# Exploratory Cluster Analysis for Josie Schafer: Part 2

Michael Flynn, Prior Analytics, LLC.

## Goals

1. Focus on developing and refining the clustering procedure.
2. Look at making the distribution of MSAs across clusters more even.
3. Focus on trying to develop more coherence with respect to demographics, higher education, and hospital elements of the cluster procedures. That is, make sure clusters make more sense.

## Notes

1. Previously focused on hierarchical clustering
2. Look at K-means clustering more. This pre-specifies the number of clusters in the data.
3. Also possible to look at principal components analysis or factor analysis.
4. Revisit the distance calculation.

- Hierarchical clustering may still make sense if we're calculating the distance using alternative methods.
- Euclidean distance will calculate distance on the basis of similarities across levels of inputs. Euclidean is the default, but other algorithms may offer better results.
- Correlation based distance calculations may make more sense as we're interested in similarities along the basis of the input measures.
- Be aware that we're looking at the `dist()` function distance calculation and then the algorithm for calculating the clusters in `hclust()`. The former handles the distance calculation and the latter handles the breakdown into different clusters.
- We want to continue rescaling the measures so that certain large values from spending variables don't overwhelm the influence of other variables not on such large scales.

5. I'm going to try another rescaling procedure, but there are some definite outliers, so reverting to the 0-1 scaling may prove to be a better option still. But some spending-based measures produce large outliers and this may help to differentiate some MSAs.

## Data Cleaning

Just like last time we read in the data and clean it and rescale the variables. Again, rescaling is done using z-score methods this time rather than the proportion-based measure. This seems to help produce more revenly distributed clusters.

```
# Read in data and select relevant variables for clusters.
# Focus is on demographic and anchor institution variables.

# Read in raw data file
data <- readxl::read_xlsx(here("data/anchor regions analysis.xlsx"))

# List of variables to include in clustering
varlist <- c("totpop_19",  # Total pop
             "popchange",  # pop change
             "medage",     # Median age
             "labfor",     # percent population in labor force
             "pov",       # percent population living in poverty
             "poc",       # people of color as percent of pop
             "highed",     # Percent population with at least bachelor's
             "forborn",    # Percent population foreign born
             "net_mig",    # Net domestic migration
             "highered_emp_qcew",
             "highered_estab_qcew",
             "hospital_emp_qcew",
             "hospital_estab_qcew",
             "inst_ipeds_enrollment_all",
             "inst_ipeds_doctoralunihighrese",
             "inst_ipeds_pellawards",
             "inst_hosp_ahacommunityhospitals",
             "inst_hosp_ahabeds",
             "inst_hosp_nihresearchfunding")

# Rescale the variables from 0-1
data.clean <- data |>
  mutate(across(all_of(varlist), # Variables to scale
                ~scale(.x),                                                      # Scale relative to m
```

```
              .names = "{col}_rescaled")) |>                                   # Add "max" suf
  dplyr::select(MSA, ends_with("_rescaled")) |>                                # select chosen
  column_to_rownames("MSA")
```

## Clustering Methods

### Hierarchical Clustering Procedure

Rescaling to z scores rather than scaling relative to the maximum of the category produces more evenly distributed clusters, all else being equal.

I'm keeping the same number of clusters (i.e. 8 clusters with sizes ranging from 5–40, increasing in increments of 5).

#### Euclidean Distance Method

This first chunk replices the Euclidean distance approach from the first phase.

```
distance <- dist(data.clean) # calculate Euclidean distance between obs

hc.tree <- hclust(distance, method = "complete") # Create cluster groupings based on dista


# List of distances to use in generating clusters
cluster.size.list <- list("5" = 5,
                          "10" = 10,
                          "15" = 15,
                          "20" = 20,
                          "25" = 25,
                          "30" = 30,
                          "35" = 35,
                          "40" = 40)

cluster.ids <- map(
  .x = seq_along(cluster.size.list),
  .f = ~ cutree(hc.tree, k = cluster.size.list[[.x]])
                 ) |>
  bind_cols()
```

```
New names:
* `` -> `...1`
* `` -> `...2`
* `` -> `...3`
* `` -> `...4`
* `` -> `...5`
* `` -> `...6`
* `` -> `...7`
* `` -> `...8`
```

```
names(cluster.ids) <- c("cluster_5", "cluster_10", "cluster_15", "cluster_20", "cluster_25

data.out.euc <- data |>
  bind_cols(cluster.ids) |>
  dplyr::select(starts_with("cluster"), Name, State, MSA, varlist)
```

```
Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
i Please use `all_of()` or `any_of()` instead.
  # Was:
  data %>% select(varlist)

  # Now:
  data %>% select(all_of(varlist))

See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
```

```
names(data.out.euc) <- c("Cluster 5",
                         "Cluster 10",
                         "Cluster 15",
                         "Cluster 20",
                         "Cluster 25",
                         "Cluster 30",
                         "Cluster 35",
                         "Cluster 40",
                         "Name",
                         "State",
                         "MSA",
                         "Total Population (2019)",
                         "Population Change",
                         "Median Age",
                         "% Population in Labor Force",
                         "% Population in Poverty",
                         "% Population People of Color",
                         "$% Population with Bachelor's Degree",
                         "% Population Foreign Born",
                         "Net Domestic Migration",
                         "Higher Education Employment",
                         "Higher Education Establishments",
                         "Hospital Employment",
                         "Hospital Establishments",
                         "Higher Education Enrollment",
                         "High Research Doctoral Degree Institutions",
                         "Total Pell Grant Amounts Awarded",
                         "Hospitals/Community Hospitals",
                         "Hospital Beds",
                         "NIH Research Funding")

    write_csv(data.out.euc,
              here::here("data/raw-data-with-cluster-ids-euclidean.csv"))
```

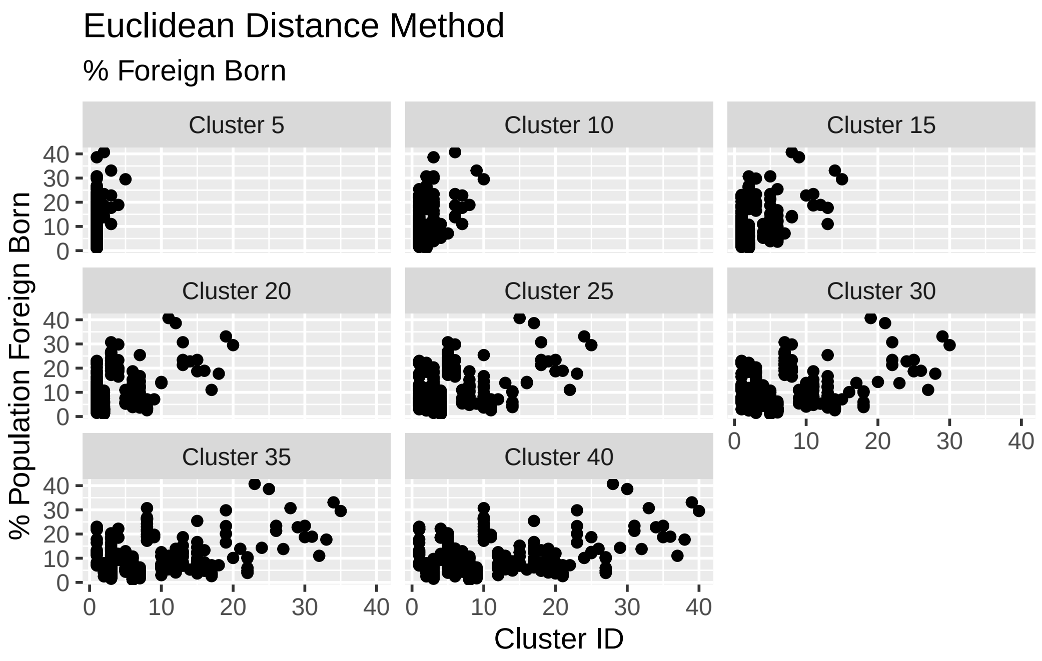
### Visualization for Euclidean Distance Method

This part includes some sample figures to show what the distribution of some of the inputs looks like across clusters.

