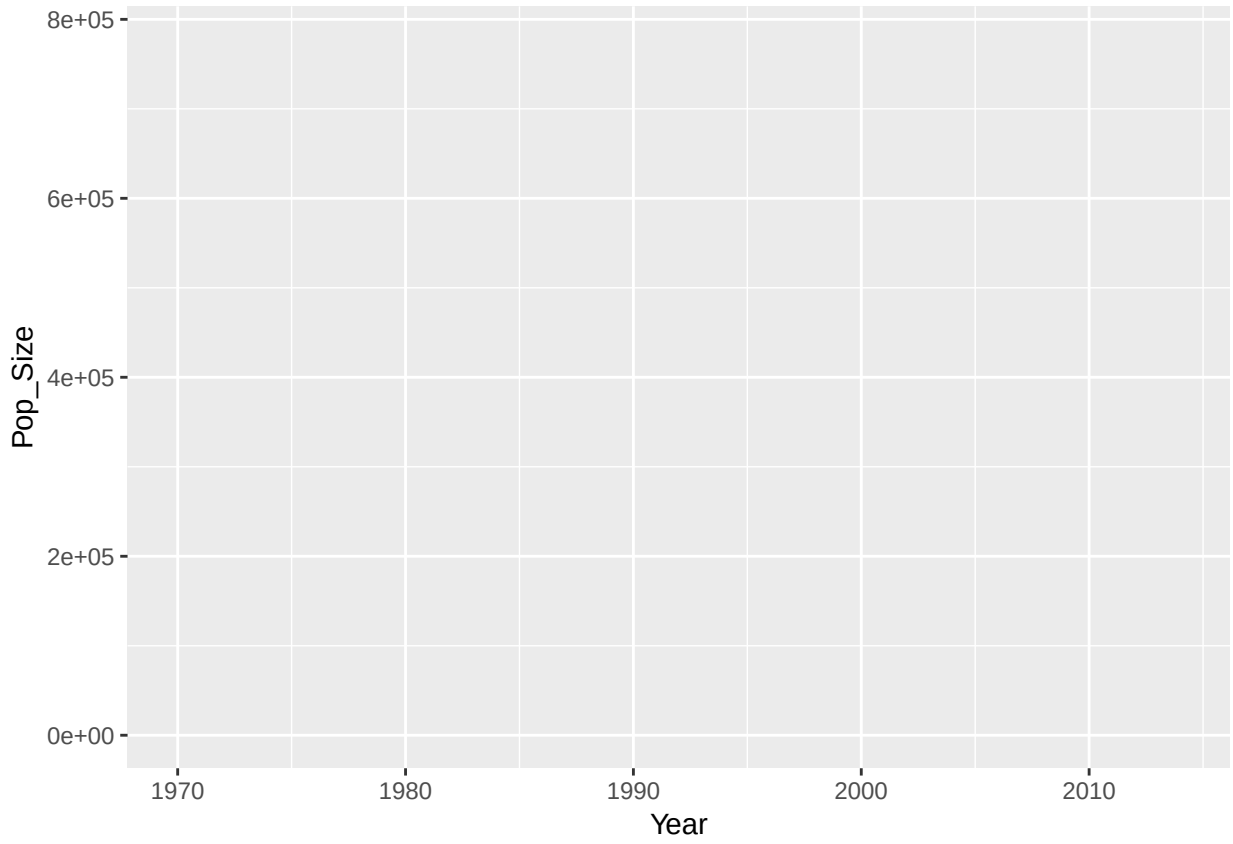
This file contains all commands of the Chapter 9, "Data Wrangling and Visualisation"

```
# load the library
library(tidyverse)
# read the data
popsize <- read_tsv("../data/FauchaldEtAl2017/pop_size.csv")
ndvi <- read_tsv("../data/FauchaldEtAl2017/ndvi.csv")
seaice <- read_tsv("../data/FauchaldEtAl2017/sea_ice.csv")
snow <- read_tsv("../data/FauchaldEtAl2017/snow.csv")
# bring data into long format
seaice <- seaice %>% gather(Month, Cover, 3:14)
```

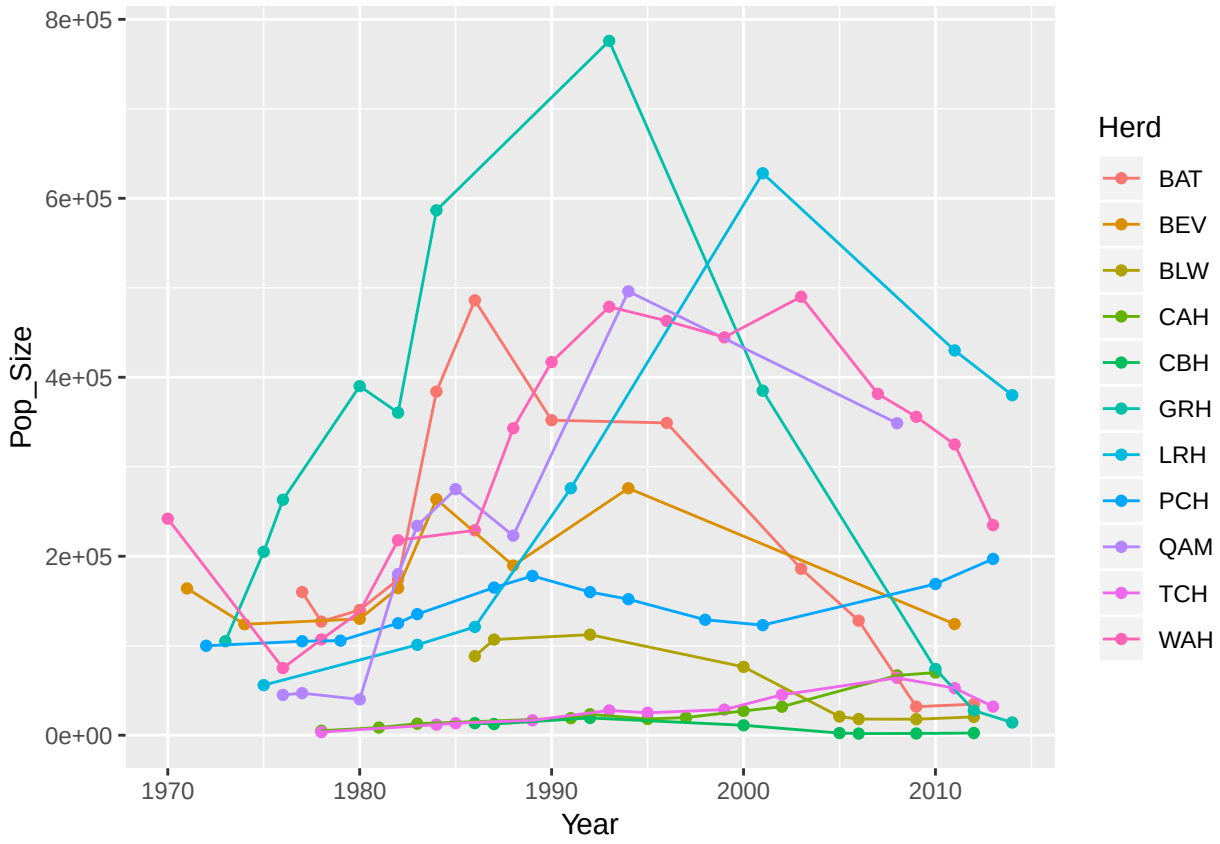
```
# build the first plot
ggplot(data = popsize)
```



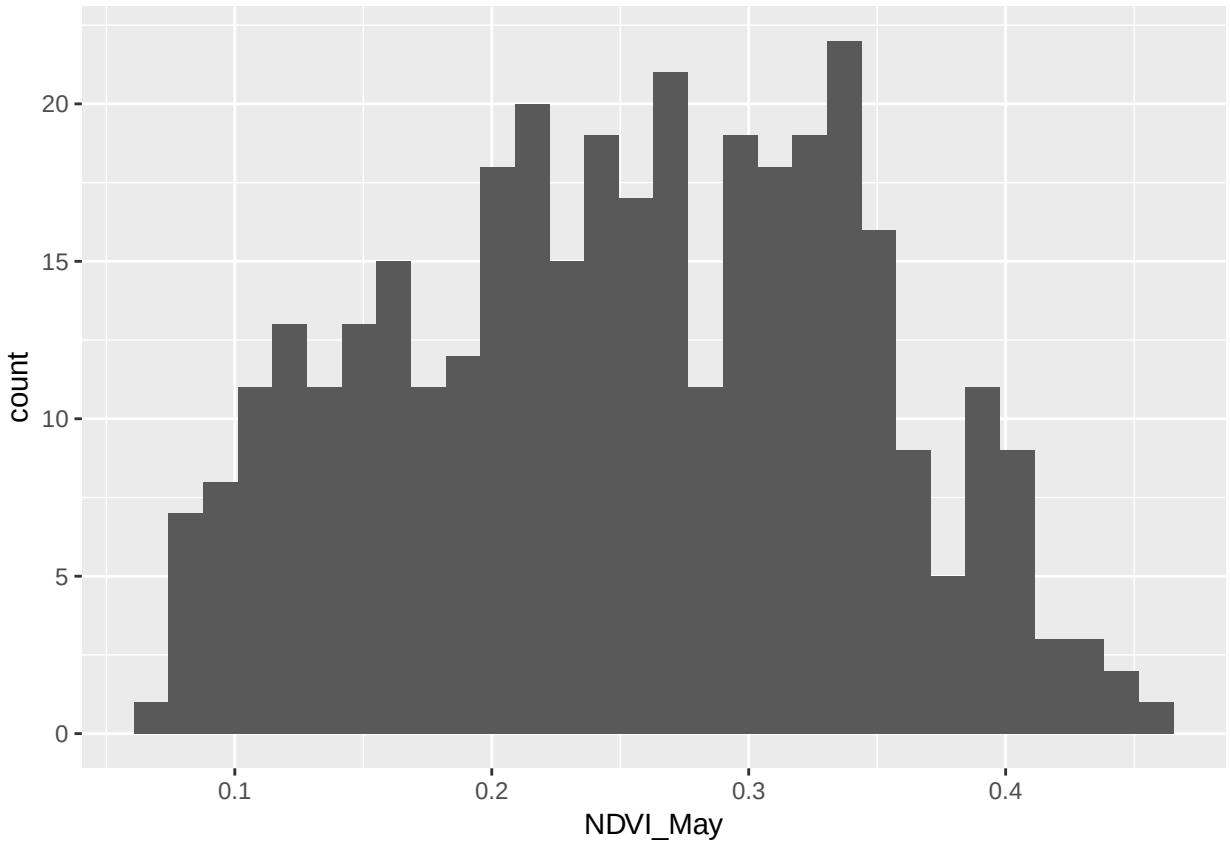
```
# add an aestethic mapping
ggplot(data = popsize) + aes(x = Year, y = Pop_Size, colour = Herd)
```



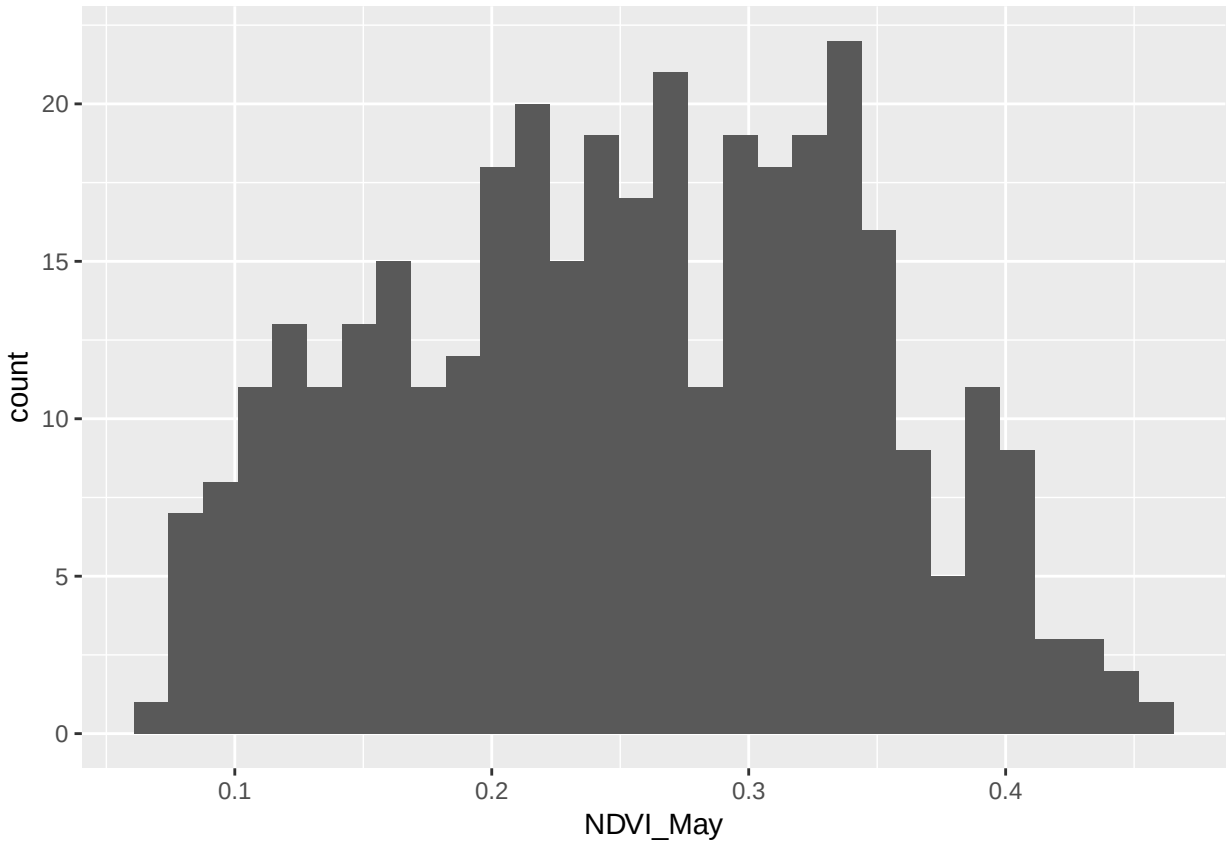

```
# add geometries
ggplot(data = popsize) +
    aes(x = Year, y = Pop_Size, colour = Herd) +
    geom_point() +
    geom_line()
```



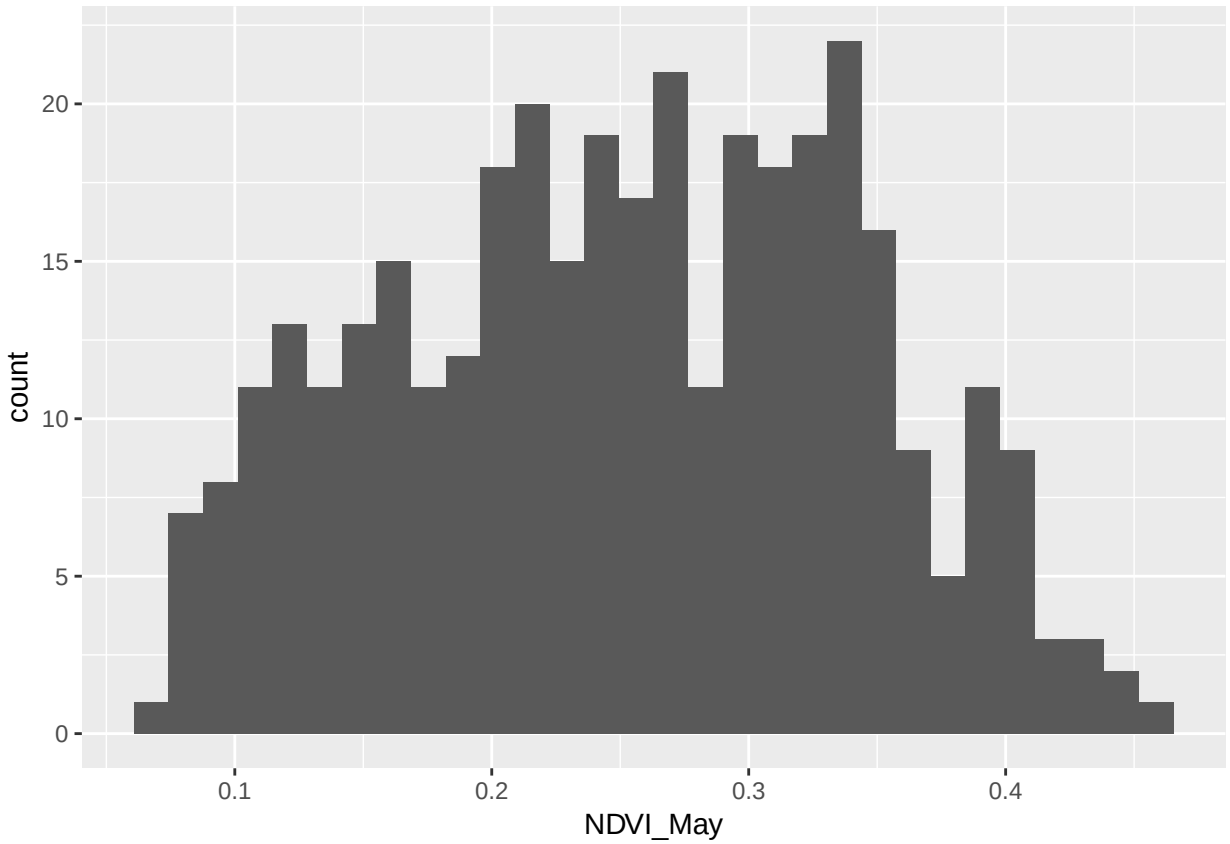

```
# plot frequency distribution (histogram)
ggplot(data = ndvi) +
  aes(x = NDVI_May) +
  geom_histogram()
```



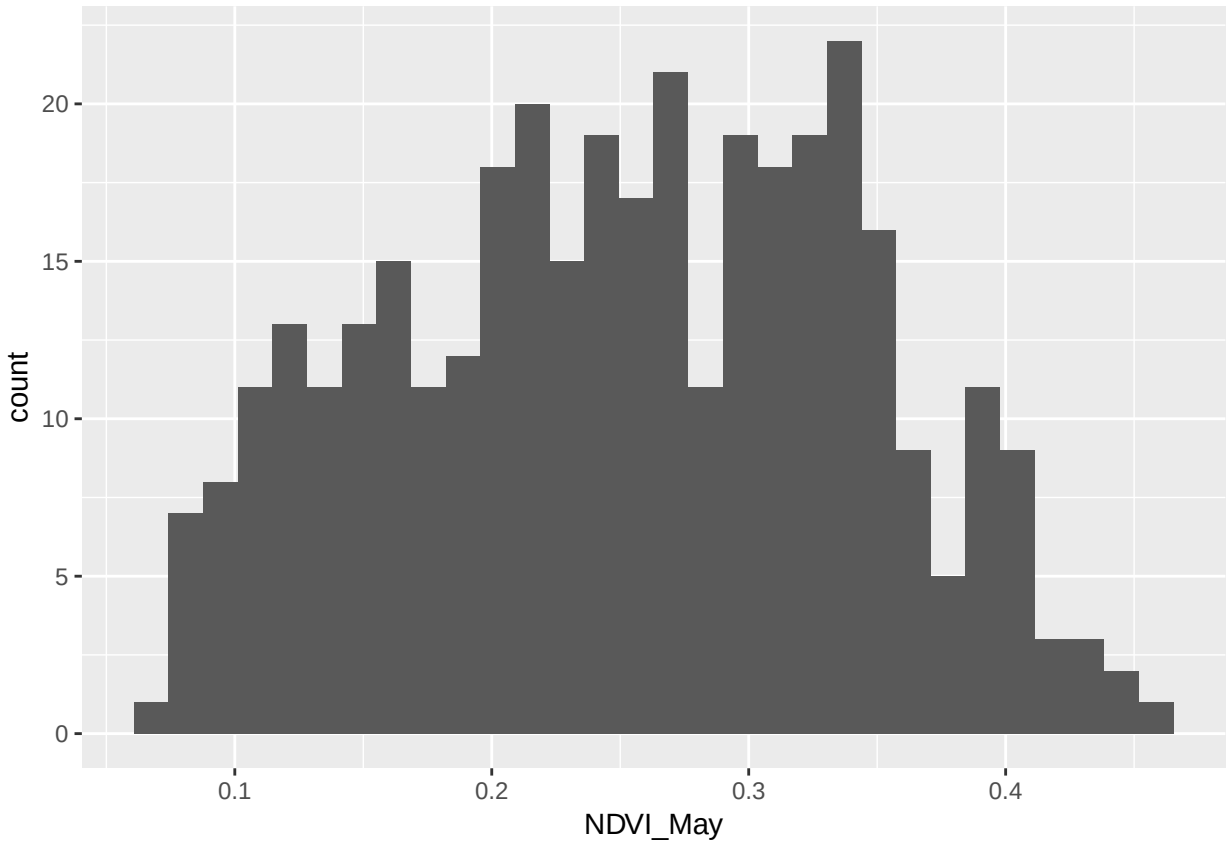

```
# "unrolled" or nested
# These three commands produce the same graph:
ggplot(data = ndvi) +
  aes(x = NDVI_May) +
  geom_histogram()
```



