P4. GROUP PROJECT REPORT

MENTAL HEALTH IN THE MIDST OF COVID

ISM 6137 // MICHAEL ZEOLLA, TATYANNA VILLEGAS, TAPIWA GWEDE

Contents

Project

RESULTS

Executive Summary

According to Centre for Disease Control (CDC), mental health refers to our “emotional, psychological and social well-being”. It is so integral in people’s lives and determines how we handle stress, relate to others, and make healthy choices from early childhood to adulthood.

Since late 2019, a highly contagious and deadly pandemic ‘COVID-19’ emerged and mental health illnesses and symptoms have never been this prevalent. It is reported that 4 in 10 adults in the United States have reported symptoms of anxiety or depression during the pandemic (year 2020), a sharp rise from 1 in 10 reported in the previous year. This new dimension creates two major relevant issues that need to be addressed efficiently in tandem with healthcare professionals, mental health experts, state/federal government officials and others.

Based on the project analysis, COVID case rate had the most significant impact (0.693 per 100k) on the percent of symptoms of anxiety and depression (SAD) in adults. Policymakers are encouraged to implement policies that focus on preventing the spread of COVID, especially if they involve closing of daycares, gyms, bars, etc which were reported to have little to no impact on SAD. Closing of non-essential businesses, stay-at-home orders and mask mandates had some impact. A deep-dive into policy conditions is recommend for states that performed the best (ND, SD, WI) and the worst (OR, NM, LA). Local experts should be consulted to consider economic or other impacts when making policy changes.

Problem Definition and Significance

Mental health illness is a multi-dimension societal issue that needs all-hands-on-deck approach. With this project we want to influence policy makers and local and state government officials to implement policies that address this issue in a concerted effort. Experts in this field believe that unattended mental health problems have a negative influence on homelessness, poverty, employment, safety, and the local economy. Cumulatively, it impacts the overall productivity of local businesses and health care costs and lead to community disruption. According to the Tacoma-Pierce County Health Department, in 2012, "the expenditures for mental health care in the U.S. cost about $83.6 billion" and much of the economic burden of mental illness emanates from indirect costs that last a lifetime.

Inconclusive and controversial studies also suggest that approximately 25% of mass murderers had exhibited a mental illness, and a Federal Bureau of Investigation (FBI) "study of 63 active-shooter incidents between 2000 and 2013 found that 25% of shooters were known to have been diagnosed with a mental illness of some kind, ranging from minor to more serious disorders". Consistently true to all this is that "tense interactions between people with mental illness and law enforcement officers often result in avoidable arrests and sometimes ending in fatal shootings by the police" (Harvard Review of Psychiatry, 1/2 - 2021 - Volume 29 - Issue 1).

Now, as symptoms of anxiety or depression during the COVID-19 pandemic are more prevalent, it is an alarm to brace for the consequences that will stay with our societies long after COVID-19 is toned-down. With a good understanding of the correlation between rising COVID-19 cases and symptoms of anxiety or depression, it is reasonable to implement policies that reduce the spread and effects of COVID-19, thereby reducing prevalence of depression in society.

Prior Literature

Even though COVID-19 is a new and unique disease, mental health during this pandemic has already been researched, studied, analyzed and there are a number of articles and literature that try to address this.

One article looks at the "Correlation between Preventive Health Behaviors and Psycho-Social Health Based on the Leisure Activities of South Koreans in the COVID-19 Crisis". In considering demographics when looking at variables for mental stress results showed higher mental health and preventative behaviors were found for the following: women, teens/ages 60+, married, and those who considered themselves healthy (Young-Jae Kim and Jeong-Hyung Cho: International Journal of Environmental Research and Public Health, 7 June 2020). Another article, "Prevalence of depression during the COVID-19 outbreak: A meta-analysis of community-based studies," showed that prevalence of depression was 25% in 2020 compared to 3.4% in 2017. Rates seem to have increased due to confinement measures imposed to control COVID-19 contamination (Juan Bueno-Notivol et al: International Journal of Clinical and Health Psychology Volume 21, Issue 1, January – April 2021) .

The study "Frontline nurses' burnout, anxiety, depression, and fear statuses and their associated factors during the COVID-19 outbreak in Wuhan, China," the article looks into what are the levels of, relationships, and differences in burnout, anxiety, depression, and fear between nurses of various sociodemographic backgrounds. This goes further into examining the interventions to improve mental health, build self-efficacy and resilience among frontline nurses who were caring for COVID-19 patients

as this warrants support from policymakers. (Deying Hua et al: EClinical Medicine; The lancet Volume 24, July 2020).

