W_t + 0.7W_{t-1}.$$

I have at first tried 30 and 20 breaks, which didn't seem to give optimum look. Then I did a little research and used Sturges formula to determine breaks which gave me 15, and then tried Freedman-Diaconis formula which gave me 60 breaks (not included because it was also not great). I decided on 15 which can be seen on figure 3.

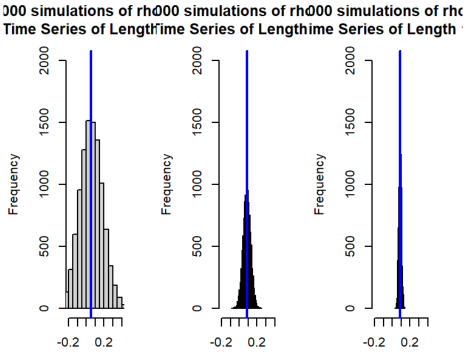


Figure 1: 30 breaks

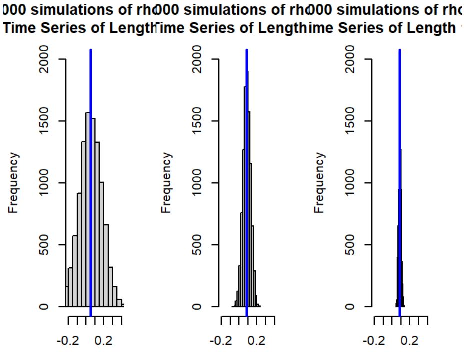


Figure 2: 20 breaks

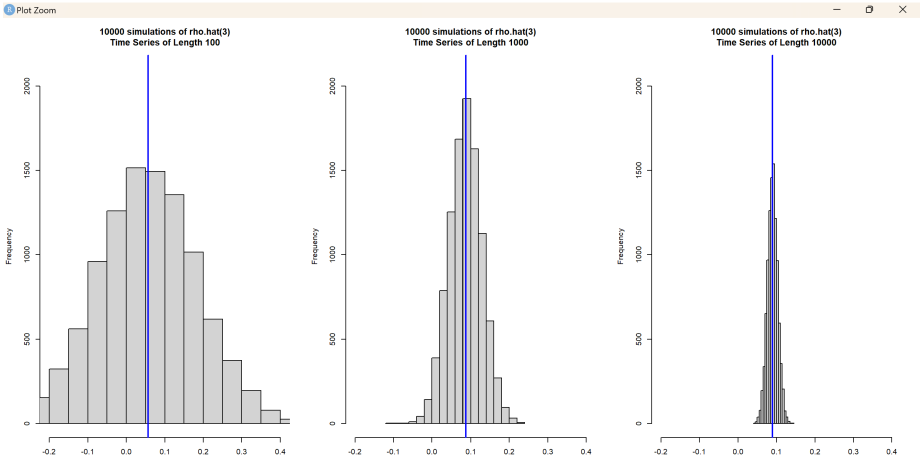


Figure 3: 15 breaks

R code:

```
# ARMA(2,1) process
phi <- c(0.2, 0.08)  # AR coefficients
theta <- 0.7         # MA coefficient
n_values <- c(100, 1000, 10000)
n_simulations <- 10000

simulate_rho <- function(n) {
  rho_values <- numeric(n_simulations)
  for (i in 1:n_simulations) {
    # simulate ARMA(2,1)
    arma_process <- arima.sim(model = list(ar = phi,
    ma = theta), n = n)
    # compute rho(3)
    rho_values[i] <- acf(arma_process, plot = FALSE)$acf[4]
  }
  return(rho_values)
}

# histograms
par(mfrow = c(1, 3))  # Arrange plots in a row
for (n in n_values) {
  rho_values <- simulate_rho(n)
  hist(rho_values, breaks = 15, main = paste("10000
  simulations of rho.hat(3) \n
  Time Series of Length", n),
       xlab = "",ylab = "Frequency",xlim = c(-0.2, 0.4),
       ylim = c(0, 2100))
  abline(v = mean(rho_values), col = "blue", lwd = 2)
}

# determining breaks
n <- length(rho_values)
sturges_breaks <- ceiling(log2(n) + 1)
print(sturges_breaks)  # 15

bin_width <- 2 * IQR(rho_values) / (n^(1/3))
fd_breaks <- ceiling(diff(range(rho_values)) / bin_width)
print(fd_breaks)  # 60 definitely not
```

The histograms indicating the distribution of the values of $\hat{\rho}(3)$;
For n=100, the distribution is wide, which tells us greater variability in the estimated autocorrelations.
For n=1000, the distribution becomes more concentrated around the true value of $\rho(3)$ which is 0.09.
For n=10000, even more closely concentrated with minimum variability.

