MATH 8090: ARMA Models-Inference, Diagnostics, and Model Selection

Whitney Huang, Clemson University

9/23-9/25/2025

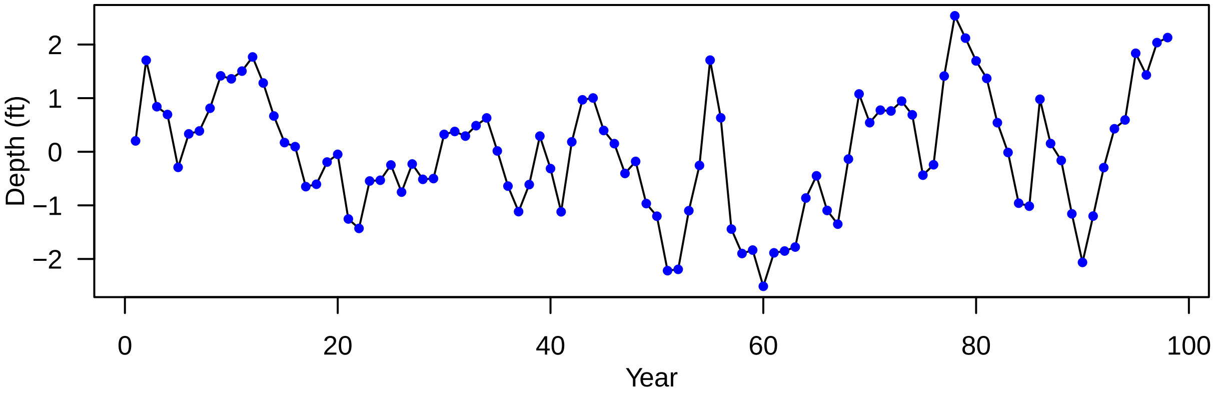
Contents

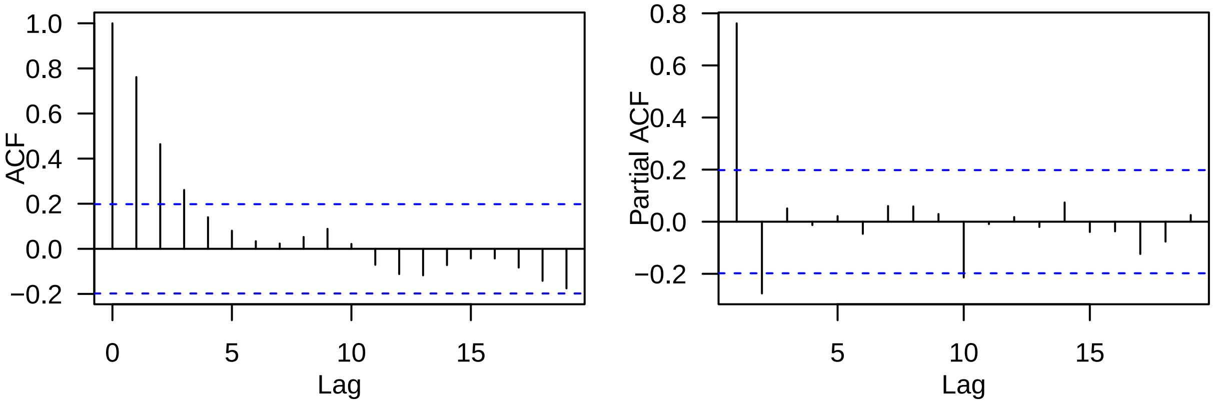
MLE

Lake Huron Example

```r
data("LakeHuron")
yr <- time(LakeHuron)
lm <- lm(LakeHuron ~ yr)

par(las = 1, mgp = c(2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
plot(lm$residuals, ylab = "Depth (ft)", xlab = "Year", type = "l")
points(lm$residuals, cex = 0.8, col = "blue", pch = 16)
acf(lm$residuals)
pacf(lm$residuals)
```





