Project 9 Model Selection

Team K - Minghang Li

May 03, 2023

```
set.seed(2023)
```

Problem 23: d-separation

(i) Write down all the variables that are d-sepparated from A given C, D

Just G (because given $A \not\perp B|D$, hence B is not d-separated from A and consequently B's descendents F and E are also not d-separated).

(ii) Indicate whether each statement is true or false and expalin your choice.

We have the Theorem (Verma & Pearl): A is d-separated from B by C *if and only if* the join distribution over all variables satisfied $A \perp B|C$.

1. $B \perp C|D$: **Wrong**.

   C and B are not d-separated because in the path $B - D - A - C$ the arrows meet head-to-head at D.

2. $G \perp E|D$: **Wrong**.

   G and E are not d-separated because in the path $E - F - B - D - A - C - G$ the arrows meet head-to-head at D.

3. $C \perp F|A$: **Correct**.

   C and F are d-separated because the arrows meet tail-to-tail at A.

4. $C \perp E|MB(C)$: **Wrong**.

   Given $MB(C) = A, D, G, C$ and E are not d-separated because in the path $E - F - B - D - A - C$ the arrows meet head-to-head at D and $D \in MB(C)$.

Load data for Problem 24 - 27.

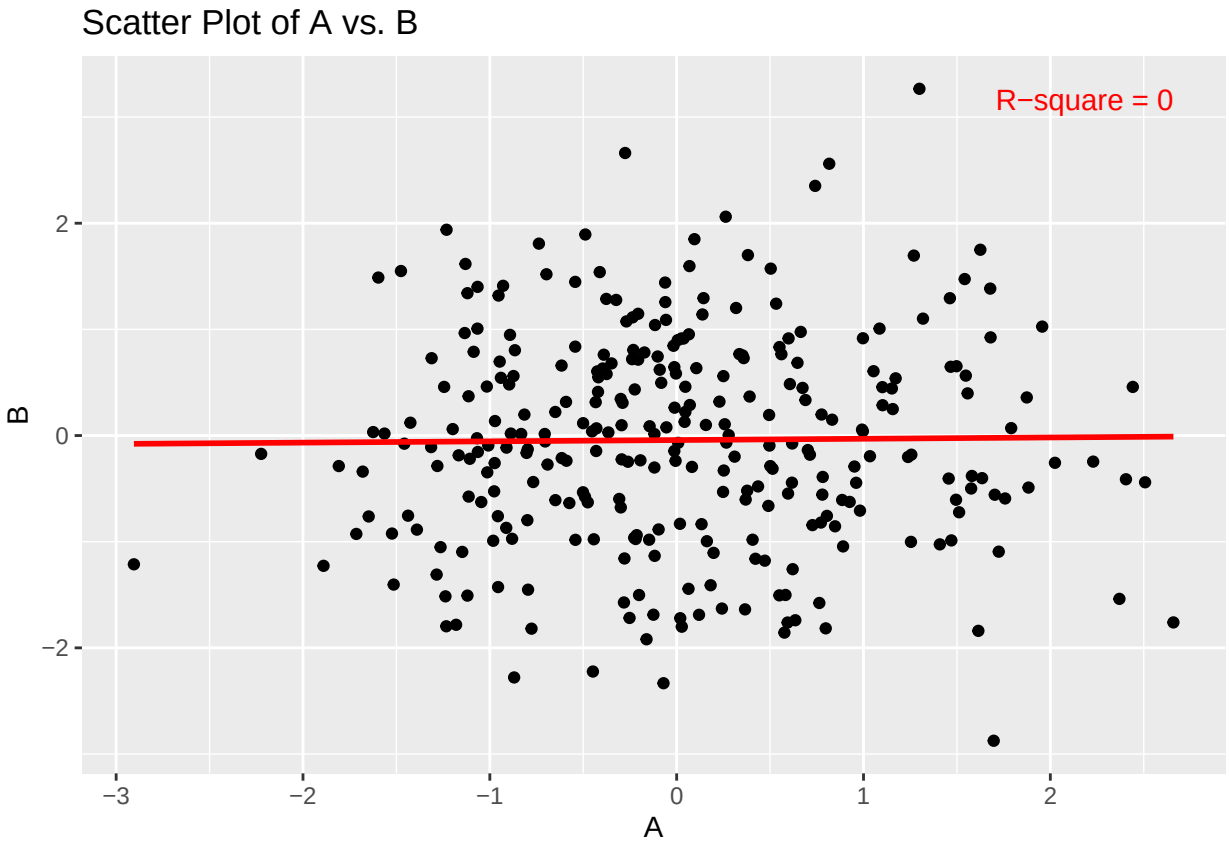
```
mvn.dag <- readRDS("MVN_DAG.rds")
head(mvn.dag)
```

```
##              A           B          C           D           E          F
## 1 -0.294720447  0.09739665 -0.6693398 -1.83015111  0.41091561  0.8663284
## 2 -0.005767173 -0.23838695  1.3541249 -0.63615467 -0.62030667  0.5347883
## 3  2.404653389 -0.41182796 -0.6750873  2.40311737  0.25173361 -1.3942278
## 4  0.763593461 -1.57721805 -2.9092631 -0.52703829 -0.81604685  1.4960640
## 5 -0.799009249 -0.79727610 -2.5411415  0.18356375  0.07764537 -1.2424275
## 6 -1.147657009 -1.09623678 -0.8384432 -0.03142255 -0.23194797 -1.0266878
```