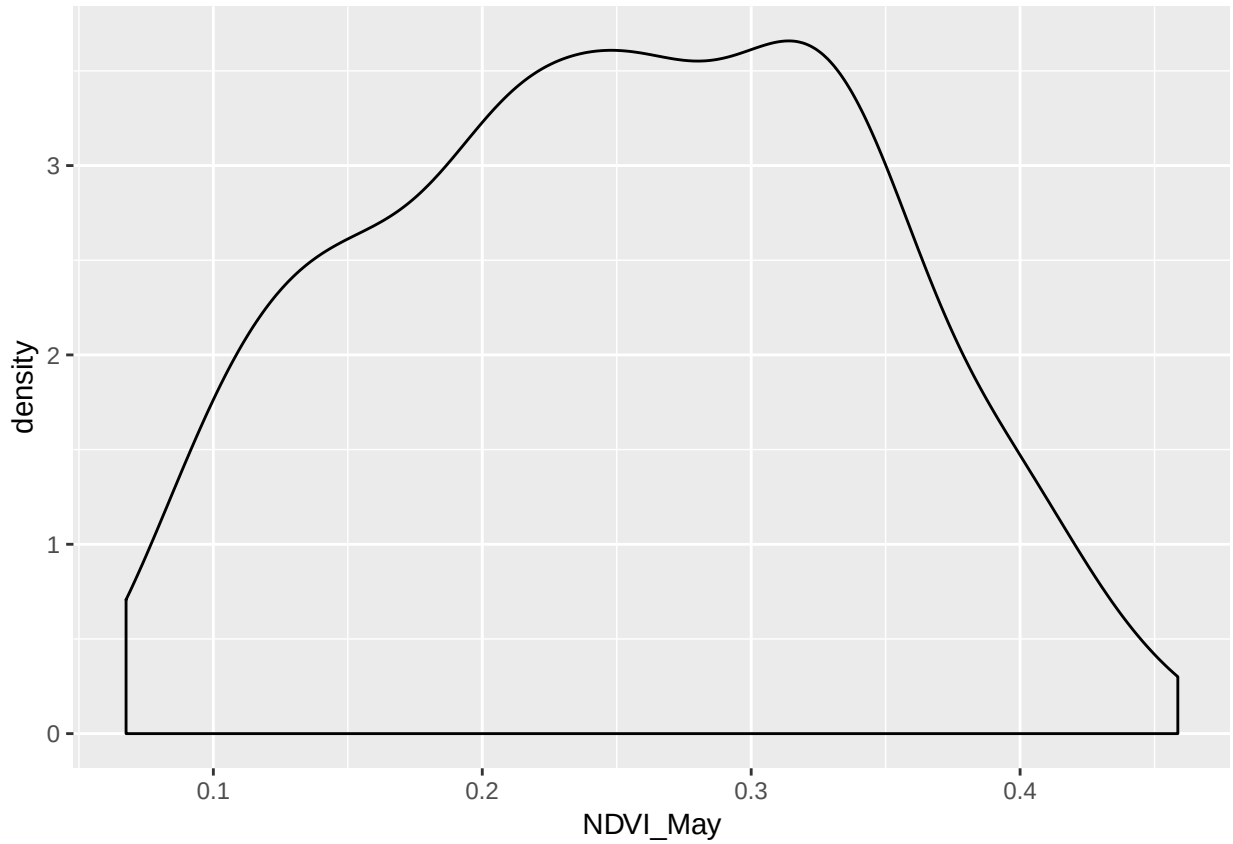

```
ggplot(data = ndvi, aes(x = NDVI_May)) +
  geom_histogram()
```



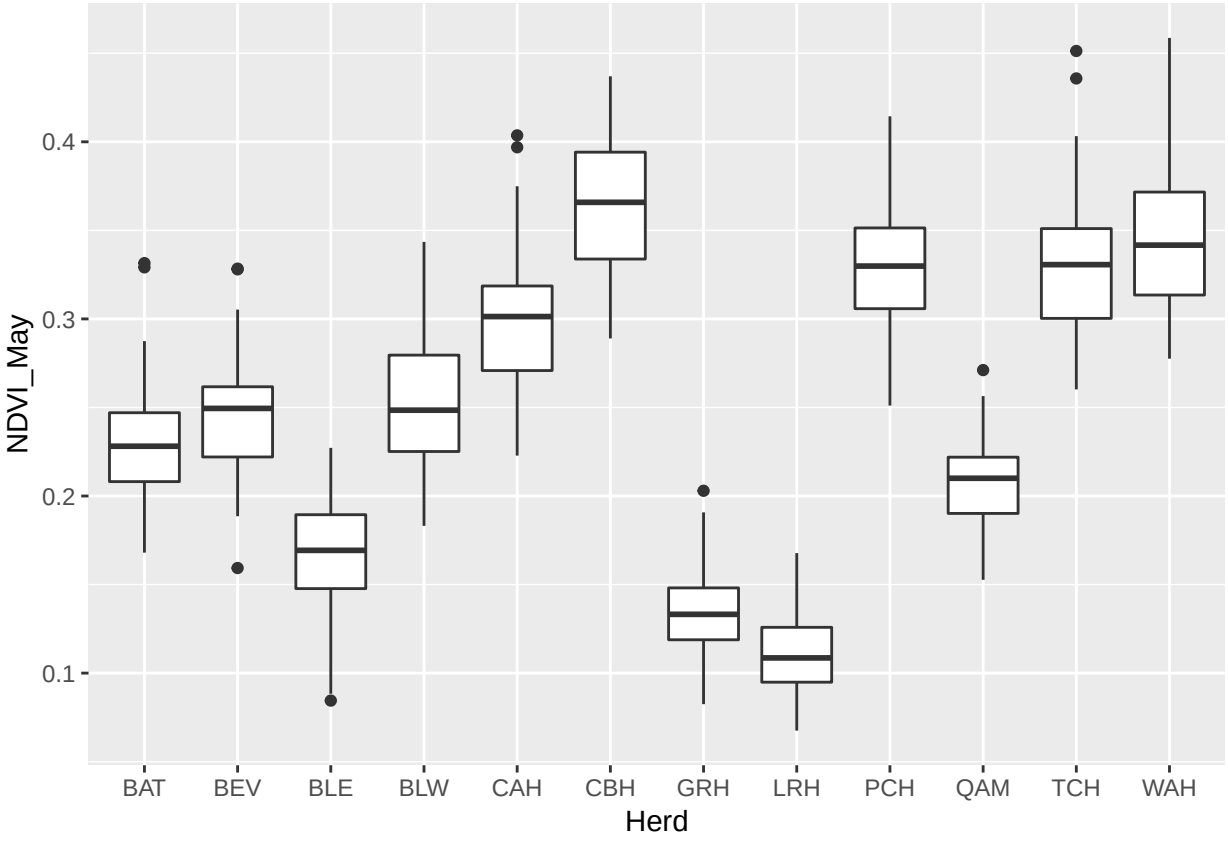

```
ggplot(data = ndvi) +
  geom_histogram(aes(x = NDVI_May))
```



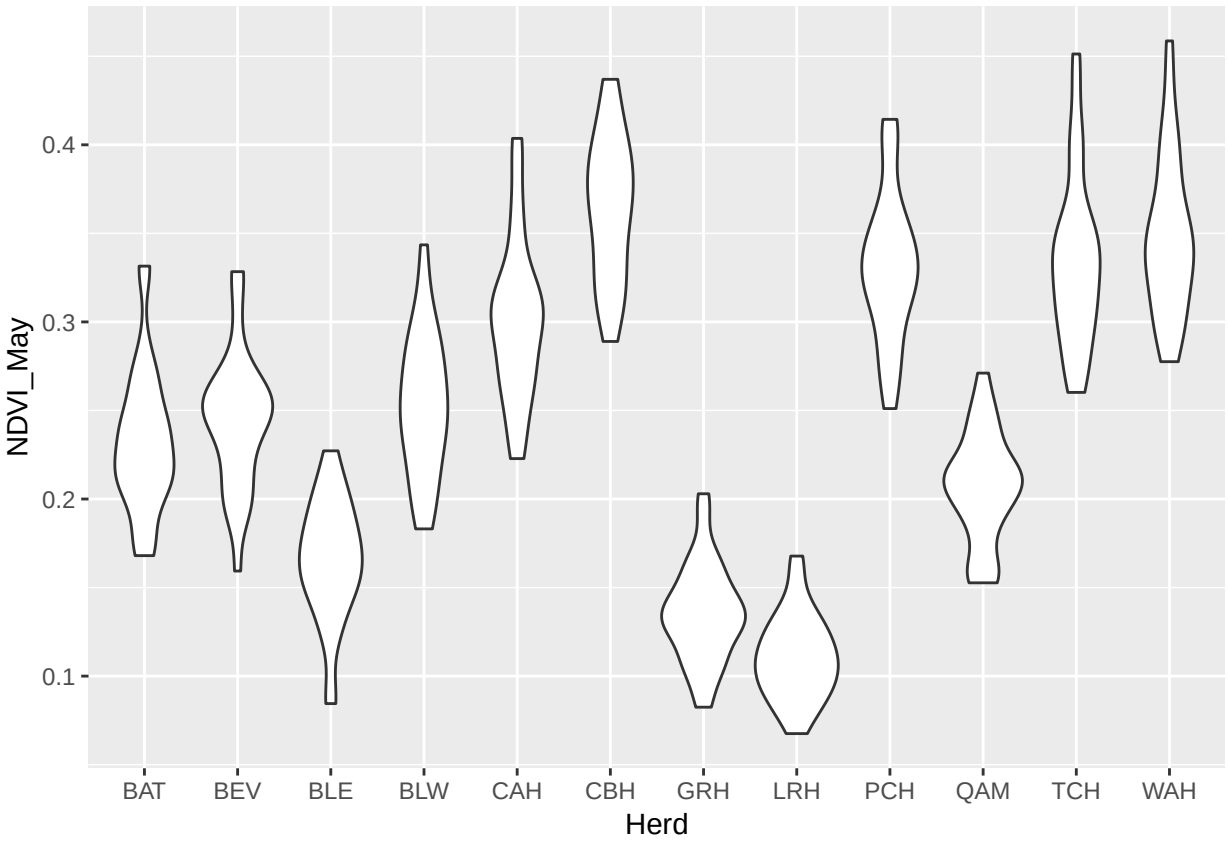

```
# density plot
ggplot(data = ndvi) +
  aes(x = NDVI_May) +
  geom_density()
```



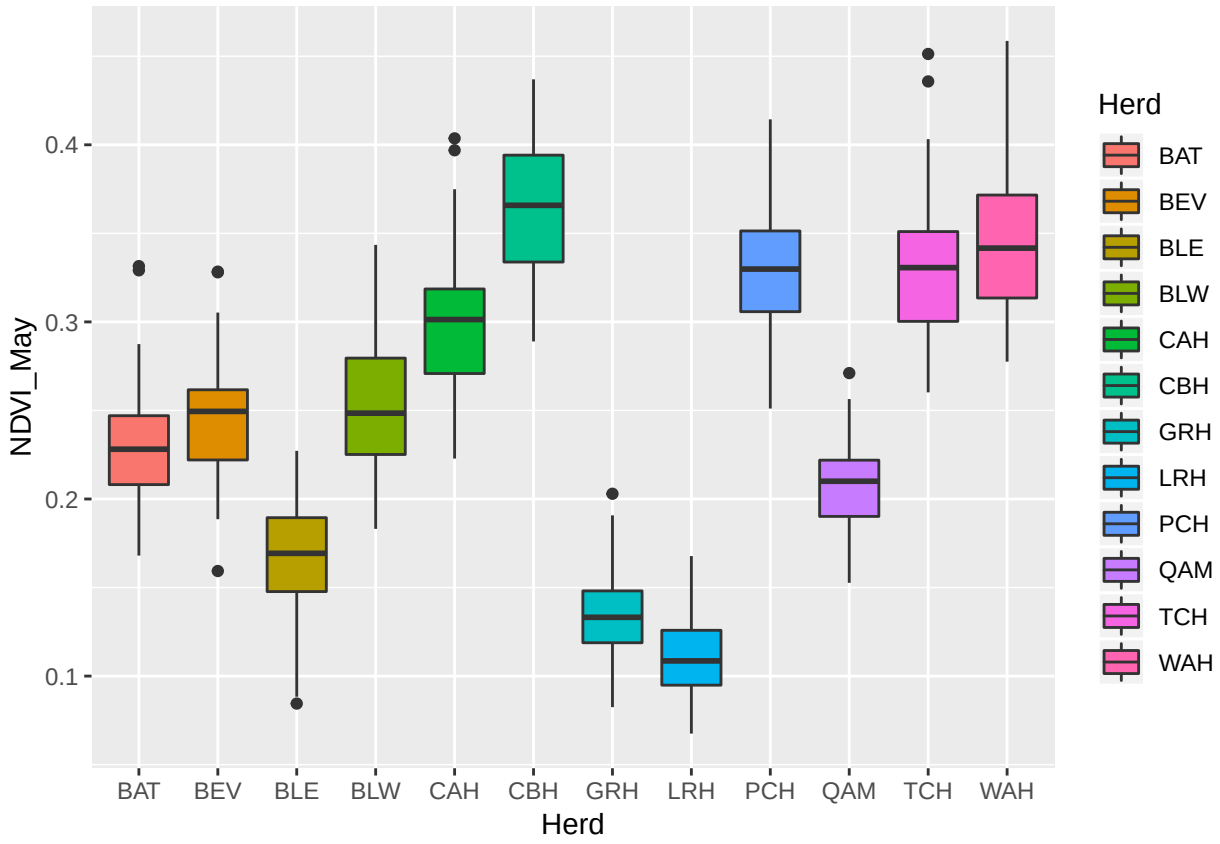

```
# assign plot to a variable
pl <- ggplot(data = ndvi) + aes(x = Herd, y = NDVI_May)
# add components to existing plot
pl + geom_boxplot()
```



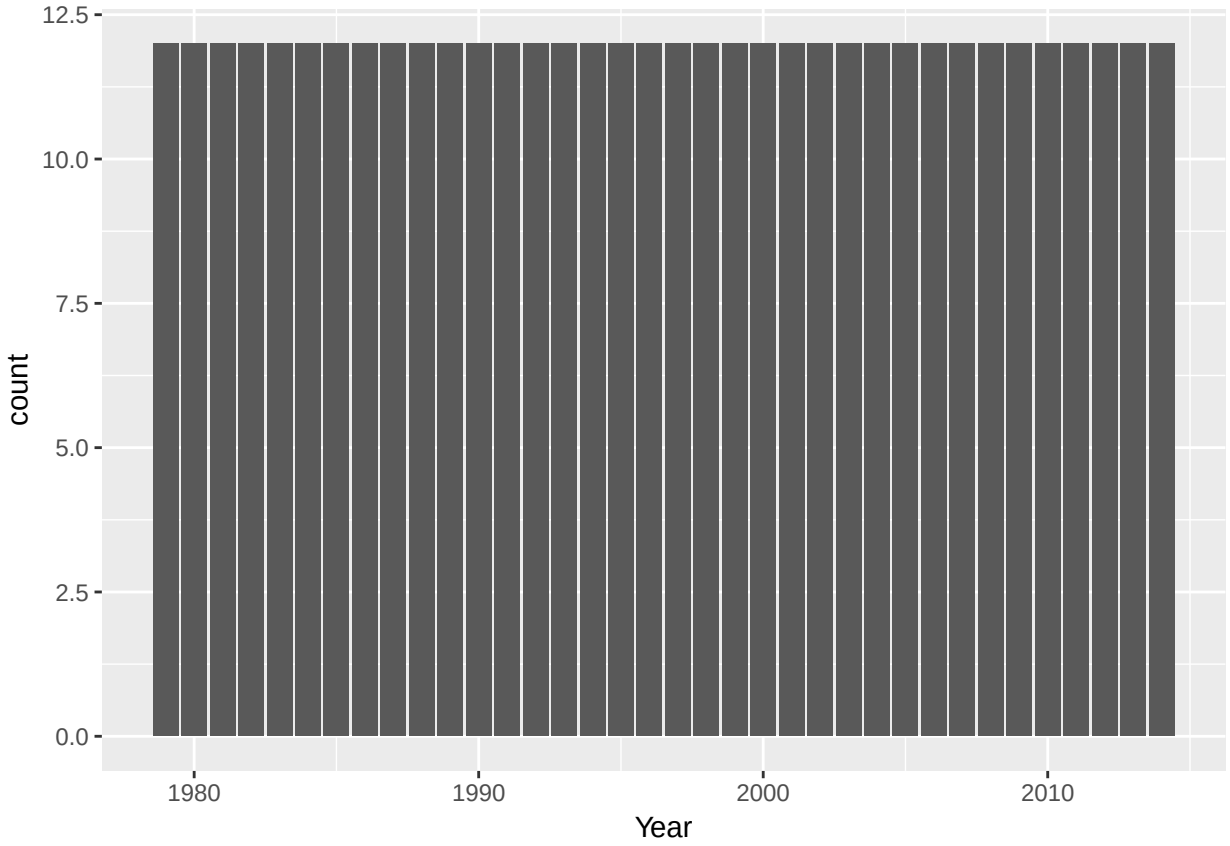

```
pl + geom_violin()
```



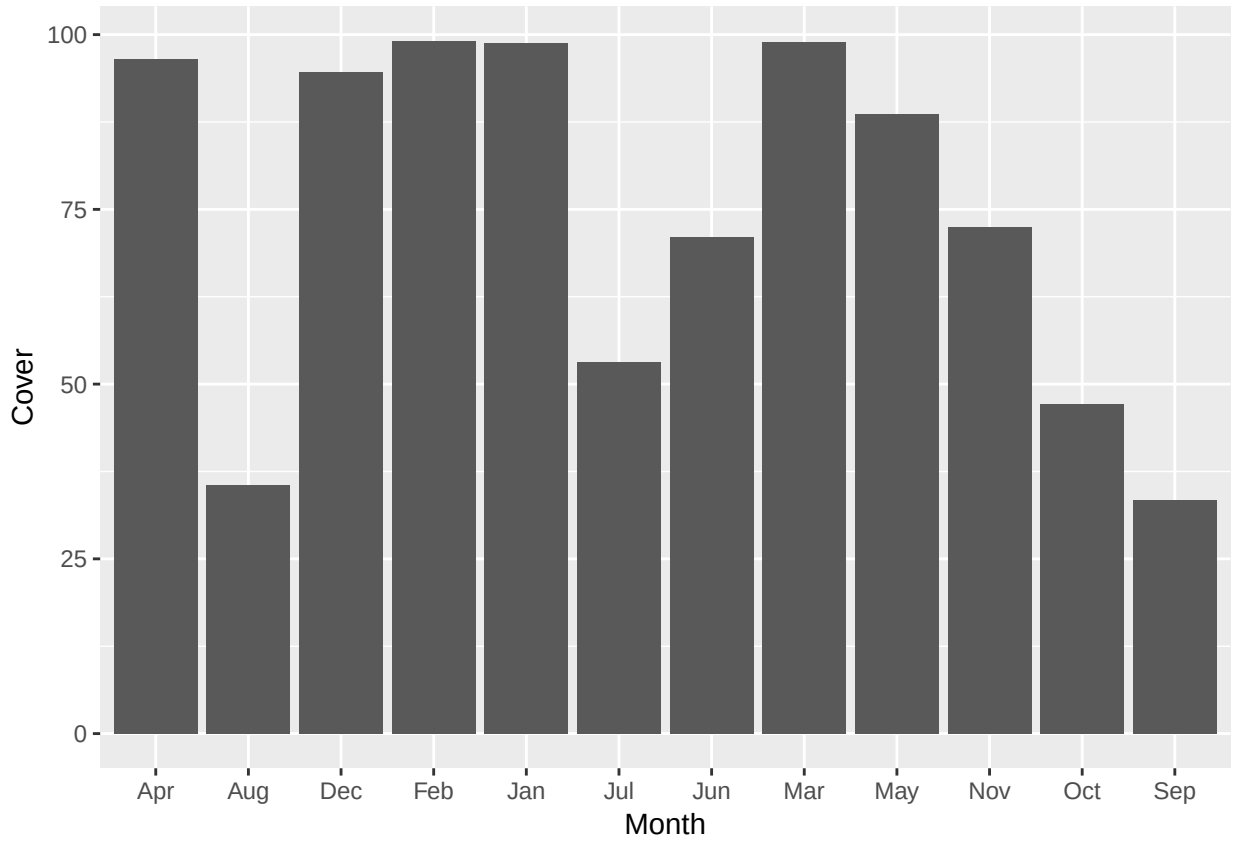

```
# change color of boxes
pl + geom_boxplot() + aes(fill = Herd)
```