Larger sample sizes lead to more accurate and precise estimates of the autocorrelation. For large n, the sample autocorrelation $\hat{\rho}(h)$ converges in probability to the true autocorrelation $\rho(h)$. This shows that increasing sample size reduces the variability in the estimates of autocorrelations.

2) For AR(2) process of

$$X_t = 0.7X_{t-1} - 0.6X_{t-2} + W_t$$

```
set.seed(4)
n <- 100
X <- arima.sim(model = list(ar = c(0.7, -0.6)), n)

par(mfrow = c(1, 2))
acf(X)
pacf(X)

rho.hat <- acf(X, plot = F)$acf[-1][1:2]
rho.hat

acf_values <- acf(X, plot = FALSE)$acf
acvf_lag0 <- var(X)
```

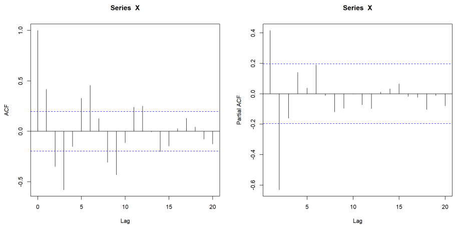
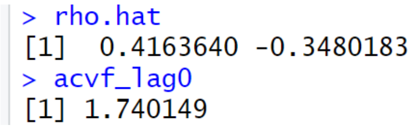


Figure 4: ACF and PACF



```
> rho.hat
[1]  0.4163640 -0.3480183
> acvf_lag0
[1] 1.740149
```

Figure 5: ACF, and sample ACVF at lag zero

```
R <- matrix(c(1, rho.hat[1], rho.hat[1], 1),2,2)
phi.hat <- solve(R,rho.hat)
phi.hat
```

$$\hat{\phi}_1 \approx 0.67897$$

$$\hat{\phi}_2 \approx -0.6307$$

These are respectable approximations to the ‘true’ values

$$\phi_1 = 0.7$$

and

$$\phi_1 = -0.6$$

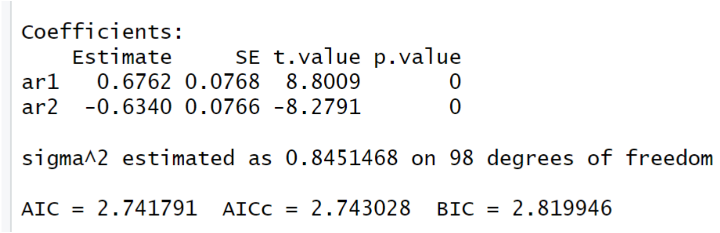
that we started with.
Yule-Walker estimator of the variance

```
gamma.0 <- acf(X, type = ’covariance’, plot = F)$acf[1]
var.W <- gamma.0*(1 - sum(rho.hat*phi.hat))
var.W
```

The results are $\hat{\gamma}(0) \approx 1.723$, and, $\hat{\sigma}^2_W \approx 0.858$.

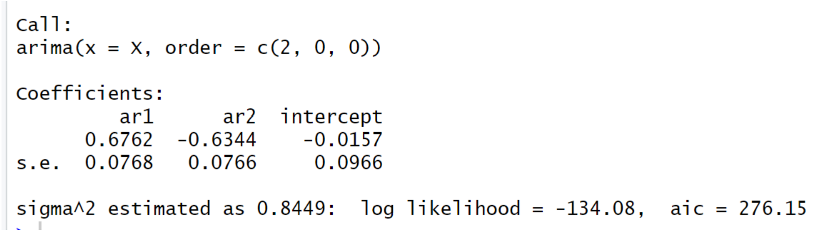
Maximum likelihood estimators with sarima:

```
 X.model <- sarima(X, 2,0,0, no.constant = TRUE, details = FALSE)
```



```
Coefficients:
    Estimate     SE t.value p.value
ar1   0.6762 0.0768  8.8009       0
ar2  -0.6340 0.0766 -8.2791       0

sigma^2 estimated as 0.8451468 on 98 degrees of freedom

AIC = 2.741791  AICc = 2.743028  BIC = 2.819946
```

