Assignment 3

Kaushik Raj V Nadar (208160499)

2024-10-16

Problem 2

```r
# Set seed for reproducibility
set.seed(100)

# Define true parameter values
phi_true <- 0.9
theta_true <- 0.5
sigma2_true <- 1

# List of sample sizes
sample_sizes <- c(50, 200, 500)

# Number of simulations to run
num_simulations <- 1000

# Function to compute performance metrics
compute_metrics <- function(estimates, true_value, se_estimate = NULL) {
  mse <- mean((estimates - true_value)^2)
  mad <- mean(abs(estimates - true_value))

  if (!is.null(se_estimate)) {
    coverage <- mean(estimates - 1.96 * se_estimate < true_value &
                     true_value < estimates + 1.96 * se_estimate)
  } else {
    coverage <- NaN  # Coverage is undefined for sigma^2
  }

  return(c(MSE = mse, MAD = mad, Coverage = coverage))
}

# Function to format results into a table
generate_table <- function(results, parameter_name) {
  kable(results,
        caption = paste("Performance Metrics for", parameter_name),
        col.names = c("MSE", "MAD", "Coverage"),
        row.names = TRUE,
        digits = 4)
}

# Initialize matrices for storing results
phi_results <- theta_results <- sigma2_results <- matrix(NA,
          nrow = length(sample_sizes), ncol = 3)
colnames(phi_results) <-
colnames(theta_results) <-
  colnames(sigma2_results) <- c("MSE", "MAD", "Coverage")
rownames(phi_results) <-
  rownames(theta_results) <-
  rownames(sigma2_results) <- paste0("n = ", sample_sizes)
```

```r
# Run simulations for each sample size
for (i in 1:length(sample_sizes)) {
  n <- sample_sizes[i]

  # Initialize vectors to store estimates and standard errors
  phi_estimates <- theta_estimates <-
    sigma2_estimates <- phi_se_estimates <-
    theta_se_estimates <- numeric(num_simulations)

  for (j in 1:num_simulations) {
    # Simulate ARMA(1,1) process
    x <- arima.sim(n = n, list(ar = phi_true, ma = theta_true))
    fit <- arima(x, order = c(1, 0, 1))

    # Store parameter estimates and standard errors
    phi_estimates[j] <- fit$coef[1]
    theta_estimates[j] <- fit$coef[2]
    sigma2_estimates[j] <- fit$sigma2
    se <- sqrt(diag(fit$var.coef))
    phi_se_estimates[j] <- se[1]
    theta_se_estimates[j] <- se[2]
  }

  # Compute performance metrics for phi, theta, and sigma^2
  phi_results[i, ] <- compute_metrics(phi_estimates, phi_true,
                                      phi_se_estimates)
  theta_results[i, ] <- compute_metrics(theta_estimates,
                            theta_true, theta_se_estimates)
  sigma2_results[i, ] <- compute_metrics(sigma2_estimates,
                                         sigma2_true)
}

# Display results for phi
generate_table(phi_results, "phi")
```

Table 1: Performance Metrics for phi

	MSE	MAD	Coverage
n = 50	0.0168	0.0917	0.909
n = 200	0.0019	0.0323	0.931
n = 500	0.0006	0.0180	0.939

```
# Display results for theta
generate_table(theta_results, "theta")
```

Table 2: Performance Metrics for theta

	MSE	MAD	Coverage
n = 50	0.0210	0.1123	0.915
n = 200	0.0042	0.0516	0.939
n = 500	0.0017	0.0325	0.946

```
# Display results for sigma^2
generate_table(sigma2_results, "sigma^2")
```

Table 3: Performance Metrics for sigma^2

	MSE	MAD	Coverage
n = 50	0.0445	0.1698	NaN
n = 200	0.0105	0.0825	NaN
n = 500	0.0043	0.0525	NaN