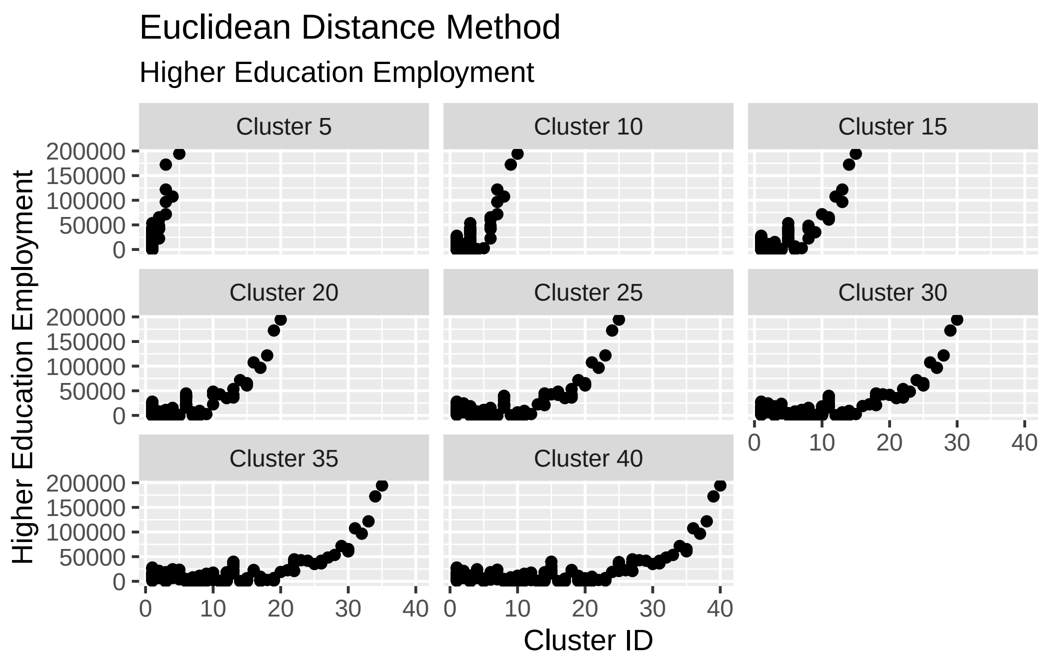
One important take away across all methods is that the more clusters we get the more likely we are to have similar average levels of a variable across clusters.

For the Euclidean distance calculation method, we end up with several very small clusters as we increase the total number of clusters. So as we move to more than 15-20 clusters we start seeing lots of clusters with only a few, or even a single, member.

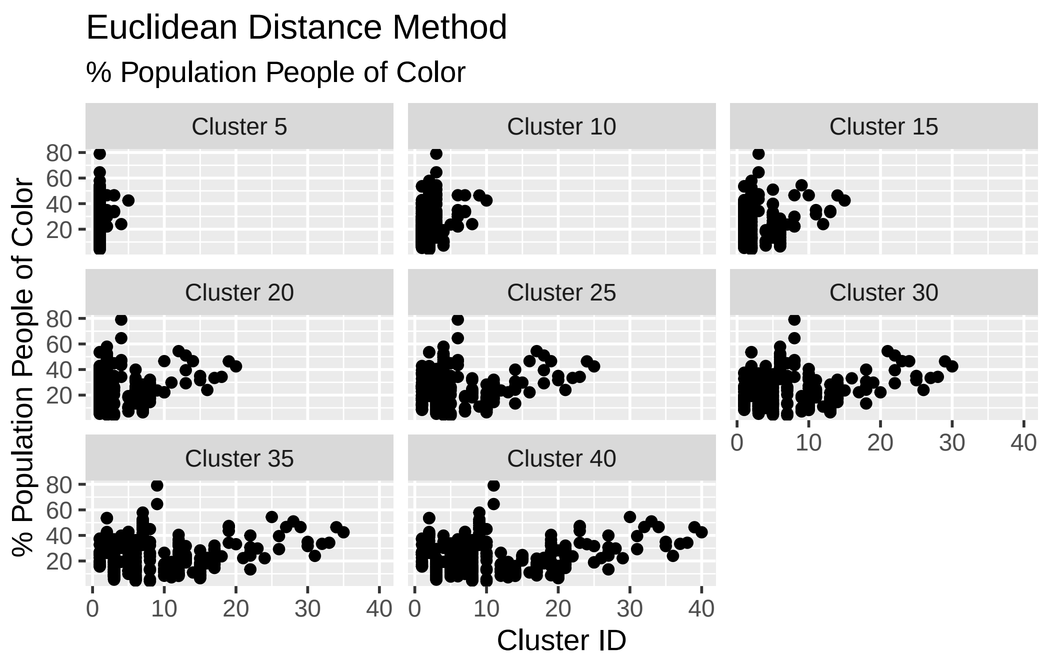
```
data.long <- data.out.euc |>
  pivot_longer(cols = starts_with("Cluster"),
               names_to = "Cluster Size",
               values_to = "Cluster ID") |>
  mutate(`Cluster Size` = factor(`Cluster Size`,
                                 levels = c("Cluster 5",
                                            "Cluster 10",
                                            "Cluster 15",
                                            "Cluster 20",
                                            "Cluster 25",
                                                          "Cluster 30",
                                                          "Cluster 35",
                                                          "Cluster 40")))


ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `% Population Foreign Born`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "Euclidean Distance Method",
       subtitle = "% Foreign Born")
```



