Final Project

Gavin Gunawardena

8/6/2021

Introduction

For this project, I will be conducting an analysis on 6 chronologically ordered agriculture datasets on harvesting and seeding. To do this, first I will convert the data into dataframes and segregate the gps coordinates of the data into intervals of 50 in order to simplify it into x/y coordinates. Next I will aggregate the data, merge it, normalize it, and then analyze it via graphs in order to solve the question of: "If a specific region of the corn field was low yielding in 2018, was it because it was seeded at a lower rate, or because it is just a poor part of the field, as indicated by yield 2017?". In other words, is the crop yield in 2017 or the seeding rate in 2018 more indicative of better crop yield in 2018?

Data Imports:

```
# 2017 Soybeans Harvest
SB_HV_2017 = "A 2017 Soybeans Harvest.csv"
SB_HV_2017.dat <- read.csv(SB_HV_2017,header=TRUE)
# 2018 Corn Seeding
C_SD_2018 = "A 2018 Corn Seeding.csv"
C_SD_2018.dat <- read.csv(C_SD_2018,header=TRUE)
# 2018 Corn Harvest
C_HV_2018 = "A 2018 Corn Harvest.csv"
C_HV_2018.dat <- read.csv(C_HV_2018,header=TRUE)
# 2019 Soybeans Harvest
SB_HV_2019 = "A 2019 Soybeans Harvest.csv"
SB_HV_2019.dat <- read.csv(SB_HV_2019,header=TRUE)
# 2020 Corn Seeding
C_SD_2020 = "A 2020 Corn Seeding.csv"
C_SD_2020.dat <- read.csv(C_SD_2020,header=TRUE)
# 2020 Corn Harvest
C_HV_2020 = "A 2020 Corn Harvest.csv"
C_HV_2020.dat <- read.csv(C_HV_2020,header=TRUE)
```

Data

After importing and reviewing the data, it seems to have many dimensions and measures related to agriculture. The data can likely be grouped into 3 distinct groups based on common measures: seeding of corn, harvesting of corn and harvesting of soy beans. Furthermore, the data can likely be aggregated based on longitude and latitude. Thus, I'm planning on aggregating and merging the data via an x/y grid created with the longitude and latitude data, and analyzing the data via the yield and applied rate measures.

Create grids in the new dataframes:

```
# Create a function to individually convert each dataframe into an x/y coordinate grid (interval = 50, 
Grid.Create <- function(x.dat, interval = 50) {
  # Function to add 2 columns to the dataframes based on their longitude and latitude variables. These 
  x.var <- ceiling(x.dat['Latitude']/interval)
  y.var <- ceiling(x.dat['Longitude']/interval)

  x.var <- replace(x.var,            # Replace values
                       x.var == 0,
                       1)
    y.var <- replace(y.var,            # Replace values
                       y.var == 0,
                       1)

  x.dat['x.coord'] <- x.var
  x.dat['y.coord'] <- y.var
  return(x.dat)
}


# Running the above function to convert the data into grids
SB_HV_2017.dat <- Grid.Create(SB_HV_2017.dat)
C_SD_2018.dat <- Grid.Create(C_SD_2018.dat)
C_HV_2018.dat <- Grid.Create(C_HV_2018.dat)
SB_HV_2019.dat <- Grid.Create(SB_HV_2019.dat)
C_SD_2020.dat <- Grid.Create(C_SD_2020.dat)
C_HV_2020.dat <- Grid.Create(C_HV_2020.dat)
```

Data Munging to remove uneccessary columns

I've noticed after looking through the data and project requirements that there is a lot of excess data that can be omitted and is not necessary for this project. I've decided to identify and remove these columns from the dataframes in order to focus the analysis on the necessary dimensions and measures for answering the question from the intro that this project is designed to answer. Columns I've decided to keep are those involving location, harvest yield, and seeding rate.

```
# Create lists of columns to keep

# Columns to keep for harvest dataframes
ColumnKeep_1 <- c('Longitude','Latitude','Yield','x.coord','y.coord')
# Columns to keep for seeding dataframes
ColumnKeep_2 <- c('Longitude','Latitude','AppliedRate','x.coord','y.coord')

# Remove the columns
SB_HV_2017_sub.dat <- subset (SB_HV_2017.dat, select = ColumnKeep_1)
C_SD_2018_sub.dat <- subset (C_SD_2018.dat, select = ColumnKeep_2)
C_HV_2018_sub.dat <- subset (C_HV_2018.dat, select = ColumnKeep_1)
SB_HV_2019_sub.dat <- subset (SB_HV_2019.dat, select = ColumnKeep_1)
C_SD_2020_sub.dat <- subset (C_SD_2020.dat, select = ColumnKeep_2)
C_HV_2020_sub.dat <- subset (C_HV_2020.dat, select = ColumnKeep_1)
```

Aggregate the data on grid cells and remove unneccesary data

Here I've aggregated all measures of the datasets by mean and count. I decided to do this on one of the datasets to figure out all of the kinks, and then create a function of what I did so that I could apply the same process to the other datasets. I made sure to make the function flexible enough that it would work for the seeding and harvesting datasets since they have different measures.

```
# Aggregate  dataframes functions since I'll need to do this a few times
Agg.SD.Create <- function(x.dat) {
  # Take in data frame, aggregate for mean and count, producing 2 dataframes, merge on x.coord and y.co
  df_agg_mean <- aggregate(x=x.dat, by=list(x.dat$x.coord, x.dat$y.coord),
FUN = mean )[,-(1:2)]
  df_agg_count <- setNames(aggregate(x=x.dat, by=list(x.dat$x.coord, x.dat$y.coord),
FUN = length)[,1:3], c("x.coord","y.coord","Obs"))
  return(merge(df_agg_mean, df_agg_count, by = c("x.coord","y.coord")))
}

# Seeding of corn
C_SD_2018_agg.dat <- Agg.SD.Create(C_SD_2018_sub.dat)
C_SD_2020_agg.dat <- Agg.SD.Create(C_SD_2020_sub.dat)
# Harvesting of Corn
C_HV_2018_agg.dat <- Agg.SD.Create(C_HV_2018_sub.dat)
C_HV_2020_agg.dat <- Agg.SD.Create(C_HV_2020_sub.dat)
# Harvesting of soybeans
SB_HV_2017_agg.dat <- Agg.SD.Create(SB_HV_2017_sub.dat)
SB_HV_2019_agg.dat <- Agg.SD.Create(SB_HV_2019_sub.dat)
```

Merge the Aggregated Dataframes

Here I actually completed a few steps in order to merge the data and prepare the data for analysis. These steps were:

Remove Longitude/Latitude: I decided to remove longitude/latitude columns from the dataframes to simplify the analysis and since a close estimate within at most 50 longitude and latitude can be ascertained via the x/y coordinates.