As China was at the center of this pandemic, one article looks at "Prevalence and Psychosocial Correlates of Mental Health Outcomes among Chinese College Students during the Coronavirus Disease (COVID-19) Pandemic." (Xinli Chi et al, August 2020). Of the 2,500 invited Chinese university students, 2,038 completed the survey and results showed that older age, knowing people who had been isolated, more ACEs (Adverse Childhood Experiences), higher level of anxious attachment, and lower level of resilience all predicted primary outcome (all $p < 0.01$). There was a high prevalence of posttraumatic, anxiety and depressive symptoms. Yet in another article, a study tried to identify what factors are relevant to the prevalence of anxiety and depression of healthcare professionals. The results showed that COVID-19 exposure, epidemiological issues, material resources, human resources, and personal factors are factors that could influence the mental health of healthcare professionals (M. D Braquehais et al, 01 July 2020).

In a May 2020 article "Mental health during the COVID-19 pandemic: Effects of stay-at-home policies, social distancing behavior, and social resources" by Marroquin, B et al, the authors ask what the relationship is between social distancing (stay-at-home + personal distancing) and social resources (perceived social support + network size) with mental health (depression, GAD, intrusive thoughts, insomnia, acute stress). In a survey of 435 people, results showed that health scores increased from February to March 2020, social support was associated with lower scores, and stay-at-home and personal distancing associated with higher scores. In Germany, one article looked at volume of calls for a telephone counseling hotline for 91 call centers from 1/1/2020 to 4/28/2020 and results indicate that the number of calls were significantly higher during times of lockdown. Stricter states had significantly higher volume than less-strict states. "Lost in lockdown? COVID-19, social distancing, and mental health in Germany". (Armbruster, S et al, May 2020).

Data Source and Preparation

There were three main data sources used for this project. Information on the topic of interest, anxiety and depression, was provided by the CDC Household Pulse Survey. It is a 20-minute survey that collects information on the frequency of anxiety and depressive symptoms of adults. Modified versions of the two-item Generalized Anxiety Disorder (GAD-2) and Patient Health Questionnaire (PHQ-2) were used. The GAD-2 and PHQ-2 are recognized as effective and useful tools for screening anxiety and depression. The specific questions as well as their previous performance can be seen below. The survey was conducted several times starting from April 23rd, 2020 to the most recent being March 29th, 2021. Weekly or biweekly estimates were gathered across groups such as age, gender, and state reporting the percentage of adults experiencing symptoms of anxiety of depression that have been shown to be associated with diagnoses of generalized anxiety disorder or major depressive disorder. Respondents were randomly selected with the sample size ranging from 40,000 to 120,000 each time.

Generalized Anxiety Disorder two-item

Over the last 7 days, how often have you been bothered by the following problems…	Not at all	Several days	More than half the days	Nearly every day
Feeling nervous, anxious, or on edge?	O	O	O	O

Not being able to stop or control worrying?	O	O	O	O

Patient Health Questionnaire two-item

Over the last 7 days, how often have you been bothered by…	Not at all	Several days	More than half the days	Nearly every day
Having little interest or pleasure in doing things?	O	O	O	O
Feeling down, depressed, or hopeless?	O	O	O	O

Each response assigned a numerical value: not at all = 0, several days = 1, more than half the days = 2, and nearly every day = 3. Scores for GAD-2 and PHQ-2 were obtained by adding the score of both questions, where a total of 3 or more has been shown to be associated with diagnoses of anxiety and major depressive disorder respectively.

Performance of Question Types as a Screening Tool for Associated Disorders

Type	Sensitivity	Specificity
GAD-2	97%	67%
PHQ-2	86%	83%

Sensitivity (the ability of a test to correctly identify those with a condition) and specificity (the ability of a test to correctly identify people without the disease) for questions types listed, showing they are effective to use as a screening tools for the topic of interest.

To analyze the impact of COVID-19 during this time, the CDC US Cases and Deaths by State dataset was used. State, local, and territorial public health departments verify and report cases and death of COVID-19 to the CDC, which aggregates the counts for states daily online. Case and death counts were collected starting January 1st, 2020 and are still being updated with the most recent as April 26th, 2021.

Since the previous datasets had the option to view data at the state level, the third dataset looks at information on the state level as well. It looks at possible factors that could have an impact on anxiety and depression levels during the pandemic, specifically state policies implemented at the time. The COVID-19 US State Policy Database contains information on statewide mandates on all 50 states as well as the District of Columbia. Over 200 policies are included in the dataset, regarding topics such as closures, stay at home orders, mask mandates, healthcare delivery, food security, unemployment and more. The dataset also included information on state characteristics, such as population, percent unemployed and number of mental healthcare professionals available.

Various data manipulations were performed to include information from all data sources within the same dataset for analysis. The final dataset used the format of time periods provided by the dependent variable across different states. As a result, the Pulse Survey was filtered to look only at values grouped by State. There was also a need to create a consistent time period length. Consecutive weekly values were grouped when possible to create consistent biweekly time periods. This also shortened the date range of values included in the analysis. Since weekly time periods were grouped together, the average was taken from their associated y-values. After grouping there were 19 total time periods across 51 regions, leading to 969 total rows.