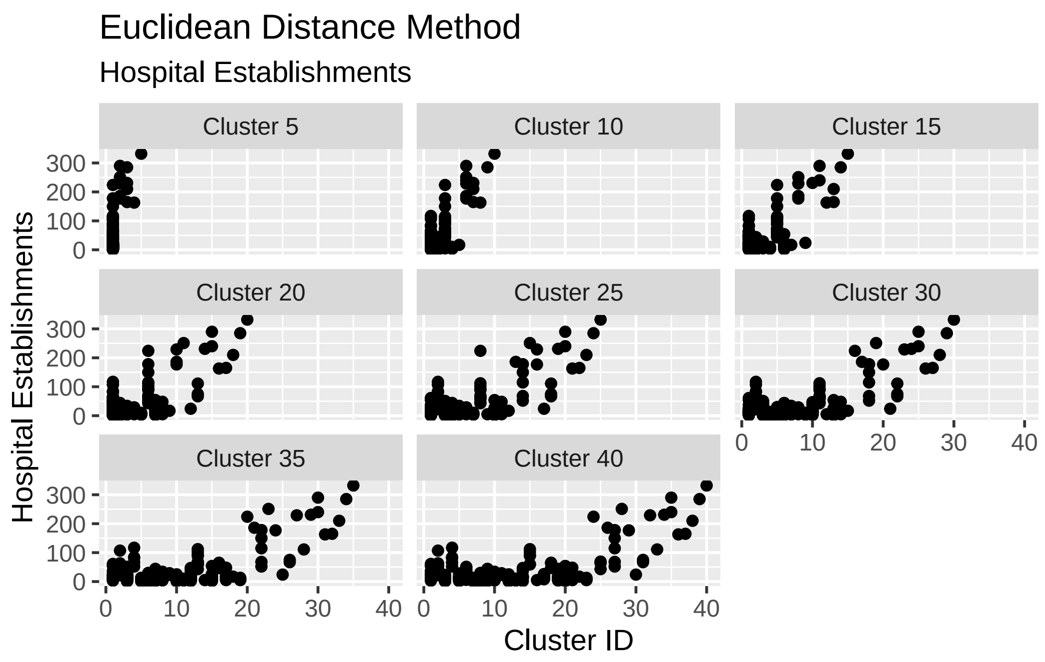
```
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `Higher Education Employment`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "Euclidean Distance Method",
       subtitle = "Higher Education Employment")
```



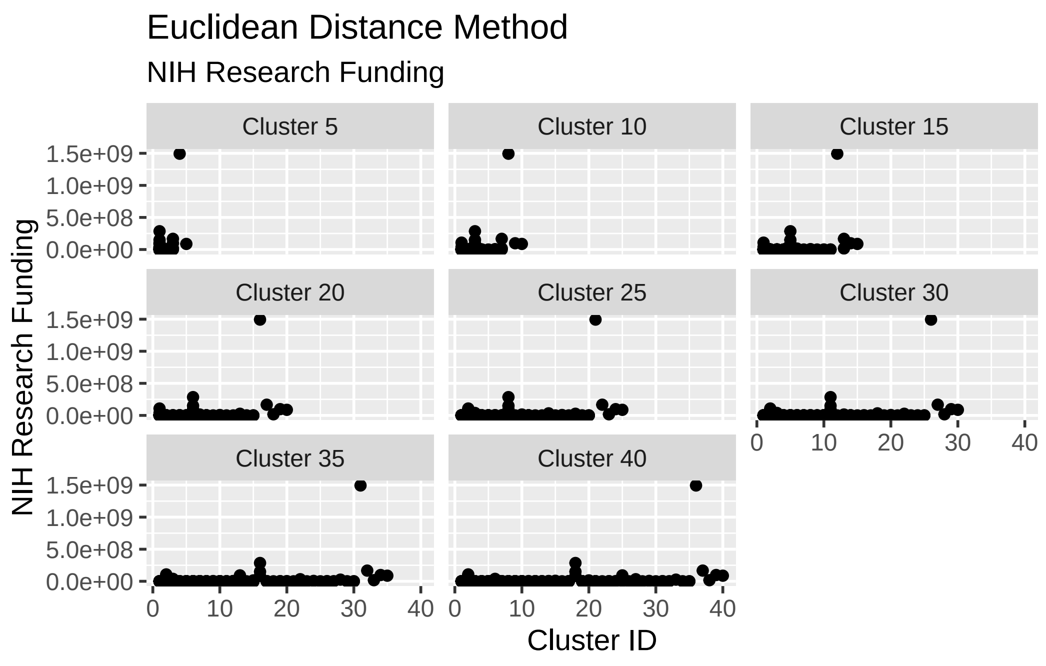
```
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `% Population People of Color`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "Euclidean Distance Method",
       subtitle = "% Population People of Color")
```



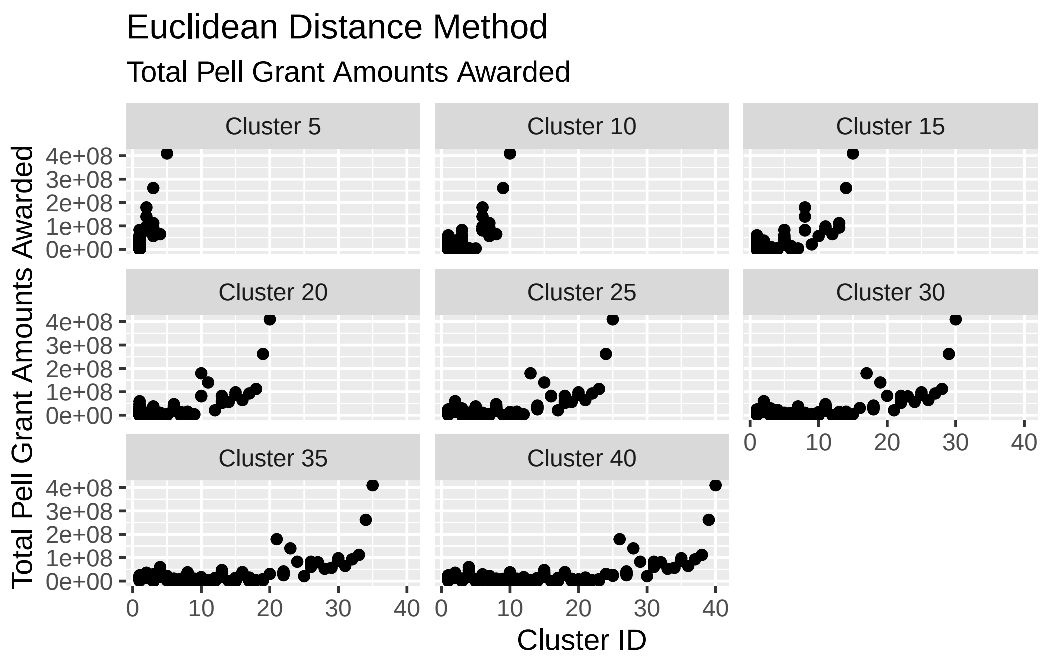
```
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `Hospital Establishments`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "Euclidean Distance Method",
       subtitle = "Hospital Establishments")
```



```
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `NIH Research Funding`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "Euclidean Distance Method",
       subtitle = "NIH Research Funding")
```



```
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `Total Pell Grant Amounts Awarded`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "Euclidean Distance Method",
       subtitle = "Total Pell Grant Amounts Awarded")
```



