Technical Appendix

Notes for appendix

The Appendix still needs more comments and more details.

For the draft, we only contain the code for plots in our IDMRD paper.

Appendix 1: Map

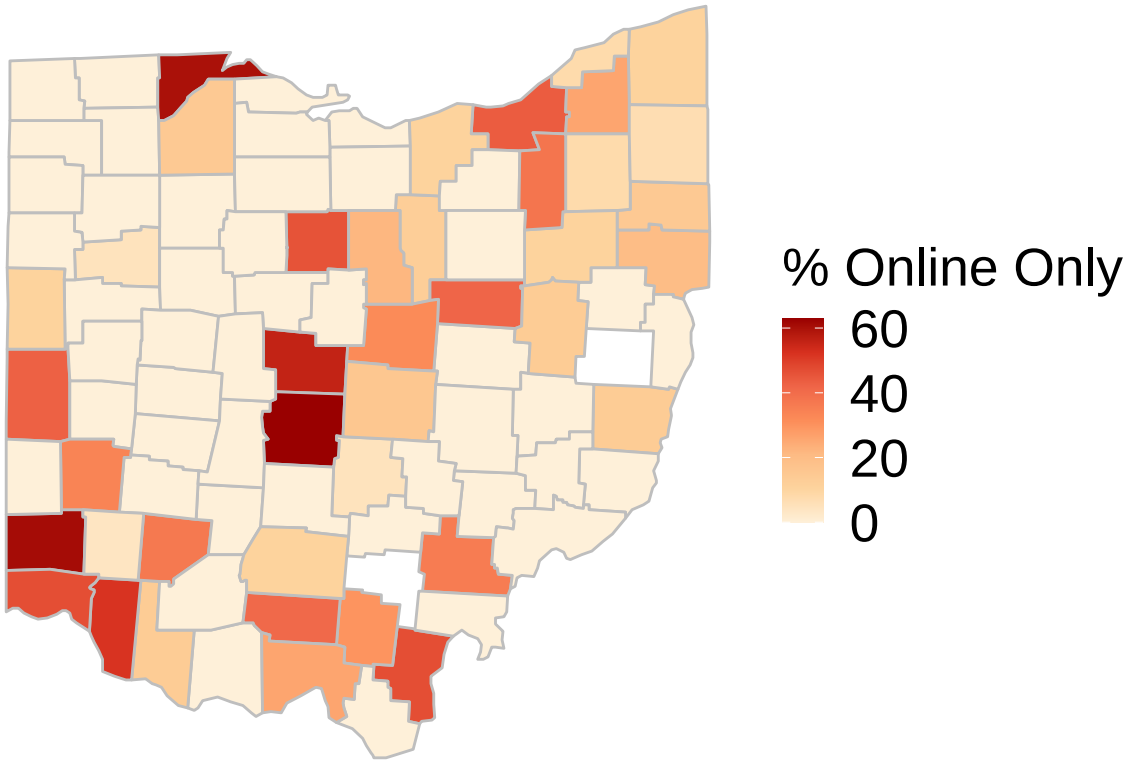
```
Sys.setlocale("LC_TIME", "English")
```

```
## [1] "English_United States.1252"
```

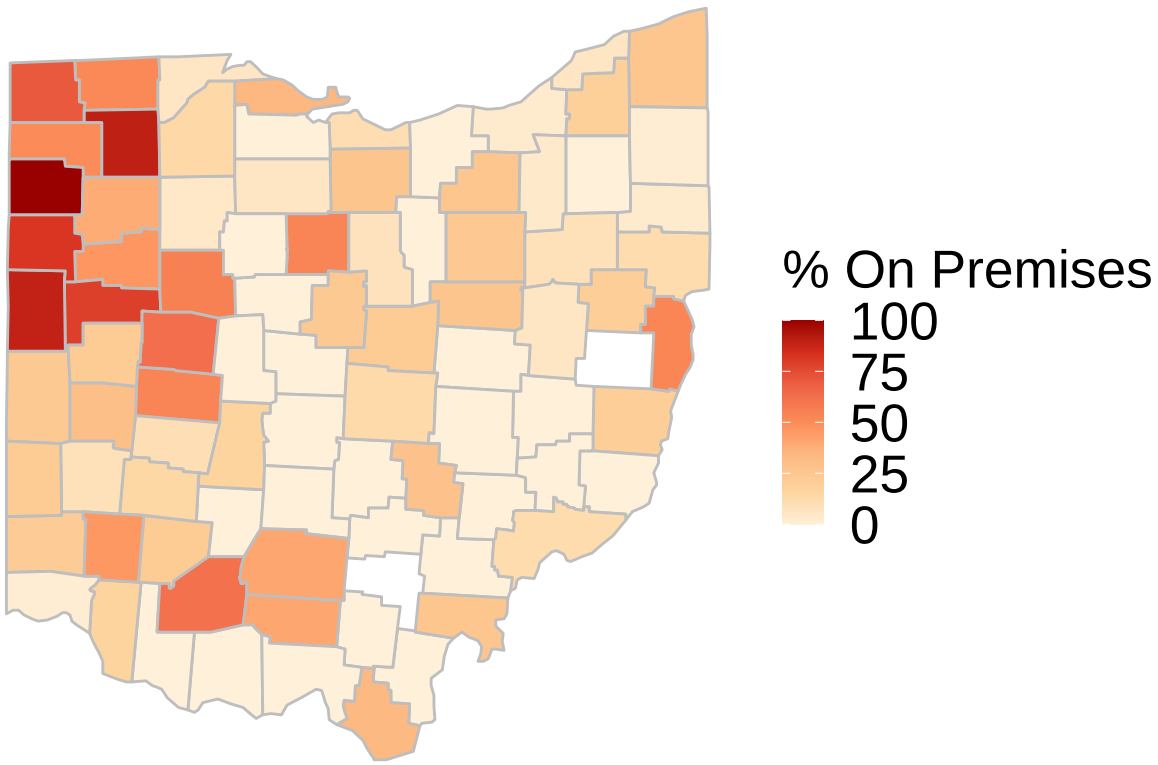
```
library(ggrepel)
library(cowplot)
library(sp)
source("step2_data_wrangle.R")
```