Problem 24: Testing for marginal correlation

```
ggplot(mvn.dag, aes(x = A, y = B)) +
  geom_point() +
  xlab("A") +
  ylab("B") +
  ggtitle("Scatter Plot of A vs. B") +
 # Add linear regression line and R-squared value to better show that there's no correlation
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  annotate("text", x = max(mvn.dag$A), y = max(mvn.dag$B),
           label = paste("R-square =", round(summary(lm(B ~ A, data = mvn.dag))$r.squared, 2)),
           hjust = 1, vjust = 1, color = "red")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



The (very scattered) plot suggests that A and B are marginally independent (no correlation, $R^2 = 0$). And it confirms with Figure 2 because it also suggests that A and B are marginally independent.

```
cor.test(mvn.dag$A, mvn.dag$B)
```

```
##
##  Pearson's product-moment correlation
##
## data:  mvn.dag$A and mvn.dag$B
## t = 0.20194, df = 298, p-value = 0.8401
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.1016784  0.1247727
## sample estimates:
##        cor
## 0.01169715
```

The result of `cor.test()` on A and B also confirms that there's no correlation between A and B (the p-value is 0.84, so we cannot reject the null hypothesis that there's no correlation between A and B).

Problem 25: Testing for partial correlation

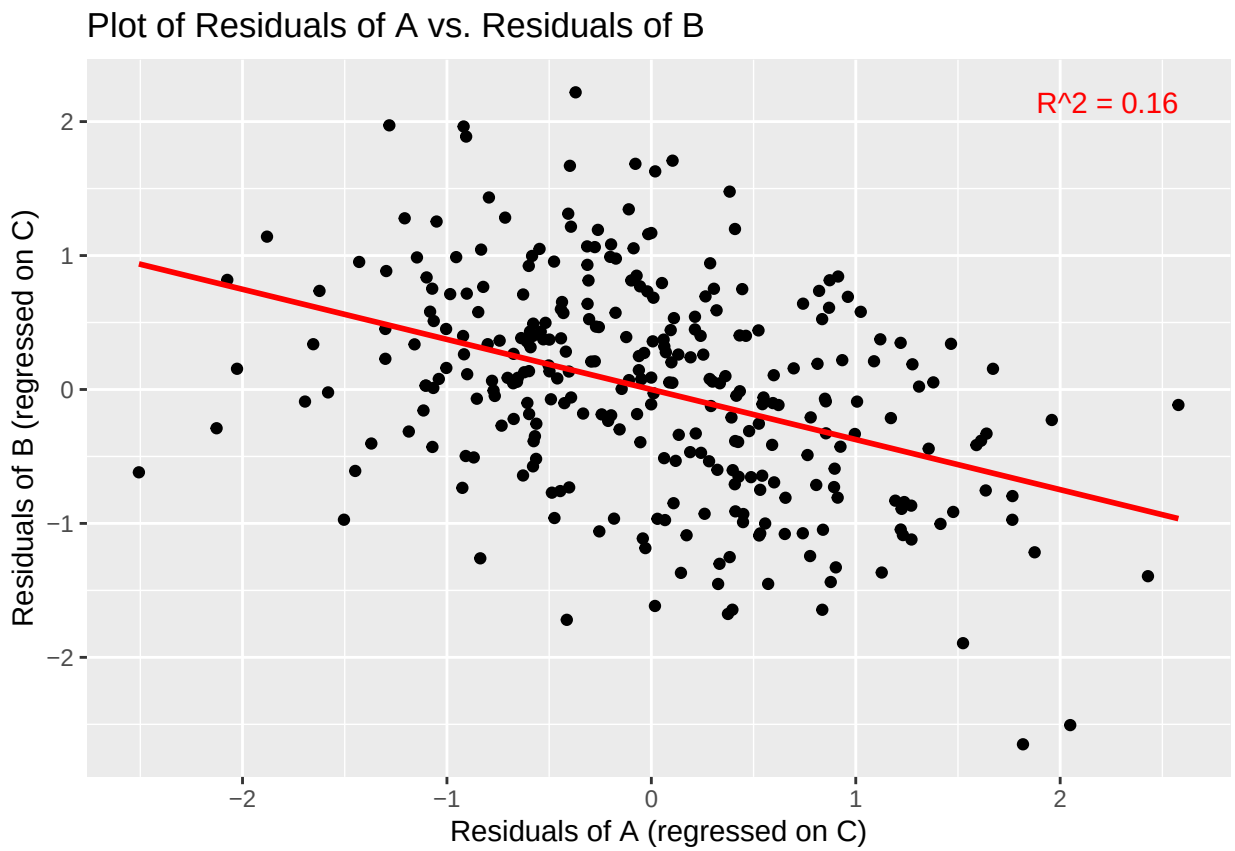