Change Column Names: Before merging the dataframes, I've decided to change the names of the columns as to make them more easily distinguishable after the merge.

*Merge: For merging the dataframes, I've decided to merge them all via R's version of the SQL Inner join. This will join all of the data of the dataframes that cover the same cells. This eliminates issues of missing data due an area of land being harvested that was never seeded again, or vice versa. Essentially we're trying to find areas of land that all of our data covers to optimize causal analysis.

Filter: Furthermore, for this analysis, I'll need to remove data where a cell has less than 30 observations. In order to complete this analysis, we need to keep a lower limit of observations, and thus, cells that were seeded or harvested less than 30 times in any of the datasets must be removed.

Reorder: Finally, I reordered the columns to better organize the merged dataset, moving observation counts to the end of the dataset and chronologically ordering the columns.

```
# Remove latitude and longitude from most of the datasets
# Seeding of corn
C_SD_2018_agg_pre_merge.dat <- subset(C_SD_2018_agg.dat, select = -c(Longitude,Latitude))
C_SD_2020_agg_pre_merge.dat <- subset(C_SD_2020_agg.dat, select = -c(Longitude,Latitude))
```

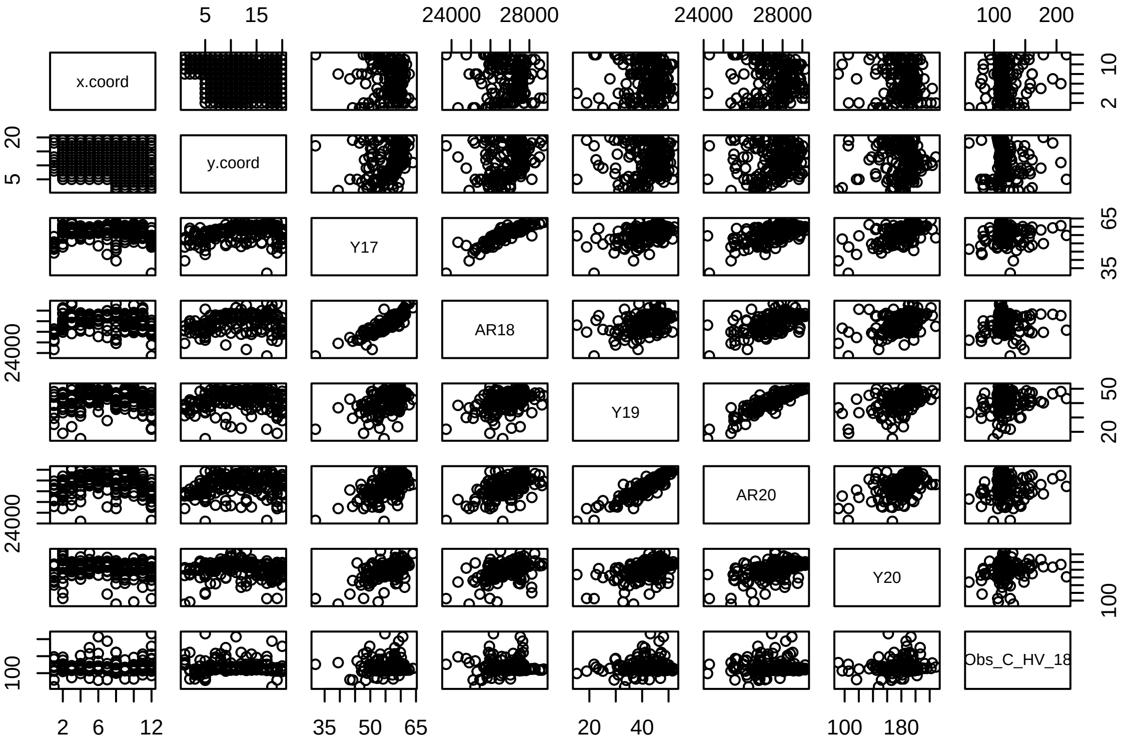
```
# Harvesting of Corn
C_HV_2018_agg_pre_merge.dat <- subset(C_HV_2018_agg.dat, select = -c(Longitude,Latitude))
C_HV_2020_agg_pre_merge.dat <- subset(C_HV_2020_agg.dat, select = -c(Longitude,Latitude))
# Harvesting of soybeans
SB_HV_2017_agg_pre_merge.dat <- subset(SB_HV_2017_agg.dat, select = -c(Longitude,Latitude))
SB_HV_2019_agg_pre_merge.dat <- subset(SB_HV_2019_agg.dat, select = -c(Longitude,Latitude))

# Change names of columns in the dataset
# Seeding of corn
C_SD_2018_agg_pre_merge2.dat <- setNames(C_SD_2018_agg_pre_merge.dat, c('x.coord','y.coord','AR18','Obs_
C_SD_2020_agg_pre_merge2.dat <- setNames(C_SD_2020_agg_pre_merge.dat, c('x.coord','y.coord','AR20','Obs_
# Harvesting of Corn
C_HV_2018_agg_pre_merge2.dat <- setNames(C_HV_2018_agg_pre_merge.dat, c('x.coord','y.coord','Y18','Obs_C
C_HV_2020_agg_pre_merge2.dat <- setNames(C_HV_2020_agg_pre_merge.dat, c('x.coord','y.coord','Y20','Obs_C
# Harvesting of soybeans
SB_HV_2017_agg_pre_merge2.dat <- setNames(SB_HV_2017_agg_pre_merge.dat, c('x.coord','y.coord','Y17','Obs
SB_HV_2019_agg_pre_merge2.dat <- setNames(SB_HV_2019_agg_pre_merge.dat, c('x.coord','y.coord','Y19','Obs



# Merge the dataframes into Combined.dat
Combined_pre_1.dat <- Reduce(function(x,y) merge(x = x, y = y, by = c("x.coord","y.coord"), all.x = FALS
       list(C_SD_2018_agg_pre_merge2.dat, C_SD_2020_agg_pre_merge2.dat, C_HV_2018_agg_pre_merge2.dat, C_



# Remove rows where observations are less than 30 in any of the datasets
Combined_pre_2.dat <- subset(Combined_pre_1.dat, Obs_C_SD_18>=30)
Combined_pre_3.dat <- subset(Combined_pre_2.dat, Obs_C_SD_20>=30)
Combined_pre_4.dat <- subset(Combined_pre_3.dat, Obs_C_HV_18>=30)
Combined_pre_5.dat <- subset(Combined_pre_4.dat, Obs_C_HV_20>=30)
Combined_pre_6.dat <- subset(Combined_pre_5.dat, Obs_SB_HV_17>=30)
Combined_pre_7.dat <- subset(Combined_pre_6.dat, Obs_SB_HV_19>=30)



# Reorder columns
Combined.dat <- subset(Combined_pre_7.dat, select = c(x.coord, y.coord, Y17, AR18, Y18, Y19, AR20, Y20,
```