Teaching method, Population and Enrollment

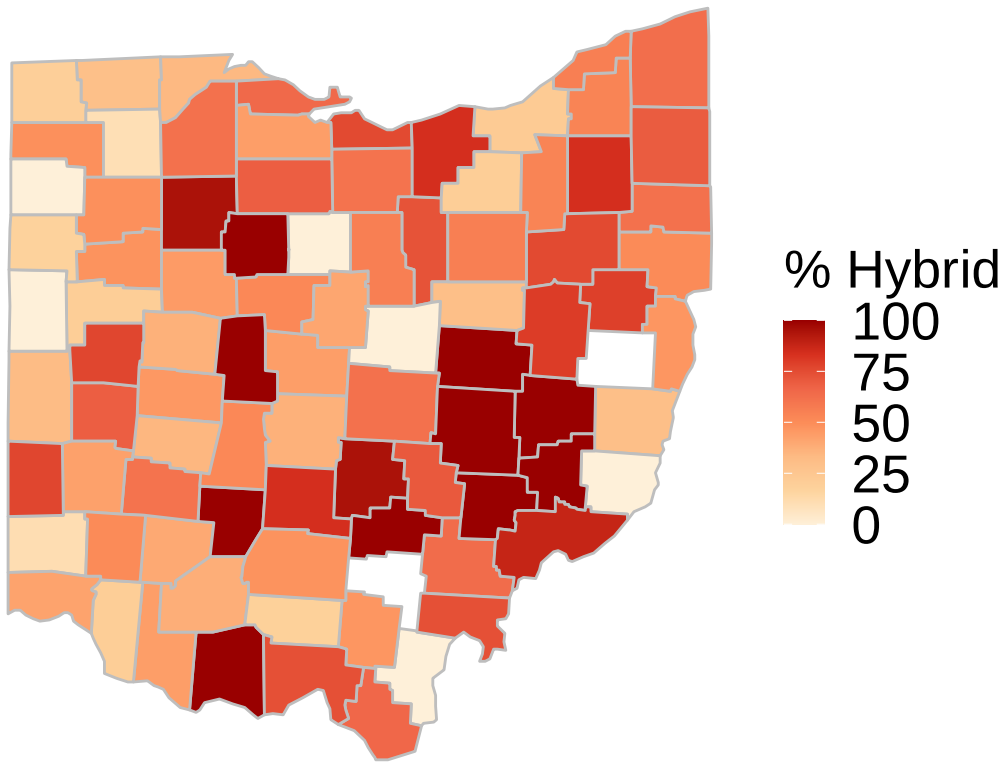
```
ohio_map <- map_data("county") %>%subset(region=="ohio")%>%
  mutate(county=toupper(subregion))%>%select(long,lat,county,group)
# create map plots
wide_teaching_enroll%>%
  left_join(ohio_map,by='county')%>%
  mutate(Online_Only= Online_Only*100)%>%
  ggplot() +
  geom_polygon(aes(x = long, y = lat, group = group, fill = Online_Only), color = "gray") +
  coord_fixed(1.3) + theme_map() +
  scale_fill_distiller(palette = "OrRd",direction = 1)+
  labs(fill='% Online Only')+
  theme(legend.text = element_text(size=20),legend.title = element_text(size=20))
```