## Correlation Method

This time I calculate distance according to the correlation method. Resources indicate that the correlation method can be better for matching similarities across dimensions, whereas Euclidean distance is going to be better at matching on the basis of volume/spending levels, regardless of similarities across particular dimensions/variables (James et al. 2021). If the goal is to generate clusters that are more readily identifiable on the basis of the three core dimensions, this is more likely to do that.

```
distance.matrix <- as.matrix(data.clean) # convert to matrix

distance <- as.dist(1 - cor(t(distance.matrix))) # calculate correlation-based

hc.tree <- hclust(distance, method = "complete") # Create cluster groupings based on dista


# List of distances to use in generating clusters
cluster.size.list <- list("5" = 5,
                          "10" = 10,
                          "15" = 15,
                          "20" = 20,
                          "25" = 25,
                              "30" = 30,
                              "35" = 35,
                              "40" = 40)

cluster.ids <- map(
  .x = seq_along(cluster.size.list),
  .f = ~ cutree(hc.tree, k = cluster.size.list[[.x]])
                 ) |>
  bind_cols()
```

```
New names:
* `` -> `...1`
* `` -> `...2`
* `` -> `...3`
* `` -> `...4`
* `` -> `...5`
* `` -> `...6`
* `` -> `...7`
* `` -> `...8`
```

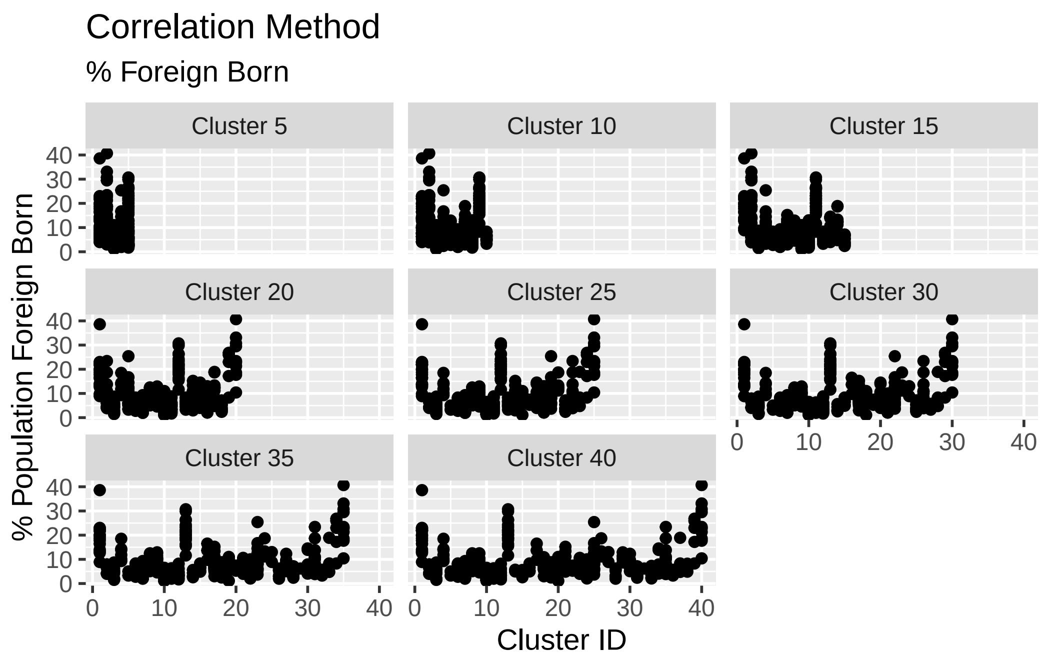
```
names(cluster.ids) <- c("cluster_5", "cluster_10", "cluster_15", "cluster_20", "cluster_25

data.out.cor <- data |>
  bind_cols(cluster.ids) |>
  dplyr::select(starts_with("cluster"), Name, State, MSA, varlist)

names(data.out.cor) <- c("Cluster 5",
                     "Cluster 10",
                     "Cluster 15",
                     "Cluster 20",
                     "Cluster 25",
                     "Cluster 30",
                     "Cluster 35",
                     "Cluster 40",
                     "Name",
                     "State",
                     "MSA",
                     "Total Population (2019)",
                     "Population Change",
                     "Median Age",
                     "% Population in Labor Force",
                     "% Population in Poverty",
                     "% Population People of Color",
                     "$% Population with Bachelor's Degree",
                     "% Population Foreign Born",
                     "Net Domestic Migration",
                     "Higher Education Employment",
                     "Higher Education Establishments",
                     "Hospital Employment",
                     "Hospital Establishments",
                     "Higher Education Enrollment",
                     "High Research Doctoral Degree Institutions",
                     "Total Pell Grant Amounts Awarded",
                     "Hospitals/Community Hospitals",
                     "Hospital Beds",
                     "NIH Research Funding")

write_csv(data.out.cor,
          here::here("data/raw-data-with-cluster-ids-correlation.csv"))
```

### Visualization for Correlation Method

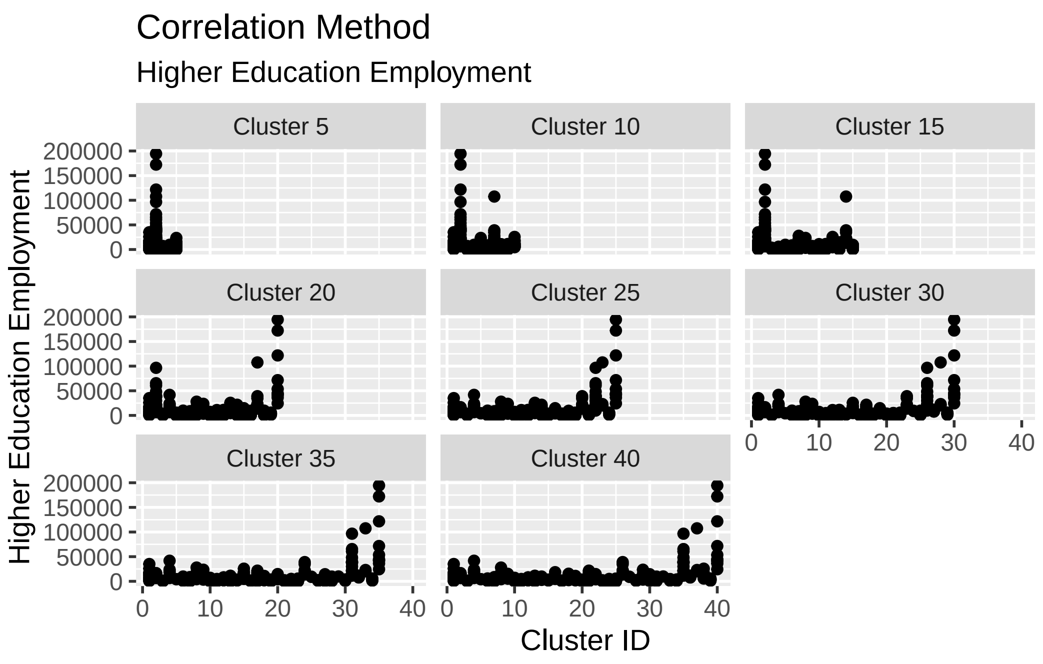
What we find below is that we get more even distribution of the MSAs across clusters. However, we also tend to find some variables have clusters with a high level of variation within cluster. Some clusters are tighly grouped on the input variable of interest while others see a very large range in input variable values.

```
data.long <- data.out.cor |>
  pivot_longer(cols = starts_with("Cluster"),
               names_to = "Cluster Size",
               values_to = "Cluster ID") |>
  mutate(`Cluster Size` = factor(`Cluster Size`,
                                 levels = c("Cluster 5",
                                            "Cluster 10",
                                            "Cluster 15",
                                            "Cluster 20",
                                            "Cluster 25",
                                            "Cluster 30",
                                            "Cluster 35",
                                            "Cluster 40")))
```