```
# Linear regression of A on C
model_A <- lm(A ~ C, data = mvn.dag)
resid_A <- residuals(model_A)

# Linear regression of B on C
model_B <- lm(B ~ C, data = mvn.dag)
resid_B <- residuals(model_B)

# Linear regression of residuals of A vs. residuals of B
model_resid <- lm(resid_B ~ resid_A)
resid_AB <- residuals(model_resid)

# Plot residuals of A vs. residuals of B, with linear regression and R-squared value
ggplot(data = data.frame(resid_A = resid_A, resid_B = resid_B, resid_AB = resid_AB), aes(x = resid_A, y = re
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  xlab("Residuals of A (regressed on C)") +
  ylab("Residuals of B (regressed on C)") +
  ggtitle("Plot of Residuals of A vs. Residuals of B") +
  annotate("text", x = max(resid_A), y = max(resid_B),
           label = paste("R^2 =", round(summary(model_resid)$r.squared, 2)),
           hjust = 1, vjust = 1, color = "red")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



There seems to be a correlation between $A|C$ and $B|C$.

```
cor.test(resid_A, resid_B)
```

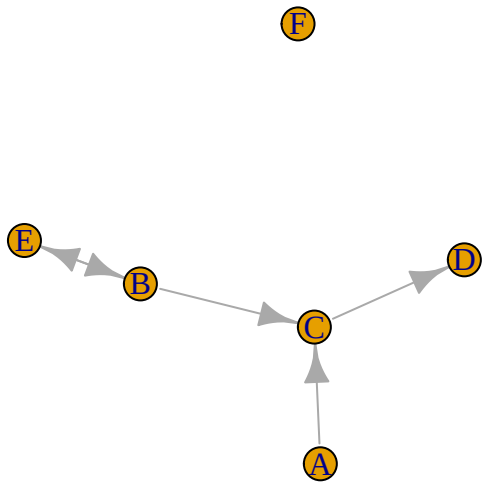
```
##
##  Pearson's product-moment correlation
##
## data:  resid_A and resid_B
## t = -7.5173, df = 298, p-value = 6.6e-13
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.4903245 -0.2995546
## sample estimates:
##       cor
## -0.3992521
```