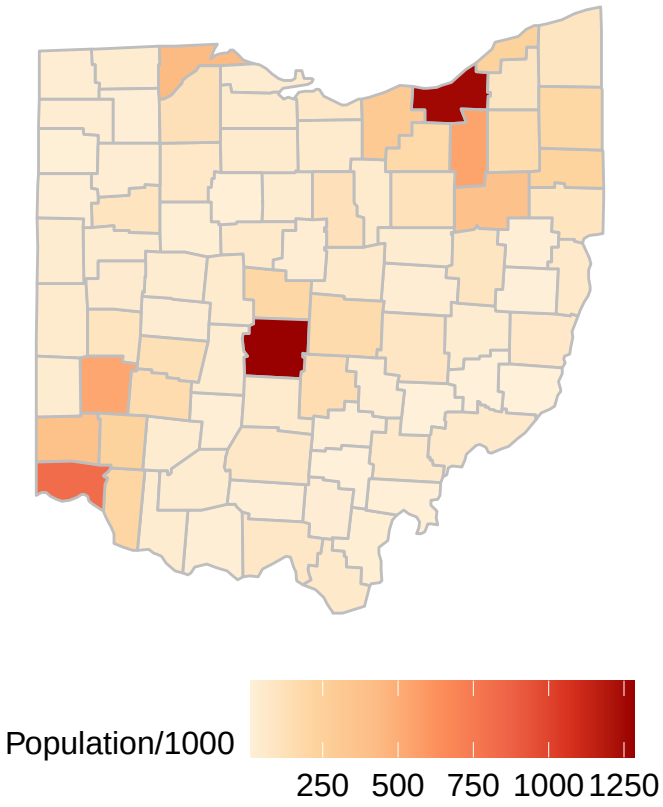

```
# create map plots
wide_teaching_enroll%>%
  left_join(ohio_map,by='county')%>%
  mutate(On_Premises= On_Premises*100)%>%
  ggplot() +
  geom_polygon(aes(x = long, y = lat, group = group, fill = On_Premises), color = "gray") +
  coord_fixed(1.3) + theme_map() +
  scale_fill_distiller(palette = "OrRd",direction = 1)+
  labs(fill='% On Premises')+
  theme(legend.text = element_text(size=20),legend.title = element_text(size=20))
```



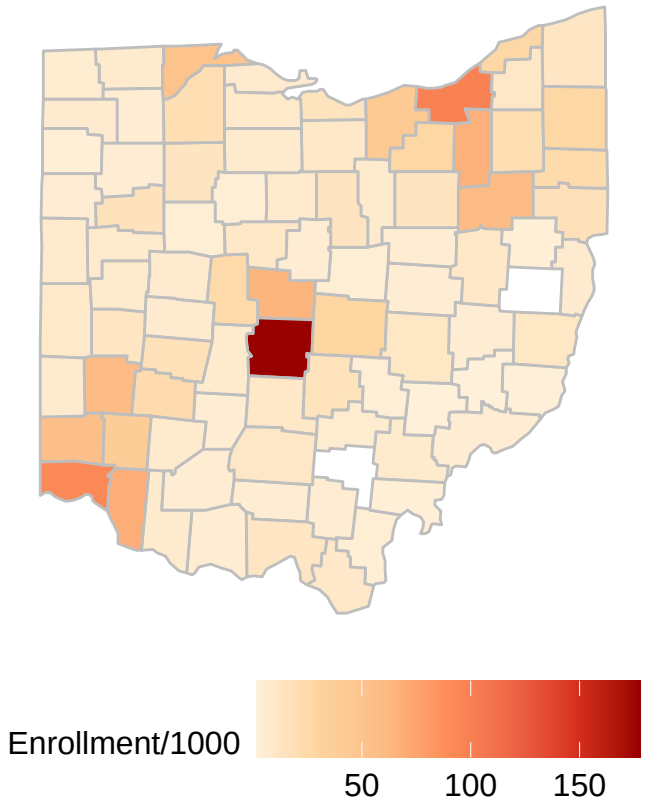

```
# create map plots for population
wide_teaching_enroll%>%
  left_join(ohio_map,by='county')%>%
  mutate(Hybrid= Hybrid*100)%>%
  ggplot() +
  geom_polygon(aes(x = long, y = lat, group = group, fill = Hybrid), color = "gray") +
  coord_fixed(1.3) +
  theme_map() +
  scale_fill_distiller(palette = "OrRd",direction = 1)+
  labs(fill='% Hybrid')+
  theme(legend.text = element_text(size=20),legend.title = element_text(size=20))
```