CSS, MLE, and CSS-MLE

```
(CSS <- arima(lm$residuals, order = c(2, 0, 0), include.mean = F,
              method = "CSS"))
```

```
##
## Call:
## arima(x = lm$residuals, order = c(2, 0, 0), include.mean = F, method = "CSS")
##
## Coefficients:
##          ar1      ar2
##       1.0020  -0.2834
## s.e.  0.0947   0.0964
##
## sigma^2 estimated as 0.4436:  part log likelihood = -99.23
```

```
(MLE <- arima(lm$residuals, order = c(2, 0, 0), include.mean = F,
              method = "ML"))
```

```
##
## Call:
## arima(x = lm$residuals, order = c(2, 0, 0), include.mean = F, method = "ML")
##
## Coefficients:
```

```
##          ar1      ar2
##       1.0050  -0.2925
## s.e.  0.0976   0.1002
##
## sigma^2 estimated as 0.4572:  log likelihood = -101.26,  aic = 208.51
```

```
(CSS_ML <- arima(lm$residuals, order = c(2, 0, 0), include.mean = F))
```

```
##
## Call:
## arima(x = lm$residuals, order = c(2, 0, 0), include.mean = F)
##
## Coefficients:
##          ar1      ar2
##       1.0050  -0.2925
## s.e.  0.0976   0.1002
##
## sigma^2 estimated as 0.4572:  log likelihood = -101.26,  aic = 208.51
```

Model selection

```
orders <- list(
  c(1, 0, 0),   # ARMA(1,0)
  c(1, 0, 1),   # ARMA(1,1)
  c(2, 0, 0),   # ARMA(2,0)
  c(2, 0, 1)    # ARMA(2,1)
  )
models <- c("ARMA(1,0)", "ARMA(1,1)", "ARMA(2,0)", "ARMA(2,1)")

fit <- lapply(orders, function(z) arima(LakeHuron, order = z, xreg = yr))
names(fit) <- models

lapply(fit, AIC)
```

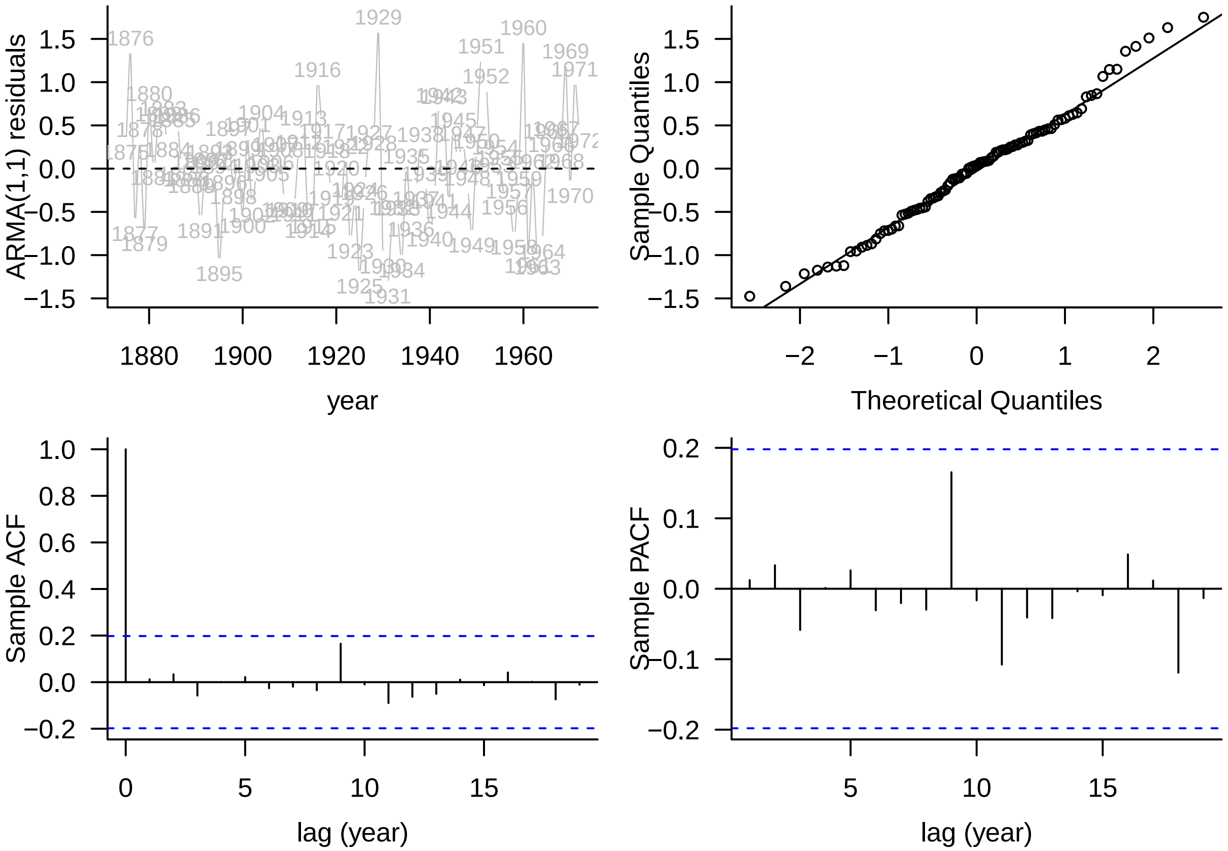
```
## $‘ARMA(1,0)‘
## [1] 218.4501
##
## $‘ARMA(1,1)‘
## [1] 212.3954
##
## $‘ARMA(2,0)‘
## [1] 212.3965
##
## $‘ARMA(2,1)‘
## [1] 214.0638
```

```
library(MuMIn)
lapply(fit, AICc)
```

```
## $‘ARMA(1,0)‘
## [1] 218.8803
##
## $`ARMA(1,1)`
## [1] 213.0476
##
## $`ARMA(2,0)`
## [1] 213.0487
##
## $`ARMA(2,1)`
## [1] 214.9868
```