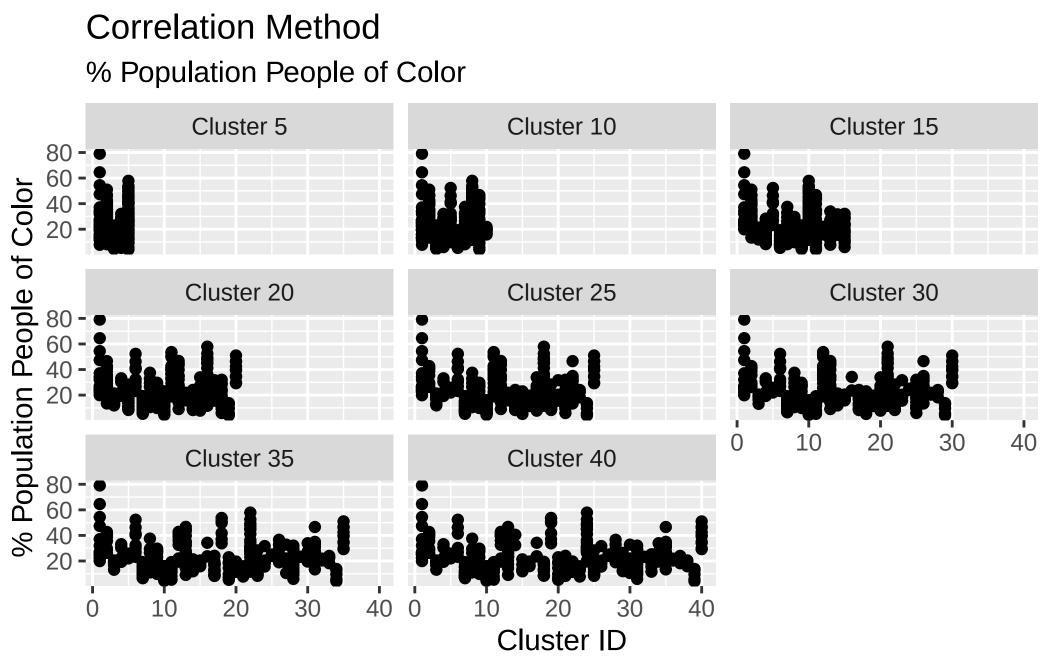
```r
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `% Population Foreign Born`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "Correlation Method",
       subtitle = "% Foreign Born")
```



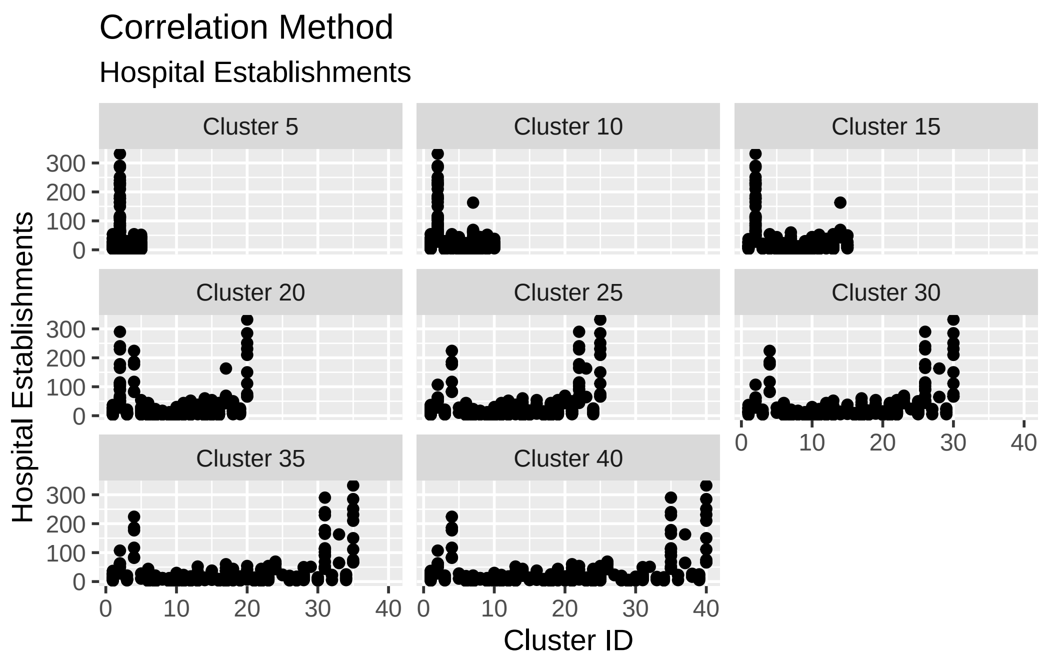
```r
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `Higher Education Employment`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "Correlation Method",
       subtitle = "Higher Education Employment")
```



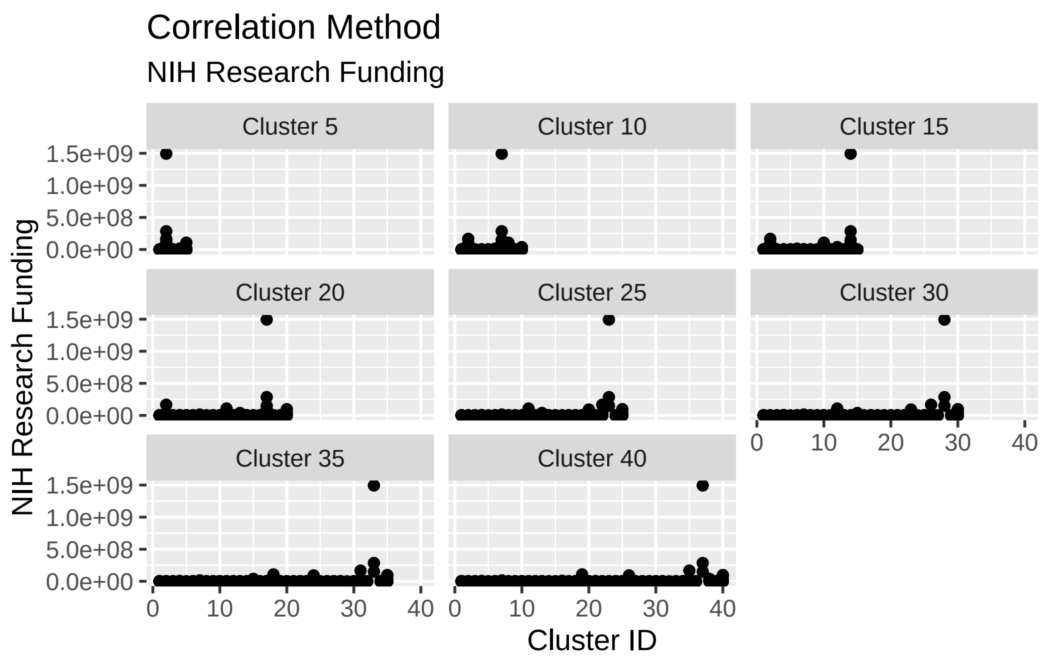
```
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `% Population People of Color`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "Correlation Method",
       subtitle = "% Population People of Color")
```