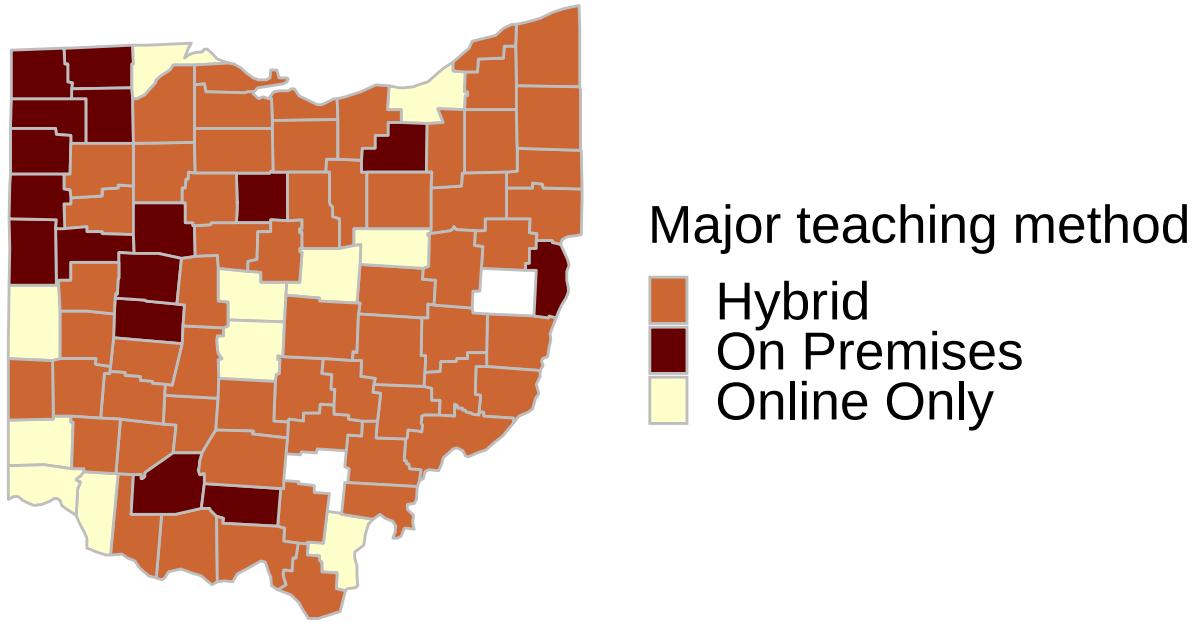

```
# create map plots
cases%>%
  distinct(COUNTY,POPULATION)%>%
  left_join(ohio_map,by=c('COUNTY'='county'))%>%
  mutate(population = POPULATION/1000)%>%
  ggplot() +
  geom_polygon(aes(x = long, y = lat, group = group, fill = population), color = "gray") +
  coord_fixed(1.3) + theme_map() +
  scale_fill_distiller(palette = "OrRd",direction = 1)+
  labs(fill='Population/1000')+
  theme(legend.text = element_text(size=12),
        legend.title = element_text(size=12),
        legend.position = "bottom",
        legend.key.size = unit(2,"lines"))
```




```
# create map plots
teachingmethod_enroll%>%
  distinct(county,county_enroll)%>%
  left_join(ohio_map,by=c('county'))%>%
  mutate(county_enroll = county_enroll/1000)%>%
  ggplot() +
  geom_polygon(aes(x = long, y = lat, group = group, fill = county_enroll), color = "gray") +
  coord_fixed(1.3) + theme_map() +
  scale_fill_distiller(palette = "OrRd",direction = 1)+
  labs(fill='Enrollment/1000')+
  theme(legend.text = element_text(size=12),legend.title = element_text(size=12),
        legend.position = "bottom",legend.key.size = unit(2,"lines"))
```



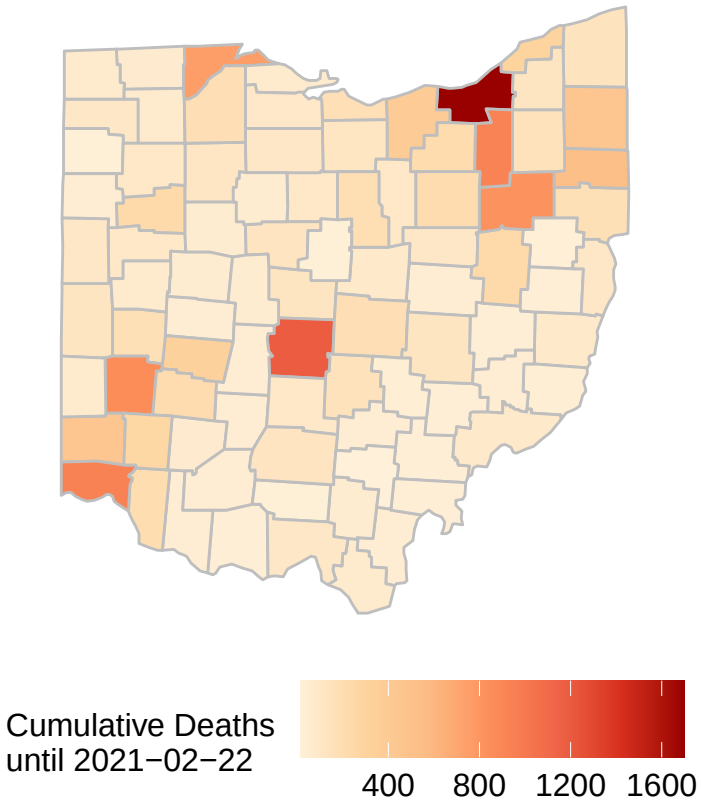

```
wide_teaching_enroll%>%
  left_join(ohio_map,by='county')%>%
  mutate(On_Premises= On_Premises*100)%>%
  ggplot() + geom_polygon(aes(x = long, y = lat, group = group, fill = as.factor(major_teaching)), colo
  coord_fixed(1.3) + theme_map() +
  scale_fill_manual(values = c(`Online Only`="#FFFFCC",
                     `Hybrid`="#CC6633",
                     `On Premises`="#660000",
                     `NA`="gray"),
                         name="Major teaching method")+
  labs(fill='% On Premises')+
  theme(legend.text = element_text(size=20),                                    legend.title = e
```