Visualize the Merged Data Frame

I'm using the y-coordinates of my grid along with the yield and applied rate data to show the relationship among data columns in the merged data. This shows the relationships between each of the measures of the combined.dat dataframe. At a glance, it seems like the closest relationships are between the yield of the 2017 soybean harvest and the applied rate of the 2018 corn seeding; as well as the yield of the 2019 soybean harvest and the 2020 corn seeding.

Key and Timeline: Y17: Yield of the 2017 Soybean Harvest AR18: Applied Rate of the 2018 Corn Seeding Y18: Yield of the 2018 Corn Harvest Y19: Yield of the 2019 Soybean Harvest AR20: Applied rate of the 2020 Corn Seeding Y20: Yield of the 2020 Corn Harvest

```
# Combined.dat
pairs(subset(Combined.dat,select = c(1,2,3,4,6,7,8,10)))
```

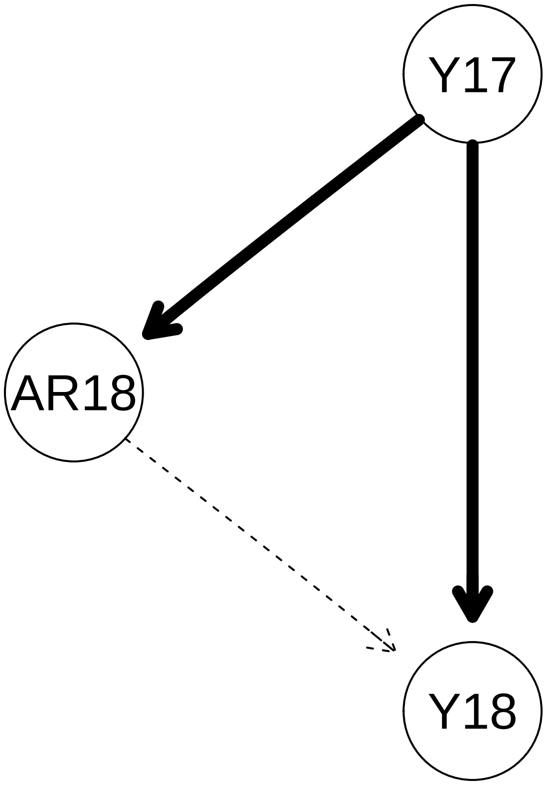


Causal Analysis

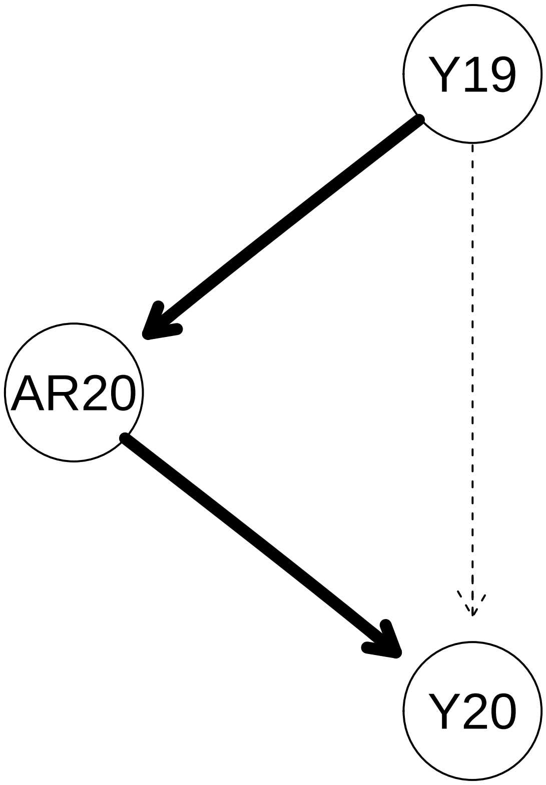
This acyclic graph shows the relationships between the different measures from the datasets. It tries to connect them to show what has a likely effect or relationship on what.

Key and Timeline: Y17: Yield of the 2017 Soybean Harvest AR18: Applied Rate of the 2018 Corn Seeding Y18: Yield of the 2018 Corn Harvest Y19: Yield of the 2019 Soybean Harvest AR20: Applied rate of the 2020 Corn Seeding Y20: Yield of the 2020 Corn Harvest

