# Assignment 4

Yuki Joyama (yj2803)

## Problem 1

## Problem 2

```r
# import file
data <- read.table("./data.txt", header = TRUE, sep = " ", stringsAsFactors = FALSE)
```

### PC algorithm

```r
# Function to calculate log likelihood
compute_log_likelihood <- function(data, adj_matrix) {
  sample_cov <- cov(data)  # Sample covariance matrix
  adj_cov <- sample_cov * adj_matrix  # Adjust covariance for graph

  # Regularization for numerical stability
  p <- ncol(data)
  regularization <- diag(10, p)
  adj_cov <- adj_cov + regularization

  # Log-likelihood calculation
  tryCatch({
    log_det <- log(det(adj_cov))  # Log determinant
    inv_cov <- solve(adj_cov)     # Inverse covariance
    log_likelihood <- -0.5 * nrow(data) * (log_det + sum(diag(inv_cov %*% sample_cov)))
    return(log_likelihood)
  }, error = function(e) {
    return(NA)  # Return NA if numerical issues occur
  })
}

# Function to calculate BIC
compute_bic <- function(pc_fit, data) {
  adj_matrix <- as(pc_fit@graph, "matrix")  # Adjacency matrix
```

1

```r
  num_params <- sum(adj_matrix)  # Number of edges (parameters)
  log_lik <- compute_log_likelihood(data, adj_matrix)

  if (is.na(log_lik)) return(Inf)  # Return infinite BIC for invalid graphs