Model Diagnostics

```r
## extract the residuals
resids <- resid(fit[[2]])
## time series plot of the residuals
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.4, 1, 0),
    mfrow = c(2, 2))
plot(yr, resids, type = "l", xlab = "year",
     ylab = "ARMA(1,1) residuals", lwd = 0.6, col = "gray")
abline(h = 0, lty = 2)
## Normal Q-Q plot for the residuals
qqnorm(resids, main = "", cex = 0.75); qqline(resids)
## Sample ACF and PACF of the residuals
acf(resids, ylab = "Sample ACF", xlab = "lag (year)", main = "")
pacf(resids, ylab = "Sample PACF", xlab = "lag (year)")
```



```
## Carry out the Ljung-Box test
Box.test(resids, lag = 10, type = "Ljung-Box", fitdf = 2)
```

```
##
##  Box-Ljung test
##
## data:  resids
## X-squared = 3.7882, df = 8, p-value = 0.8757
```

Monte Carlo Simulation

```
N = 1000
n = 100
phi <- c(0.6, 0.35)
sim1 <- replicate(N, arima.sim(n = n, list(ar = phi)))
```

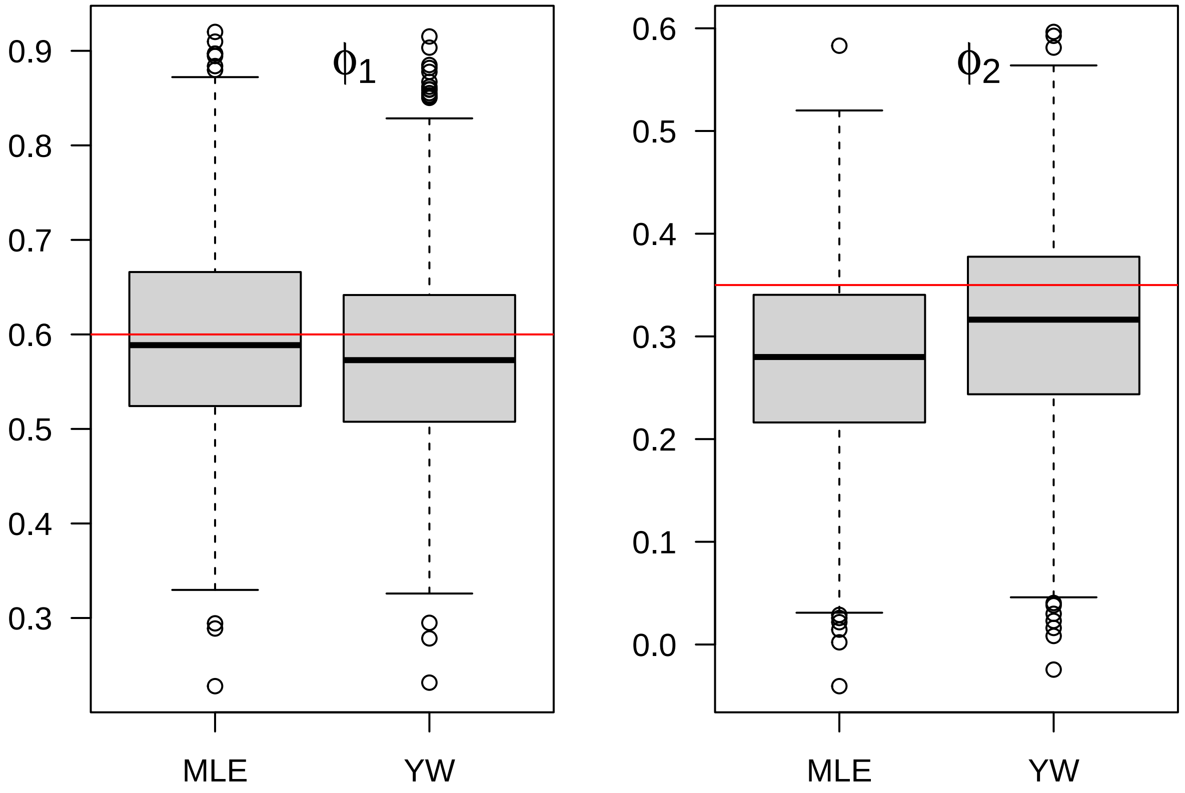
```
yw <- apply(sim1, 2, ar, aic = F, order.max = 2, method = "yule-walker")
mle <- apply(sim1, 2, ar, aic = F, order.max = 2, method = "mle")
```

```