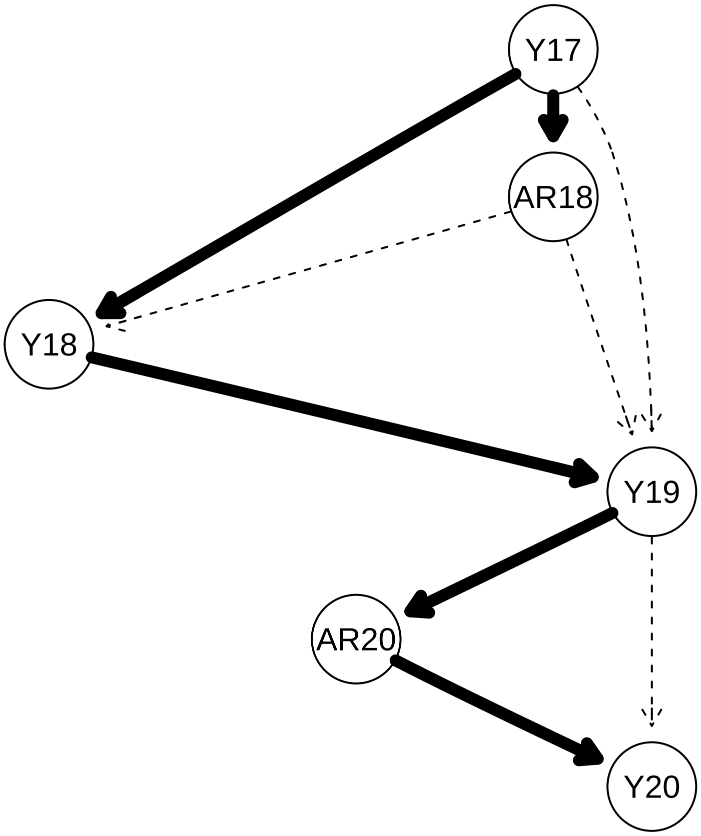
```
#acyclic Graph
# Check for required libraries
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
if (!requireNamespace("Rgraphviz", quietly = TRUE))
    install.packages("Rgraphviz")
if (!requireNamespace("bnlearn", quietly = TRUE))
    install.packages("bnlearn")
library(bnlearn)
# Graph the data
modela.dag <- model2network("[Y17][AR18|Y17][Y18|AR18:Y17]")
fit1 = bn.fit(modela.dag, Combined.dat[,c('Y17','AR18','Y18')])
#fit1
strengtha <- arc.strength(modela.dag, Combined.dat[,c('Y17','AR18','Y18')])
strength.plot(modela.dag, strengtha)
```



```
modelb.dag <- model2network("[Y19][AR20|Y19][Y20|AR20:Y19]")
fit2 = bn.fit(modelb.dag, Combined.dat[,c('Y19','AR20','Y20')])
#fit2
strengthb <- arc.strength(modelb.dag, Combined.dat[,c('Y19','AR20','Y20')])
strength.plot(modelb.dag, strengthb)
```



```
model1.dag <- model2network("[Y17][AR18|Y17][Y18|AR18:Y17][Y19|Y17:AR18:Y18][AR20|Y19][Y20|AR20:Y19]")
fit3 = bn.fit(model1.dag, Combined.dat[,c('Y17','AR18','Y18','Y19','AR20','Y20')])
#fit3
strength1 <- arc.strength(model1.dag, Combined.dat[,c('Y17','AR18','Y18','Y19','AR20','Y20')])
strength.plot(model1.dag, strength1)
```



Normalize the Data

The data will need to be normalized in order to be properly analyzed. When measures are of different scales, they often do not contribute equally to an analysis. Normalizing the data will adjust it so that each measure follows the same scale, improving the analysis by making the measures relate to each other more equally. Furthermore, in order to optimally normalize the data and limit unaccounted for variables, I'll normalize the original datasets as well as aggregate and merge them again to check for changes in the graphs.

After conducting some research, I've decided to normalize the data via Z scores as that seems to be one of the most common methods. It is known for handling outliers well by giving them slightly more weight than the rest of the data, anda from the pairs plot created earlier, it seems like these datasets have a lot of outliers. Regarding the calculation for Z-Score, I'm using the standard calculation by first calculating the mean and standard deviation of the yields and applied rates of each dataset, and then calculating the z score for each individual yield or applied rate by subtracting the value by the mean and dividing this new value by the standard deviation.

source: https://www.statology.org/how-to-normalize-data-in-r/

Z Score Formula Used:

$$Mean = \overline{y}_j = \frac{\sum_{i=1}^{N_i} y_{ij}}{N_i}$$

$$SampleStandardDeviation = s_j^2 = \frac{\sum_{i=1}^{N_i} (y_{ij} - \overline{y}_j)^2}{N_i - 1}$$

$$z - score = z_{ij} = \frac{y_{ij} - \overline{y}_j}{s_j}$$

Skewness and Kurtosis

Along with following the previous steps but with the included step of normalizing the data, before I merged the data, I analyzed it by checking for skewness and kurtosis. These two formulas take in the observation count, mean, and sample standard deviation of the data. The results shown below, indicate a mix of skewed and non-skewed data as well as a high kurtosis. This indicates that not all of the datasets are normalized around their means, and that their tails are heavy, indicating outliers in the data.

Skewness and Kurtosis formulas:

$$Skewness = g_1 = \frac{\sum_{i=1}^{N}(Y_i - \bar{Y})^3/N}{s^3}$$

$$kurtosis = \frac{\sum_{i=1}^{N}(Y_i - \bar{Y})^4/N}{s^4} - 3$$

```
# Create functions to calculate Z Scores, Skewness, and Kurtosis
# Z score formula
ZScore.Solve <- function(x.dat) {
  #mean calculation
 y_hat <- mean(x.dat)
  #Standard deviation calculation
 s_dev <- sd(x.dat)
  #Z-score calculation
 z_score <- (x.dat - y_hat)/sqrt(s_dev)
 return(z_score)
}
#This skewness and kurtosis calculation function uses a more manual calculation of the mean and standar
SkewnessKurtosis <- function(arrayOfValues){
  mean.x <- 0
  stDev.x <- 0
  sum.x <- 0
  skewness.x <- 0
  kurtosis.x <- 0
  runningCalcTotal1.x <- 0 #standard deviation
  runningCalcTotal2.x <- 0 #skewness
  runningCalcTotal3.x <- 0 #kurtosis
  n <- 0
  #Mean calculation
  for (i in 1:length(arrayOfValues)){
    if(!is.na(arrayOfValues[i])){
      sum.x <- sum.x + arrayOfValues[i]
      n <- n+1
    }
  }
  mean.x <- sum.x/n

  #Standard Deviation Calculation
  for (i in 1:length(arrayOfValues)){
    if(!is.na(arrayOfValues[i])){
      runningCalcTotal1.x <- runningCalcTotal1.x + ((arrayOfValues[i]-mean.x) * (arrayOfValues[i]-mean.
    }
  }
  stDev.x <- sqrt(runningCalcTotal1.x/(n-1))
  #Fisher-Pearson  skewness Calculation
  for (i in 1:length(arrayOfValues)){
    if(!is.na(arrayOfValues[i])){
      runningCalcTotal2.x <- (runningCalcTotal2.x + ((arrayOfValues[i]-mean.x) * (arrayOfValues[i]-mean
     }
  }
  skewness.x <- ((runningCalcTotal2.x/n) / stDev.x^3)

  #kurtosis Calculation
    for (i in 1:length(arrayOfValues)){
    if(!is.na(arrayOfValues[i])){
      runningCalcTotal3.x <- (runningCalcTotal3.x + ((arrayOfValues[i]-mean.x) * (arrayOfValues[i]-mean
     }
  }
  kurtosis.x <- ((runningCalcTotal3.x/n) / stDev.x^4)
  c(skewness.x,kurtosis.x)
}


# Normalize the data
# First start with versions of the datasets that are filtered
SB_HV_2017_sub_Norm_Z.dat <- SB_HV_2017_sub.dat
C_SD_2018_sub_Norm_Z.dat <- C_SD_2018_sub.dat
C_HV_2018_sub_Norm_Z.dat <- C_HV_2018_sub.dat
SB_HV_2019_sub_Norm_Z.dat <- SB_HV_2019_sub.dat
C_SD_2020_sub_Norm_Z.dat <- C_SD_2020_sub.dat
C_HV_2020_sub_Norm_Z.dat <- C_HV_2020_sub.dat

#Normalize via the newly created Z-score function
SB_HV_2017_sub_Norm_Z.dat$Yield <- ZScore.Solve(SB_HV_2017_sub_Norm_Z.dat$Yield)
C_SD_2018_sub_Norm_Z.dat$AppliedRate <- ZScore.Solve(C_SD_2018_sub_Norm_Z.dat$AppliedRate)
C_HV_2018_sub_Norm_Z.dat$Yield <- ZScore.Solve(C_HV_2018_sub_Norm_Z.dat$Yield)
SB_HV_2019_sub_Norm_Z.dat$Yield <- ZScore.Solve(SB_HV_2019_sub_Norm_Z.dat$Yield)
C_SD_2020_sub_Norm_Z.dat$AppliedRate <- ZScore.Solve(C_SD_2020_sub_Norm_Z.dat$AppliedRate)
C_HV_2020_sub_Norm_Z.dat$Yield <- ZScore.Solve(C_HV_2020_sub_Norm_Z.dat$Yield)

# Check for skewness and kurtosis of the data
print(c("Dataset","Skewness","Kurtosis"))
```

```
## [1] "Dataset"  "Skewness" "Kurtosis"
```

```
print(c('Soybean_Harvest_2017:',SkewnessKurtosis(SB_HV_2017_sub_Norm_Z.dat$Yield)))
```

```
## [1] "Soybean_Harvest_2017:" "4.36786682249339"      "113.387590121002"
```

```
print(c('Corn_Seeding_2018:',SkewnessKurtosis(C_SD_2018_sub_Norm_Z.dat$AppliedRate)))
```

```
## [1] "Corn_Seeding_2018:" "0.661506470222465"  "72.0313082162357"
```

```
print(c('Corn_Harvest_2018:',SkewnessKurtosis(C_HV_2018_sub_Norm_Z.dat$Yield)))
```

```
## [1] "Corn_Harvest_2018:" "9.4084930746947"    "221.471742912075"
```

```
print(c('Soybean_Harvest_2019:',SkewnessKurtosis(SB_HV_2019_sub_Norm_Z.dat$Yield)))
```

```
## [1] "Soybean_Harvest_2019:" "2.46112836996902"      "100.49489453292"
```

```
print(c('Corn_Seeding_2020:',SkewnessKurtosis(C_SD_2020_sub_Norm_Z.dat$AppliedRate)))
```

```
## [1] "Corn_Seeding_2020:" "-1.52833929161938"  "63.0017855621816"
```

```
print(c('Corn_Harvest_2020:',SkewnessKurtosis(C_HV_2020_sub_Norm_Z.dat$Yield)))
```

```
## [1] "Corn_Harvest_2020:" "9.96745128813082"   "230.221714478411"
```