  n <- nrow(data)  # Sample size
  p <- ncol(data)  # Number of variables
  bic <- -2 * log_lik + log(n) * (num_params + p)  # BIC formula
  return(bic)
}

# Generate a sequence of alpha values
alphas <- seq(0.001, 0.1, by = 0.001)

# Fit PC algorithm and compute BIC for each alpha
bic_values <- sapply(alphas, function(alpha) {
  pc_fit <- pc(
    suffStat = list(C = cor(data), n = nrow(data)),
    indepTest = gaussCItest,
    alpha = alpha,
    labels = colnames(data),
    verbose = FALSE
  )
  compute_bic(pc_fit, data)
})

# Find optimal alpha
optimal_alpha <- alphas[which.min(bic_values)]
cat("Optimal alpha based on BIC:", optimal_alpha, "\n")
```
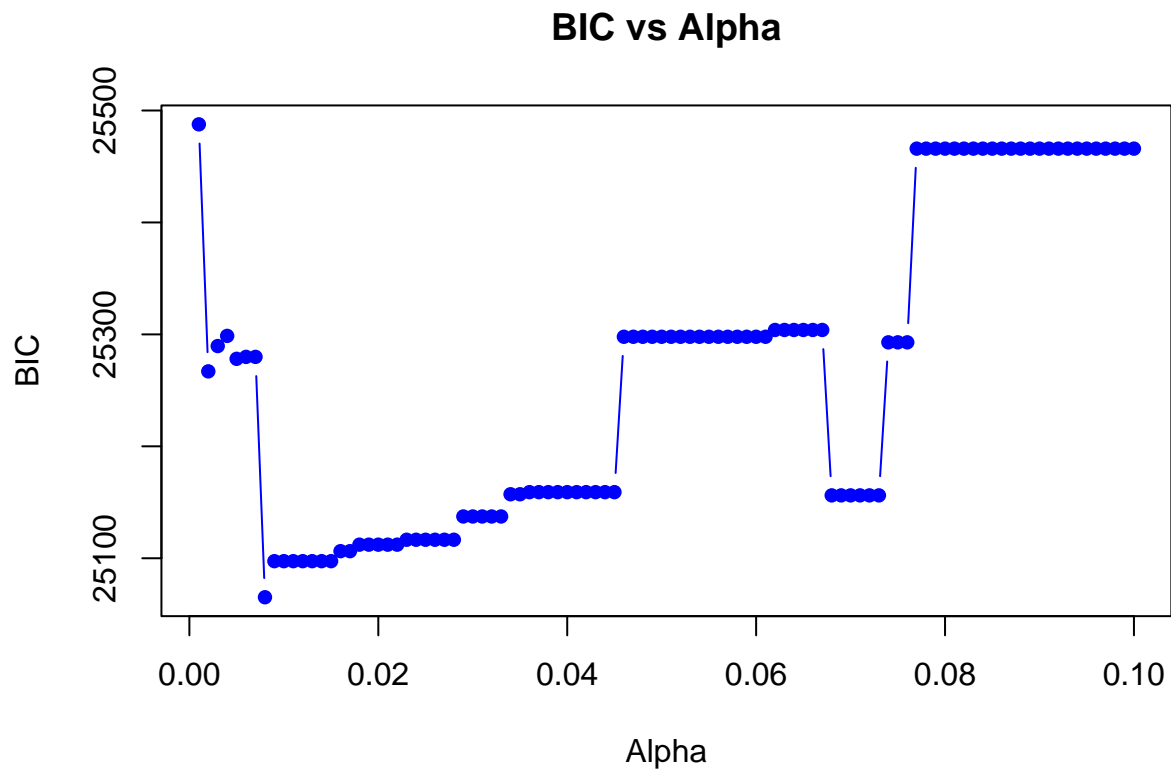
```
## Optimal alpha based on BIC: 0.008
```

```r
# Plot BIC vs Alpha
plot(
  alphas, bic_values, type = "b", pch = 16, col = "blue",
  xlab = "Alpha", ylab = "BIC",
  main = "BIC vs Alpha"
)
```
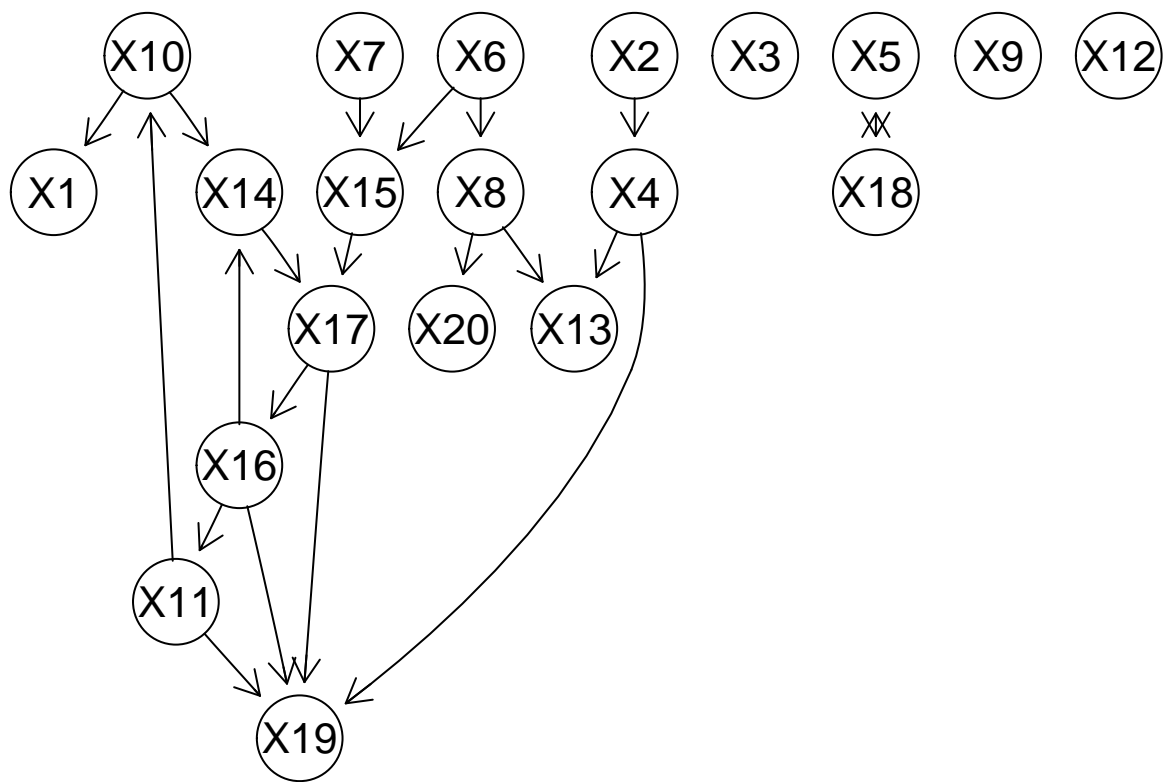
## BIC vs Alpha



```r
# Run pc algorithm with alpha = 0.001
pc.fit.001 <- pc(suffStat = list(C = cor(data), n = nrow(data)),
          indepTest = gaussCItest,
          alpha=0.001, labels = colnames(data), verbose = TRUE)


# alpha = 0.008
pc.fit.008 <- pc(suffStat = list(C = cor(data), n = nrow(data)),