Problem 3

```r
# Load the cmort dataset
data(cmort)
# Fit AR(2) model using OLS
reg1 <- ar.ols(cmort, order = 2)

# Print model summary
cat("OLS Estimates:\n")
```

```
OLS Estimates:
```

```r
print(reg1)
```

```


Call:
ar.ols(x = cmort, order.max = 2)

Coefficients:
     1       2
0.4286  0.4418

Intercept: -0.04672 (0.2527)

Order selected 2  sigma^2 estimated as  32.32
```

```r
# Calculate forecasts for 8 week horizon using OLS model
forecasts <- forecast(reg1, h = 8)

# Print forecasts
print(forecasts)
```

```
         Point Forecast    Lo 80    Hi 80    Lo 95     Hi 95
1979.769       87.59986 80.31444 94.88529 76.45777  98.74196
1979.788       86.76349 78.83713 94.68985 74.64117  98.88581
1979.808       87.33714 78.19426 96.48002 73.35431 101.31997
1979.827       87.21350 77.48222 96.94478 72.33079 102.09621
1979.846       87.41394 77.09205 97.73583 71.62798 103.19990
1979.865       87.44522 76.71082 98.17963 71.02837 103.86208
1979.885       87.54719 76.45775 98.63662 70.58736 104.50701
1979.904       87.60471 76.23805 98.97136 70.22091 104.98850
```

```r
# Extract point forecasts and prediction intervals
point_forecasts <- forecasts$mean
lower_pi <- forecasts$lower[, 2]  # 95% lower prediction interval
upper_pi <- forecasts$upper[, 2]  # 95% upper prediction interval

# Print forecasts and prediction intervals
cat("\nForecasts and 95% Prediction Intervals:\n")
```

```
Forecasts and 95% Prediction Intervals:
```

```r
for (i in 1:8) {
  cat(sprintf("Week %d: %.4f (%.4f, %.4f)\n", i,
      point_forecasts[i], lower_pi[i], upper_pi[i]))
}
```

```
Week 1: 87.5999 (76.4578, 98.7420)
Week 2: 86.7635 (74.6412, 98.8858)
Week 3: 87.3371 (73.3543, 101.3200)
Week 4: 87.2135 (72.3308, 102.0962)
Week 5: 87.4139 (71.6280, 103.1999)
Week 6: 87.4452 (71.0284, 103.8621)
Week 7: 87.5472 (70.5874, 104.5070)
Week 8: 87.6047 (70.2209, 104.9885)
```

```r
# Fit AR(2) model using Yule-Walker
reg2 <- ar.yw(cmort, order = 2)

cat("\nYule-Walker Estimates:\n")
```

```
Yule-Walker Estimates:
```

```r
print(reg2)
```

```
Call:
ar.yw.default(x = cmort, order.max = 2)

Coefficients:
     1       2
0.4339  0.4376

Order selected 2  sigma^2 estimated as  32.84
```

```r
# Compare estimates and standard errors
# Create a data frame for the comparison table
comparison_df <- data.frame(
  Parameter = c("AR1", "AR2"),
  OLS_Estimate = c(reg1$ar[1], reg1$ar[2]),
  OLS_SE = c(reg1$asy.se.coef$ar[1], reg1$asy.se.coef$ar[2]),
  YW_Estimate = c(reg2$ar[1], reg2$ar[2]),
  YW_SE = sqrt(diag(reg2$asy.var.coef))
)

# Display the comparison table
kable(comparison_df,
      col.names = c("Parameter",
      "OLS Estimate", "OLS SE", "Yule-Walker Estimate",
    "Yule-Walker SE"),
      caption = "Comparison of Estimates and Standard Errors",
      digits = 4)
```

Table 4: Comparison of Estimates and Standard Errors

Parameter	OLS Estimate	OLS SE	Yule-Walker Estimate	Yule-Walker SE
AR1	0.4286	0.0398	0.4339	0.04
AR2	0.4418	0.0398	0.4376	0.04

```r
# Display variance estimates
cat("Variance estimate (OLS):", sprintf("%.2f", reg1$var.pred), "\n")
```

```
Variance estimate (OLS): 32.32
```

```r
cat("Variance estimate (Yule-Walker):", sprintf("%.2f", reg2$var.pred))
```

```
Variance estimate (Yule-Walker): 32.84
```

The results are consistent using 2 methods