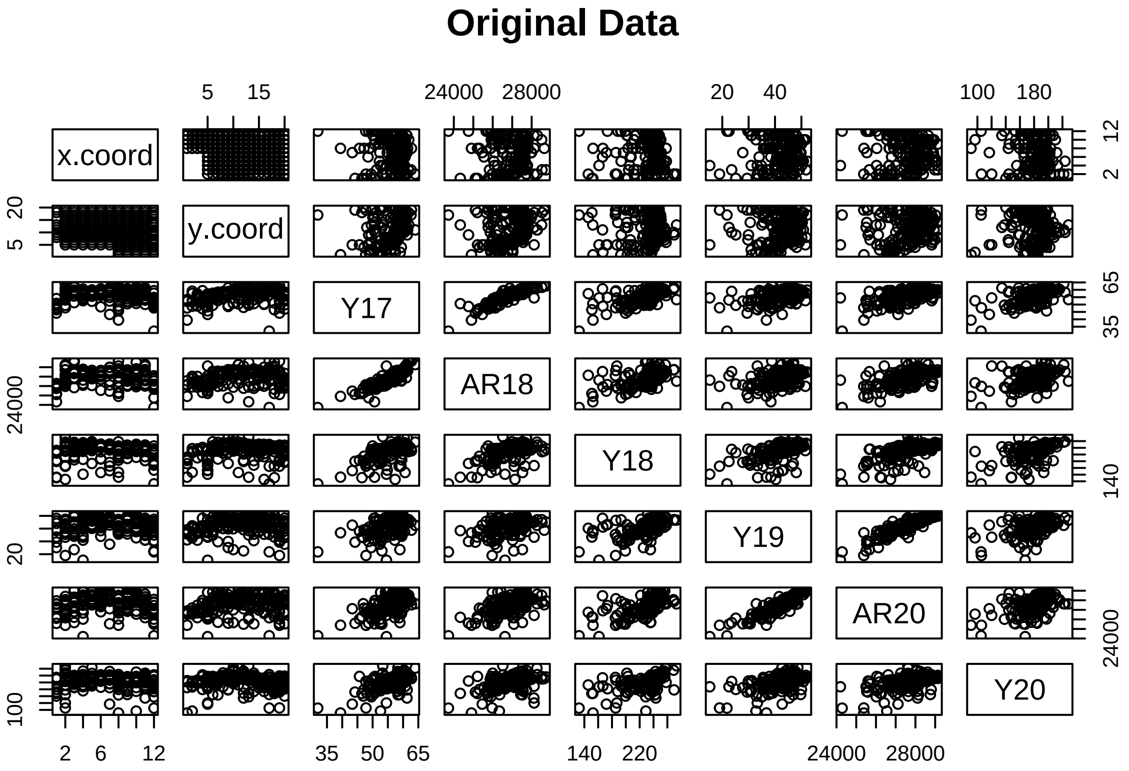
```
# Aggregate the Data via the aggregation function I created earlier
# Seeding of corn
C_SD_2018_agg_Norm_Z.dat <- Agg.SD.Create(C_SD_2018_sub_Norm_Z.dat)
C_SD_2020_agg_Norm_Z.dat <- Agg.SD.Create(C_SD_2020_sub_Norm_Z.dat)
# Harvesting of Corn
C_HV_2018_agg_Norm_Z.dat <- Agg.SD.Create(C_HV_2018_sub_Norm_Z.dat)
C_HV_2020_agg_Norm_Z.dat <- Agg.SD.Create(C_HV_2020_sub_Norm_Z.dat)
# Harvesting of soybeans
SB_HV_2017_agg_Norm_Z.dat <- Agg.SD.Create(SB_HV_2017_sub_Norm_Z.dat)
SB_HV_2019_agg_Norm_Z.dat <- Agg.SD.Create(SB_HV_2019_sub_Norm_Z.dat)

# Remove latitude and longitude from the datasets
# Seeding of corn
C_SD_2018_agg_Norm_Z_pre_merge.dat <- subset(C_SD_2018_agg_Norm_Z.dat, select = -c(Longitude,Latitude))
C_SD_2020_agg_Norm_Z_pre_merge.dat <- subset(C_SD_2020_agg_Norm_Z.dat, select = -c(Longitude,Latitude))
# Harvesting of Corn
C_HV_2018_agg_Norm_Z_pre_merge.dat <- subset(C_HV_2018_agg_Norm_Z.dat, select = -c(Longitude,Latitude))
C_HV_2020_agg_Norm_Z_pre_merge.dat <- subset(C_HV_2020_agg_Norm_Z.dat, select = -c(Longitude,Latitude))
# Harvesting of soybeans
SB_HV_2017_agg_Norm_Z_pre_merge.dat <- subset(SB_HV_2017_agg_Norm_Z.dat, select = -c(Longitude,Latitude
SB_HV_2019_agg_Norm_Z_pre_merge.dat <- subset(SB_HV_2019_agg_Norm_Z.dat, select = -c(Longitude,Latitude

# Change names of columns in the datasets
# Seeding of corn
C_SD_2018_agg_Norm_Z_pre_merge2.dat <- setNames(C_SD_2018_agg_Norm_Z_pre_merge.dat, c('x.coord','y.coor
C_SD_2020_agg_Norm_Z_pre_merge2.dat <- setNames(C_SD_2020_agg_Norm_Z_pre_merge.dat, c('x.coord','y.coor
# Harvesting of Corn
C_HV_2018_agg_Norm_Z_pre_merge2.dat <- setNames(C_HV_2018_agg_Norm_Z_pre_merge.dat, c('x.coord','y.coor
C_HV_2020_agg_Norm_Z_pre_merge2.dat <- setNames(C_HV_2020_agg_Norm_Z_pre_merge.dat, c('x.coord','y.coor
# Harvesting of soybeans
SB_HV_2017_agg_Norm_Z_pre_merge2.dat <- setNames(SB_HV_2017_agg_Norm_Z_pre_merge.dat, c('x.coord','y.co
SB_HV_2019_agg_Norm_Z_pre_merge2.dat <- setNames(SB_HV_2019_agg_Norm_Z_pre_merge.dat, c('x.coord','y.co



# Merge the dataframes into a single dataframe
Combined_Norm_Z_pre_1.dat <- Reduce(function(x,y) merge(x = x, y = y, by = c("x.coord","y.coord"), all.
# Remove rows where observations are less than 30 in any of the datasets
Combined_Norm_Z_pre_2.dat <- subset(Combined_Norm_Z_pre_1.dat, Obs_C_SD_18>=30)
Combined_Norm_Z_pre_3.dat <- subset(Combined_Norm_Z_pre_2.dat, Obs_C_SD_20>=30)
Combined_Norm_Z_pre_4.dat <- subset(Combined_Norm_Z_pre_3.dat, Obs_C_HV_18>=30)
Combined_Norm_Z_pre_5.dat <- subset(Combined_Norm_Z_pre_4.dat, Obs_C_HV_20>=30)
Combined_Norm_Z_pre_6.dat <- subset(Combined_Norm_Z_pre_5.dat, Obs_SB_HV_17>=30)
Combined_Norm_Z_pre_7.dat <- subset(Combined_Norm_Z_pre_6.dat, Obs_SB_HV_19>=30)


# Reorder columns
Combined_Norm_Z.dat <- subset(Combined_Norm_Z_pre_7.dat, select = c(x.coord, y.coord, Y17, AR18, Y18, Y
```

Graph the Normalized Data and Compare with the Original Data

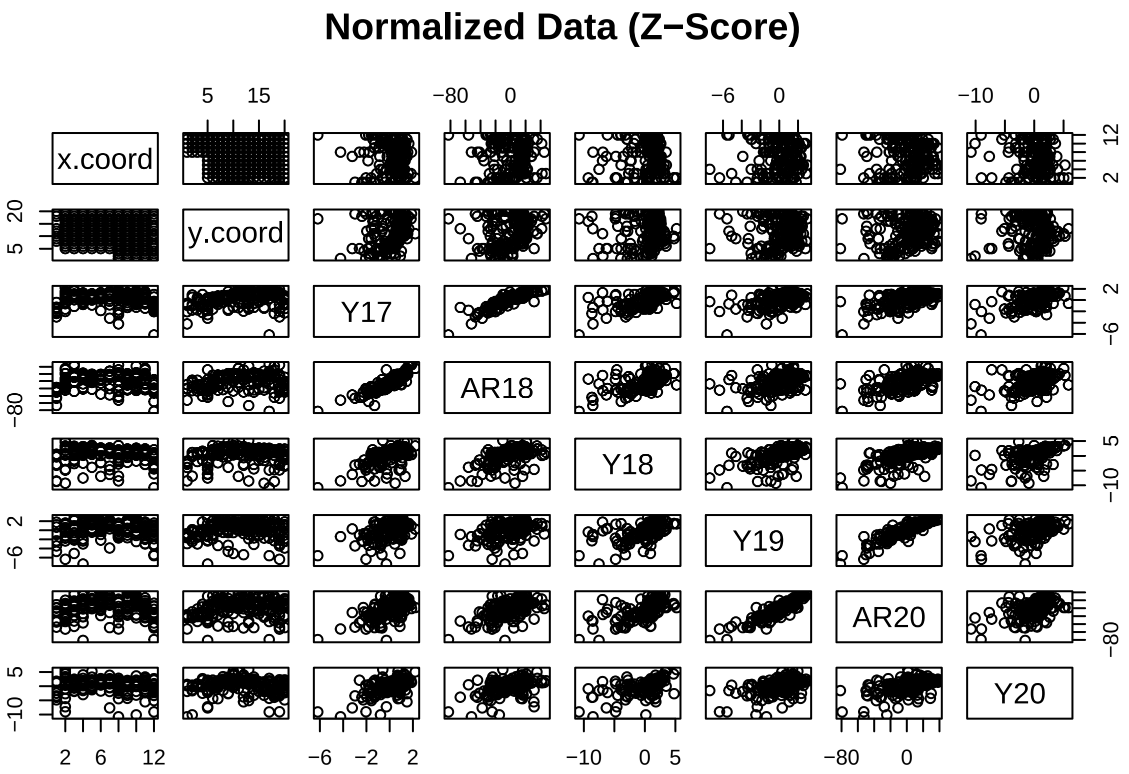
As shown below, normalizing the data did not make much of a difference in the pairs plot nor the acyclic graphs. In fact, the plots created with the original data compared to the z-score normalized data are almost exactly the same.