          indepTest = gaussCItest,
          alpha=0.008, labels = colnames(data), verbose = TRUE)


# alpha = 0.1
pc.fit.1 <- pc(suffStat = list(C = cor(data), n = nrow(data)),
          indepTest = gaussCItest,
          alpha=0.1, labels = colnames(data), verbose = TRUE)


# Plot CPDAG
plot(pc.fit.001@graph)
```
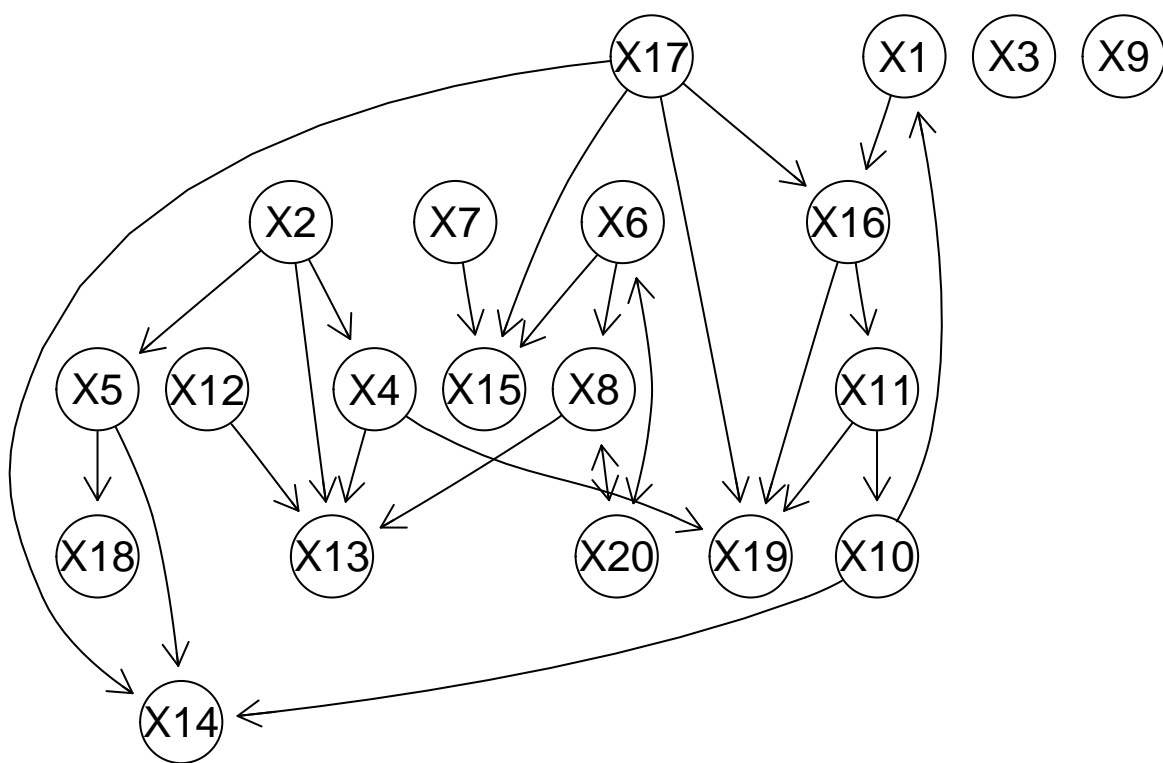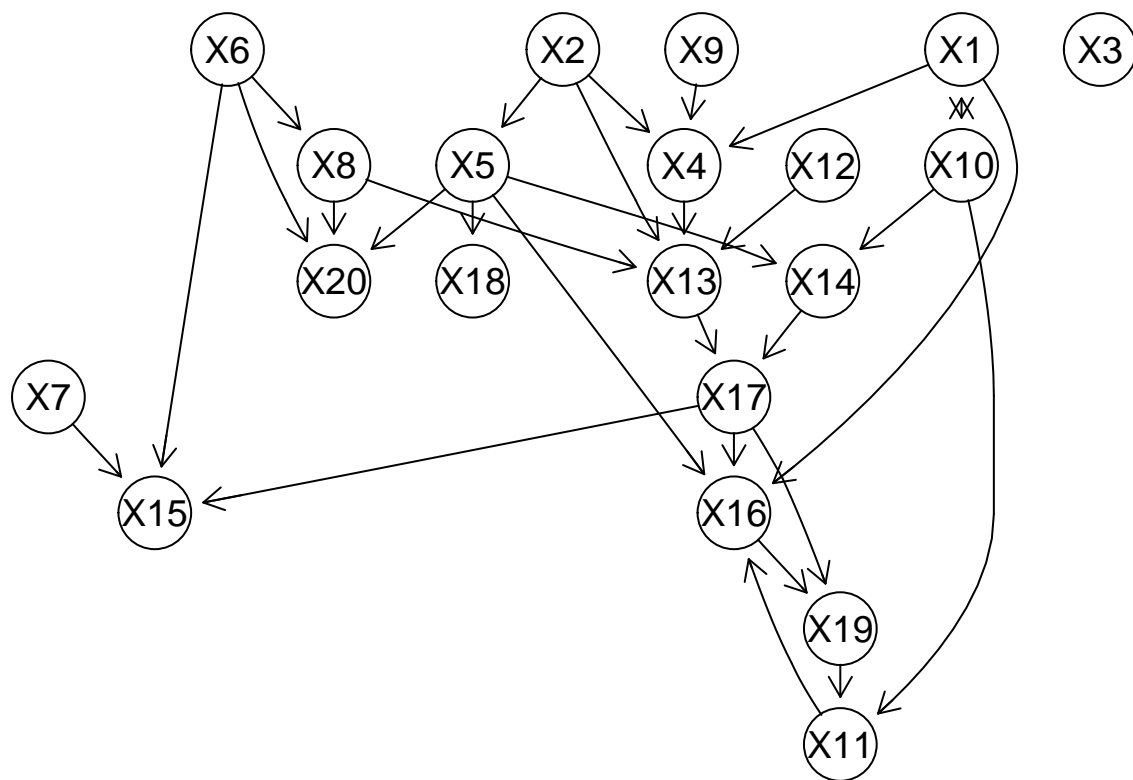
```
plot(pc.fit.008@graph)
```

```
plot(pc.fit.1@graph)
```

```
# Check the number of edges
pc.fit.001
```

```
## Object of class 'pcAlgo', from Call:
## pc(suffStat = list(C = cor(data), n = nrow(data)), indepTest = gaussCItest,
##     alpha = 0.001, labels = colnames(data), verbose = TRUE)
## Number of undirected edges:  1
## Number of directed edges:    19
## Total number of edges:       20
```

```
pc.fit.008
```

```
## Object of class 'pcAlgo', from Call:
## pc(suffStat = list(C = cor(data), n = nrow(data)), indepTest = gaussCItest,
##     alpha = 0.008, labels = colnames(data), verbose = TRUE)
## Number of undirected edges:  2
## Number of directed edges:    23
## Total number of edges:       25
```

```
pc.fit.1
```

```