Covid deaths during fall semester and death proportion during fall semester

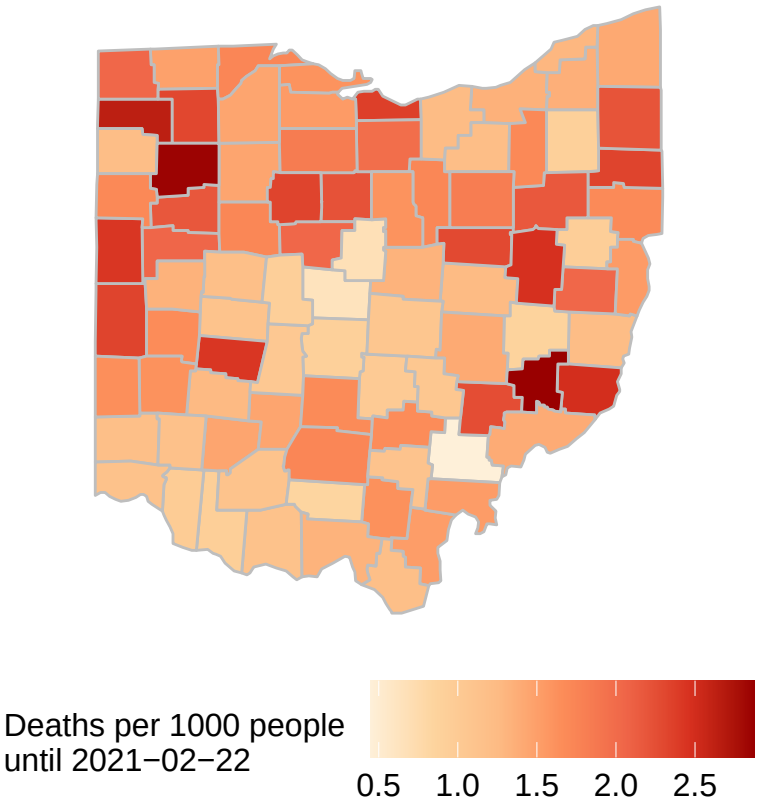
```
getLabelPoint <- # Returns a county-named list of label points
function(county) {Polygon(county[c('long', 'lat')])@labpt}
centroids = by(ohio_map, ohio_map$county, getLabelPoint)# Returns list
centroids2 <- do.call("rbind.data.frame", centroids)# Convert to Data Frame
centroids2$county = str_to_title(rownames(centroids))
names(centroids2) <- c('clong', 'clat', "county") # Appropriate Header
```

```
death_prop%>%
  left_join(ohio_map,by=c("COUNTY"='county'))%>%
  ggplot() +
  geom_polygon(aes(x = long, y = lat, group=group,fill = CUMDEATHS), color = "gray")+
  coord_fixed(1.3) + theme_map() +
  scale_fill_distiller(palette = "OrRd",direction = 1)+
  labs(fill='Cumulative Deaths \nuntil 2021-02-22')+
  theme(legend.text = element_text(size=12),
        legend.title = element_text(size=12),legend.position = "bottom",
        legend.key.size = unit(2,"lines"))
```




```
ggsave("cumdeath.png",width = 5, height = 5)

death_prop%>%
  left_join(ohio_map,by=c("COUNTY"='county'))%>%
  ggplot() +
  geom_polygon(aes(x = long, y = lat, group=group,fill = death_per_1000),
               color = "gray") +
  coord_fixed(1.3) + theme_map() +
  scale_fill_distiller(palette = "OrRd",direction = 1)+
  labs(fill='Deaths per 1000 people \nuntil 2021-02-22')+
  theme(legend.text = element_text(size=12),
        legend.title = element_text(size=12),
        legend.position = "bottom",
        legend.key.size = unit(2,"lines"))
```




```
ggsave("deathprop.png",width = 5, height = 5)
```

Appendix 2: Death Incidence

Data Process

```
library(tidyverse)
library(lubridate)
require(scales)
library(readxl)
cases_by_age <- read_excel("OhiobyAge.xlsx")
rolling_age_cases <- cases_by_age %>%
  mutate(youth_prop_roll = zoo::rollmean(`00_19/total(%)`, k = 7, fill = NA),
         all_roll = zoo::rollmean(`00_80+`, k = 7, fill = NA))
colors <- c("Total Daily Cases" = "black",
            "0-19 Age / Total Cases (%)" = "gray")
coeff <- 200
cases_by_age_long <- cases_by_age %>%
  gather(age_group, percent_cases,
         `00_19/total(%)`:`80+/total(%)`,
         factor_key=TRUE) %>%
  group_by(age_group) %>%
  mutate(roll_percent_cases= zoo::rollmean(percent_cases, k = 7, fill = NA))
county_policy_wide$major_teaching <- factor(county_policy_wide$major_teaching,
                                            levels = c("On Premises","Hybrid","Online Only"))
```

```r
# see when the intesection happens
date.intercept <- as.Date("2020-11-24")
# add 95% confidence bans
confidence_level <- .95
z_cl <- qnorm(confidence_level)
# case_policy_wide
case_policy_wide <- cases %>%
  left_join(county_policy_wide[,c("county","major_teaching","Online_Only","Hybrid","On_Premises")],
            by = c("COUNTY" = "county")) %>%
  mutate(death_prop = CUMDEATHS/POPULATION)
opendate_cases <- case_policy_wide%>%
  inner_join(major_reopening%>%select(COUNTY,major_opendate),by=c('COUNTY'))
# Box Plots in Fall semester
library(PMCMRplus)
require(DescTools)
fall_cases <- opendate_cases %>%
  filter(DATE >= major_opendate & DATE <= as.Date("2020/12/15")) %>%
  group_by(COUNTY) %>%
  arrange(DATE) %>%
  filter(row_number()==1 | row_number()==n()) %>%
  mutate(death_incidence = diff(CUMDEATHS),
         death_incidence_per_1000 = death_incidence*1000/POPULATION) %>%
  distinct(COUNTY,POPULATION,major_teaching,
           death_incidence,death_incidence_per_1000)
fall_major_teaching.aov <- aov(death_incidence_per_1000 ~ major_teaching,
                               data = fall_cases)
summary(fall_major_teaching.aov) # p-value of .012
```