Key and Timeline: Y17: Yield of the 2017 Soybean Harvest AR18: Applied Rate of the 2018 Corn Seeding Y18: Yield of the 2018 Corn Harvest Y19: Yield of the 2019 Soybean Harvest AR20: Applied rate of the 2020 Corn Seeding Y20: Yield of the 2020 Corn Harvest

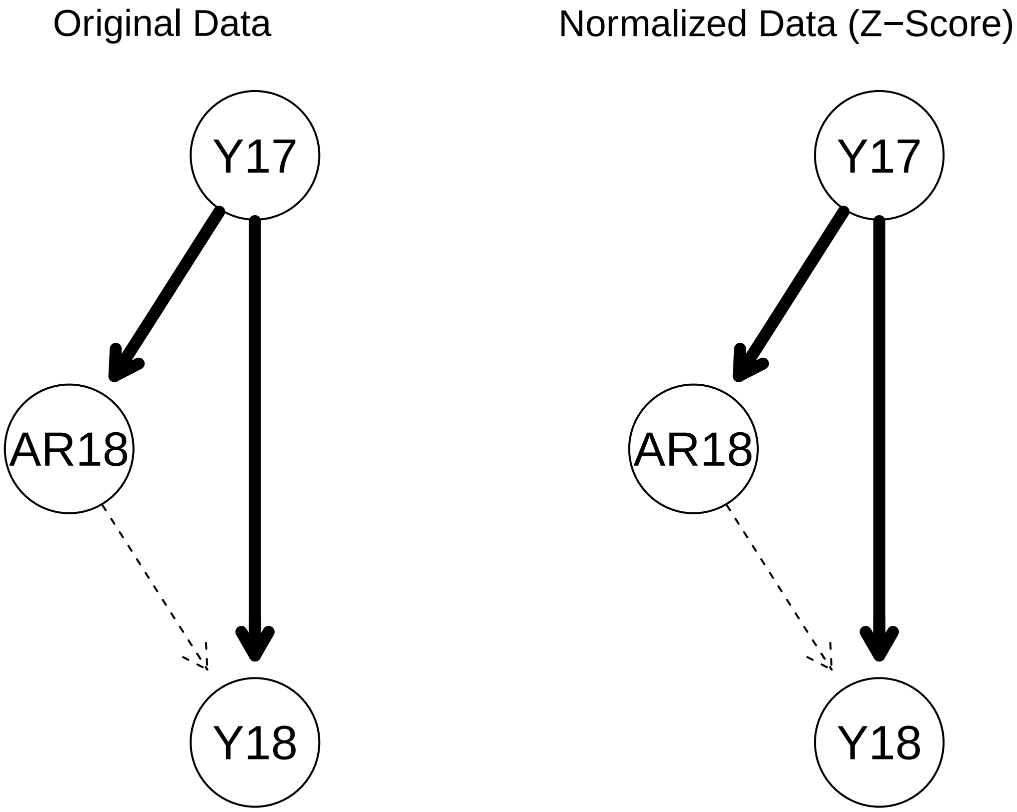
```
#Pairs Plot Comparison
pairs(subset(Combined.dat,select = c(1:8)), main = "Original Data")
```



```
pairs(subset(Combined_Norm_Z.dat,select = c(1:8)), main = "Normalized Data (Z-Score)")
```



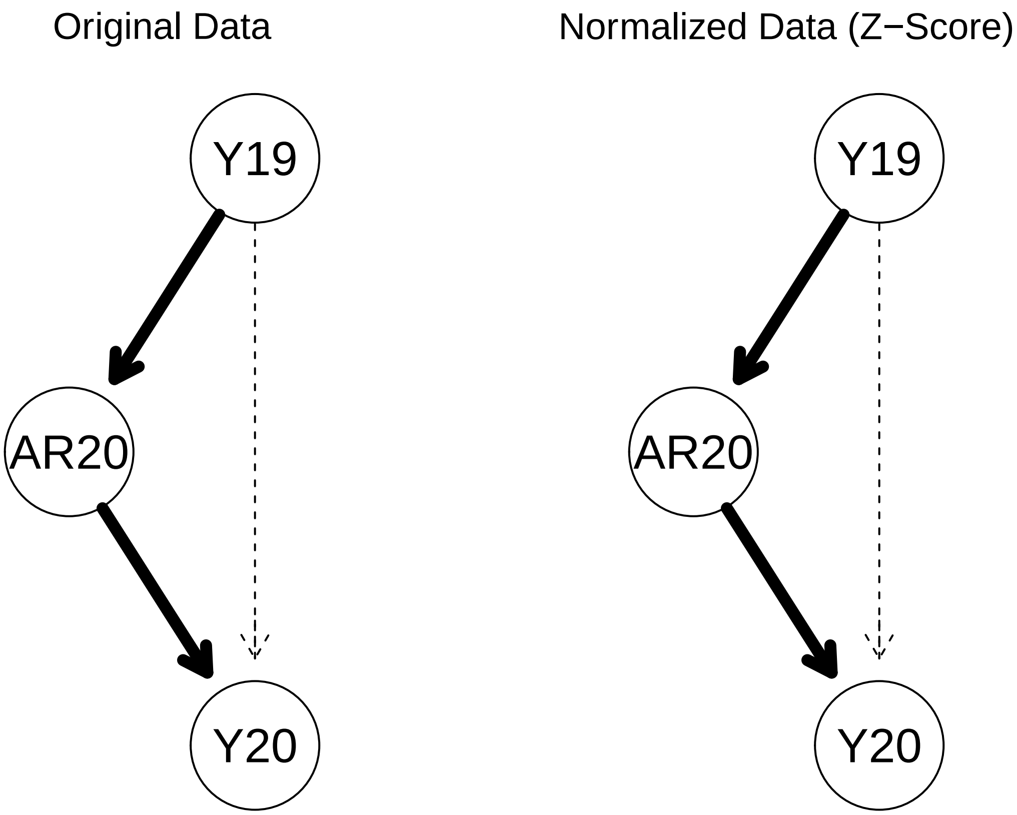
```
#Acyclic Graph
# Check for required libraries
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
if (!requireNamespace("Rgraphviz", quietly = TRUE))
    install.packages("Rgraphviz")
if (!requireNamespace("bnlearn", quietly = TRUE))
    install.packages("bnlearn")
library(bnlearn)
# Graph the data
modela.dag <- model2network("[Y17][AR18|Y17][Y18|AR18:Y17]")
fit1_Norm_Z = bn.fit(modela.dag, Combined_Norm_Z.dat[,c('Y17','AR18','Y18')])
#fit1
strengtha_Norm_Z <- arc.strength(modela.dag, Combined_Norm_Z.dat[,c('Y17','AR18','Y18')])
# set the plotting area into a 1*2 array
par(mfrow = c(1, 2))
strength.plot(modela.dag, strengtha, main = "Original Data")
strength.plot(modela.dag, strengtha_Norm_Z, main = "Normalized Data (Z-Score)")
```