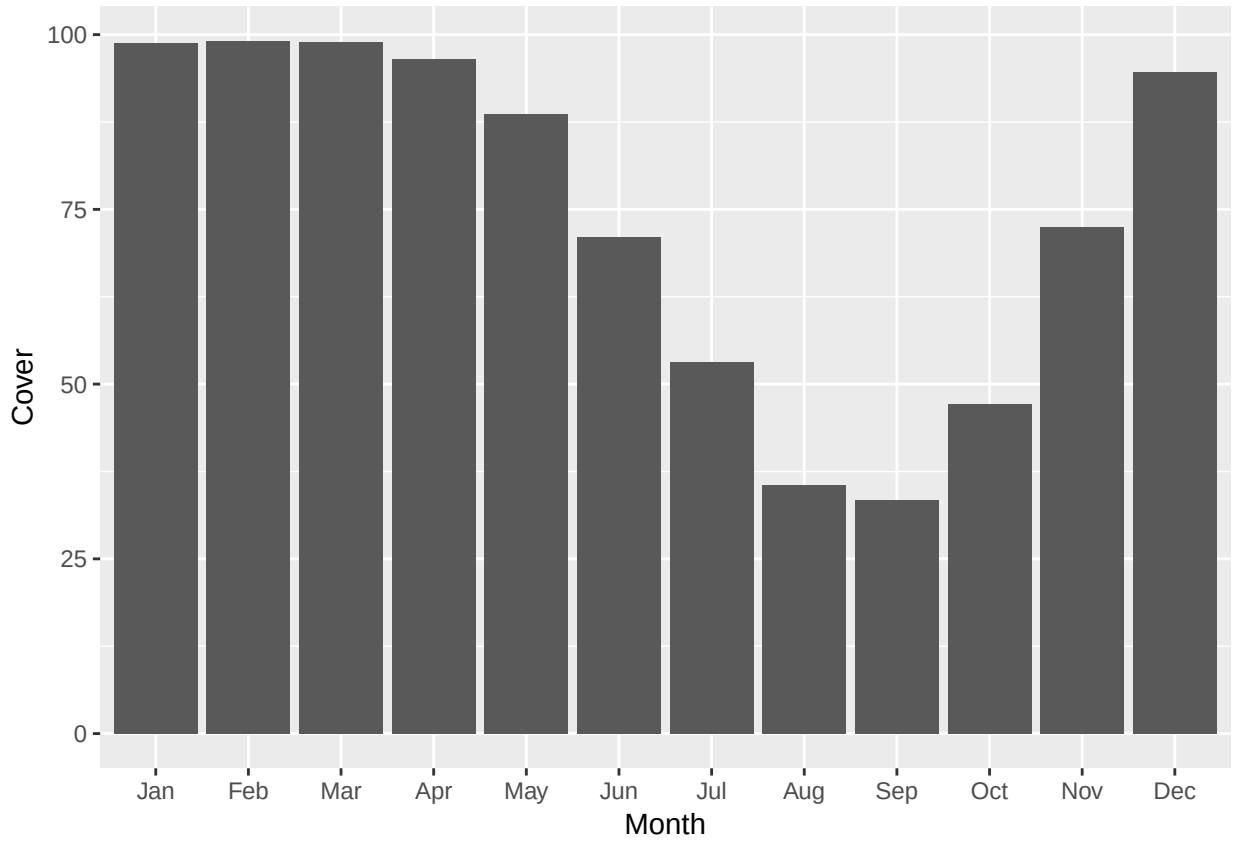

```
# barplot (count data)
ggplot(data = seaice %>% filter(Herd == "WAH")) +
  aes(x = Year) +
  geom_bar()
```



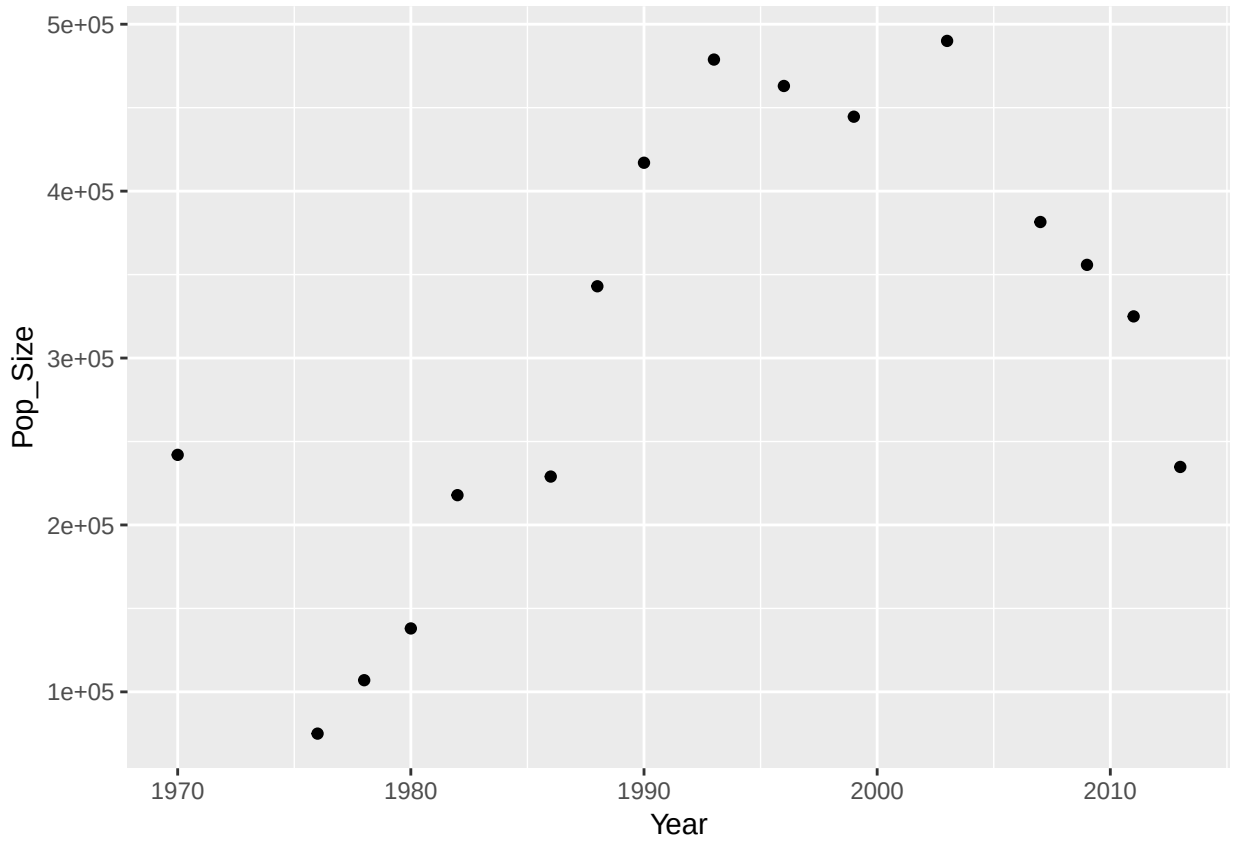

```
# map data to columns (note alphabetical order of x-axis)
ggplot(data = seaice %>%
         filter(Herd == "WAH", Year == 1990)) +
  aes(x = Month, y = Cover) +
  geom_col()
```



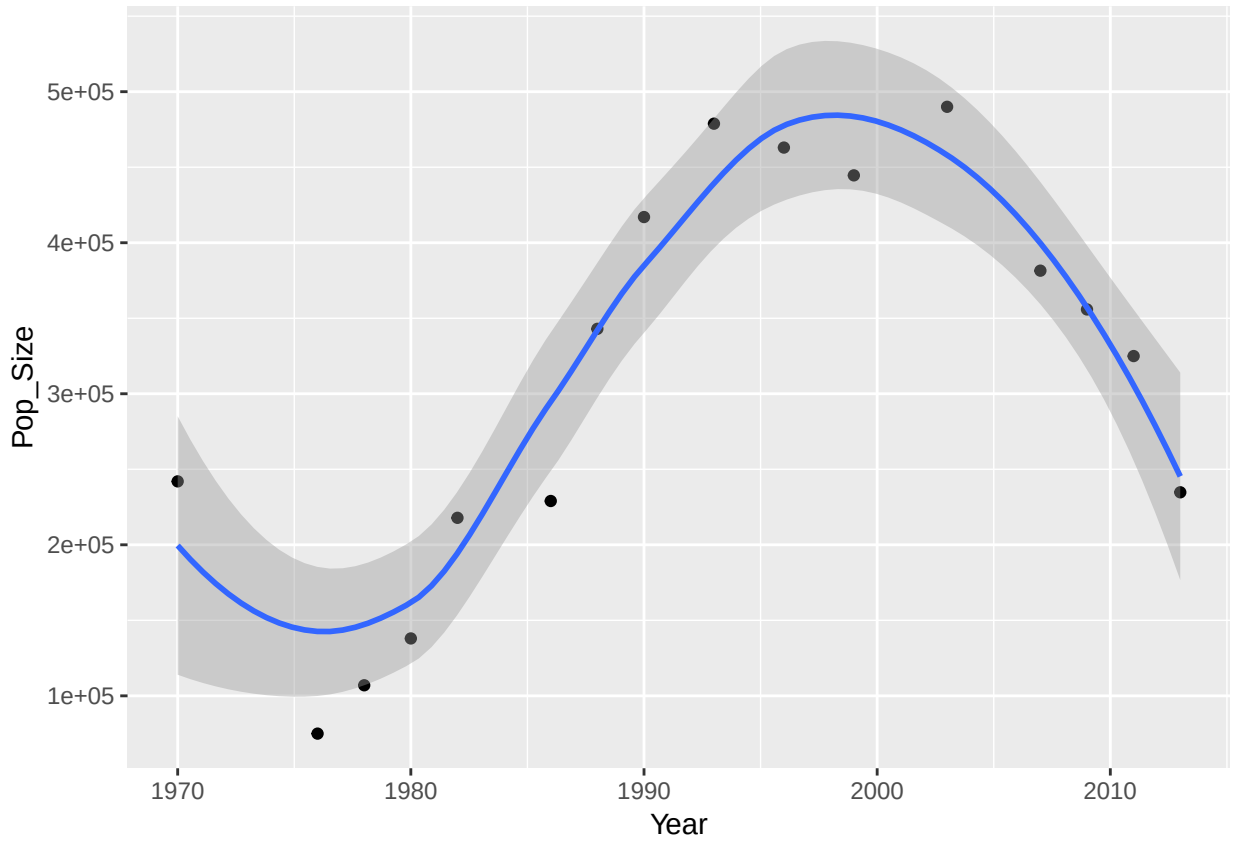

```
# display bars in chronolocigal order
# convert data into factor and set to three-letter abbreviation of months
seaice$Month <- factor(seaice$Month, month.abb)
ggplot(data = seaice %>%
         filter(Herd == "WAH", Year == 1990)) +
  aes(x = Month, y = Cover) +
  geom_col()
```



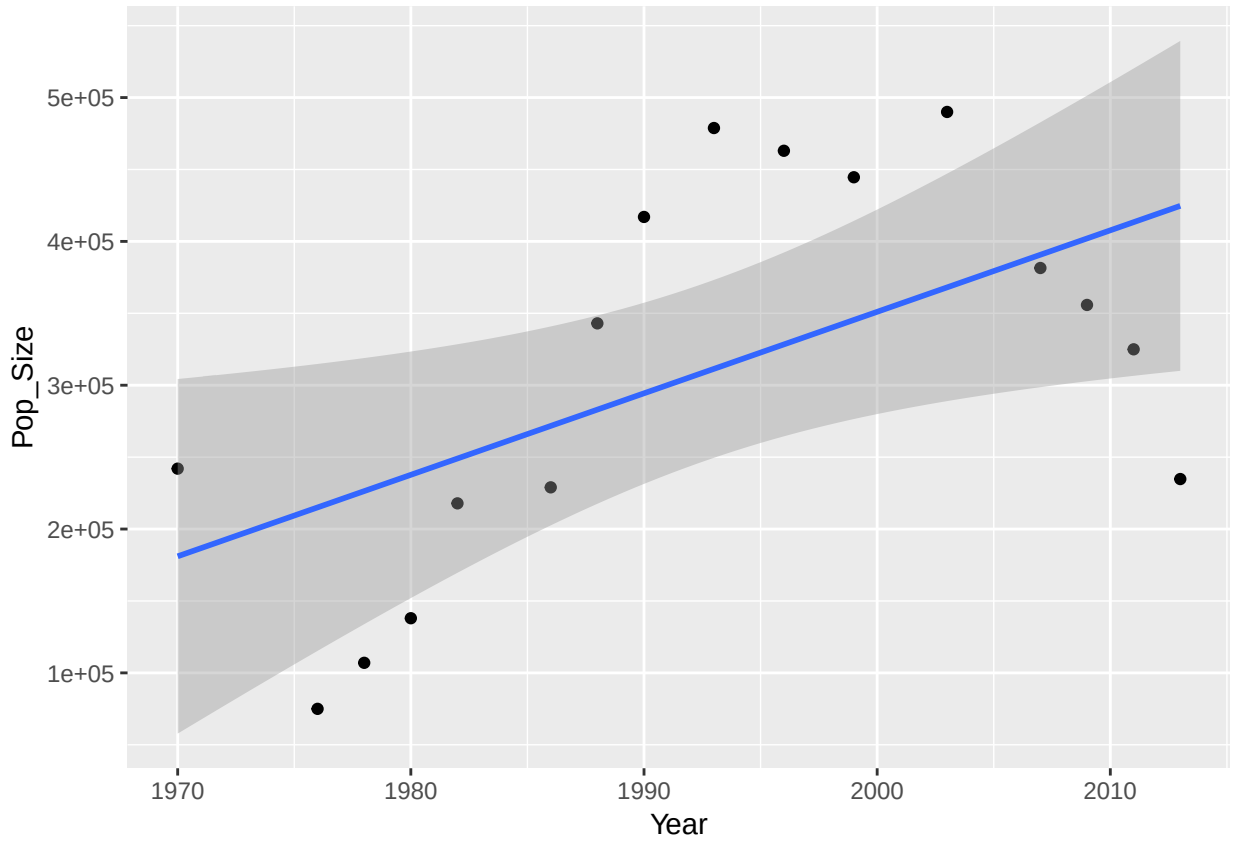

```
# scatterplots
pl <- ggplot(data = popsize %>%
               filter(Herd == "WAH")) +
  aes(x = Year, y = Pop_Size) +
  geom_point()
# show plot assigned to variable
show(pl)
```



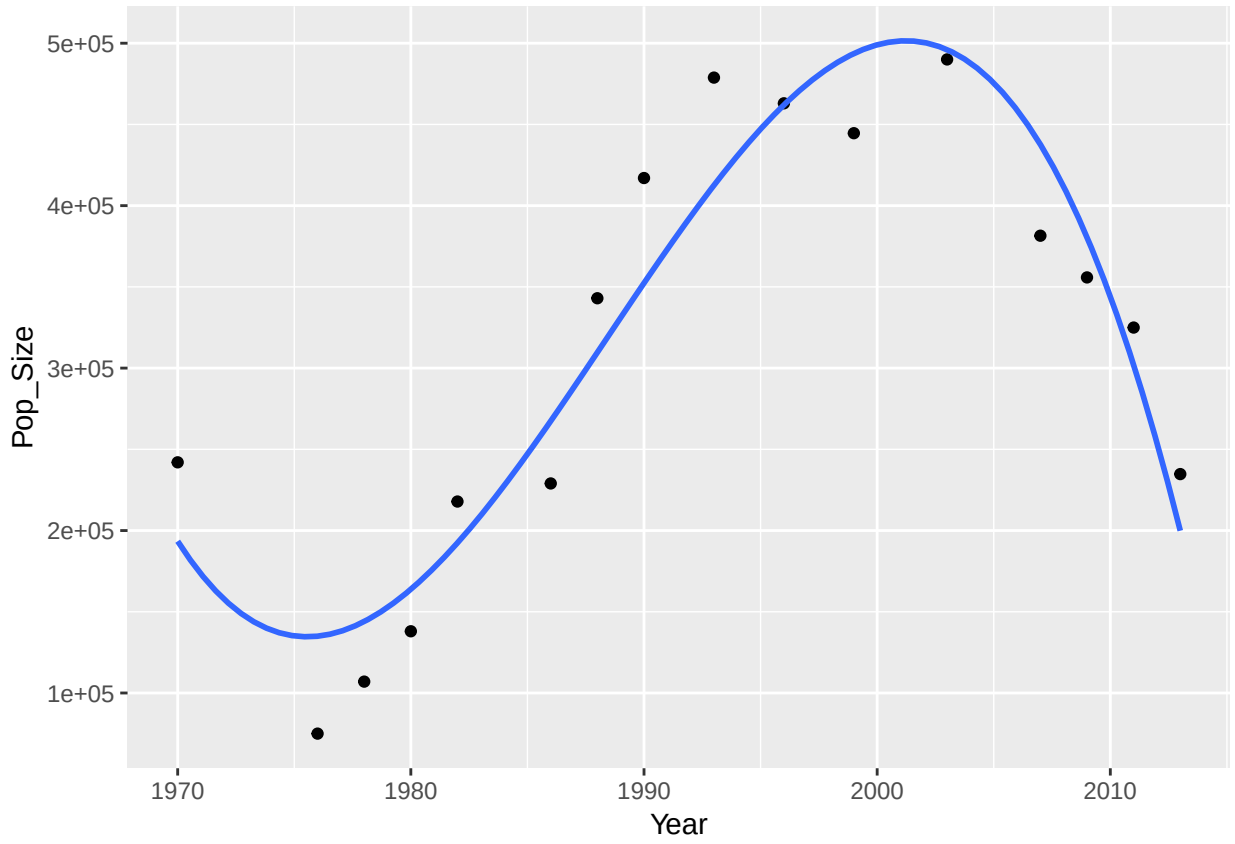

```
# add smoothing funtion
pl + geom_smooth()
```