Between cases and deaths, the former was chosen as it reflected the state of COVID-19 apart from just high-risk groups. Daily values for new cases were aggregated for a time period range. However, as the new cases column from the raw data could sometimes be negative (if more cases were disproven than initially reported on that day), the number of new cases was calculated from the difference in total cases from the start and end date of a time period. The case number was then divided

by the state population (more recent 2019 estimates were used instead of 2018 values from raw data) and multiplied by 100,000 to create the rate of cases (per 100k).

State policies within a time period were coded as binary (0/1) values after looking at the start and end dates (if applicable) of that policy. The number of days a state policy applied to a certain time period was calculated. Then an arbitrary 'cut-off' values was determined – in this case, if there were any days within the time period where the state policy applied, it would be marked as 1, otherwise 0.

Variable Choice

The final selection of independent (IV) and dependent variables (DV) are listed below. The majority of variables were taken from the policy dataset, though the selection focused more on factors of isolation, as it featured as a major topic in previous literature and helped narrow down the large amount of policies available.

DV: Percentage of Pulse Survey responders experiencing symptoms of Anxiety or Depression

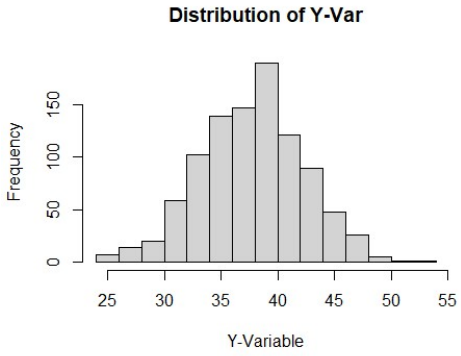
Predictor	Effect	Rationale
Are_Day_Cares_Closed?	+	When day cares are closed, parents who relied on these services must watch over their children in within the household.
StayAtHome_Order?	+	As they restrict people from leaving their homes for non-essential activities, there's an increased sense of restriction on the individual's activities.
Are_NonEssential_Businessess_Closed?	+	People are limited on doing the activities that they enjoy.
Are_Restaurants_Closed?	+	Restaurants are a source of pleasure and socialization for many.
Are_Gyms_Closed?	+	Exercise is essential part of maintaining good mental-health and many rely on gyms as a place to exercise.
Are_Bars_Closed?	+	Bars are also a source of pleasure and socialization for many.
Allowed_Audio_Telehealth	−	Having telehealth as an option can relieve those who are worried about seeing a doctor in person.
Expansion_of_Telehealth_Medicaid	−	More people having access to telehealth would be helpful during the tough times of the pandemic.
Mask_Mandate_All	+/−	Having a mask mandate can relieve people's concerns on preventative behaviors of COVID-19, while others may feel restricted being forced to wear a mask.
Mask_Mandate_Employees	+/−	Having a mask mandate can relieve people's concerns on preventative behaviors of COVID-19, while others may feel restricted being forced to wear a mask.
Mental_Health_Workers (per 100k)	−	Having more mental health workers would give people the opportunity to improve their mental health.
Cases (per 100k)	+	A higher number of cases in the state will heighten fears about the virus.
Months	+/−	Winter months are likely to be higher due to seasonal depression in comparison to others.

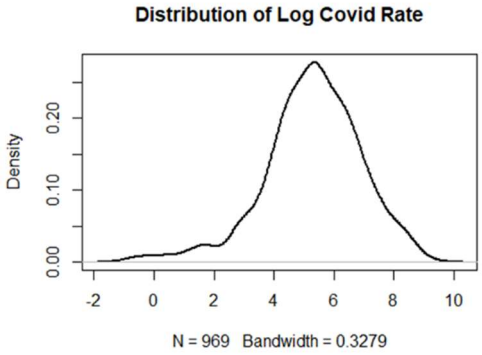
State	+/−	Y-variable likely to vary across different states based on policies implemented.

Final selection of independent variables listed in the table in regard to dependent variable listed at the top. Variable names listed here are more descriptive than as they appear in R code.

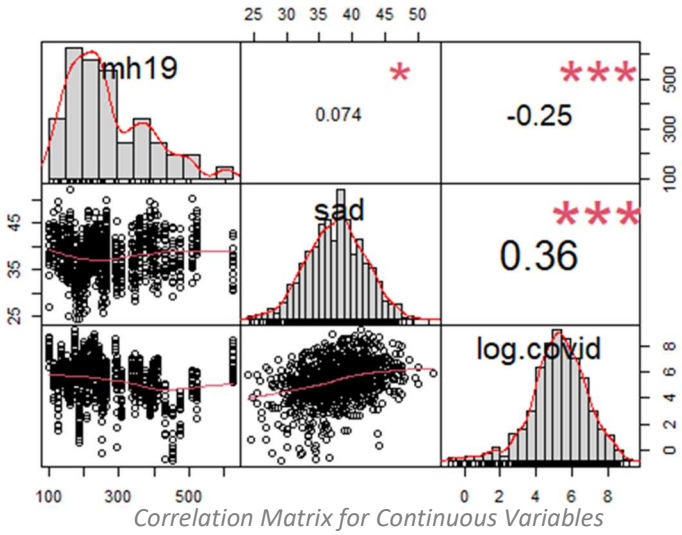
Descriptive Analysis and Data Visualizations