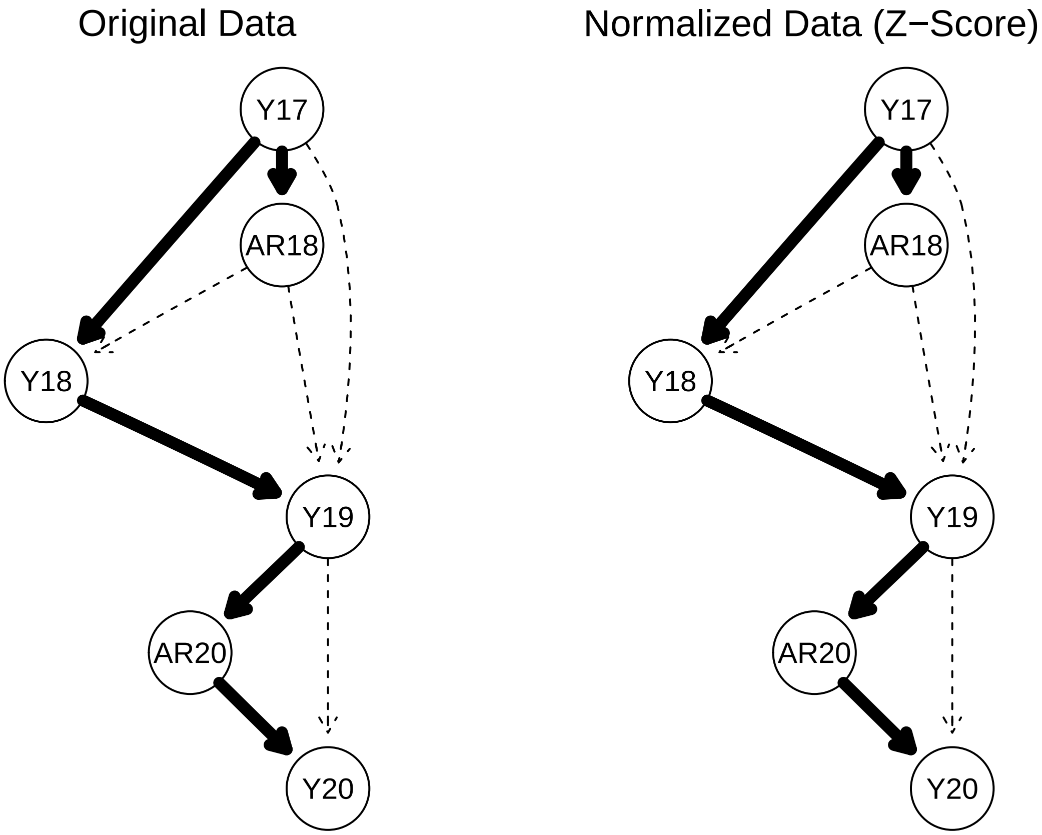
Original Data

Normalized Data (Z−Score)

```r
modelb.dag <- model2network("[Y19][AR20|Y19][Y20|AR20:Y19]")
fit2_Norm_Z = bn.fit(modelb.dag, Combined_Norm_Z.dat[,c('Y19','AR20','Y20')])
#fit2
strengthb_Norm_Z <- arc.strength(modelb.dag, Combined_Norm_Z.dat[,c('Y19','AR20','Y20')])
# set the plotting area into a 1*2 array
par(mfrow = c(1, 2))
strength.plot(modelb.dag, strengthb, main = "Original Data")
strength.plot(modelb.dag, strengthb_Norm_Z, main = "Normalized Data (Z-Score)")
```



```
model1.dag <- model2network("[Y17][AR18|Y17][Y18|AR18:Y17][Y19|Y17:AR18:Y18][AR20|Y19][Y20|AR20:Y19]")
fit3_Norm_Z = bn.fit(model1.dag, Combined_Norm_Z.dat[,c('Y17','AR18','Y18','Y19','AR20','Y20')])
#fit3
strength1_Norm_Z <- arc.strength(model1.dag, Combined_Norm_Z.dat[,c('Y17','AR18','Y18','Y19','AR20','Y2
# set the plotting area into a 1*2 array
par(mfrow = c(1, 2))
strength.plot(model1.dag, strength1, main = "Original Data")
strength.plot(model1.dag, strength1_Norm_Z, main = "Normalized Data (Z-Score)")
```



Original Data

Normalized Data (Z−Score)

Conclusion

The original question that this analysis was designed to answer was: “If a specific region of the corn field was low yielding in 2018, was it because it was seeded at a lower rate, or because it is just a poor part of the field, as indicated by yield 2017?”

As shown in the pairs plots, the strongest relationships seem to be between yield of a given area of land on a given year and the seeding rate of the same area of land during the next year, indicating that farmers are trying to optimize their crop yield by seeding at a higher rate on areas of land that they experienced higher crop yield on in the previous year. Other relationships shown in the pairs plot are not strong enough to make further inferences

The acyclic graphs allow for further inferences to be made my assisting in a causal analysis. These indicate, similar to the pairs plot, a relationship between yield of regions of land on a given year and average rate of seeding on the same regions of land during the next year. They not only show this relationship, but also a relationship between yield of regions of land between 2017, 2018, and 2019. Oddly enough, it doesn’t show this same relationship between years 2019 and 2020, but instead showing the average yield of crops in 2019 on a given region of land affecting the average rate of seeding in 2020 and the average rate of seeding in 2020 in turn, affecting the average yield of crops in 2020.

Thus, to answer the original question, if a specific region of the corn field was low yielding in 2018, it was likely because it was a poor part of land and not because it was seeded at a lower rate.