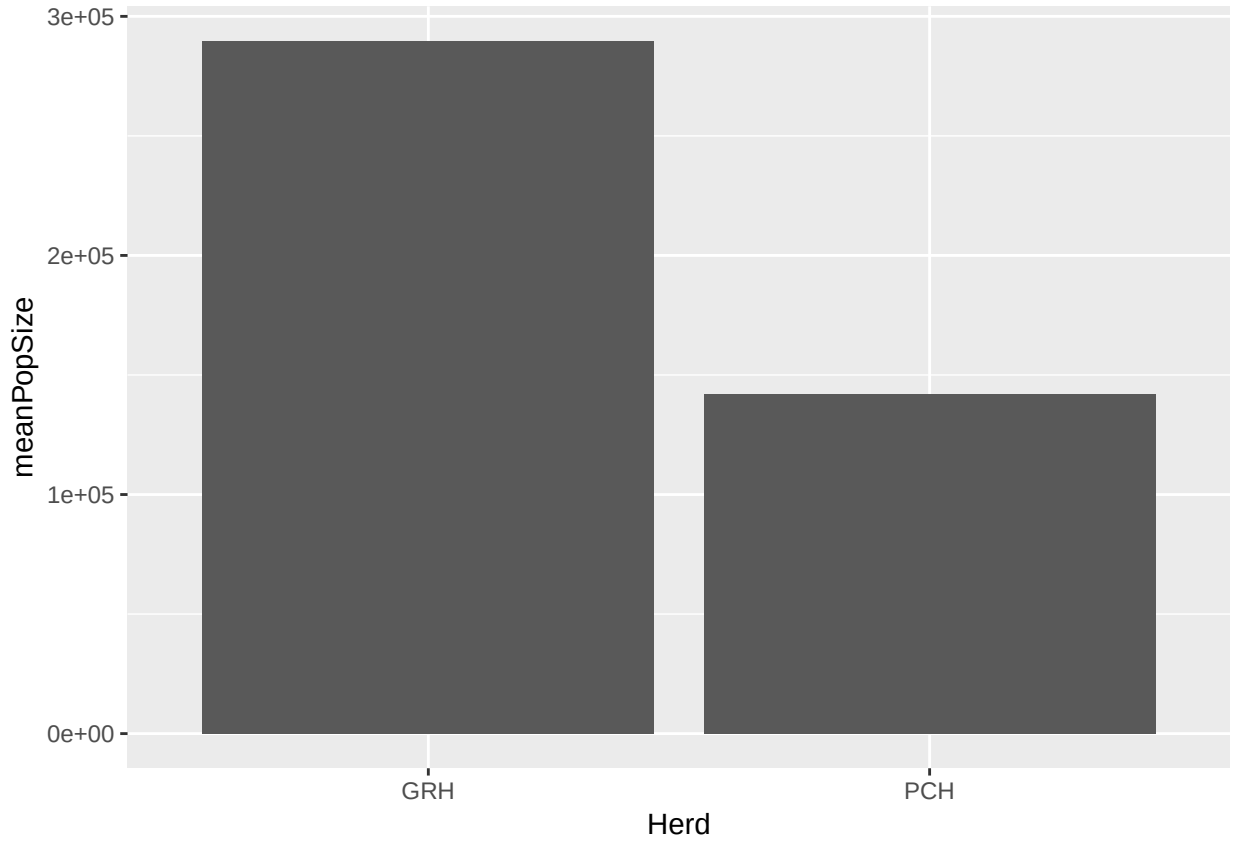

```
# use a linear model
pl + geom_smooth(method = "lm")
```



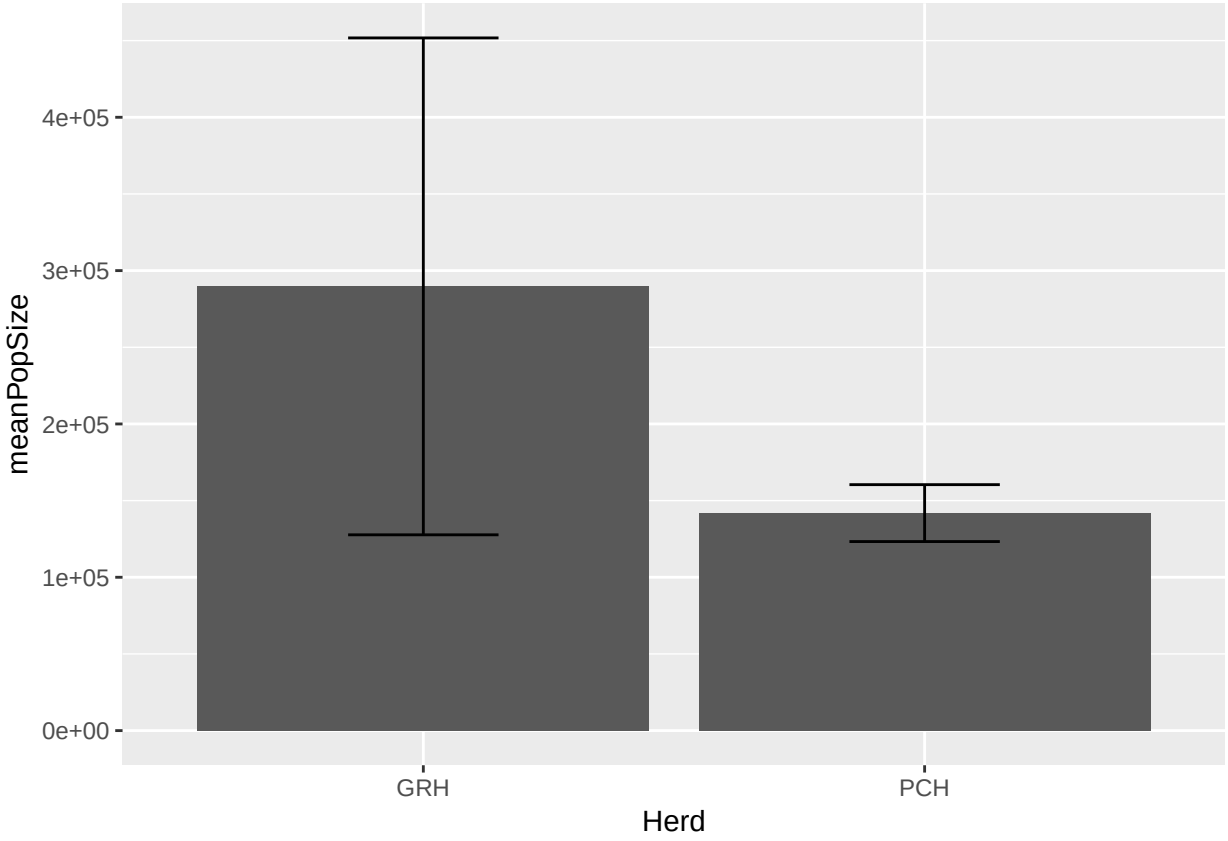

```
# use a polynomial regression
pl + geom_smooth(method = "lm",
                 formula = y ~ poly(x, 3), se = FALSE)
```



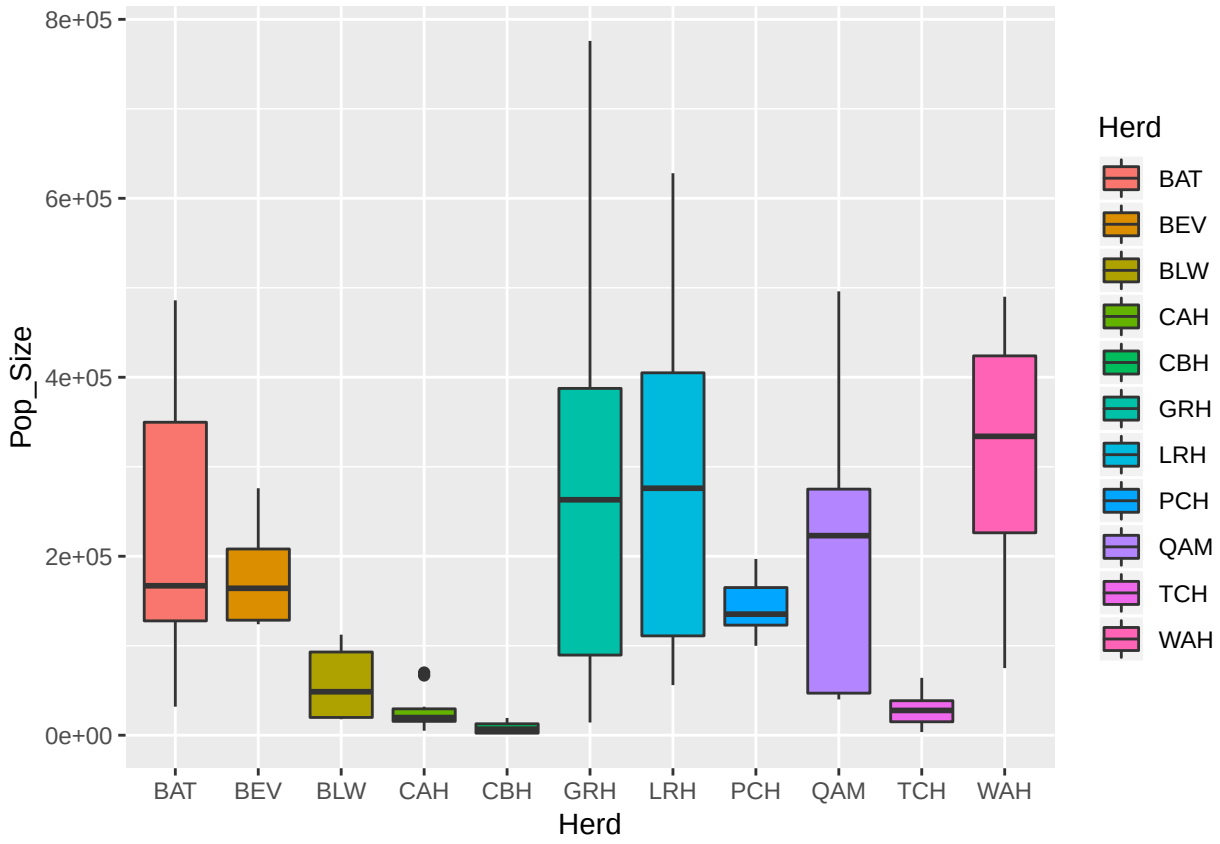

```
 # calculate summary stats and errors
stats <- popsize %>% filter(Herd %in% c("GRH", "PCH")) %>%
          group_by(Herd) %>%
          summarise(
                meanPopSize= mean(Pop_Size),
                SD = sd(Pop_Size),
                N = n(),
                SEM = SD/sqrt(N),
                CI = SEM * qt(0.975, N-1))
# bar plot without error bars
ggplot(data = stats) +
    aes(x = Herd, y = meanPopSize) +
    geom_col()
```



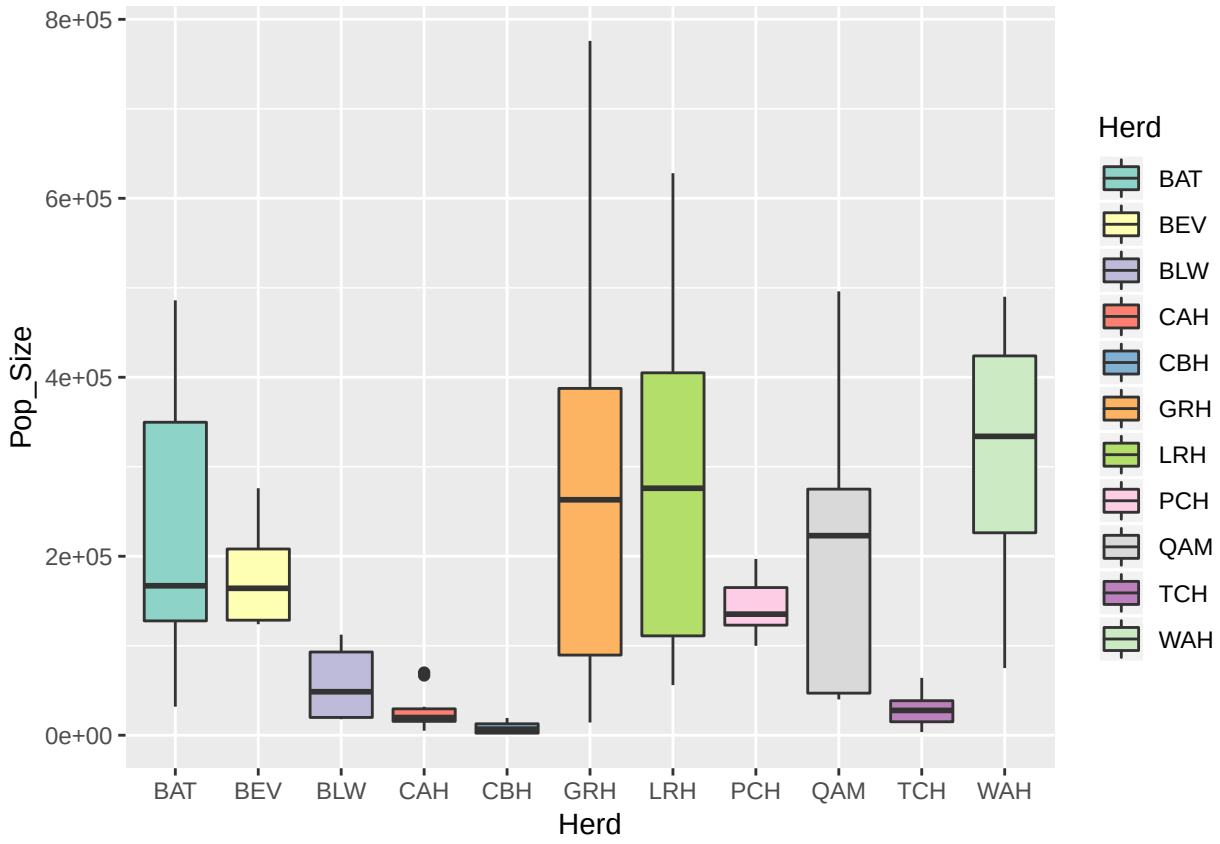

```
# set up aesthetic mapping for confidence intervals
limits <- aes(ymax = stats$meanPopSize + stats$CI,
              ymin = stats$meanPopSize - stats$CI)
# plot including confidence intervals
ggplot(data = stats) +
    aes(x = Herd, y = meanPopSize) +
    geom_col() +
    geom_errorbar(limits, width = .3)
```



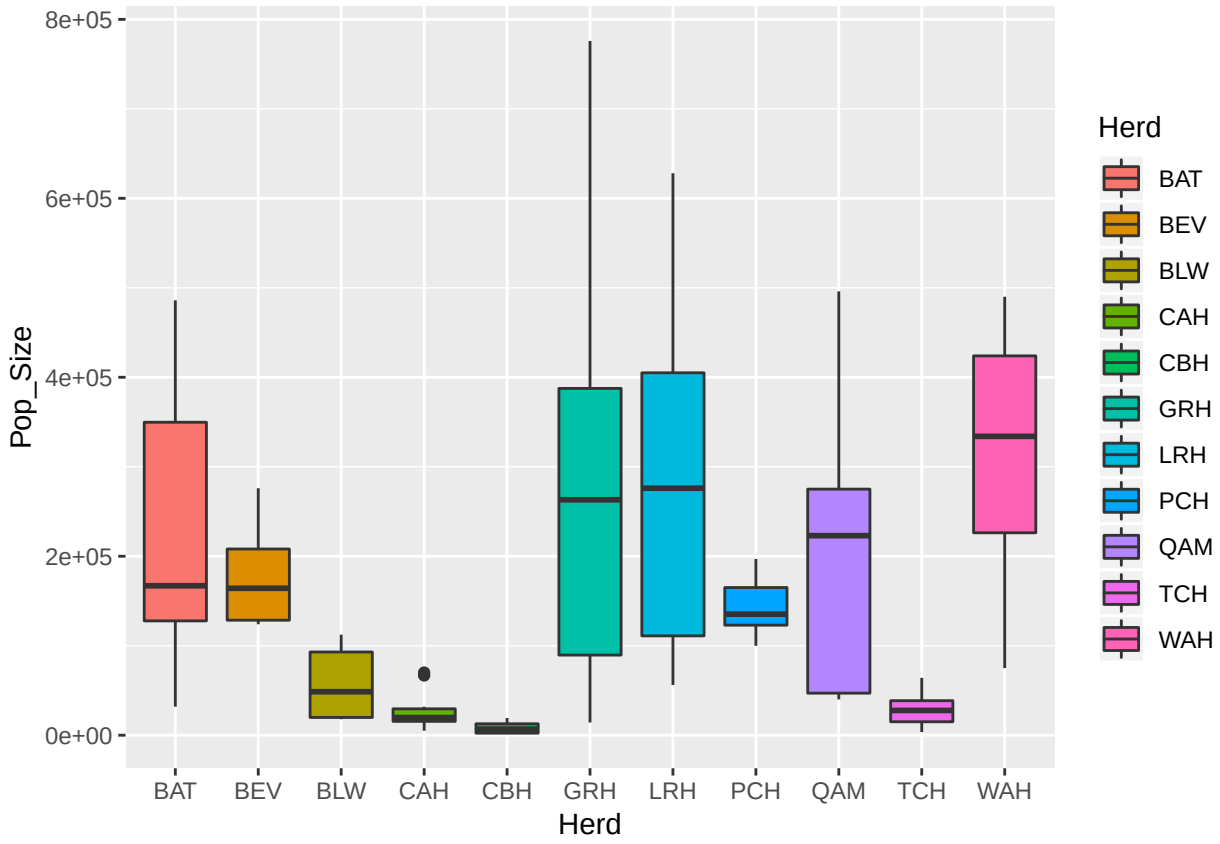

```
# set fill color of boxes by using scales
pl <- ggplot(data = popsize,
             aes(x = Herd, y = Pop_Size, fill = Herd)) +
  geom_boxplot()
show(pl)
```