With a p-value of `6.6e-13`, we can reject the null hypothesis that $A|C$ and $B|C$ are not correlated, hence A and B are not conditionally independent given C. This also confirms with the uderlying DAG in Figure 2 because A and B explain away each other given C.

Problem 26: Running the PC algorithm

```
res <- pc(suffStat = list(C=cor(mvn.dag), n=nrow(mvn.dag)),
          indepTest = gaussCItest,
          alpha = 0.01,
          labels = colnames(mvn.dag))
```

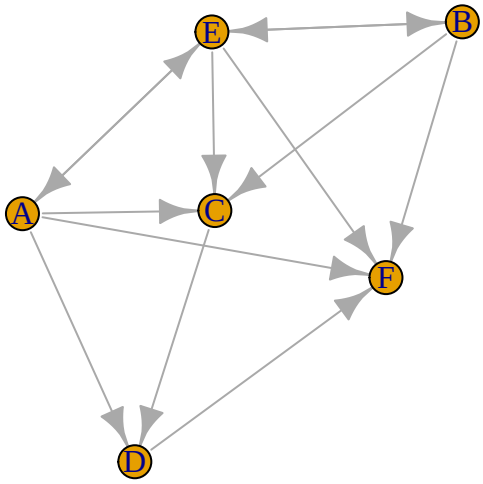
```
iplotPC(res)
```



The PC algorithm has successfully learned the correct graph. The colliders are also successfully identified.

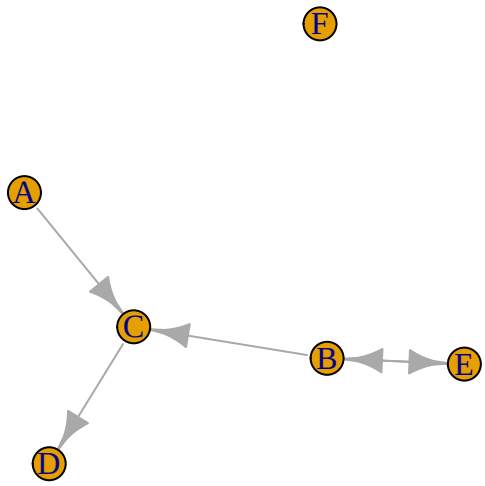
If the significance level α increase, more edges can get accepted, making the graph become more dense. (The code block below showed a very unrealistic setting)

```
not_reasonable_res <- pc(suffStat = list(C=cor(mvn.dag), n=nrow(mvn.dag)),
          indepTest = gaussCItest,
          alpha = 0.9,
          labels = colnames(mvn.dag))
iplotPC(not_reasonable_res)
```



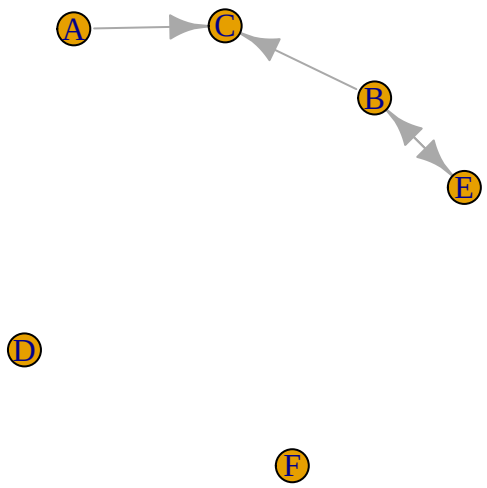
Problem 27: Running the partition MCMC algorithm

```
score <- scoreparameters(scoretype = "bge", mvn.dag)
maxBN <- learnBN(score, algorithm = "orderIter")
plot.igraph(graph.adjacency(maxBN$CPDAG))
```