yw_phi <- t(array(unlist(lapply(yw, function(x) x$ar)), dim = c(2, N)))
mle_phi <- t(array(unlist(lapply(mle, function(x) x$ar)), dim = c(2, N)))

par(mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0), mfrow = c(1, 2))
```

```
boxplot(yw_phi[, 1], mle_phi[, 1], xaxt = "n")
abline(h = 0.6, col = "red")
axis(1, at = 1:2, labels = c("MLE", "YW"))
legend("top", legend = expression(phi[1]), bty = "n", cex = 1.5)

boxplot(yw_phi[, 2], mle_phi[, 2], xaxt = "n")
abline(h = 0.35, col = "red")
axis(1, at = 1:2, labels = c("MLE", "YW"))
legend("top", legend = expression(phi[2]), bty = "n", cex = 1.5)
```



```
apply(yw_phi, 2, mean); apply(yw_phi, 2, sd)
```

```
## [1] 0.5938021 0.2769123
```

```
## [1] 0.10781354 0.09581546
```

```
apply(mle_phi, 2, mean); apply(mle_phi, 2, sd)
```

```
## [1] 0.5762923 0.3126342
```

```
## [1] 0.10327422 0.09878245
```

```
sqrt(mean((yw_phi[, 1] - 0.6)^2))
```

```
## [1] 0.1079377
```

```
sqrt(mean((mle_phi[, 1] - 0.6)^2))
```

```
## [1] 0.1059101
```

```
roots <- t(apply(yw_phi, 1, function(x) Mod(polyroot(c(1, -x[1], -x[2])))))
check <- apply(roots, 1, function(x) ifelse(x[1] > 1 && x[2] > 1, 0, 1))
```