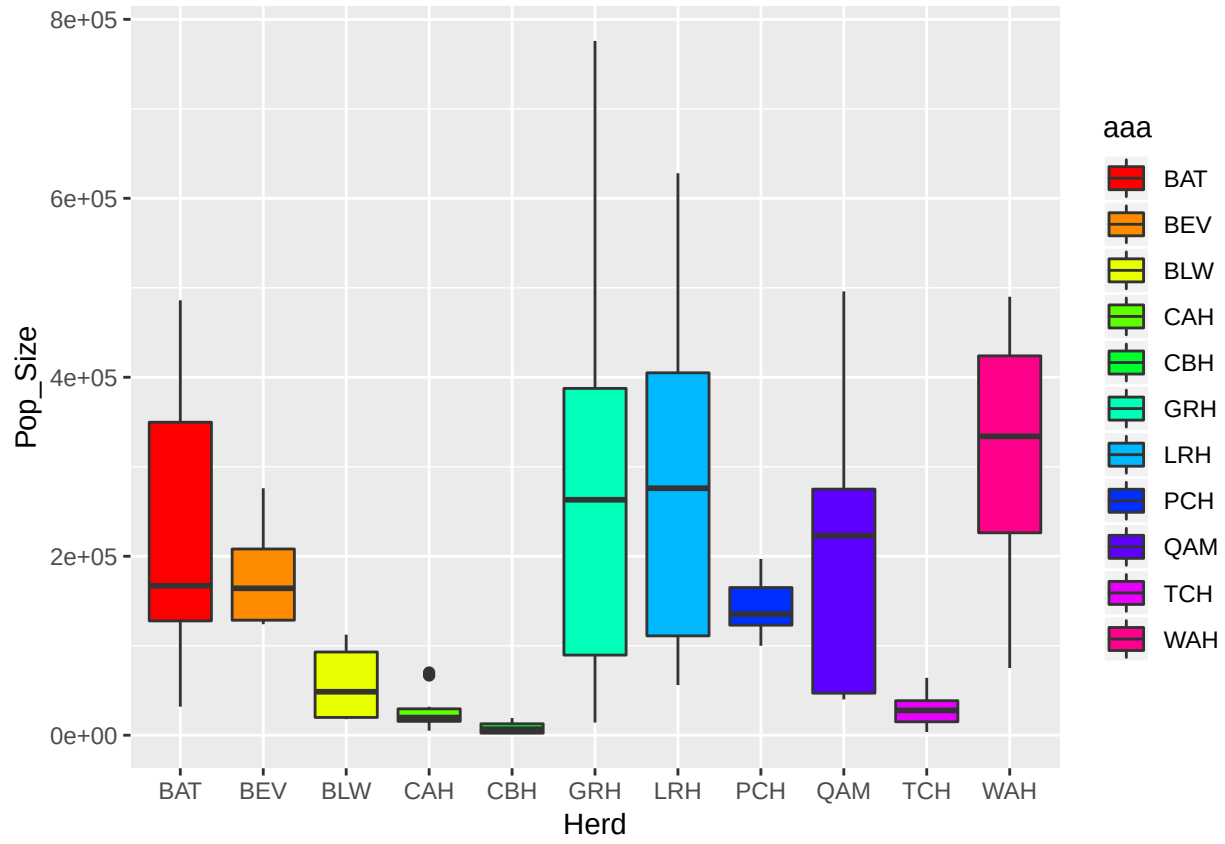

```
# choose a palette from Color Brewer
pl + scale_fill_brewer(palette = "Set3")
```



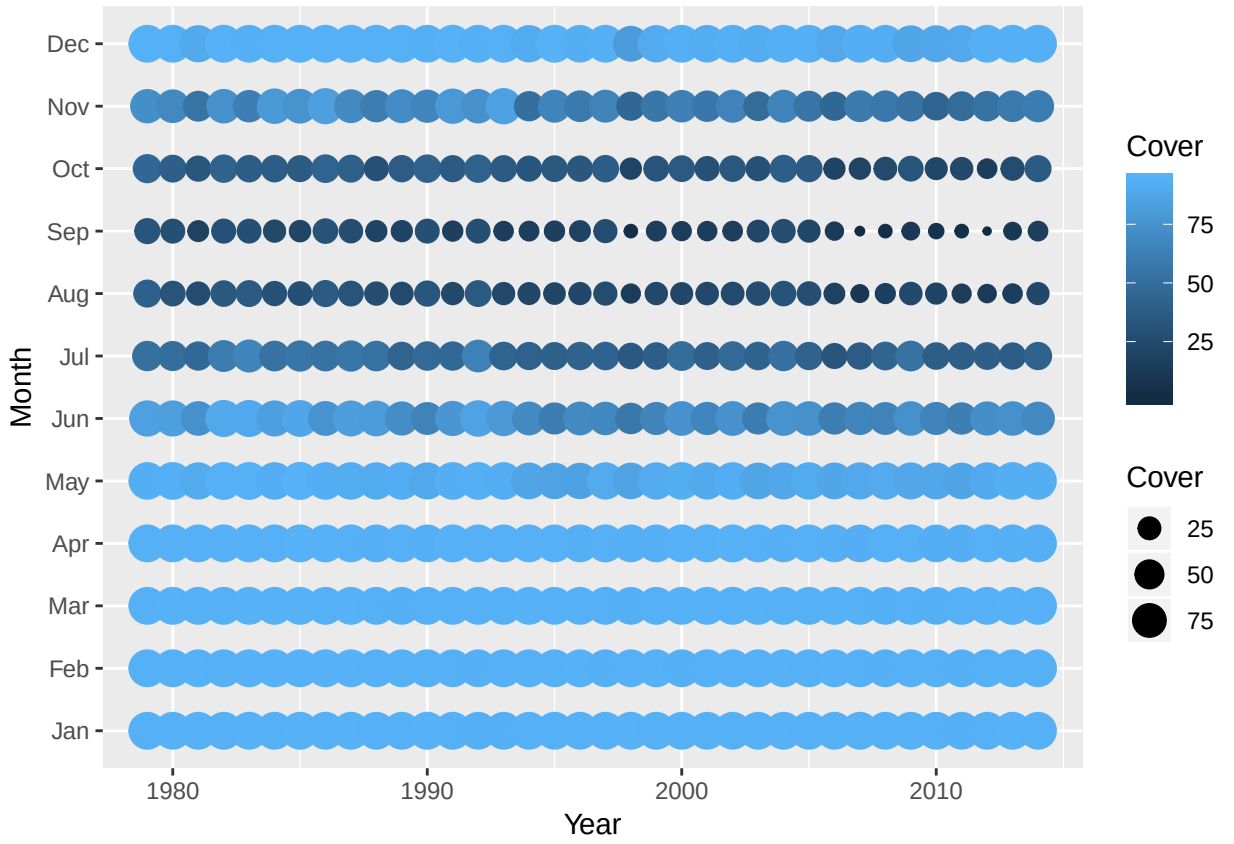

```
# palette based on hue
pl + scale_fill_hue()
```



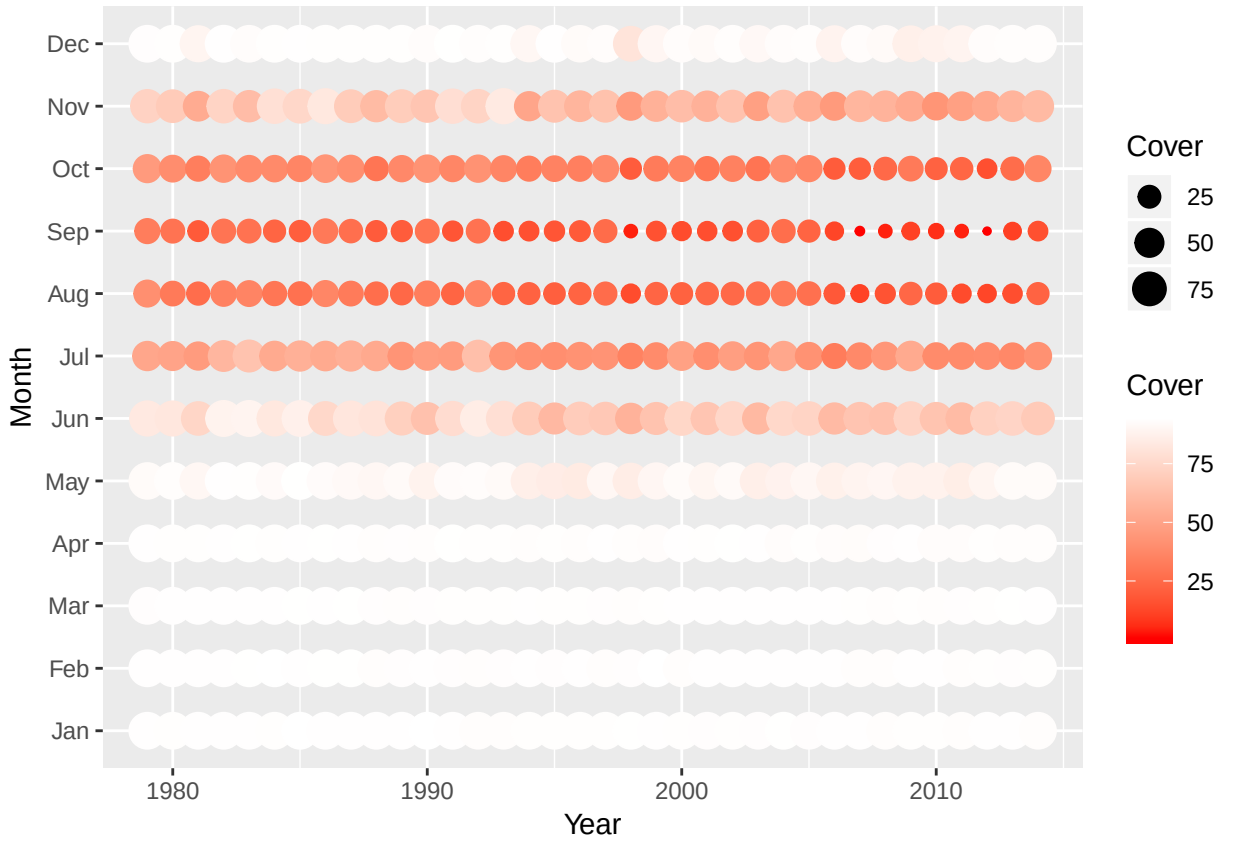

```
# manually set values and rename the legend
pl + scale_fill_manual(values = rainbow(11),
                       name = "aaa")
```



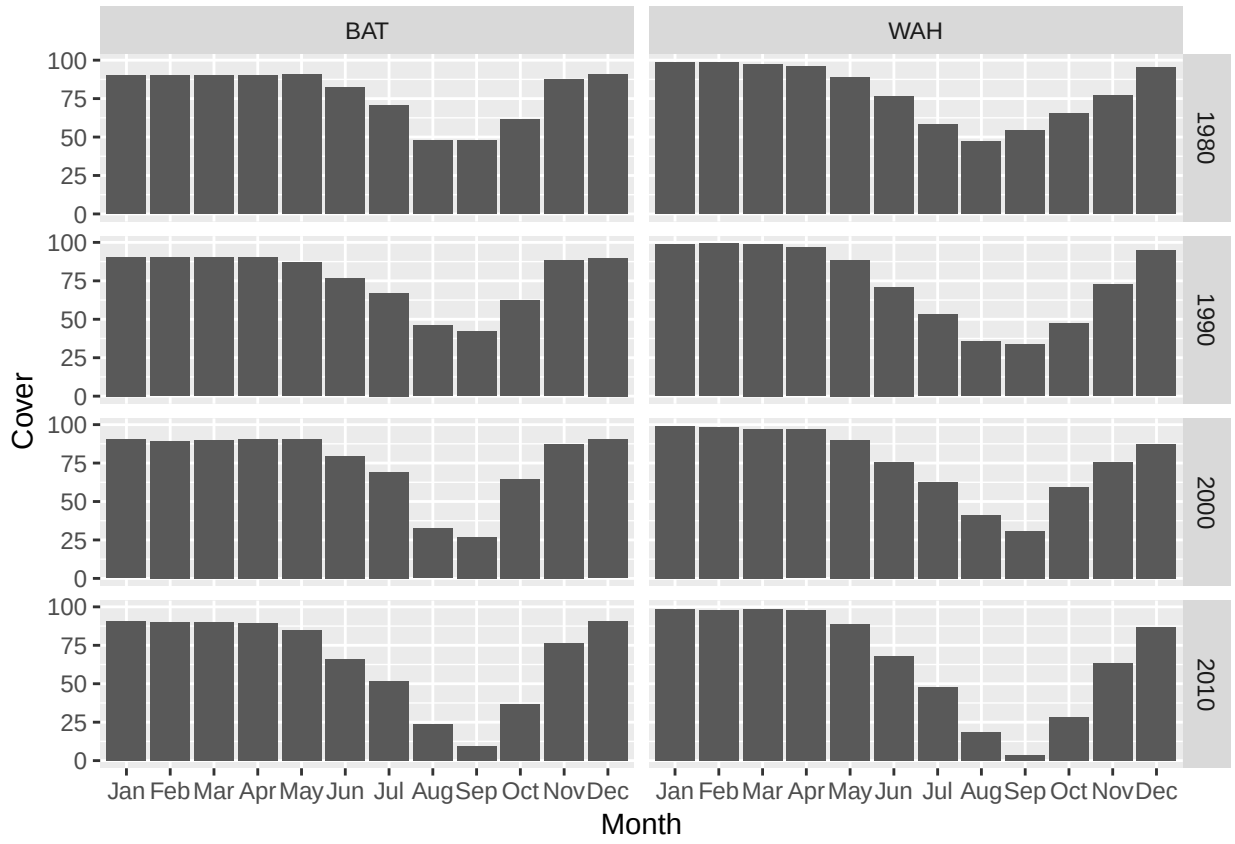

```
# apply scales to manipulate color and size of aestethic mappings
pl <- ggplot(data = seaice %>% filter(Herd == "BEV")) +
  aes(x = Year, y = Month, colour = Cover,
      size = Cover) +
  geom_point()
show(pl)
```



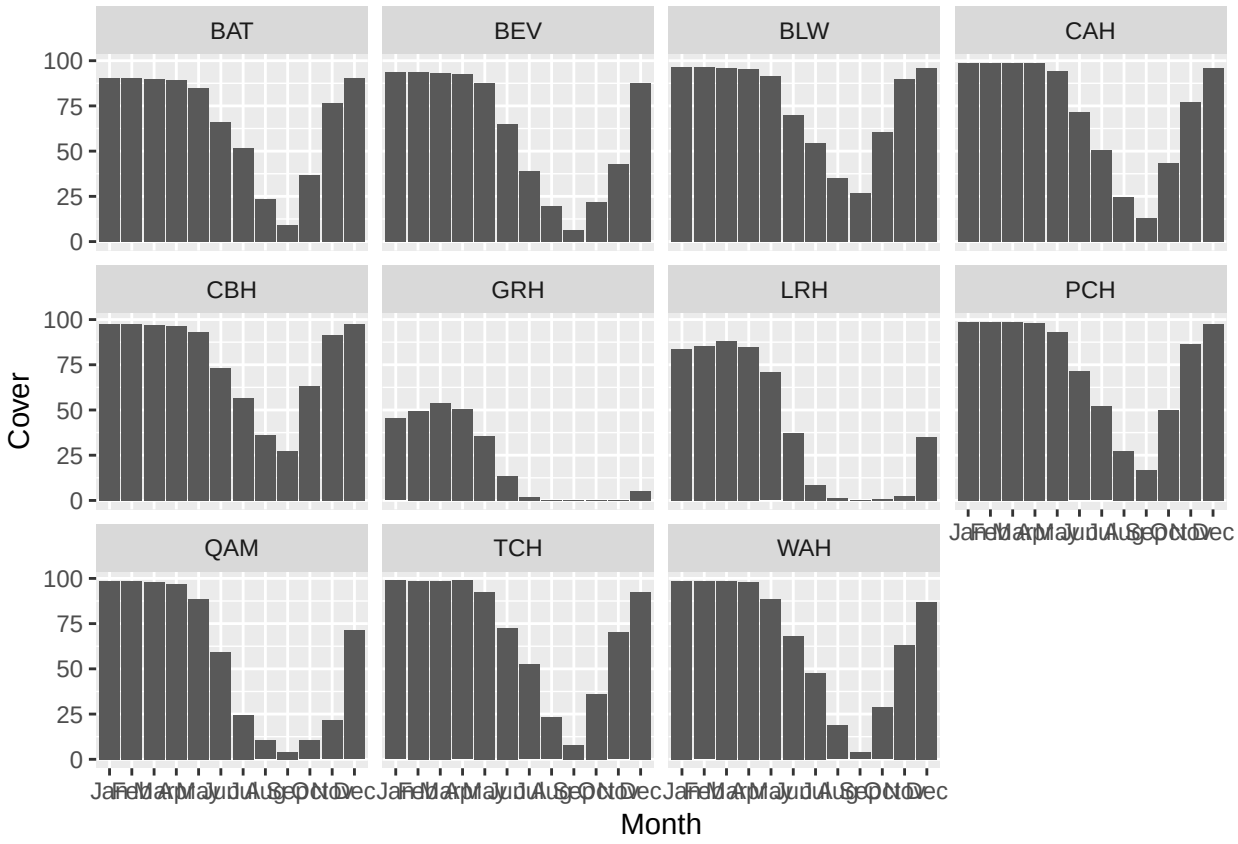

```
# change color of continuous gradient
pl + scale_color_gradient(high = "white", low = "red")
```



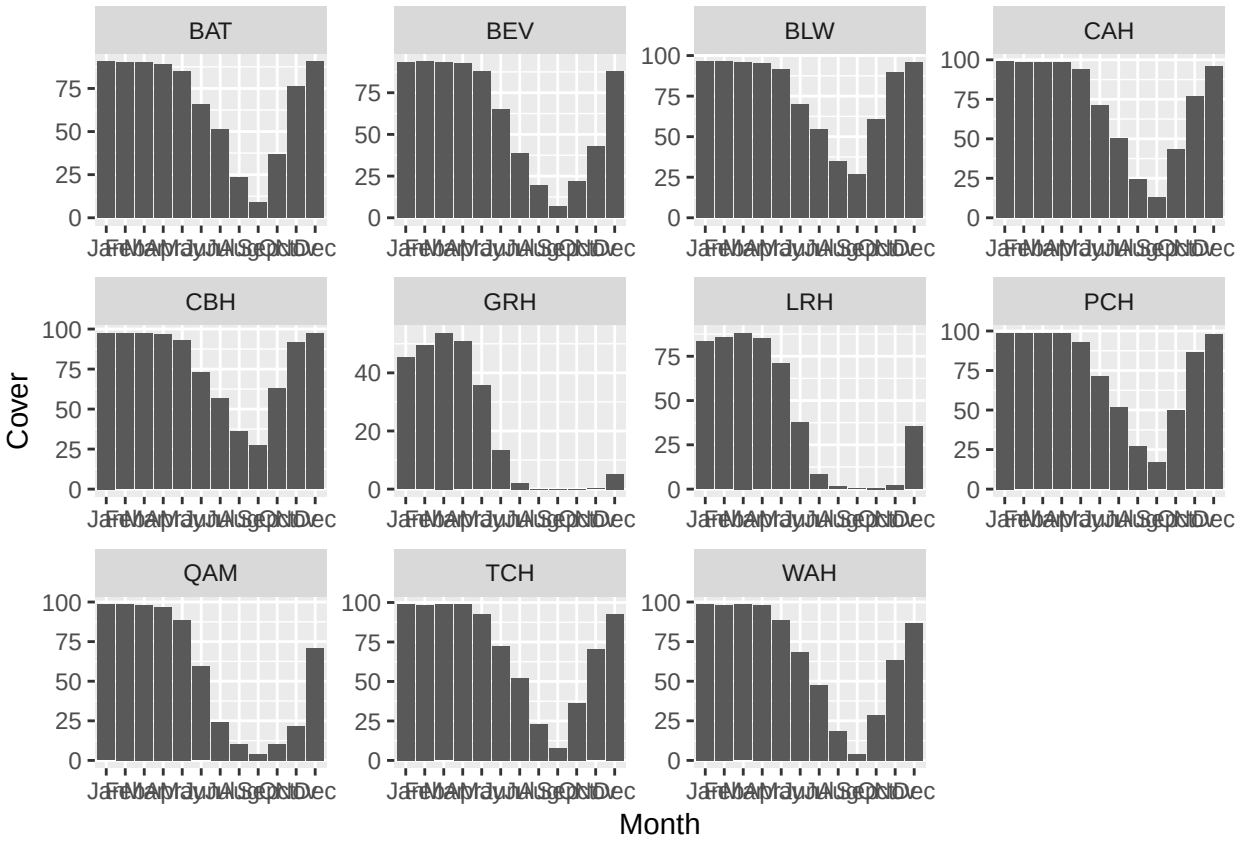

```
# facetting with identical scale of axis, including missing data
ggplot(data = seaice %>%
         filter(Herd %in% c("WAH", "BAT"),
                Year %in% c(1980, 1990, 2000, 2010))) +
  aes(x = Month, y = Cover) +
  geom_col() +
  facet_grid(Year~Herd)
```