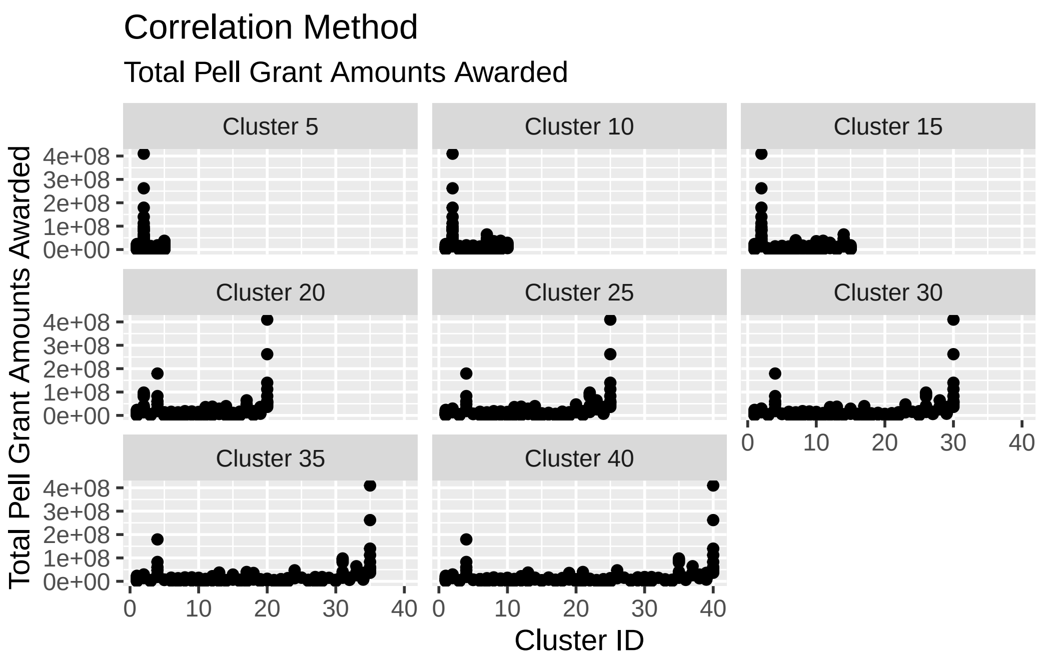
```r
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `Hospital Establishments`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "Correlation Method",
       subtitle = "Hospital Establishments")
```



```
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `NIH Research Funding`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "Correlation Method",
       subtitle = "NIH Research Funding")
```



```
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `Total Pell Grant Amounts Awarded`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "Correlation Method",
       subtitle = "Total Pell Grant Amounts Awarded")
```



**K-Means Method**

K-Means is another method for calculating clusters, but it requires us to specify the exact number of clusters we want at the outset. While we play with different cluster sizes above, the hierarchical clustering method is generating a range of clusters from 1–N in size and allows us to specify different cut points along that range. Here we have to run the method 8 times to obtain the same 8 different cluster sizes/groupings.

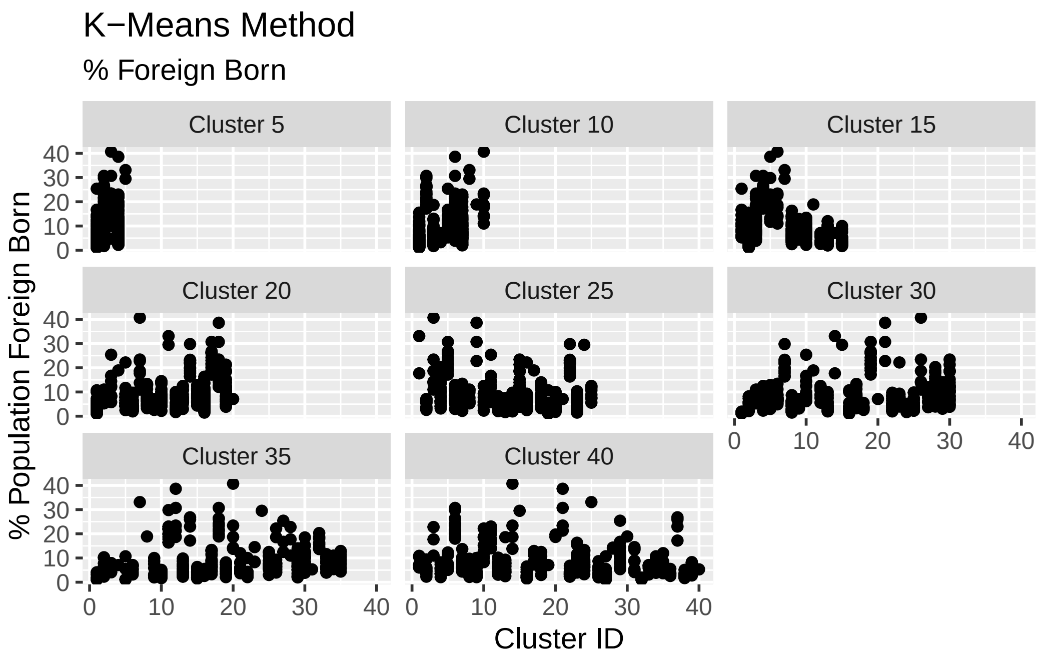
```
# List of distances to use in generating clusters
cluster.size.list <- list("5" = 5,
                          "10" = 10,
                          "15" = 15,
                          "20" = 20,
                          "25" = 25,
                          "30" = 30,
                          "35" = 35,
                          "40" = 40)

# Need to run the kmeans procedure multiple times to generate clusters
cluster.ids <- map(
  .x = seq_along(cluster.size.list),
  .f = ~ kmeans(data.clean,
                 centers = cluster.size.list[[.x]])
  )

cluster.ids.combined <- bind_cols(cluster.ids[[1]]$cluster,
                                  cluster.ids[[2]]$cluster,
                                  cluster.ids[[3]]$cluster,
                                  cluster.ids[[4]]$cluster,
                                  cluster.ids[[5]]$cluster,
                                  cluster.ids[[6]]$cluster,
                                  cluster.ids[[7]]$cluster,
                                  cluster.ids[[8]]$cluster)
```

```
New names:
* `` -> `...1`
* `` -> `...2`
* `` -> `...3`
* `` -> `...4`
* `` -> `...5`
* `` -> `...6`
* `` -> `...7`
* `` -> `...8`
```