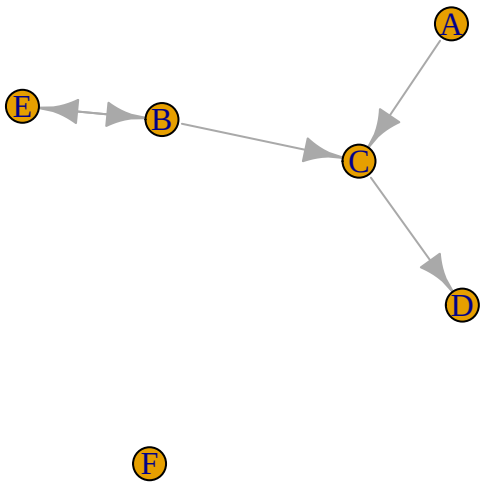


Testing different values of hyperparameter α_μ.

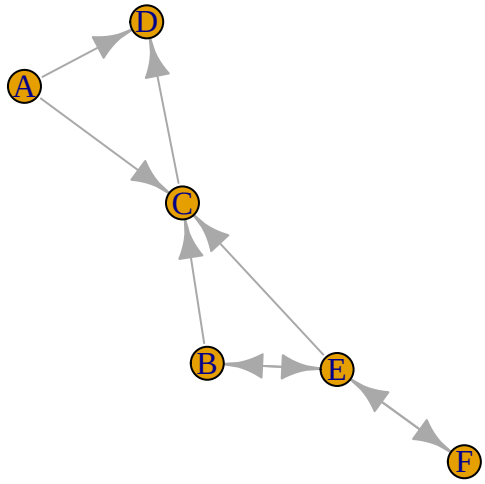
```
score <- scoreparameters(scoretype = "bge", mvn.dag, bgepar = list(am=1e-3, aw=NULL))
maxBN <- learnBN(score, algorithm = "orderIter")
plot.igraph(graph.adjacency(maxBN$CPDAG))
```



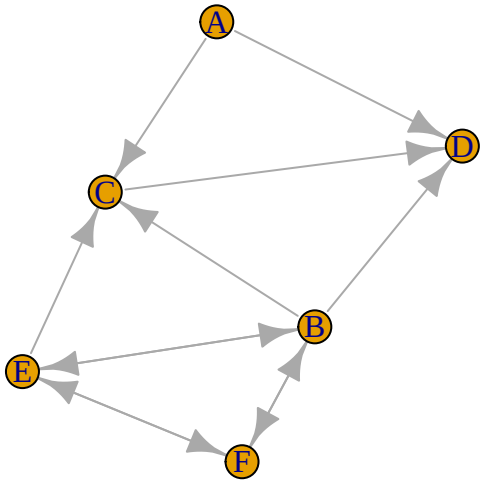

```
score <- scoreparameters(scoretype = "bge", mvn.dag, bgepar = list(am=1e-2, aw=NULL))
maxBN <- learnBN(score, algorithm = "orderIter")
plot.igraph(graph.adjacency(maxBN$CPDAG))
```




```
score <- scoreparameters(scoretype = "bge", mvn.dag, bgepar = list(am=1e2, aw=NULL))
maxBN <- learnBN(score, algorithm = "orderIter")
plot.igraph(graph.adjacency(maxBN$CPDAG))
```




```
score <- scoreparameters(scoretype = "bge", mvn.dag, bgepar = list(am=1e3, aw=NULL))
maxBN <- learnBN(score, algorithm = "orderIter")
plot.igraph(graph.adjacency(maxBN$CPDAG))
```