Figure 6: sarima results



```
Call:
arima(x = X, order = c(2, 0, 0))

Coefficients:
         ar1      ar2  intercept
      0.6762  -0.6344    -0.0157
s.e.  0.0768   0.0766     0.0966

sigma^2 estimated as 0.8449:  log likelihood = -134.08,  aic = 276.15
```

Figure 7: arima results

from sarima:

$$\hat{\phi}_1 \approx 0.67$$

$$\hat{\phi}_2 \approx -0.63$$

The fitted model from arima is:

$$Y_t = 0.6762 Y_{t-1} - 0.6344 Y_{t-2} - 0.0157 + W_t,$$

where the white noise term with variance $\hat{\sigma}^2 \approx 0.8449$.

From sarima:

$$Y_t = 0.6762 Y_{t-1} - 0.6340 Y_{t-2} + W_t,$$

where the white noise term with variance $\hat{\sigma}^2 \approx 0.8451$. Note: I have attached the sarima plots to the end since it is another pdf from R. The standardized residuals seem to concentrate around zero. ACF seems to have a small spike on lag 9. As for QQ plot, points on the plot lie mostly along a line so I would say it is normally distributed.

```
Recreated ljung box
X2 <- sarima(X, 2,0,0)           # fit an AR(2) model
X2.resid <- X2$fit$residuals             # store the residuals
X2.st.resid <- X2.resid/sqrt(X2$fit$sigma2)


LBtest = NULL
```

```
for(H in 3:20){
  LBtest[H]=Box.test(X2.st.resid, lag = H,
  type='Ljung-Box', fitdf = 2)$p.value
  }
plot(LBtest, ylim = c(0,1))
grid(nx = NULL, ny = NULL, col = 'lightgray',
lty = 'dotted',lwd = par('lwd'), equilogs = TRUE)
abline(h = 0.05, lty = 'dashed', col='blue')
```