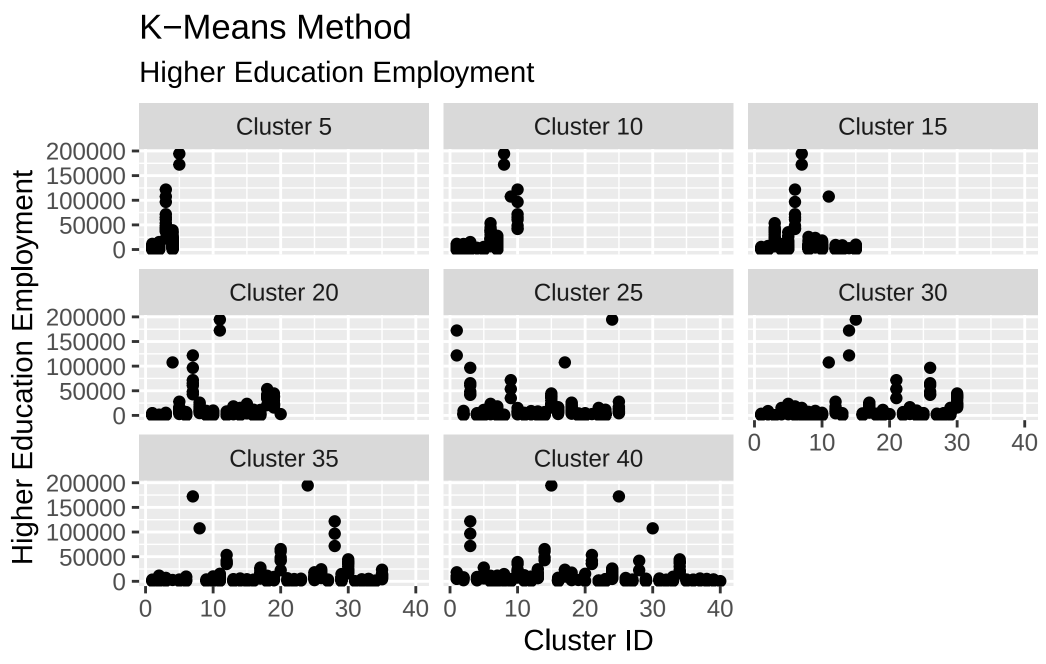
```
names(cluster.ids.combined) <- c("Cluster 5",
                                 "Cluster 10",
                                 "Cluster 15",
                                 "Cluster 20",
                                 "Cluster 25",
                                 "Cluster 30",
                                 "Cluster 35",
                                 "Cluster 40")


data.out.kmeans <- data |>
  bind_cols(cluster.ids.combined) |>
  dplyr::select(starts_with("Cluster"), Name, State, MSA, varlist)

names(data.out.kmeans) <- c("Cluster 5",
                     "Cluster 10",
                     "Cluster 15",
                     "Cluster 20",
                     "Cluster 25",
                     "Cluster 30",
                     "Cluster 35",
                     "Cluster 40",
                     "Name",
                     "State",
                     "MSA",
                     "Total Population (2019)",
                     "Population Change",
                     "Median Age",
                     "% Population in Labor Force",
                     "% Population in Poverty",
                     "% Population People of Color",
                     "$% Population with Bachelor's Degree",
                     "% Population Foreign Born",
                     "Net Domestic Migration",
                     "Higher Education Employment",
                     "Higher Education Establishments",
                     "Hospital Employment",
                     "Hospital Establishments",
                     "Higher Education Enrollment",
                     "High Research Doctoral Degree Institutions",
                     "Total Pell Grant Amounts Awarded",
                     "Hospitals/Community Hospitals",
                     "Hospital Beds",
                     "NIH Research Funding")

write_csv(data.out.kmeans,
          here::here("data/raw-data-with-cluster-ids-kmeans.csv"))
```

### K-Means Visualization

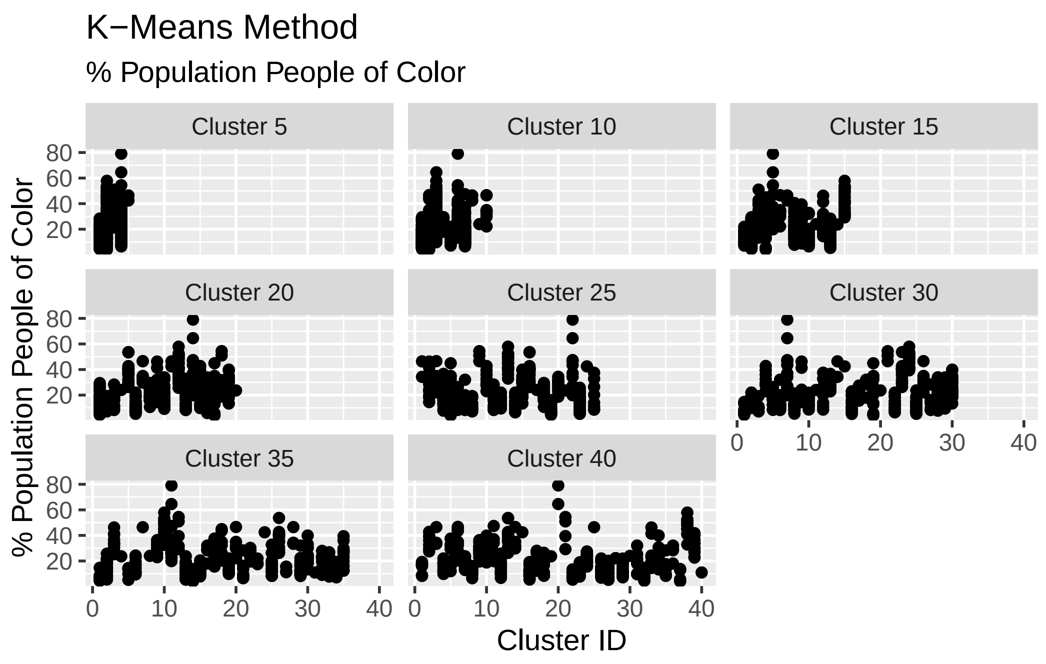
```
data.long <- data.out.kmeans |>
  pivot_longer(cols = starts_with("Cluster"),
               names_to = "Cluster Size",
               values_to = "Cluster ID") |>
  mutate(`Cluster Size` = factor(`Cluster Size`,
                                 levels = c("Cluster 5",
                                            "Cluster 10",
                                            "Cluster 15",
                                            "Cluster 20",
                                            "Cluster 25",
                                                            "Cluster 30",
                                                            "Cluster 35",
                                                            "Cluster 40")))

ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `% Population Foreign Born`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "K-Means Method",
       subtitle = "% Foreign Born")
```