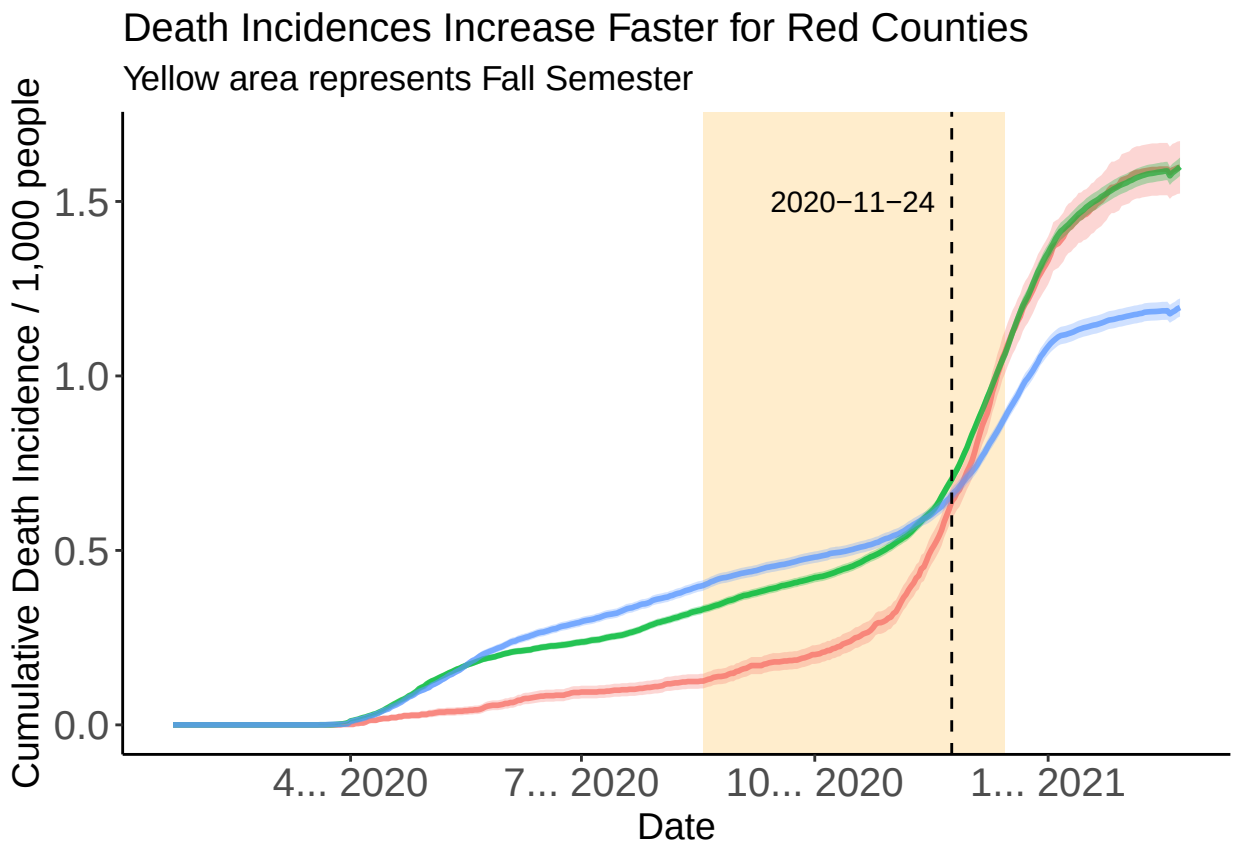
```
##                Df Sum Sq Mean Sq F value  Pr(>F)
## major_teaching  2  1.653  0.8264   5.205 0.00761 **
## Residuals      76 12.067  0.1588
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
stat.test <- PostHocTest(fall_major_teaching.aov, method = "duncan")$major_teaching%>%
  as.data.frame()%>%
  rownames_to_column("group") %>%
  separate(group,"-", into = c("group1","group2")) %>%
  mutate(pval = round(pval,3),
         p = case_when(pval <= .01~ "**",
                       pval <= .05 ~ "*",
                       TRUE ~ "NS"))%>%
  select(group1, group2, pval, p)
library(ggpubr)
```