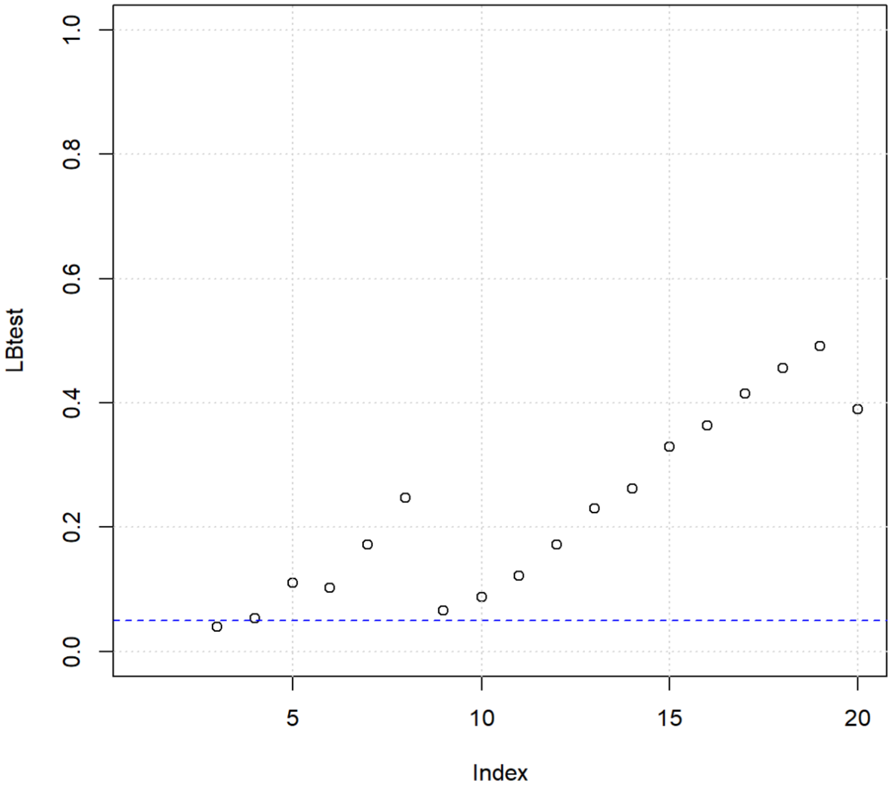
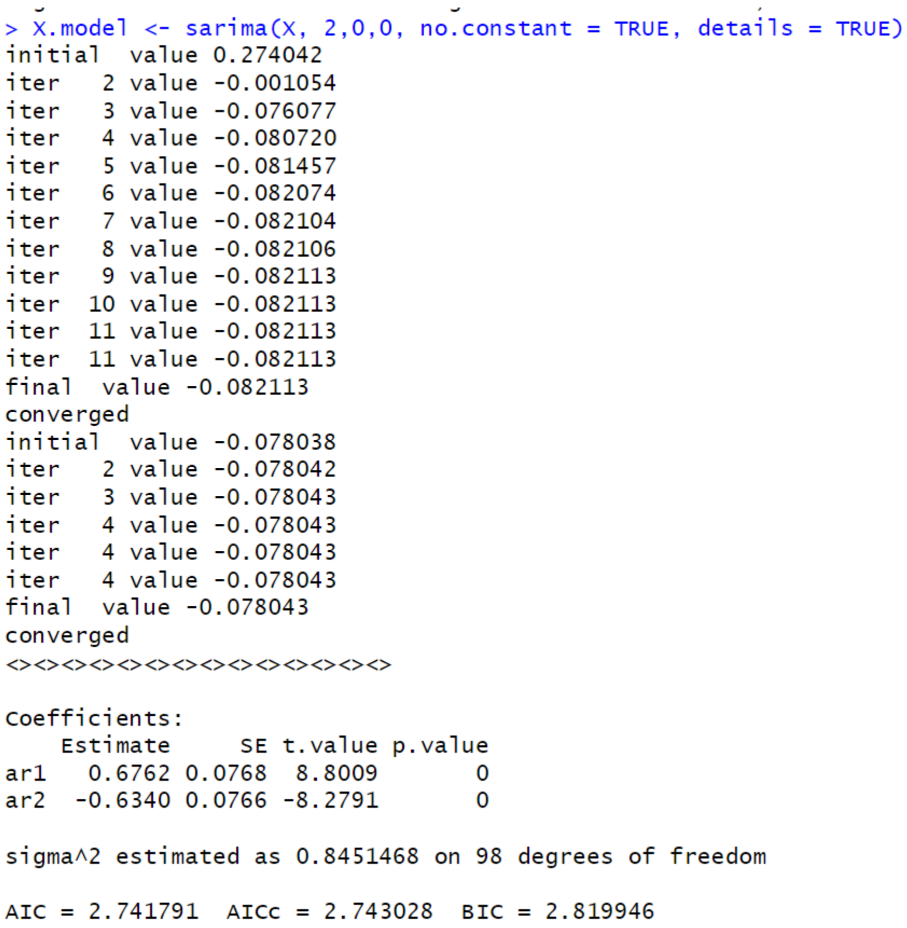


Figure 8: Ljung Box

Starting from lag 4 and on it looks like p values are above the treshold (which I also put in as 0.05) and this is the wanted result, we fail to reject.