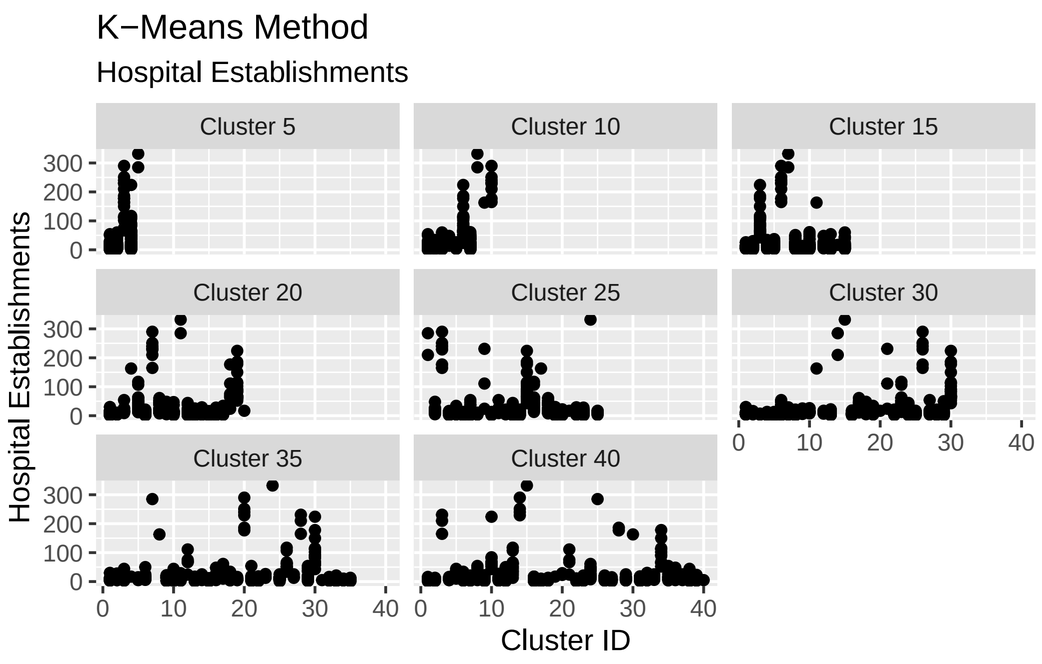
```
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `Higher Education Employment`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "K-Means Method",
       subtitle = "Higher Education Employment")
```



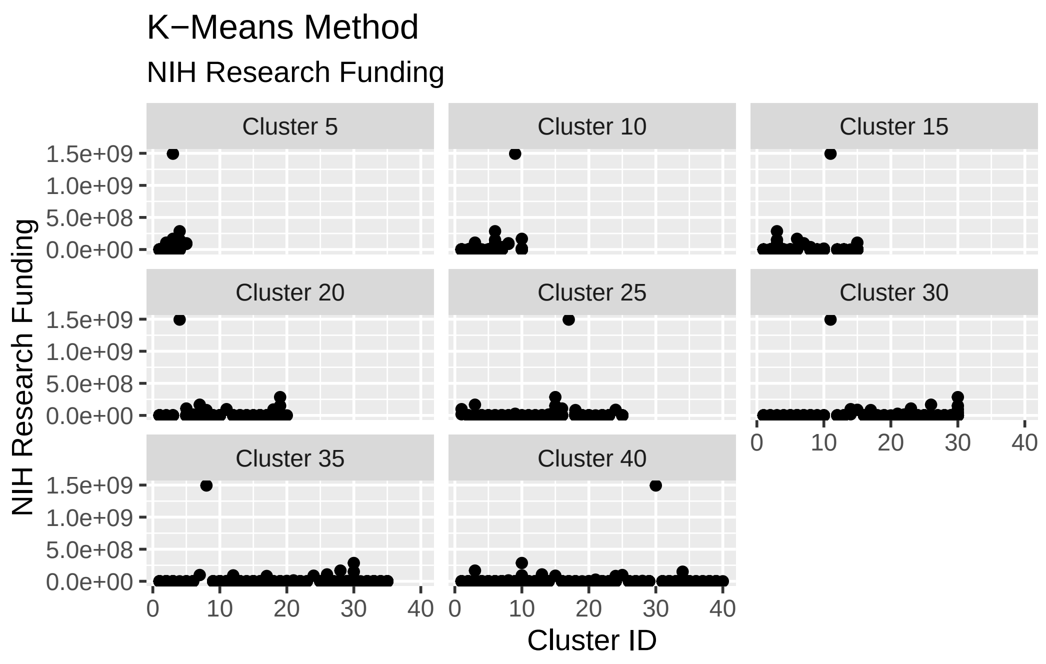
```r
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `% Population People of Color`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "K-Means Method",
       subtitle = "% Population People of Color")
```



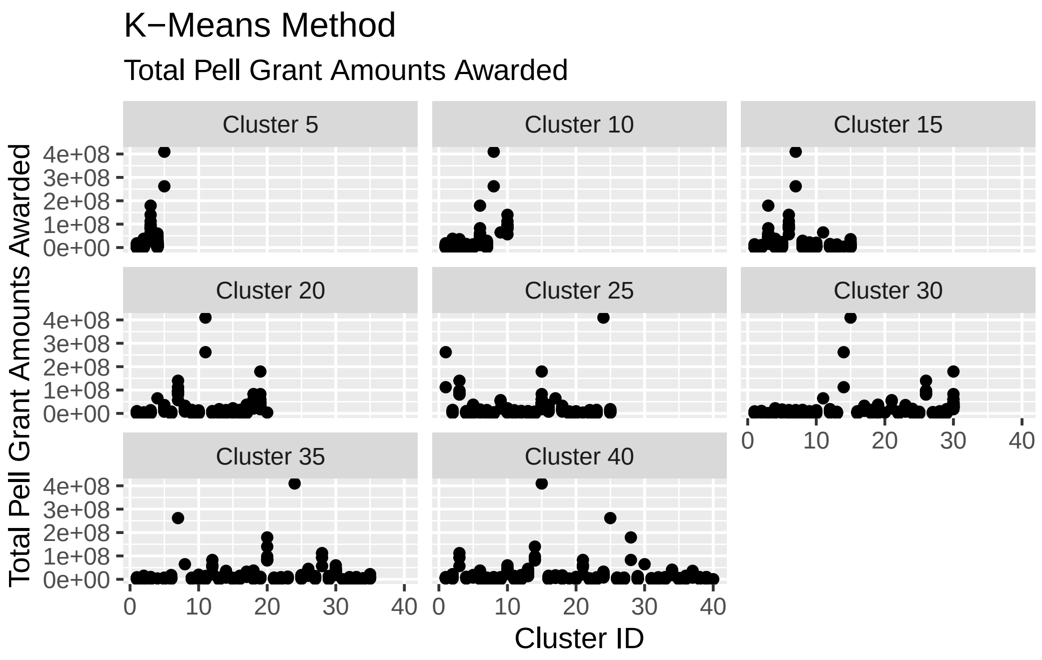
```
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `Hospital Establishments`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "K-Means Method",
       subtitle = "Hospital Establishments")
```



```
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `NIH Research Funding`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "K-Means Method",
       subtitle = "NIH Research Funding")
```



```
ggplot(data = data.long) +
  geom_point(aes(x = `Cluster ID`, y = `Total Pell Grant Amounts Awarded`)) +
  facet_wrap(. ~ `Cluster Size`) +
  labs(title = "K-Means Method",
       subtitle = "Total Pell Grant Amounts Awarded")
```



James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2021. *An Introuction to Statistical Leaning: With Applications in r*. Second Edition. Springer.