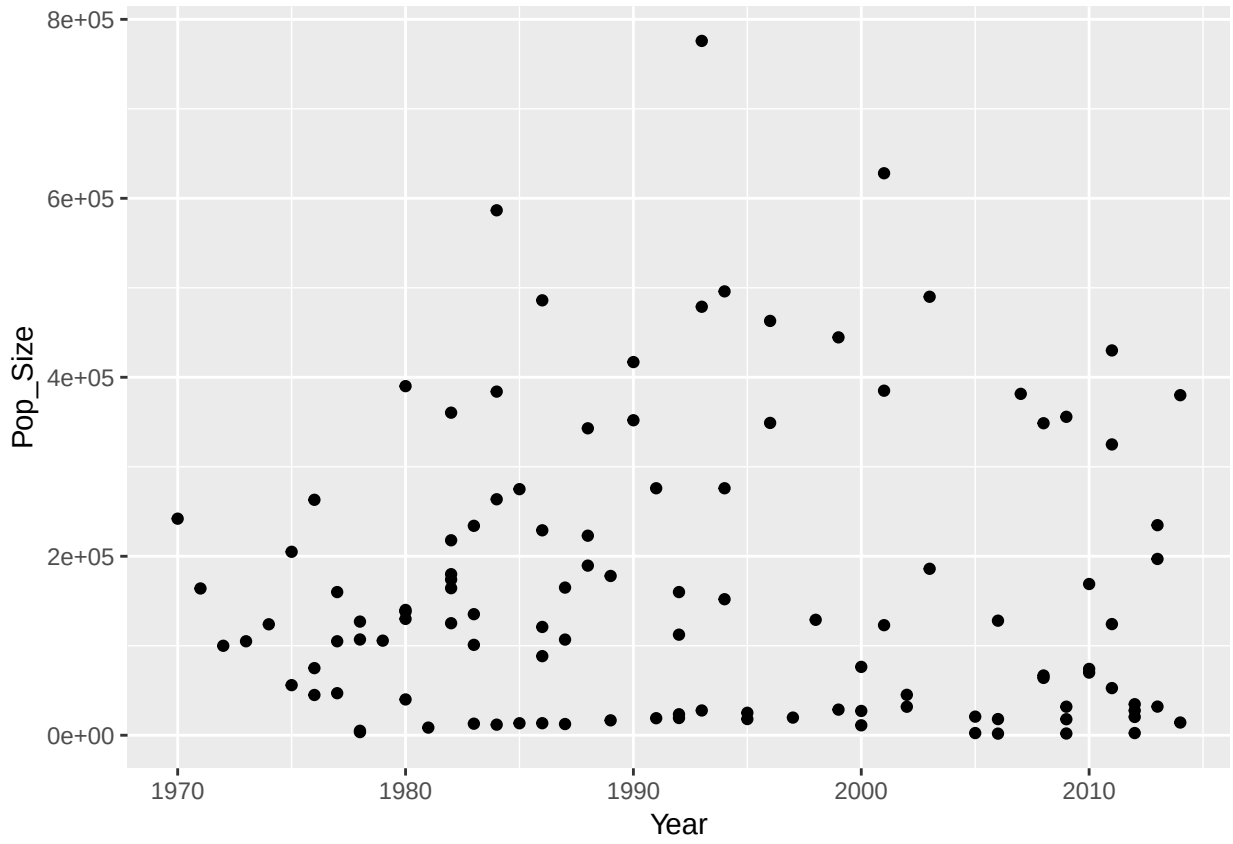

```
# facetting, ommit missing data
ggplot(data = seaice %>% filter(Year == 2010)) +
  aes(x = Month, y = Cover) +
  geom_col() +
  facet_wrap(~Herd)
```



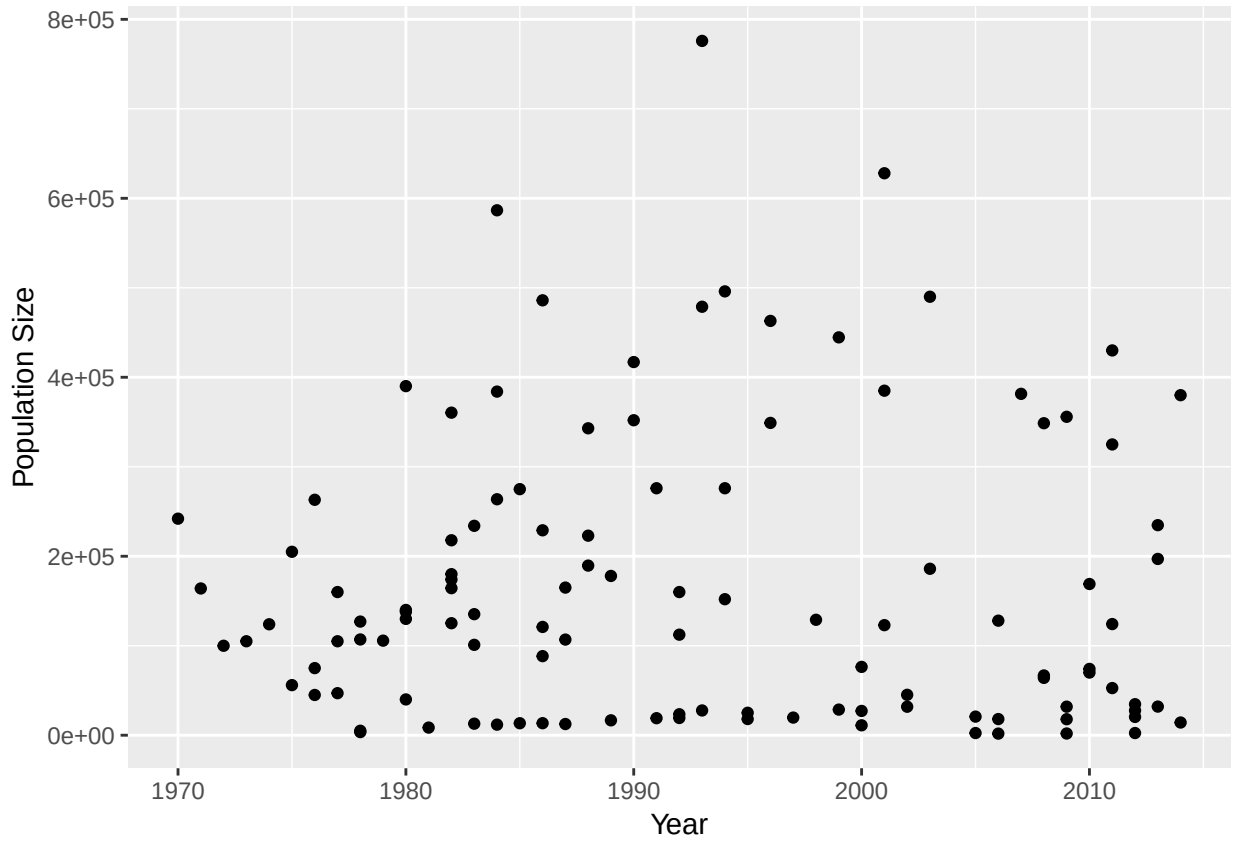

```
# facetting, ommit missing data, adjusted scale of axes
ggplot(data = seaice %>% filter(Year == 2010)) +
  aes(x = Month, y = Cover) +
  geom_col() +
  facet_wrap(~Herd, scales = "free")
```



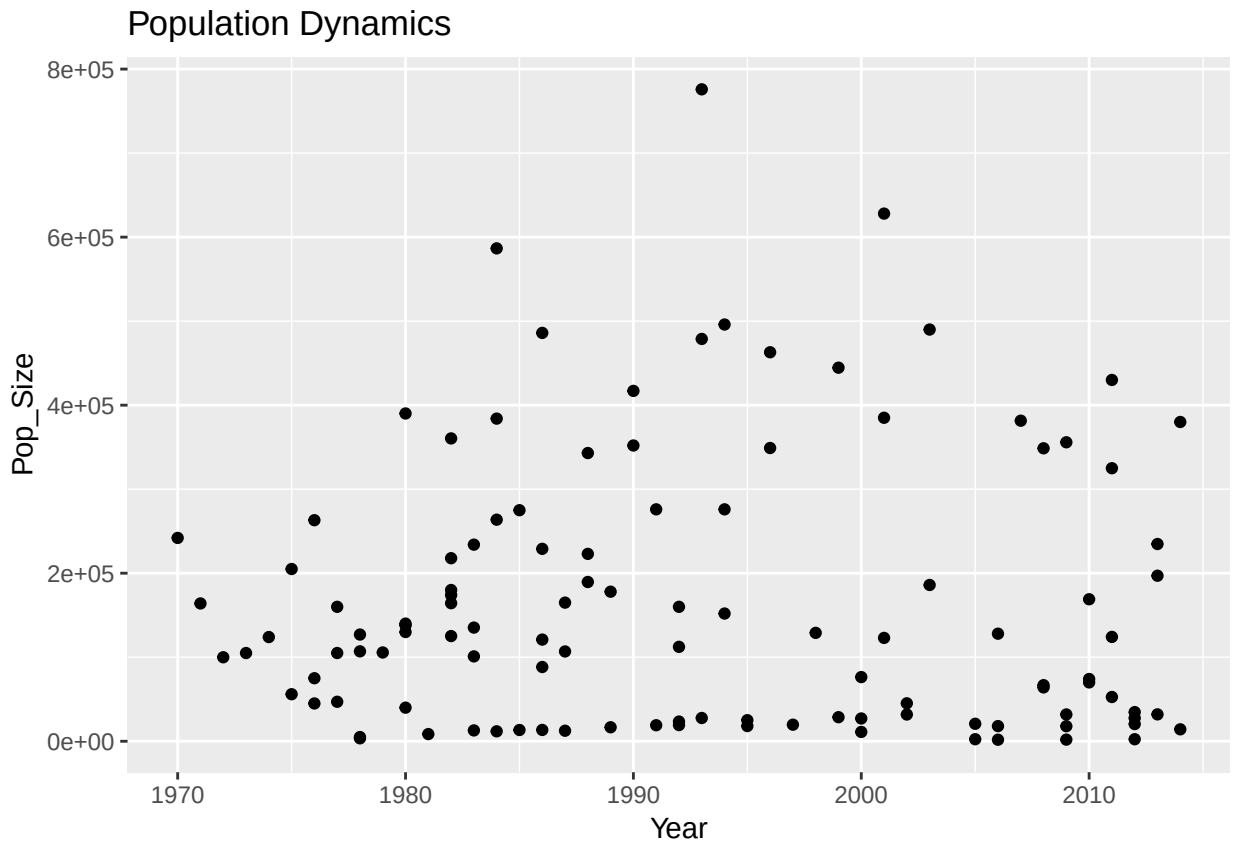

```
# changing labels and title
pl <- ggplot(data = popsize) +
  aes(x = Year, y = Pop_Size) +
  geom_point()
pl + xlab("Year")
```



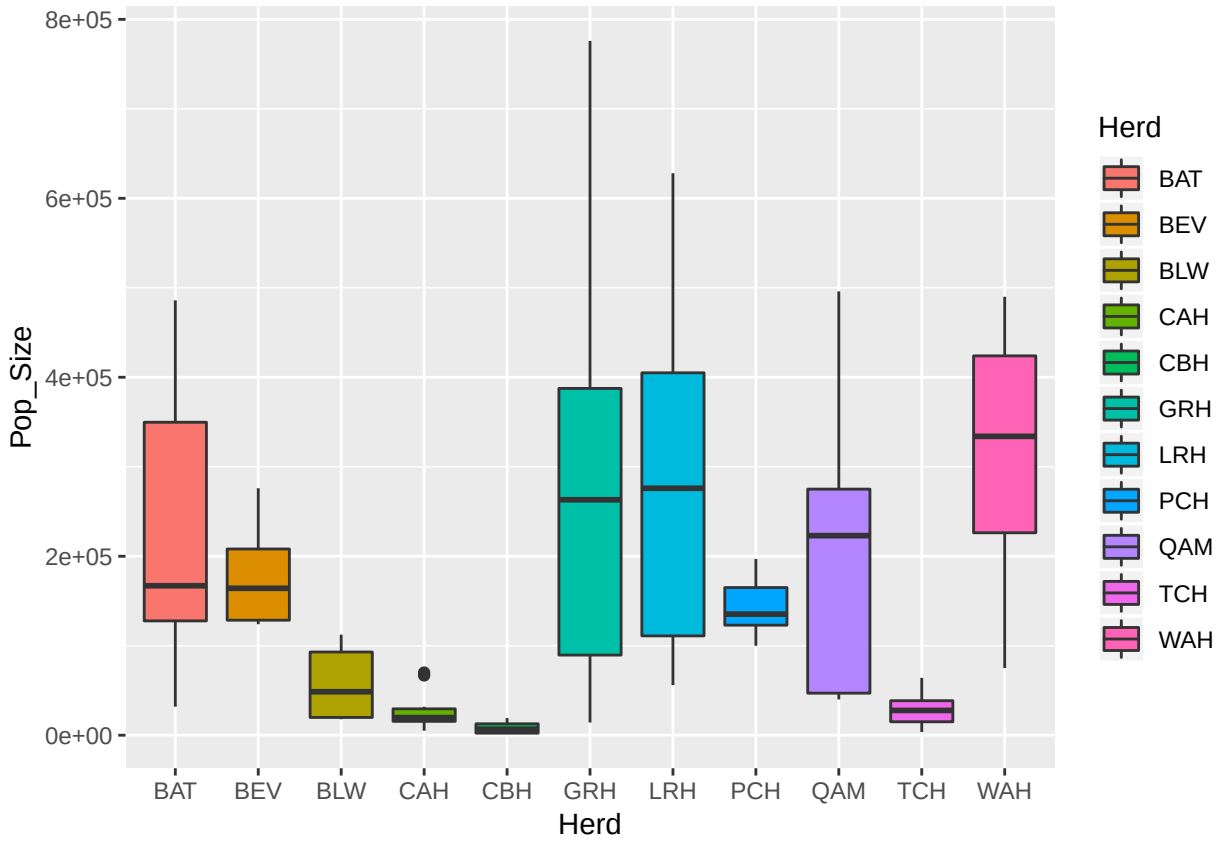

```
pl + ylab("Population Size")
```



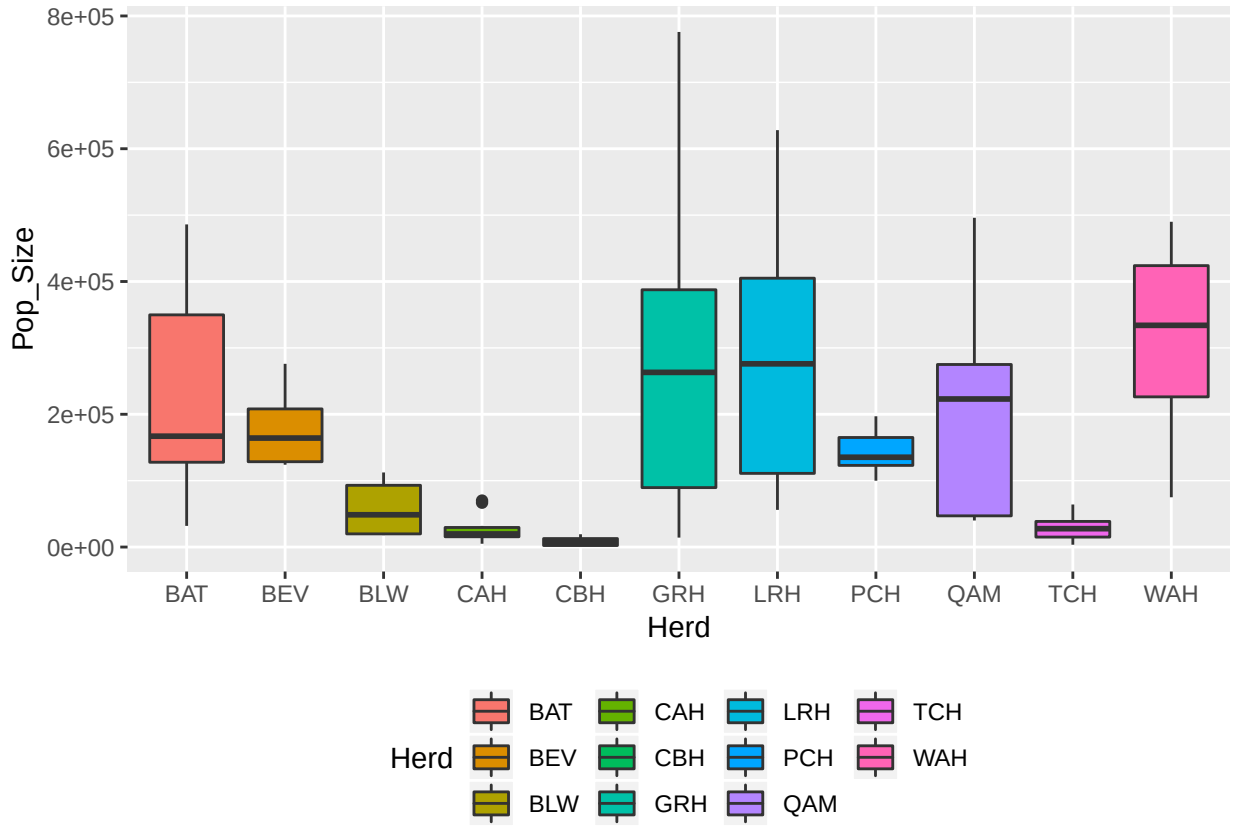

```
pl + ggtitle("Population Dynamics")
```