When beginning descriptive analysis, the first steps taken were to see how certain variables were distributed—especially the dependent variable, the percentage of Pulse survey responders experiencing symptoms of anxiety and/or depression. Its distribution was fairly normal. Another relevant variable was the COVID-19 rate variable (count of cases per 100,000 population). Since this is a count variable, the expectation was for it to be distributed to a Poisson curve, which it certainly was. A log transformation on COVID rate was applied and did a successful job normalizing it. When comparing the correlations of these three variables, no pairs of variables are highly correlated so it will be safe to include these in our analysis and avoid multicollinearity. "Sad" refers to "symptoms of anxiety/depression," which is what our dependent variable is coded as in our datab

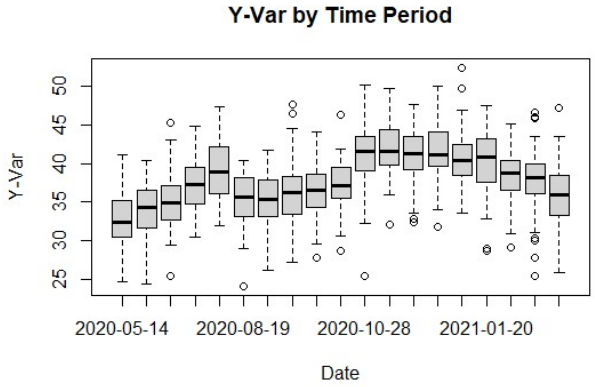


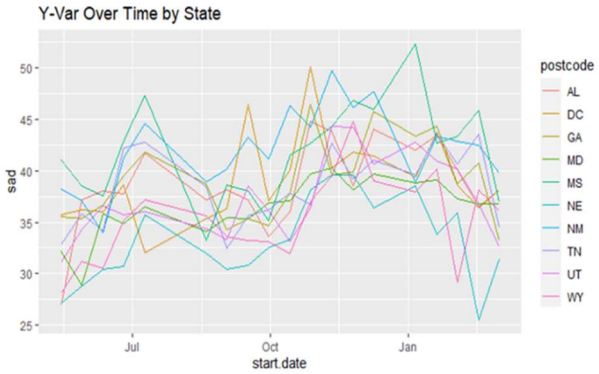


Because our data is over a span of time, it is important to analyze how time might influence our data. When we plot the average dependent variable for each of the 19 time periods, the values for the most part stay between 35 and 45 percent with no clear linear trend. However, there still appears to be some sort of pattern. There is a bit of an "M" shape for increases and decreases. Since these are the averages for all 51 states pooled together, we examine to see if the separated-out states experience common trends. The states dip and dive in ways different from each other, so it is difficult to say that all states experience this same trend. However, since there is still reason to believe there is some sort of seasonal trend.

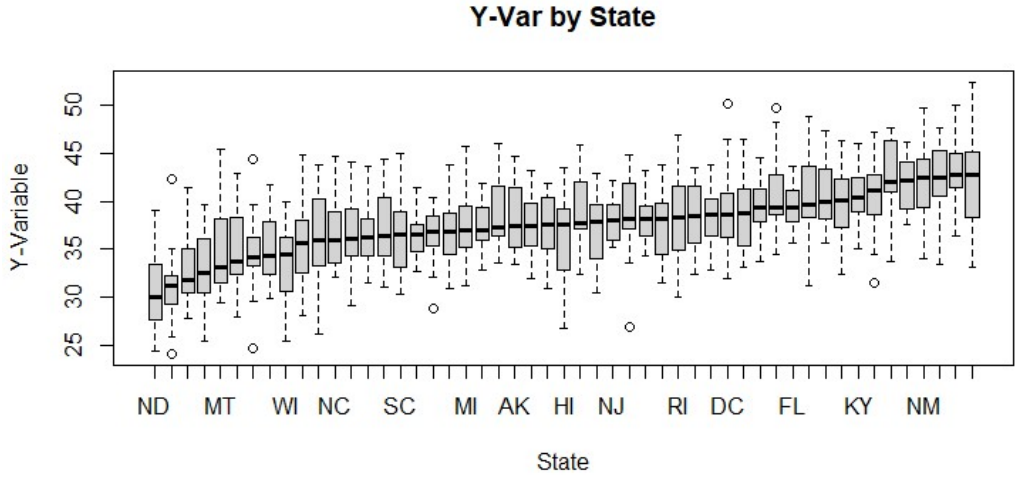


Correlation Matrix for Continuous Variables





Our data is grouped by 51 "states" (50 states plus Washington, DC). A boxplot of the dependent variable for each state confirms that the percentage of survey responders with symptoms does vary across each state. This will be important to consider when modeling the data because since we know each state behaves differently, we cannot pool all the data together without distinguishing the state level.