## Object of class 'pcAlgo', from Call:
## pc(suffStat = list(C = cor(data), n = nrow(data)), indepTest = gaussCItest,
##       alpha = 0.1, labels = colnames(data), verbose = TRUE)
## Number of undirected edges:   1
## Number of directed edges:     28
## Total number of edges:        29
```

I used the PC algorithm to estimate CPDAGs for a range of $\alpha$ values (0.001 to 0.1) to identify the optimal $\alpha$ that minimizes the Bayesian Information Criterion (BIC). For each $\alpha$, the graph structure $G(\alpha)$ was used to compute the log-likelihood $-2\ell(\hat{\Sigma}_{G'}, \hat{\mu})$, where $\ell(\cdot)$ is the log-likelihood of a Gaussian model based on the graph-constrained covariance matrix. The BIC for each graph was computed as $-2\ell + \log(n)(k+p)$, with $k$ representing the number of edges and $p$ the number of variables.

The optimal $\alpha = 0.008$ was selected as the value minimizing BIC. This model is considered to have an optimal balance between complexity and fit.

Estimated CPDAGs were plotted for specific $\alpha$ values (0.001, 0.008, 0.1), and the number of edges was examined for each plot. We see that the smaller $\alpha$ value led to sparser graph with fewer edges, and the larger $\alpha$ value led to denser graphs with more edges.