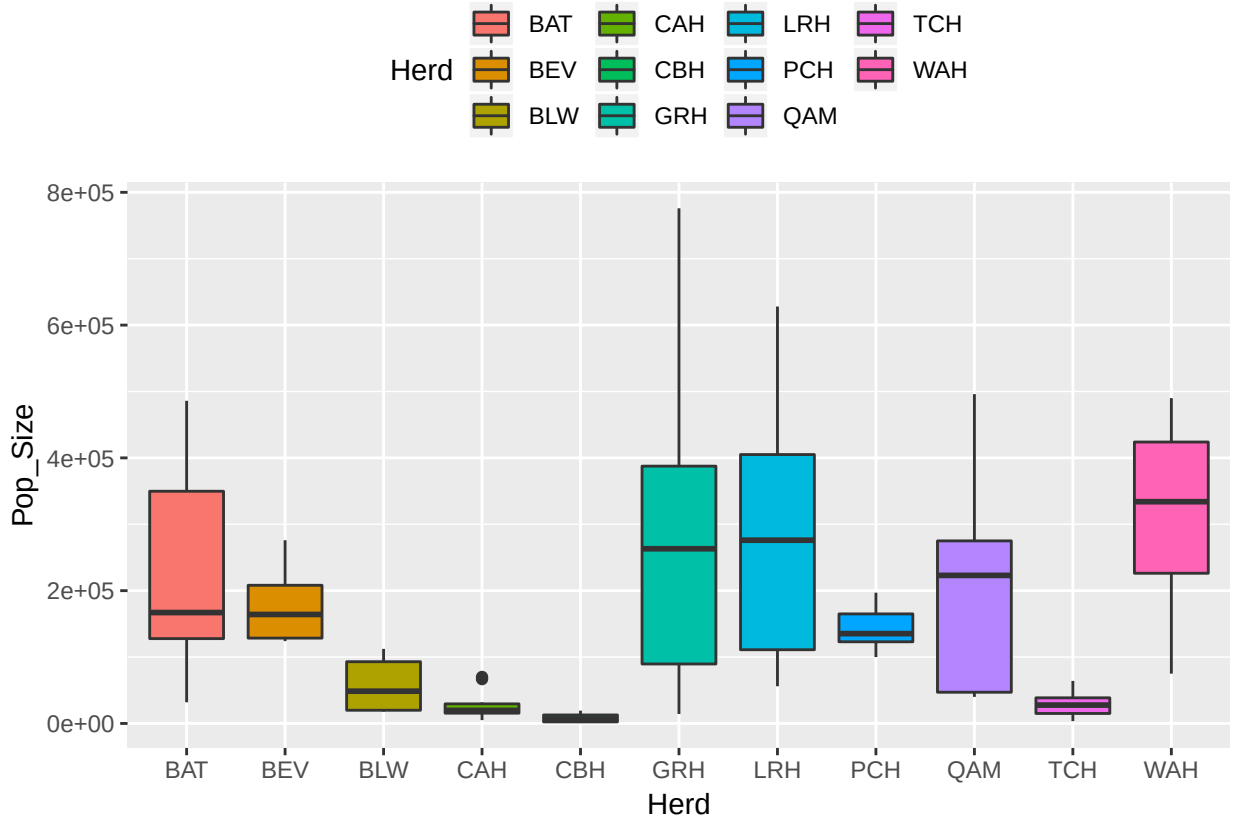

```
# legends
pl <- ggplot(data = popsize) +
  aes(x = Herd, y = Pop_Size, fill = Herd) +
  geom_boxplot()
# default
show(pl)
```



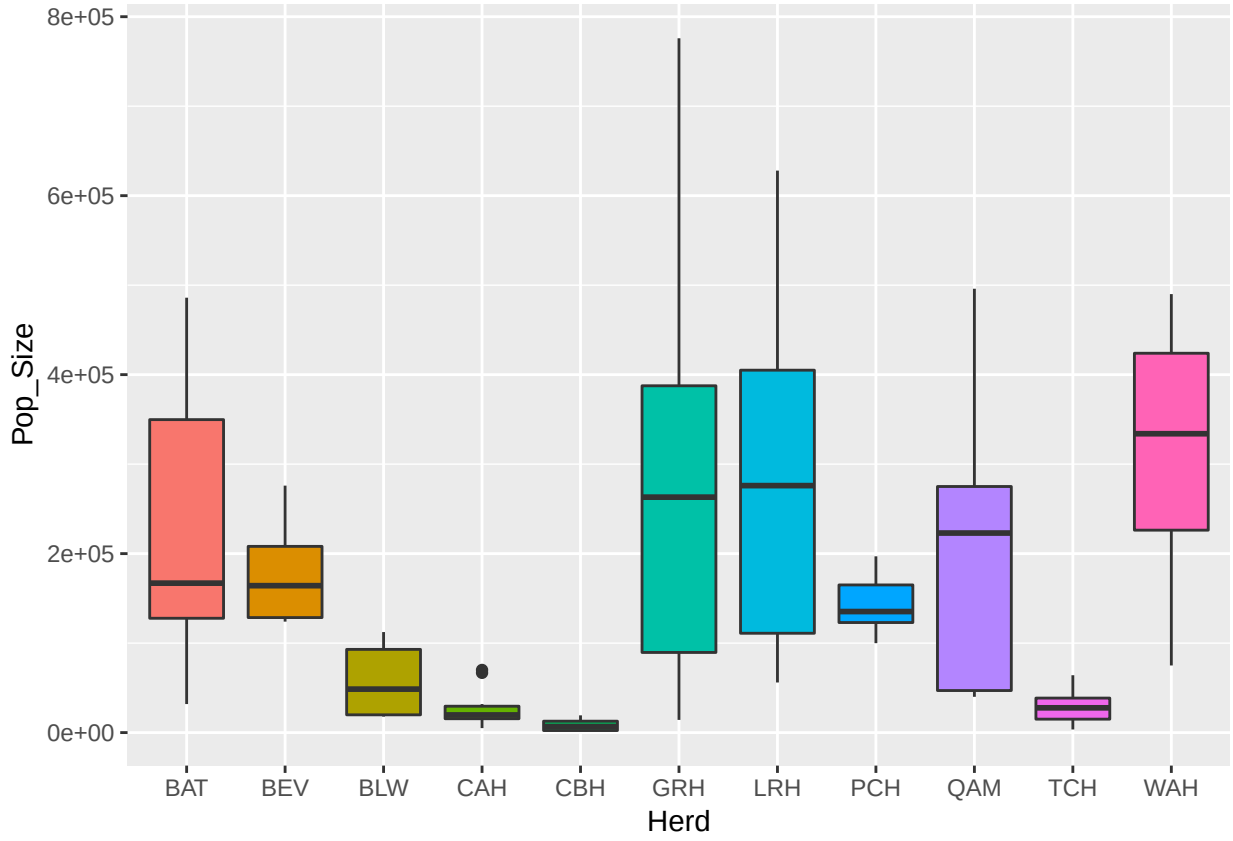

```
# move legend
pl + theme(legend.position = "bottom")
```



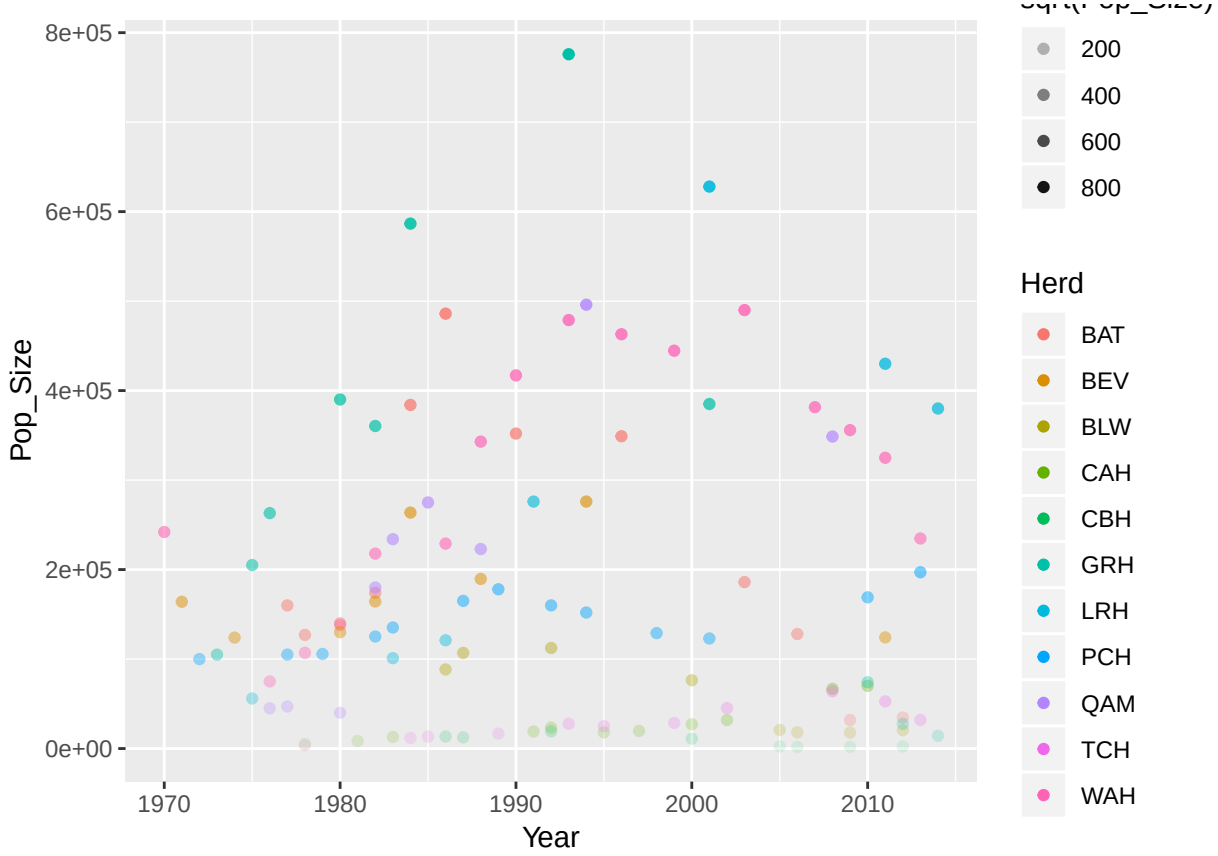

```
pl + theme(legend.position = "top")
```



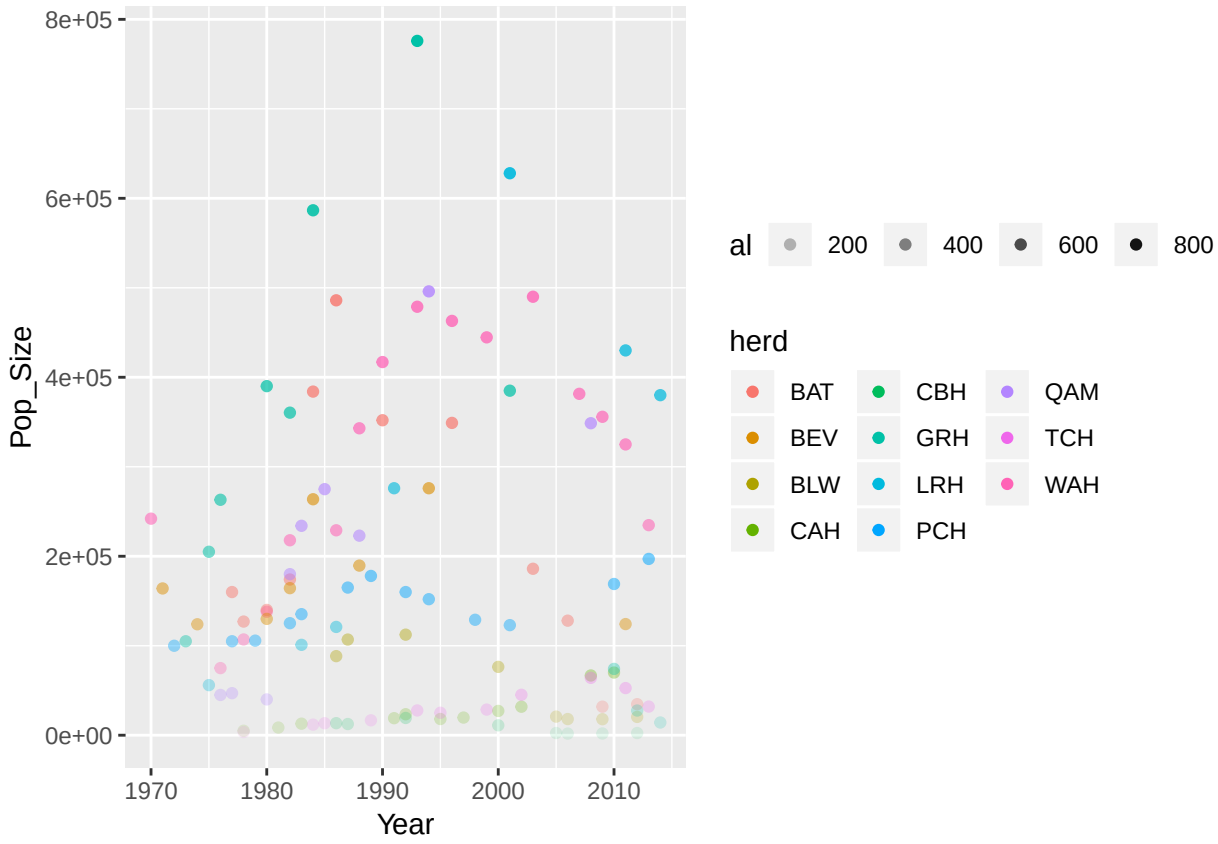
```
# remove legend
pl + theme(legend.position = "none")
```



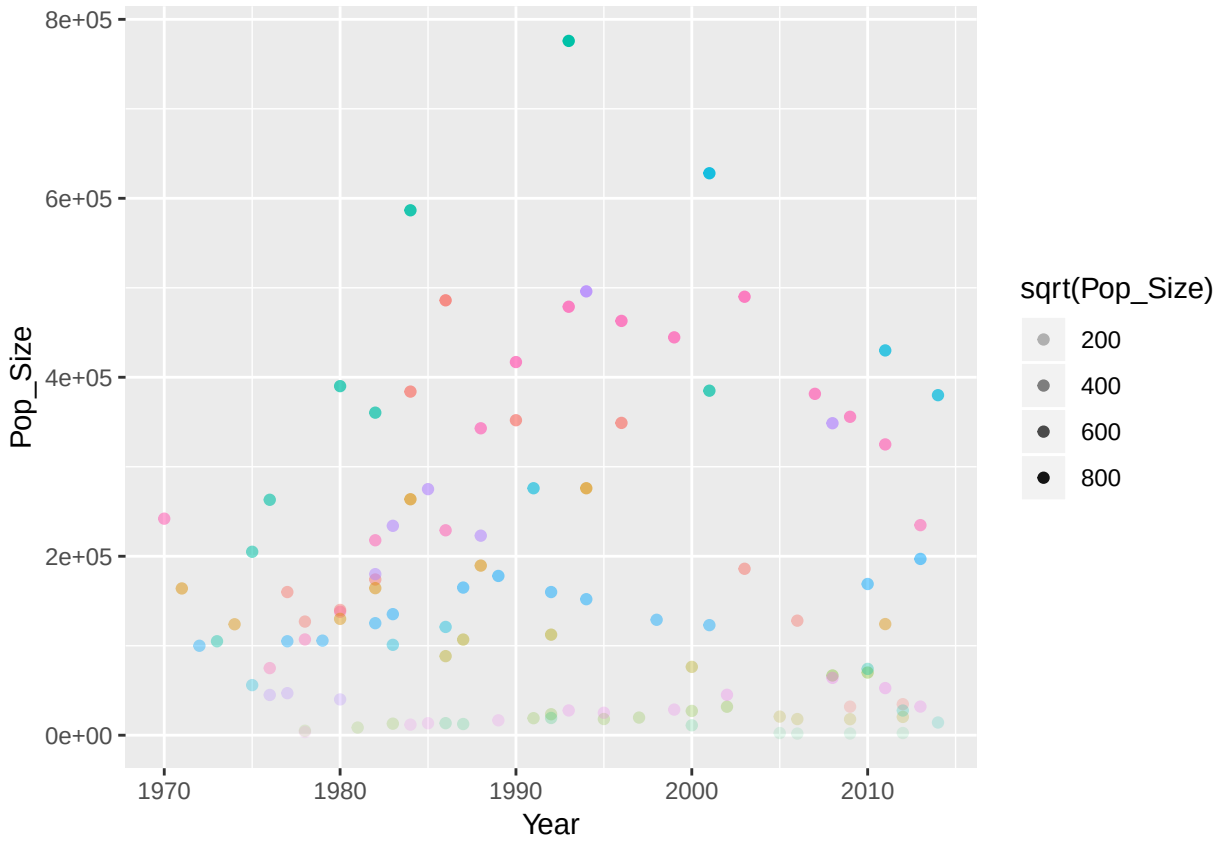

```
# legend guides
pl <- ggplot(data = popsize) +
  aes(x = Year, y = Pop_Size, colour = Herd,
      alpha = sqrt(Pop_Size)) +
  geom_point()
show(pl)
```



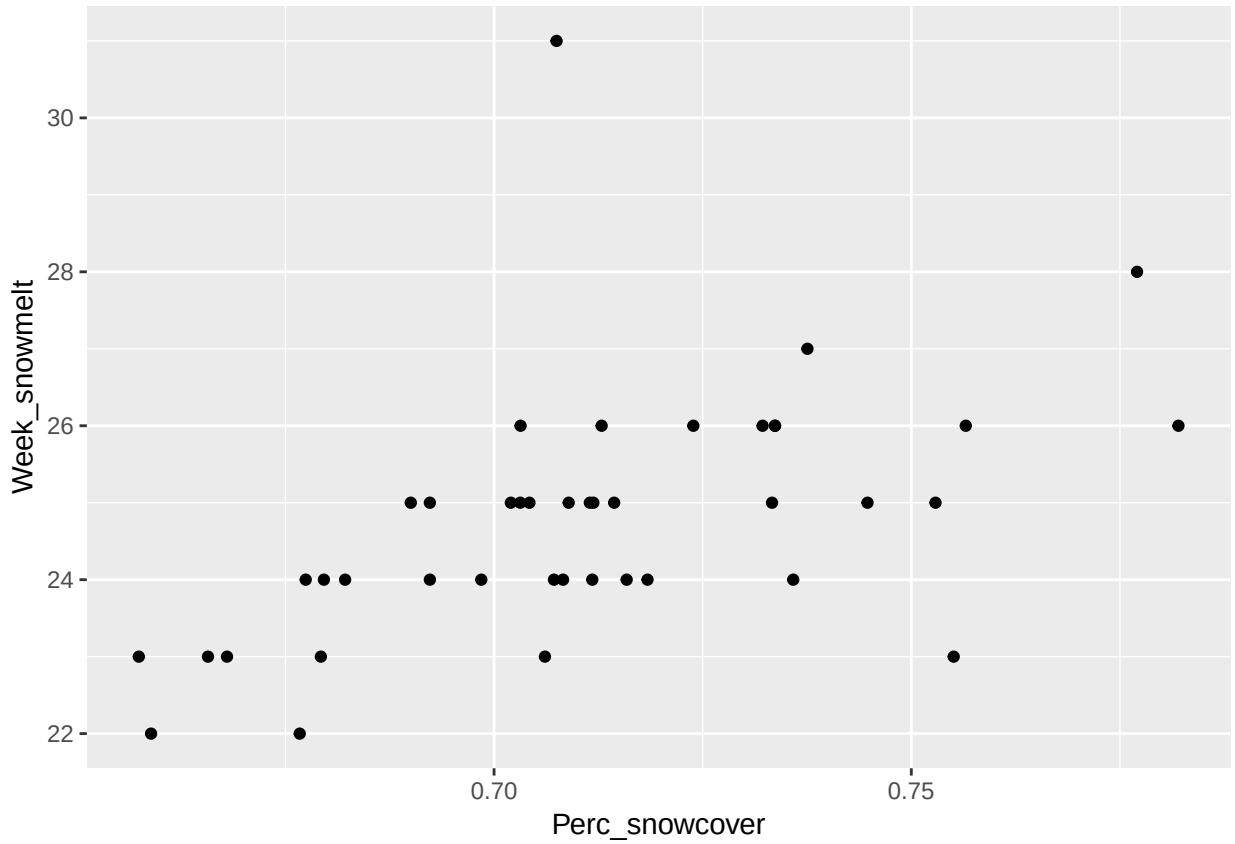

```
pl + guides(colour = guide_legend(nrow = 4,
                                  title = "herd"),
            alpha = guide_legend(direction = "horizontal",
                                 title =  "al"))
```