Models

With a dataset grouped by 51 states and 19 time periods, a panel model where the upper level is state and the lower level is time period is appropriate. There were four models that we examined.

- Panel model with fixed effect on state:
- Panel model with random effect on state:
- Panel model with a two-way effect on state and time:
- Panel model with fixed effect on state, with addition of two lag variables:

The lag variables are by time period. For example, Florida had dependent variable values of 36.30, 40.35, and 40.75, respectively, in the first three time periods. For time period three, Florida's dependent variable value is 40.75, its lag1 value is 40.35 (previous time period), and its lag2 value is 36.30 (two time periods ago). This is to account for any relationship between previous values and the current value, since theoretically the mental health of the previous time period could have an impact on the next.

When picking a final model for interpretation, we focused on deciding between models one and four. Both are fixed effects models, with the only difference being the inclusion of lag variables. Fixed effects seemed more appropriate with the rationale that we are observing the effects of state policies within the 51 states. On top of this, the 51 states are not random observations from a random sample; the 51 states are the 51 states. To decide between our two fixed effect panel models, we first compared the coefficients of the two models. How strong are the lag predictors?

The lag variables are by time period. For example, Florida had dependent variable values of 36.30, 40.35, and 40.75, respectively, in the first three time periods. For time period three, Florida's dependent variable value is 40.75, its lag1 value is 40.35 (previous time period), and its lag2 value is 36.30 (two time periods ago). This is to account for any relationship between previous values and the current value, since theoretically the mental health of the previous time period could have an impact on the next.

When picking a final model for interpretation, we focused on deciding between models one and four. Both are fixed effects models, with the only difference being the inclusion of lag variables. Fixed effects seemed more appropriate with the rationale that we are observing the effects of state policies within the 51 states. On top of this, the 51 states are not random observations from a random sample; the 51 states are the 51 states. To decide between our two fixed effect panel models, we first compared the coefficients of the two models. How strong are the lag predictors?

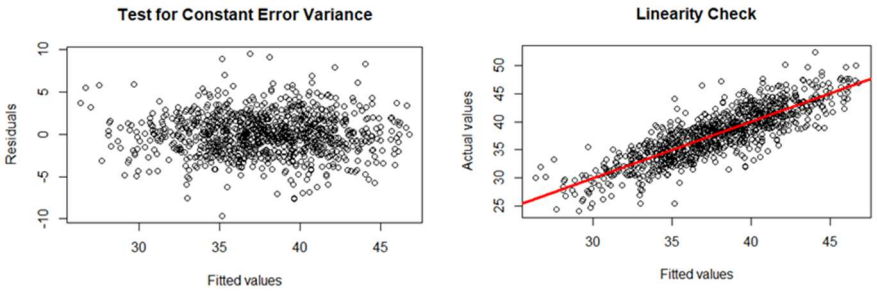
	Dependent variable:	
	sad	
	(1)	(2)
month2	-1.733*** (0.374)	-1.623*** (0.389)
month3	-3.426*** (0.465)	-3.231*** (0.499)
month5	-5.567*** (0.686)	
month6	-3.889*** (0.537)	-3.133*** (0.721)
month7	-0.808* (0.442)	-0.384 (0.545)
month8	-3.988*** (0.491)	-3.684*** (0.535)
month9	-3.619*** (0.418)	-3.318*** (0.493)
month10	-2.803*** (0.386)	-2.476*** (0.467)
month11	1.136*** (0.355)	1.337*** (0.399)
month12	0.923*** (0.356)	0.901** (0.360)
cldaycr	-0.934 (0.617)	0.990 (1.627)
clbsns	0.106 (0.719)	
clgym	-0.123 (0.373)	-0.102 (0.457)
clbar	-0.007 (0.343)	-0.172 (0.431)
tlhlaud	-1.705 (1.957)	
stayhome	0.090 (0.292)	0.076 (0.520)
fm_all	0.317 (0.349)	0.489 (0.412)
fm_emp	0.530 (0.400)	0.298 (0.484)
clinrst	-0.248 (0.452)	0.553 (0.560)
log(covid.rate)	0.693*** (0.150)	0.739*** (0.174)
lag1		-0.004 (0.035)
lag2		0.036 (0.035)
Observations	969	867
R2	0.547	0.472
Adjusted R2	0.511	0.427
F Statistic	54.134*** (df = 20; 898)	37.560*** (df = 19; 797)
Note:		*p<0.1; **p<0.05; ***p<0.01

Stargazer Output of Marginal Effects of Predictors

The coefficients for the two lag variables are very small, meaning the previous time periods have close to no predictive power on the current time period's dependent variable value. Because of this, we will interpret the first panel fixed effects model (shown under "1" in the Stargazer RStudio output).