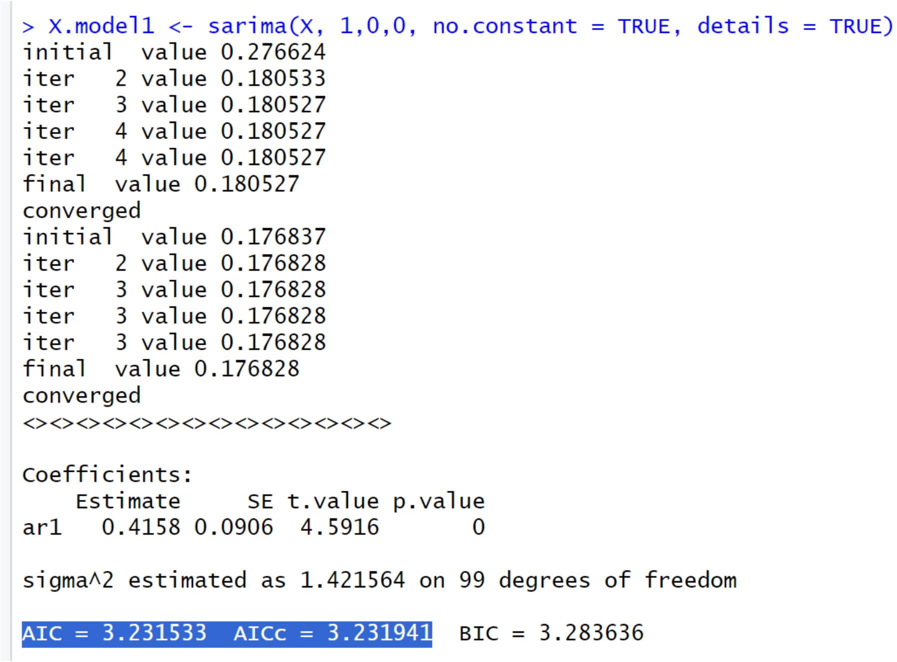
For AR(2), as I calculated with sarima before AIC = 2.741791 and AICc = 2.743028 from;



```
> X.model <- sarima(X, 2,0,0, no.constant = TRUE, details = TRUE)
initial  value 0.274042 
iter   2 value -0.001054
iter   3 value -0.076077
iter   4 value -0.080720
iter   5 value -0.081457
iter   6 value -0.082074
iter   7 value -0.082104
iter   8 value -0.082106
iter   9 value -0.082113
iter  10 value -0.082113
iter  11 value -0.082113
iter  11 value -0.082113
final  value -0.082113 
converged
initial  value -0.078038 
iter   2 value -0.078042
iter   3 value -0.078043
iter   4 value -0.078043
iter   4 value -0.078043
iter   4 value -0.078043
final  value -0.078043 
converged
<><><><><><><><><><><><><><>

Coefficients:
    Estimate     SE t.value p.value
ar1   0.6762 0.0768  8.8009       0
ar2  -0.6340 0.0766 -8.2791       0

sigma^2 estimated as 0.8451468 on 98 degrees of freedom 

AIC = 2.741791  AICc = 2.743028  BIC = 2.819946
```