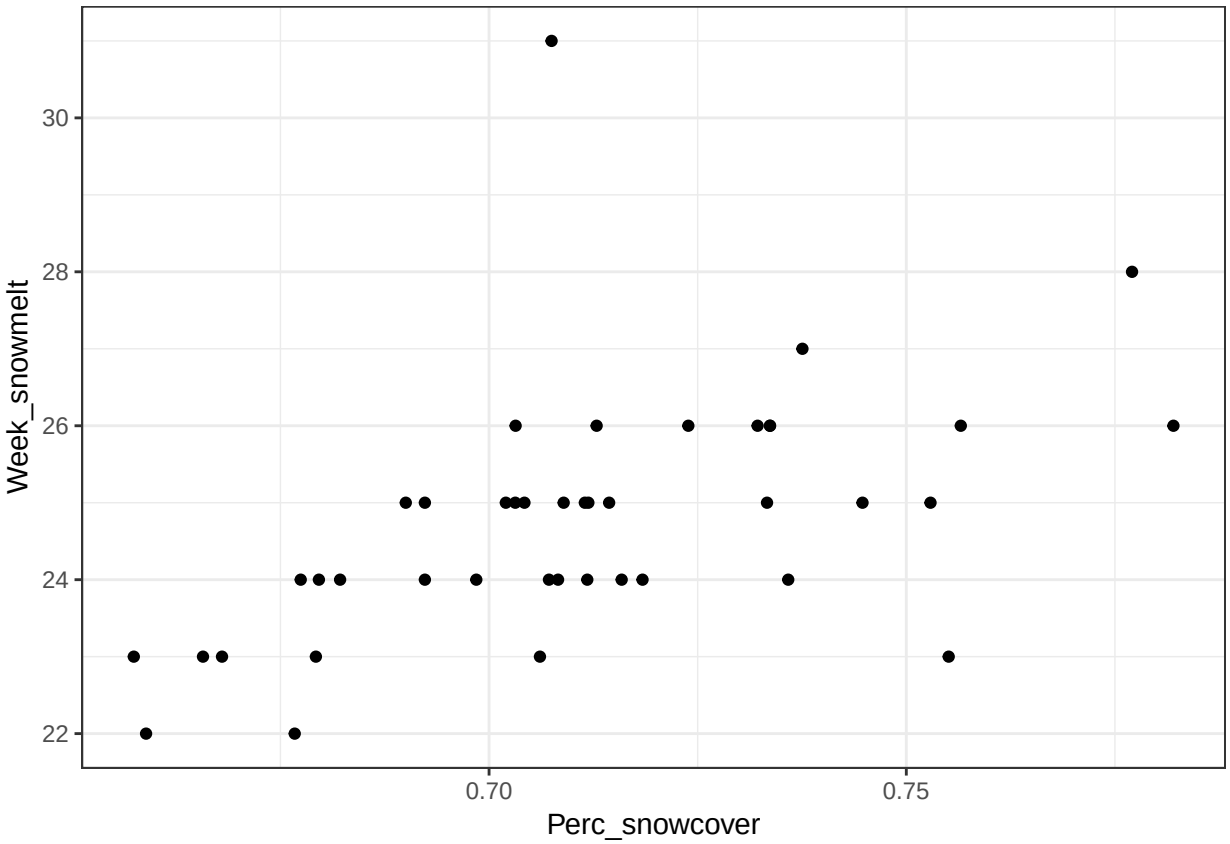

```
# suppress only one legend
pl + guides(colour = "none")
```



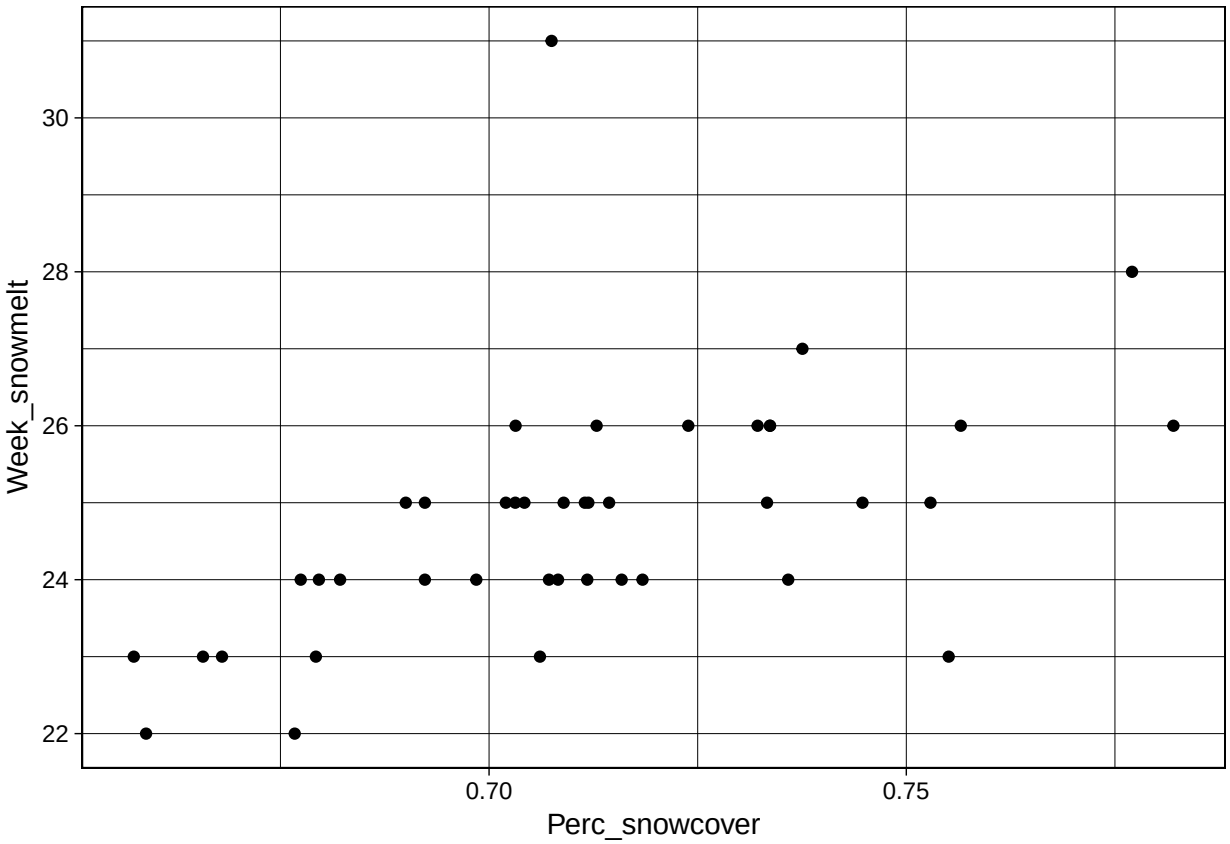

```
# themes
pl <- ggplot(data = snow %>%
    filter(Herd == "CAH"),
    aes(y = Week_snowmelt, x = Perc_snowcover)) +
    geom_point()
# default theme with grey background and white gridlines
show(pl)
```



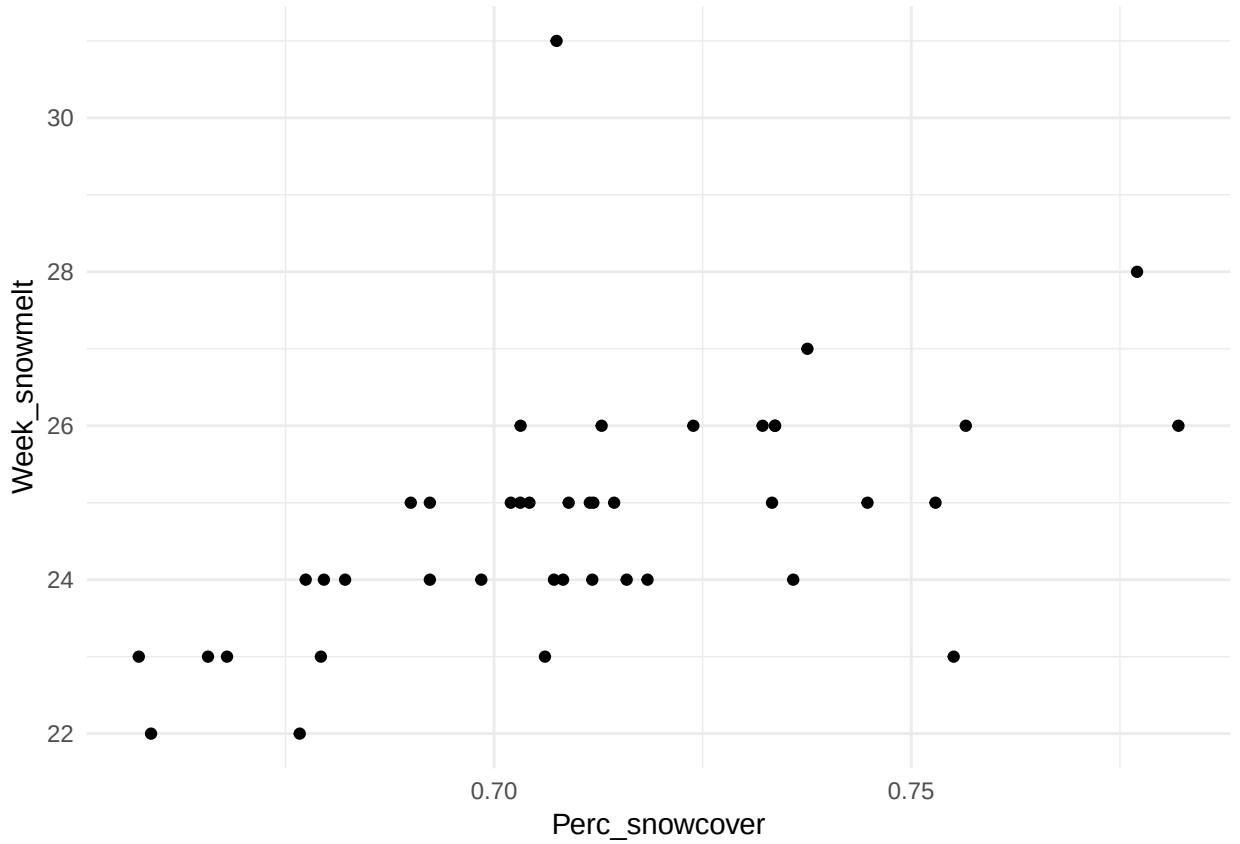

```
# black and white (light background)
pl + theme_bw()
```



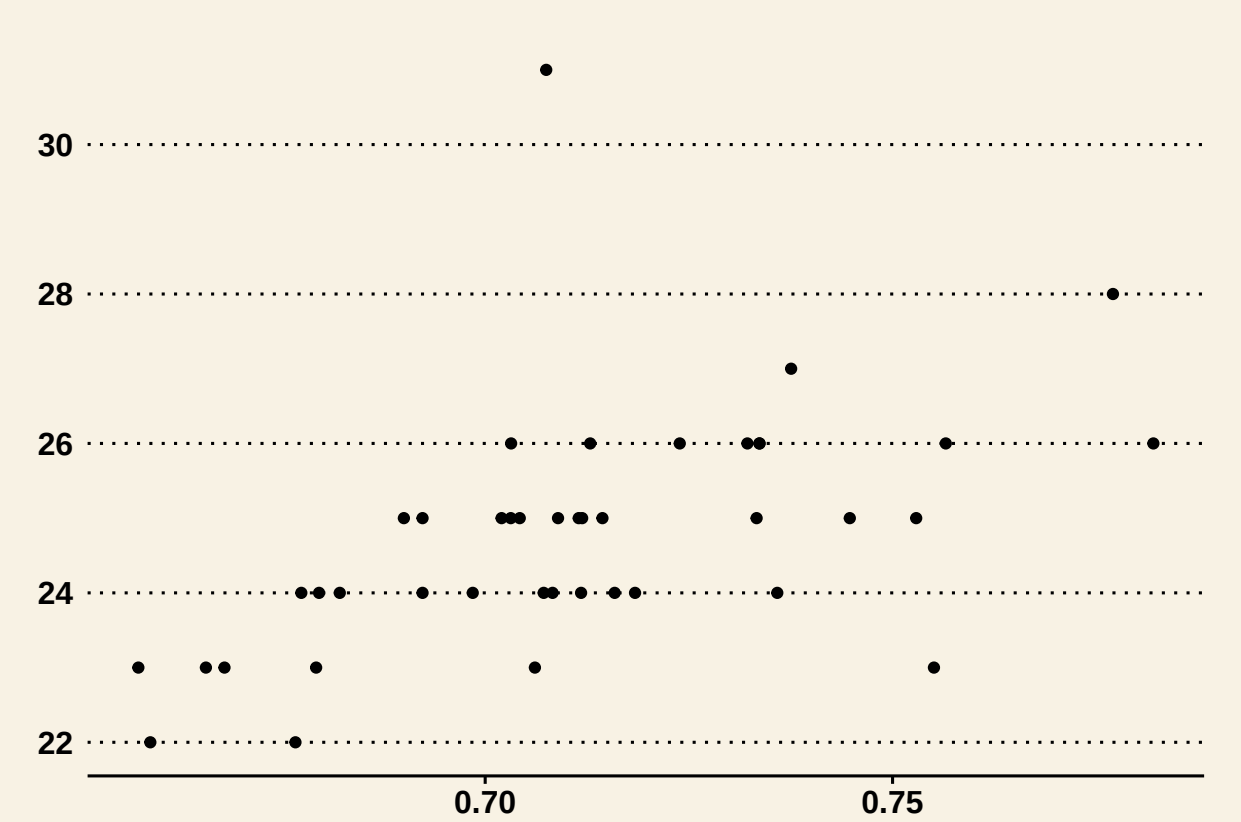

```
# line draw
pl + theme_linedraw()
```



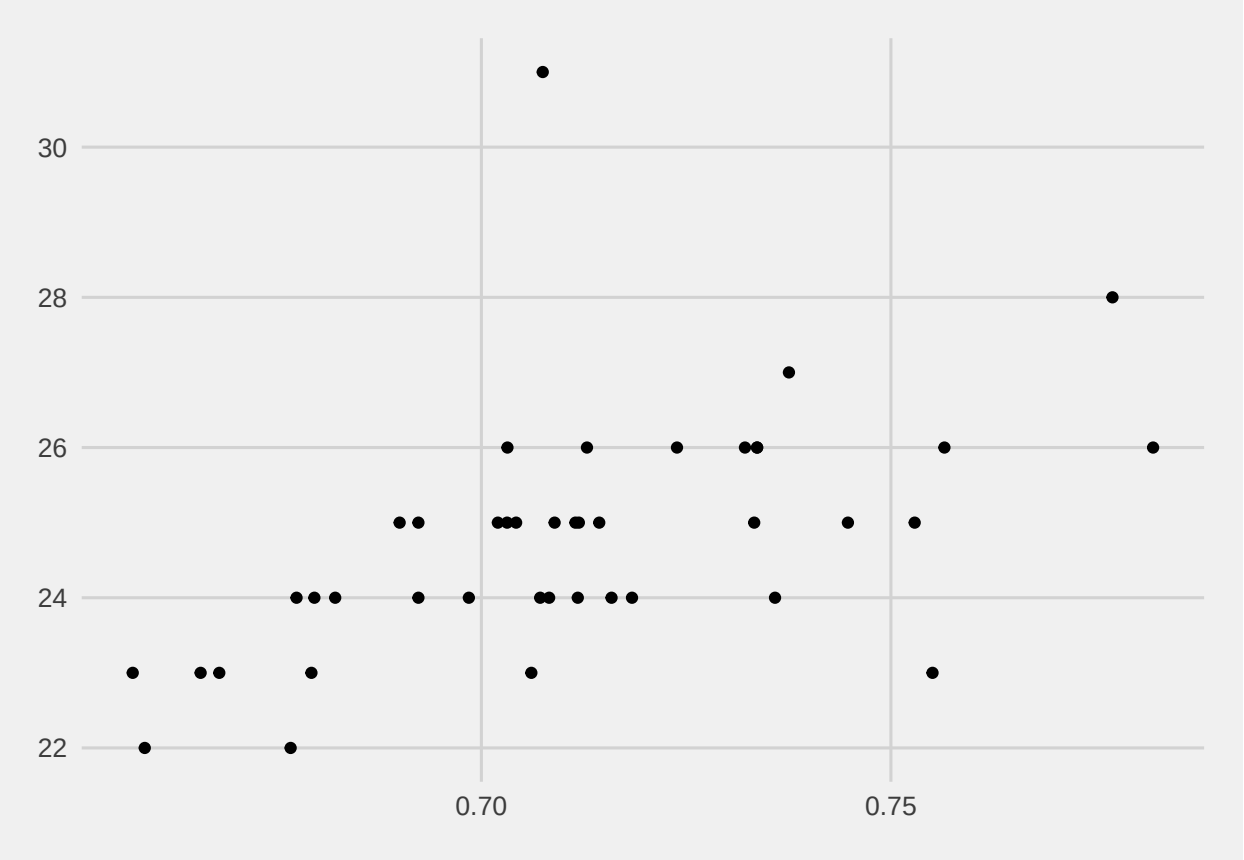

```
# minimalistic theme
pl + theme_minimal()
```




```
# load additional themes
library(ggthemes)
# Wall Street Journal
show(pl + theme_wsj())
```




```
# Five thirty-eight
show(pl + theme_fivethirtyeight())
```




```
# setting features
# use color as an aesthetic mapping, associated with Herd
pl <- ggplot(data = popsize) +
  aes(x = Year, y = Pop_Size, colour = Herd) +
  geom_point()
# set color to be red for all points
pl <- ggplot(data = popsize) +
  aes(x = Year, y = Pop_Size) +
  geom_point(colour = "red")
```

```
# saving plot as test.pdf in the sandbox
ggsave(filename = "../sandbox/test.pdf", plot = pl,
       width = 3, height = 4)
```

```
# select numerical column headers, or headers with white space using back ticks
popsize %>% filter(Year > 1979, Year < 1985) %>% spread(Year, Pop_Size) %>%  select(Herd, `1980`)
```

```
# # A tibble: 9 x 2
#   Herd  `1980`
#   <chr>  <int>
# 1 BAT   140000
# 2 BEV   130000
# 3 CAH       NA
# 4 GRH   390100
# 5 LRH       NA
# 6 PCH       NA
# 7 QAM    40000
```

```
# 8 TCH       NA
# 9 WAH   138000
```

```
# ungroup elements
popsize %>% group_by(Herd, Year) %>% tally() %>% ungroup()
```

```
# # A tibble: 114 x 3
#    Herd   Year     n
#    <chr> <int> <int>
#  1 BAT    1977     1
#  2 BAT    1978     1
#  3 BAT    1980     1
#  4 BAT    1982     1
#  5 BAT    1984     1
#  6 BAT    1986     1
#  7 BAT    1990     1
#  8 BAT    1996     1
#  9 BAT    2003     1
# 10 BAT    2006     1
# # ... with 104 more rows
```