Quality Checks

Assumptions for homoscedasticity, linearity, multivariate collinearity, no autocorrelation, and no multicollinearity were checked, and all were met except for multicollinearity. The assumption of homoscedasticity looks for a constant error variance, which can be identified in a residual versus fitted values plot. If the residuals get larger or smaller as the fitted values increase, this means that the model's variance is not constant. In the residuals versus fitted values plot for our model, no pattern in the residuals is seen and the variance is constant. Homoscedasticity is passed. The relatively flat plot across zero also suggests linearity in the residuals.



Linearity refers to the assumption that there is a linear relationship between the predictor variables and the dependent variable. As mentioned above, the residuals versus fitted values plot confirmed linearity in the model. This assumption can also be checked by seeing if the relationship between actual dependent variable values and fitted values is linear.

Multivariate normality checks if the residuals are normally distributed. A histogram of the residuals confirms this, as well as a straight, linear result from a quantile-quantile plot.

Autocorrelation checks to see if residuals in the model are correlated because of the response variable's values being dependent on each other. A Durbin-Watson test for linear autocorrelation was conducted for our model, where a score close to two is indicative of no autocorrelation. The Durbin-Watson statistic for our model was 2.101, so this assumption was passed.

Multicollinearity checks if certain predictors are strongly correlated with each other enough to the point of creating bias. This assumption is usually checked by viewing the Variance Inflation Factor (VIF) of each predictor in R, although this test was given an error due to aliases. Aliases occur in VIF tests if one variable is perfectly collinear with another. An alias check shows that the issue is in levels of our state fixed effect, specifically for Wyoming and West Virginia. We are not sure what could cause this, especially since values are different for those two states by policy. Because dropping these two states would inaccurately represent the data, we chose to leave this alone.

Recommendations

Interpretations are from the coefficients from the Models section of the report. There are four policies that increase the percentage of survey responders experiencing symptoms: the closing of essential businesses, stay-at-home orders, mask mandates for all people, and mask mandates for employees. The implementation of these policies increased the dependent variable by 0.106, 0.090, 0.317, and 0.530 percentage points, respectively. For clarification, this is not a percentage increase, but

rather a marginal increase in our dependent variable, which is a percentage. These effects are not that significant; however, it is important to note that in a state with millions of people, a fraction of a percent could be the difference maker of tens of thousands of people experience symptoms of anxiety and/or depression.

The coefficient for the log of COVID-19 cases per 100,000 is 0.693. This means that for a 1% increase in cases, the dependent variable on average increases by 0.693. This is a very significant effect and it is understandable that the uncertainty and fear of the virus could have this kind of effect. For perspective, case rates during times of "waves" are increasing at rates far higher than 1%. COVID-19 case rate is the most significant predictor that we have.

With this in mind, we would recommend state policy makers to:

- Consider implementing some of these state policies to mitigate the spread of COVID-19. Case rate is the strongest predictor of the dependent variable, so this effort will both mitigate spread while also improving mental health of the population. The effects from state policies are not as significant, so there is less worry about consequential effects on mental health.
- Especially consider implementing the closing of day cares, gyms, bars, and restaurants, which are variables with negative effects on the dependent variable (which is a good thing) before considering the closing of essential businesses, stay-at-home orders, and mask mandates.
- Examine the states with the best (North Dakota, South Dakota, Wisconsin, etc.) and worst (Oregon, New Mexico, Louisiana, etc.) fixed effects on the dependent variable. What do these states have in common and why might they influence mental health?

Ultimately, we would recommend consulting with an economist before implementing these policies. Our model examines these policies' effects on mental health, but these policies affect more than just mental health. Early in the pandemic, for example, the unemployment levels skyrocketed and a lot of damage was done to the stock market after many of these policies were implemented. It is important to examine what effects these policies may have on things other than mental health.

References

"Anxiety and Depression." *Centers for Disease Control and Prevention,* 5 May 2021, https://www.cdc.gov/nchs/covid19/pulse/mental-health.htm

Arroll, Bruce et al. "Screening for depression in primary care with two verbally asked questions: cross sectional study." *BMJ (Clinical research ed.)* vol. 327, no. 7424, 15 November 2003, doi:10.1136/bmj.327.7424.1144

"Learn About Mental Health." *Centers for Disease Control and Prevention,* 26 January 2018, https://www.cdc.gov/mentalhealth/learn/index.htm

Metzl, Jonathan M. MD, PhD;, et al. "Mental Illness, Mass Shootings, and the Future of Psychiatric Research into American Gun Violence." *Harvard Review of Psychiatry,* vol. 29, no. 1, 2 January 2021, doi: 10.1097/HRP.0000000000000280

"Learn About Mental Health." Centers for Disease Control and Prevention, 26 January 2018, https://www.cdc.gov/mentalhealth/learn/index.htm