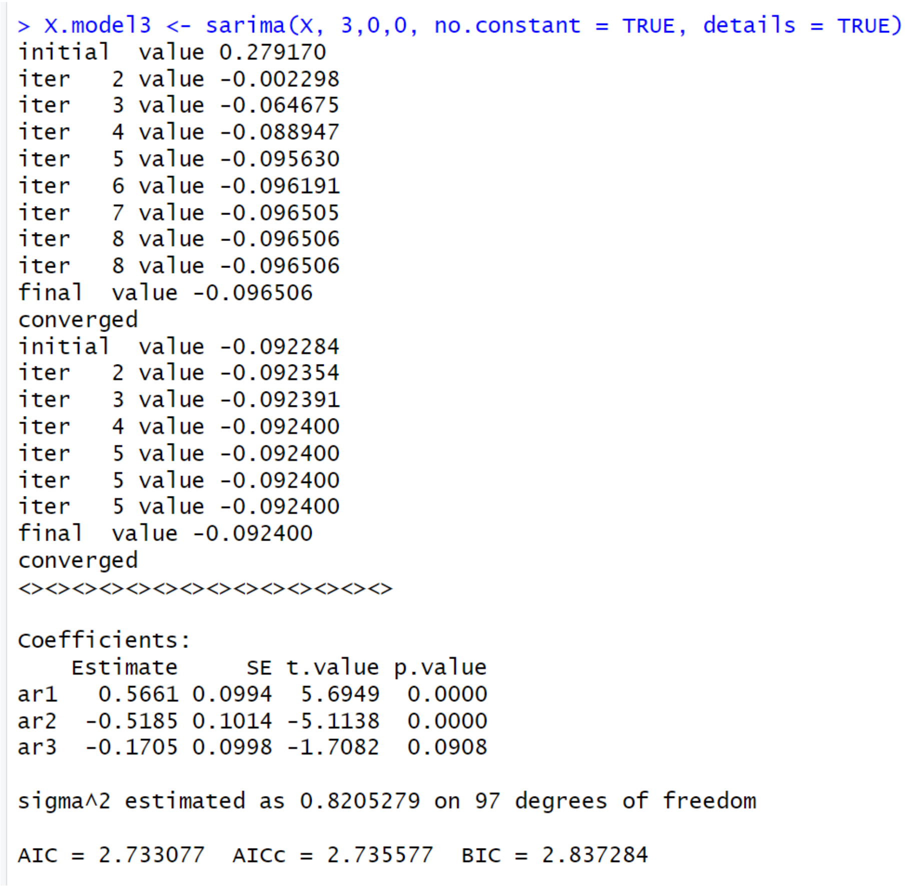
Figure 9: AR(2) AIC and AICc

For AR(1),



```
> X.model1 <- sarima(X, 1,0,0, no.constant = TRUE, details = TRUE)
initial  value 0.276624
iter   2 value 0.180533
iter   3 value 0.180527
iter   4 value 0.180527
iter   4 value 0.180527
final  value 0.180527
converged
initial  value 0.176837
iter   2 value 0.176828
iter   3 value 0.176828
iter   3 value 0.176828
iter   3 value 0.176828
final  value 0.176828
converged
<><><><><><><><><><><><><><>

Coefficients:
    Estimate     SE t.value p.value
ar1   0.4158 0.0906  4.5916       0

sigma^2 estimated as 1.421564 on 99 degrees of freedom

AIC = 3.231533  AICc = 3.231941  BIC = 3.283636
```

Figure 10: AR(1) AIC and AICc



```
> X.model3 <- sarima(X, 3,0,0, no.constant = TRUE, details = TRUE)
initial  value 0.279170
iter   2 value -0.002298
iter   3 value -0.064675
iter   4 value -0.088947
iter   5 value -0.095630
iter   6 value -0.096191
iter   7 value -0.096505
iter   8 value -0.096506
iter   8 value -0.096506
final  value -0.096506
converged
initial  value -0.092284
iter   2 value -0.092354
iter   3 value -0.092391
iter   4 value -0.092400
iter   5 value -0.092400
iter   5 value -0.092400
iter   5 value -0.092400
final  value -0.092400
converged
<><><><><><><><><><><><><><>

Coefficients:
    Estimate     SE t.value p.value
ar1   0.5661 0.0994  5.6949  0.0000
ar2  -0.5185 0.1014 -5.1138  0.0000
ar3  -0.1705 0.0998 -1.7082  0.0908

sigma^2 estimated as 0.8205279 on 97 degrees of freedom

AIC = 2.733077  AICc = 2.735577  BIC = 2.837284
```

Figure 11: AR(3) AIC and AICc

Since AIC and AICc choose the model with the smallest value, and AR(3) gives the smallest AIC with 2.733077 and AICc with 2.735577, AR(3) is a better fit.

3) I am using The AR(2) process from the first assignment:

$$X_t - 0.8X_{t-1} + 0.2X_{t-2} = W_t,$$

$$\text{AIC}(\beta) = -2\log L\left(\beta, \frac{1}{n}S(\beta)\right) + 2(p+q+1).$$

The penalty term in the formula will increase to account for the extra parameter.

$$k = 3 \quad (2 \text{ AR coefficients} + \text{variance}).$$

$$\text{AIC}_{\text{no mean}} = -2 \cdot (-200) + 2 \cdot 3 = 406$$

With Mean:

$$k = 4 \quad \text{(2 AR coefficients + mean + variance).}$$

$$\text{AIC}_{\text{with mean}} = -2 \cdot (-200) + 2 \cdot 4 = 408$$

The difference is 2 which makes sense since the additional parameter of mean is added.