Death Prop Over Time by the Majority Teaching Method

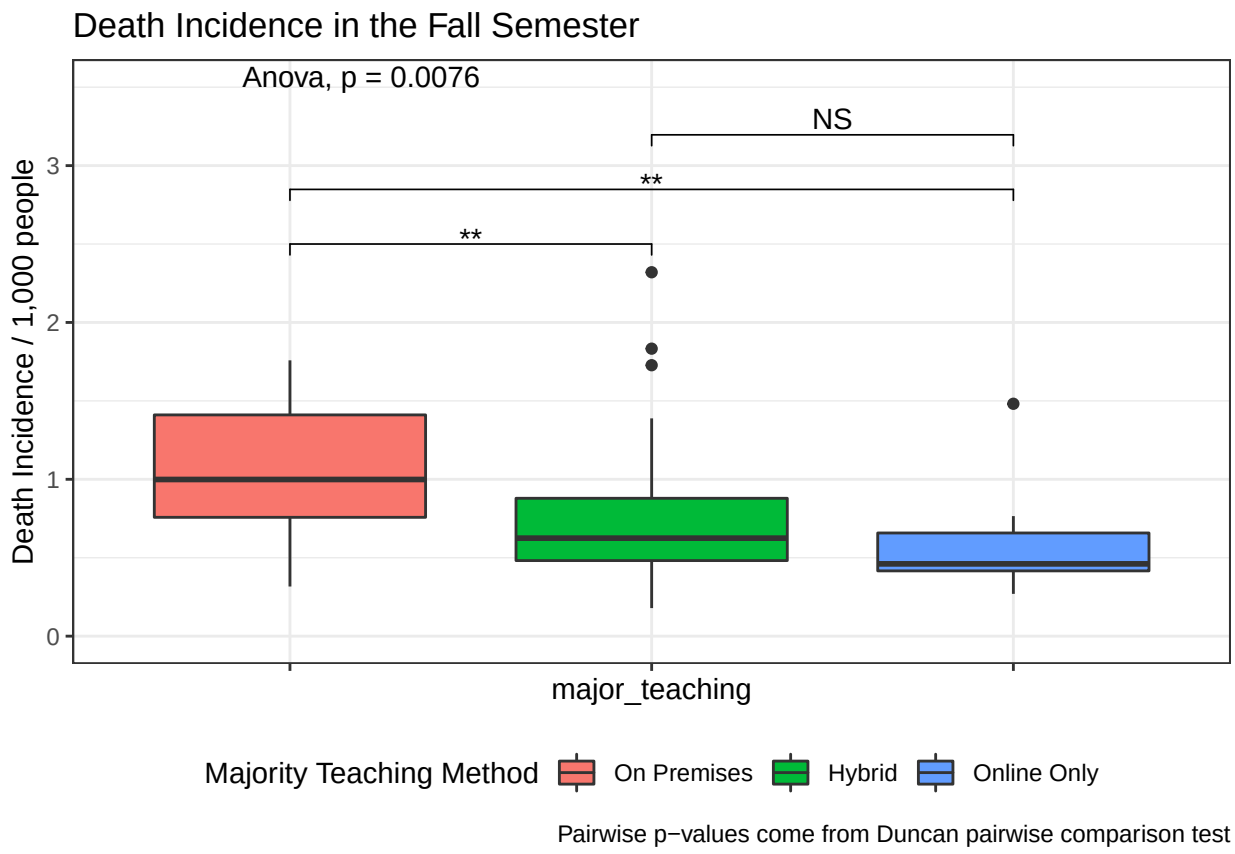
```r
case_policy_wide%>%
  group_by(DATE, major_teaching) %>%
  drop_na(major_teaching)%>%
  summarise(total_deaths = sum(CUMDEATHS),
            total_pop = sum(POPULATION),
            death_prop = total_deaths/total_pop,
            death_prop_upper = death_prop + z_cl*sqrt(death_prop*(1 - death_prop)/total_pop),
```

```
          death_prop_lower = death_prop - z_cl*sqrt(death_prop*(1 - death_prop)/total_pop),
          .groups = "drop") %>%
ggplot(aes(x = DATE, y = death_prop*1000, group = major_teaching))+
geom_rect(data=case_policy_wide[1,],
          aes(xmin=as.Date("2020/08/18"), xmax=as.Date("2020/12/15"),
              ymin=-Inf,ymax=Inf),
          color = NA,alpha=0.2, show.legend = F, fill = "orange") +
geom_line(aes(color = major_teaching),size = 1, alpha = .8) +
geom_ribbon(aes(ymin = 1000*death_prop_lower, ymax = 1000*death_prop_upper,
                fill= major_teaching),
            alpha = .3, show.legend = F)+
geom_vline(xintercept = date.intercept, linetype = "dashed") +
annotate("text",x = date.intercept,y = 1.5,
         label = date.intercept,
         hjust = 1.1) +
theme_bw() +
ggtitle("Death Incidences Increase Faster for Red Counties ")+
labs(x = "Date", y = "Cumulative Death Incidence / 1,000 people",
     subtitle = "Yellow area represents Fall Semester",
     color = "Majority Teaching Method") +
theme(legend.position = "")+
theme(legend.title = element_text(size=13),
      legend.text = element_text(size=13),
      axis.title = element_text(size=14),
      axis.text = element_text(size=15),
      legend.background = element_rect(fill = alpha("orange",0.0)),
      legend.key.size = unit(1.4,"lines"),title = element_text(size=12.9))+
theme(axis.line = element_line(colour = "black"),
  panel.grid.major = element_blank(),
  panel.grid.minor = element_blank(),
  panel.border = element_blank(),
  panel.background = element_blank())
```