Raifman, Julia, Nocka, Kristen, Jones, David, Bor, Jacob, Lipson, Sarah, Jay, Jonathan, … Galea, Sandro. COVID-19 US State Policy Database. Ann Arbor, MI: Inter-university Consortium for Political and Social Research [distributor], 2021-04-28. https://doi.org/10.3886/E119446V84

Sapra, Amit, et al. “Using Generalized Anxiety Disorder-2 (GAD-2) and GAD-7 in a Primary Care Setting.” Cureus, vol. 12, no. 5, 21 May 2020, doi: 10.7759/cureus.8224

“Unattended Mental Health’s Impact on Society.” Tacoma-Pierce County Health Department, March 2016, https://www.tpchd.org/home/showpublisheddocument?id=664

“United States COVID-19 Cases and Deaths by State over Time.” Centers for Disease Control and Prevention, 28 April 2021, https://data.cdc.gov/Case-Surveillance/United-States-COVID-19-Cases-and-Deaths-by-State-o/9mfq-cb36

Appendix

Creation of “SAD” Variable

```
rm(list=ls())                          #clear environment if needed
gc()                                   #garbage collection
setwd("~/SDM Project/Y-Variable")      #set working directory to where file is
located (CTRL+Shift+H)

library(rio)
df=import("PulseDataY.csv")            #load file into dataframe
colnames(df)=tolower(make.names(colnames(df)))
df = subset(df, indicator=="Symptoms of Anxiety Disorder or Depressive
Disorder"
              & group=="By State")     #filter indicator for chosen y and by
state

uniqState = length(unique(df$state)) #there are 51 uniq states in dataset
uniqState

attach(df)
t = as.numeric(df$state == df$subgroup)          #are state and subgroup
columns the same, after filtering?
temp = as.data.frame(t,stringsAsFactors=F)
table(temp)                                      #yes, they are (only 1s/Trues
show, no Falses that columns don't match)
df = df[, -c(1:2,4)]                             #remove unnesscary columns

#find number of days for each period
#confirmed hour input is same for all dates; no need to include hour
df$time.period.end.date = as.Date(df$time.period.end.date, format="%m/%d/%Y")
df$time.period.end.date = as.POSIXct.Date(df$time.period.end.date,
format="%m/%d/%Y")
df$time.period.start.date = as.Date(df$time.period.start.date,
format="%m/%d/%Y")
df$time.period.start.date = as.POSIXct.Date(df$time.period.start.date,
format="%m/%d/%Y")
str(df)

df$time.period.length =
difftime(df$time.period.end.date,df$time.period.start.date, units=c('days'))
table(df$time.period.length) #days are in weekly or bi-weekly format
(assuming 'weekends' are not included)

newdata = df[order(as.Date(df$time.period.start.date, format="%m/%d/%Y")),]
#order dataset by earliest startdate
temp = (rle(as.character(newdata$time.period.length)))          #find
consecutive 5/12 groups
```

```r
tPeriods = data.frame(temp[["lengths"]], temp[["values"]],stringsAsFactors=F)
table(df$time.period.start.date)                                    #notice each
time period has 51 values for each state
tPeriods$temp...lengths... = tPeriods$temp...lengths.../uniqState     #divide
by # of uniq states to reflect true length
View(tPeriods)                                                      #(1) 12
period for all states, (11) 5 periods, then (14) 12 periods
plot(df$time.period.start.date,df$time.period.length)           #plot reflects
table of 5/12 over time

# View week split, method 1, table temp
temp = subset(df, state=="Florida")
temp =
subset(temp,select=c(time.period.start.date,time.period.end.date,time.period.
length))
View(temp)

# View week split, method 2, table t
t = as.data.frame(table(newdata$time.period.start.date))
t = cbind(t,table(newdata$time.period.end.date))
t = t[, -c(2,4)] #drop frequency
count = 0
t$newCol = NA
for (i in 1:nrow(tPeriods)) {
  n = tPeriods$temp...lengths...[i]
  for (j in 1:n) {
    count = count + 1
    t$newCol[count] = tPeriods$temp...values...[i]
  }
}
View(t)

#Combining the dataset
a = temp$time.period.start.date[1] #decide to remove first 2 time periods
b = temp$time.period.start.date[2] #to be able to work with even # of 5
periods
newdata = subset(newdata, time.period.start.date!=a &
time.period.start.date!=b) #remove from whole data
temp = subset(temp, time.period.start.date!=a & time.period.start.date!=b)
#remove from week split
w = subset(temp, time.period.length==5)          #extract all the 5 day time
periods from week split
weekly = subset(newdata, time.period.length==5) #and whole data


uniqState = unique(newdata$state)  #create uniq array of states
biweekly = data.frame(seq(1,(nrow(w)/2)*length(uniqState)))   #create
dataframe of length new biweekly periods * each state
colnames(biweekly) = c("state")    #rename first column, as values will be
replaced later
biweekly$state = NA                #set column values as blank for now
biweekly$start.date = w$time.period.start.date[1]  #set column values in
dateformat for now
biweekly$end.date = w$time.period.end.date[1]
biweekly$value = NA

count = 0    #for each row in biweekly dataset
```

```
# repeat the same process for each state
for (i in uniqState) {
  #print(i)
  #for every 2 rows in weekly dataset, collect dates and value
  for (j in seq(1,nrow(w),2)){
    count = count + 1
    r = which(newdata$state==i &
newdata$time.period.start.date==w$time.period.start.date[j])
    s = which(newdata$state==i &
newdata$time.period.start.date==w$time.period.start.date[j+1])

    biweekly$state[count] = i
    biweekly$start.date[count] =  w$time.period.start.date[j]
    biweekly$end.date[count] = w$time.period.end.date[j+1]
    biweekly$value[count] = (newdata$value[r]+newdata$value[s])/2
    #print(paste(i,r,s))
    #print(paste(i,(newdata$value[r]+newdata$value[s])/2))
  }
}

#rbind biweekly datatset with following 12 period rows of regular
bw = subset(temp, time.period.length==12) #extract remaining 12 day time
periods from week split
regular = subset(newdata, time.period.length==12) #and whole data
regular = regular[, c(1,5:7)] #keep state, date and value columns only
colnames(regular) = c("state","start.date","end.date","value") #rename date
columns
#5 time periods from biweekly + 14 from regular = 19 periods, *51 states =
969 rows expected
df = rbind(biweekly,regular)
df = df[order(as.Date(df$start.date, format="%m/%d/%Y")),]  #order by
startdate
weeksplit = subset(df, state=="Florida")                              #extract
new weeksplit
weeksplit = weeksplit[, c(2:3)]                                          #keep
only the dates
library("writexl")
write_xlsx(df,"Y-Var.xlsx")
```