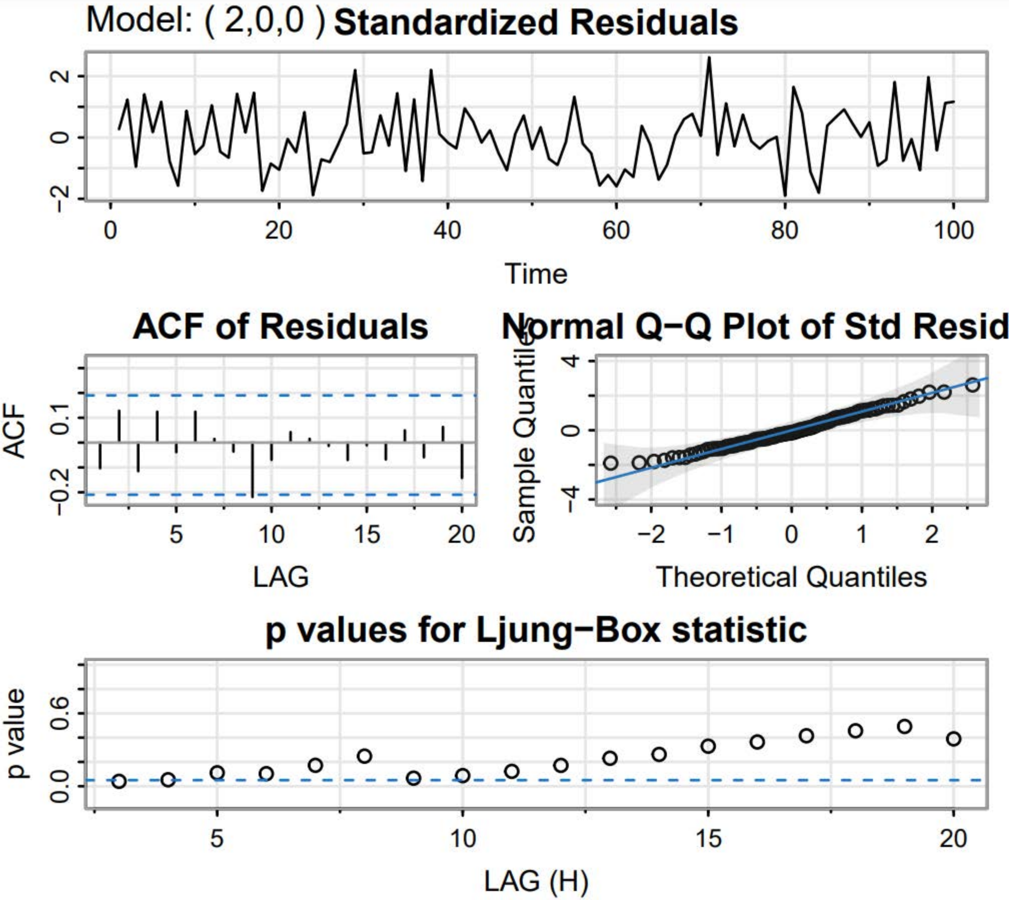
I calculated log likelihood with sarima

```
set.seed(123)
n <- 200
ar_params <- c(0.8, -0.2)
x <- arima.sim(model = list(ar = ar_params), n = n)

#AR(2) model without mean
fitzmean <- sarima(x, 2, 0, 0, no.constant = TRUE,
details = TRUE)

#AR(2) model with mean
fitmean <- sarima(x, 2, 0, 0, no.constant = FALSE,
details = TRUE)

#log likelihoods
loglikzmean <- fitzmean$fit$loglik # -271.9306
loglikmean <- fitmean$fit$loglik  # -271.9306
```

Since including a mean adds an extra parameter to the model, it can lead to a penalty in the AIC and AICc. If the true process has a non-zero mean, including a mean in the model can improve the fit to the data, potentially leading to a decrease in the AIC and AICc despite the added parameter. So it depends on the trade-off between the penalty for the extra parameter and the improvement in the model fit. In terms of dataset, it is a small dataset, the penalty for adding a mean is greater, making it less likely for AICc to decrease when including a mean. Considering the AICc formula;

$$\text{AICc}(\beta) = -2 \log L\left(\beta, \frac{1}{n} S(\beta)\right) + 2(p + q + 1) \cdot \frac{n}{n - p - q - 2}$$

Adding more parameters will cause more severe penalty and may outweigh the improvement in model fit.


Model: (2,0,0) Standardized Residuals
2
0
−2
0
20
40
60
80
100
Time
ACF of Residuals
ACF
0.1
−0.2
5
10
15
20
LAG
Normal Q−Q Plot of Std Resid
Sample Quantiles
4
0
−4
−2
−1
0
1
2
Theoretical Quantiles
p values for Ljung−Box statistic
p value
0.6
0.0
5
10
15
20
LAG (H)