Pairwise

```
ggplot(fall_cases,aes(y = death_incidence_per_1000, x = major_teaching)) +
  geom_boxplot(aes(fill = major_teaching))+
  stat_compare_means(method = "anova")+
  stat_pvalue_manual(stat.test, label = "p",y.position = 2.5, step.increase = 0.15)+
  ylim(c(0,3.5))+
  theme_bw()+
  labs(y = "Death Incidence / 1,000 people",
       fill = "Majority Teaching Method",
       title = "Death Incidence in the Fall Semester",
       caption = "Pairwise p-values come from Duncan pairwise comparison test") +
  theme(legend.position = "bottom",
        axis.text.x=element_blank())
```



Appendix 3: Exponential growth model

Data process

```
cases_slope <- read.csv("county_splines.csv", header = T)%>%
  select(COUNTY,DATE,POPULATION,CUMDEATHS,log_tot_deaths,
         tot.slope,NEWDEATHS,rev_NEWDEATHS,log_new_deaths,new.slope)
cases_slope$DATE <- as.Date(cases_slope$DATE)
# get majority teaching method wide_teaching_enroll
cases_slope_teach <-death_teaching%>%
  select(-DATE,-POPULATION,-CUMDEATHS,-NEWDEATHS)%>%
  distinct()%>%
  right_join(cases_slope,by=c("COUNTY"))
write.csv(cases_slope_teach,"cases_slope_teach.csv",row.names = F)
## ordering the teaching method factor to ensure the color order
cases_slope_teach$major_teaching <- factor(cases_slope_teach$major_teaching,
                                           levels = c("On Premises","Hybrid","Online Only"))
cases_slope_teach$DATE <- as.Date(cases_slope_teach$DATE)
maxB1 <- cases_slope_teach%>%
  group_by(COUNTY)%>%
  filter(DATE >= as.Date("2020-08-18") & DATE<=as.Date("2020-12-15"))%>%
  summarise(max_B1 = max(new.slope))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
avgB1 <- cases_slope_teach%>%
  group_by(COUNTY)%>%
  filter(DATE >= as.Date("2020-08-18") & DATE<=as.Date("2020-12-15"))%>%
  summarise(avg_B1 = mean(new.slope))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
## avg3w_B0 ## average B0 of the first 3 weeks of school reopening
## avg1w_2w_B0 ## OR average B0s between  2020-08-18 -7days and +14days
##[before the rate bounce back around the dashed line]
## avg3w_bf_B0 ## OR average B0s between  2020-08-18 -21days and 2020-08-18
##[before the rate bounce back around the dashed line]
avgB0 <- cases_slope_teach%>%
  group_by(COUNTY)%>%
  filter(DATE > as.Date("2020-08-18") & DATE<as.Date(major_opendate)+21)%>%
  summarise(avg3w_B0 = mean(new.slope))%>%
  left_join(cases_slope_teach%>%
  group_by(COUNTY)%>%
  filter(DATE > as.Date("2020-08-18")-7 & DATE<as.Date("2020-08-18")+14)%>%
  summarise(avg1w_2w_B0 = mean(new.slope)),by="COUNTY")%>%
  left_join(cases_slope_teach%>%
  group_by(COUNTY)%>%
  filter(DATE < as.Date("2020-08-18") & DATE>=as.Date("2020-08-18")-21)%>%
  summarise(avg3w_bf_B0 = mean(new.slope)),by="COUNTY")
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
cases_slope_teach_agg <- cases_slope_teach %>%
  drop_na(major_teaching)%>%
  group_by(DATE, major_teaching) %>%
  summarise(total_new_deaths = sum(rev_NEWDEATHS), .groups = "drop") %>%
  mutate(log_new_deaths = log(total_new_deaths + 1)) %>%
  group_by(major_teaching) %>%
  mutate(smooth.spline = smooth.spline(DATE,log_new_deaths,df = 398/28)$y,
         B = predict(smooth.spline(DATE,log_new_deaths,df = 398/28),deriv = 1)$y)
week3_after_start <- as.Date("2020/08/18") + 21
ggplot(cases_slope_teach_agg, aes(x = DATE, color = major_teaching)) +
  geom_line(aes(y = B), size = 1) +
  geom_rect(data = cases_slope_teach_agg[1,],
            aes(xmin=as.Date("2020/08/18"), xmax=as.Date("2020/12/15"),
                ymin=-Inf,ymax=Inf),
            color = NA,alpha=0.2, show.legend = F, fill = "orange") +
  geom_vline(xintercept = week3_after_start, lty = 2) +
  annotate("text",label = week3_after_start,
           x = week3_after_start, y = .05, hjust = 1.1)+
  labs(x = "Date", y = "Exponential Growth Coefficient",
       color = "Majority Teaching Method",
       caption = "Smoothing window set to every 4 weeks",
       subtitle = "Yellow area represents the fall semester (08/18 - 12/15)") +
  theme(legend.position = "bottom")+
  theme(axis.line = element_line(colour = "black"),
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
```

```
  panel.border = element_blank(),
  panel.background = element_blank())+
theme(axis.text = element_text(size=15),
      axis.title=element_text(size=15),
      legend.text = element_text(size=13))
```