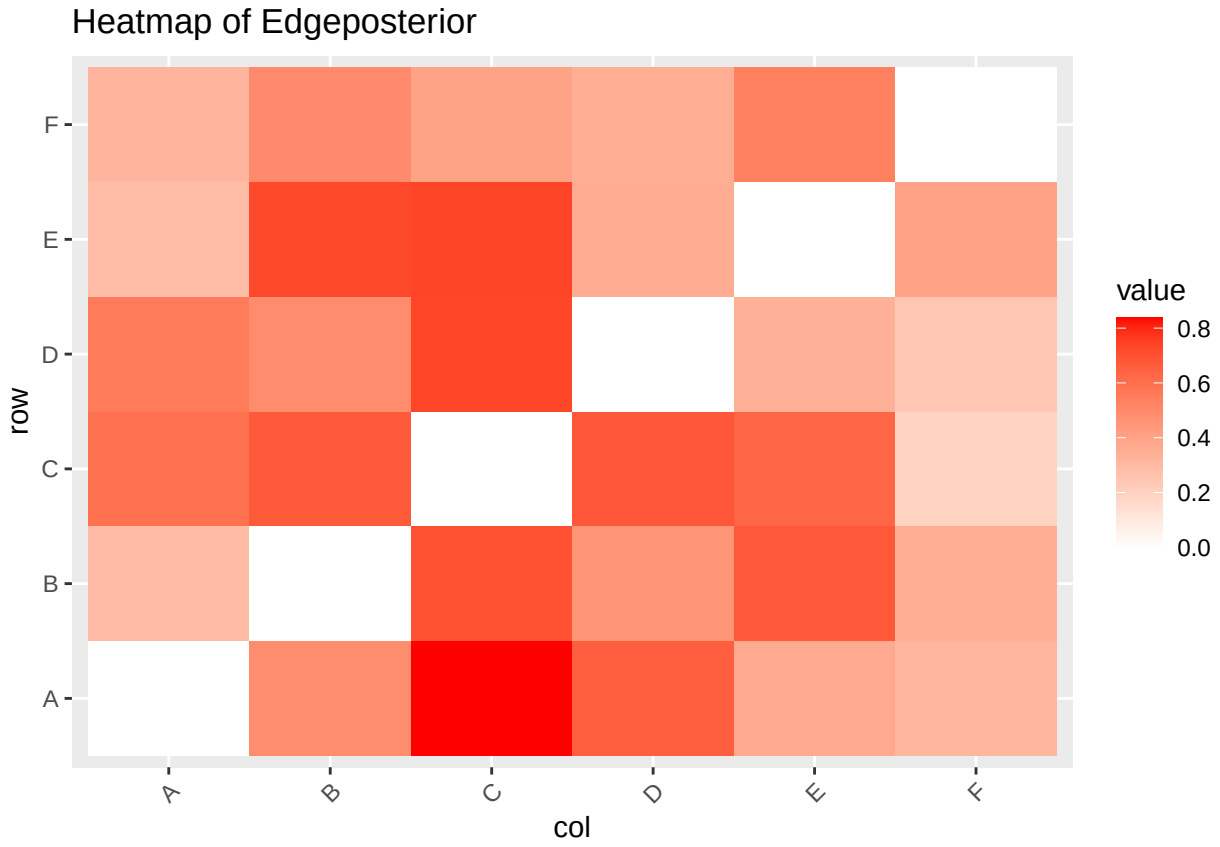


We see that as α_μ increases, the graph become denser and denser. The colliders are preserved for every learned graph though.

```
partition_sample <- sampleBN(score, algorithm = "partition", startspace = maxBN$endspace)
edgeposterior <- edgep(partition_sample, pdag = TRUE)
```

```
# Reshape data to long format
edgeposterior_df <- melt(as.matrix(edgeposterior), varnames = c("row", "col"))

# Create heatmap of edgeposterior
ggplot(edgeposterior_df, aes(x = col, y = row, fill = value)) +
  geom_tile() +
  scale_fill_gradient(low = "white", high = "red") +
  ggtitle("Heatmap of Edgeposterior") +
  scale_x_discrete(limits = c("A", "B", "C", "D", "E", "F")) +
  scale_y_discrete(limits = c("A", "B", "C", "D", "E", "F")) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```


Heatmap of Edgeposterior
row
F
E
D
C
B
A
A
B
C
D
E
F
col
value
0.8
0.6
0.4
0.2
0.0