Creation of "Covid Rate" Variable

```
#Load the CDC Cases Dataset
rm(list=ls())
library(rio)
CDC_set <- import("CDC_Cases_Final.xlsx")
#Use "df" dataframe from Tatyanna's code as the final Pulse dataset


#Create a subset from 5/14/2020 to 3/15/2021, the range of what our final
dataset will be
CDC_set$Date <- as.Date(CDC_set$submission_date)
covid <- subset(CDC_set, CDC_set$Date > as.Date("2020-05-13")
                & CDC_set$Date <= as.Date("2021-03-15"))

#I notice that we have some negative values for new_case...why?
zeros <- subset(covid, covid$new_case < 0)
```

```r
nrow(zeros) #56 observations of zero or below
hist(zeros$new_case) #Most are slightly below zero but some are near -35,000
#QUOTE FROM THE CDC WEBSITE ON THE DATASET:
#"A jurisdiction might even report a negative number of probable cases on a
given day...
#...if more probable cases were disproven than were initially reported on
that day."

#Perhaps instead of looking at the sum of new_case for the date range, I
could look at...
#...total_cases at the start and the end dates and subtract them? I will do
this.

unique(df$state) #51 values for Pulse (50 states + Washington DC)
unique(covid$state) #60 values for Covid (50 states + terrorities like DC,
Micronesia, Guam, etc.)
#The final number of areas that we use depends on what our policy dataset
looks like.
#The policy dataset also looks at 51 areas (50 + Washington DC), so for now I
will do that.
#Eventually if we want to drop DC and look at only 50 states, we can adjust
accordingly.


#Create a vector of the 51 state codes (including District of Columbia as
"DC")
state_codes <- c('AL',  'AK',   'AZ',   'AR',   'CA',   'CO',   'CT',   'DC',
'DE', 'FL',   'GA',   'HI',   'ID',
                 'IL',  'IN',   'IA',   'KS',   'KY',   'LA', 'ME', 'MD',
'MA',   'MI',   'MN',   'MS',   'MO',
                 'MT',  'NE',   'NV',   'NH',   'NJ',   'NM',   'NY',   'NC',
'ND',   'OH',   'OK', 'OR', 'PA',
                 'RI',  'SC',   'SD',   'TN',   'TX',   'UT',   'VT',   'VA',
'WA',   'WV',   'WI',   'WY')
#Repeat the state codes 19 times (for 19 time periods)
state <- rep(state_codes, 19)
state <- sort(state)

#Create a vector of the start dates
start_dates <- as.Date(c('2020-05-14', '2020-05-28', '2020-06-11', '2020-06-
25', '2020-07-09', '2020-08-19',
                         '2020-09-02', '2020-09-16', '2020-09-30', '2020-10-
14', '2020-10-28', '2020-11-11',
                         '2020-11-25', '2020-12-09', '2021-01-06', '2021-01-
20', '2021-02-03', '2021-02-17',
                         '2021-03-03'))
start_date <- rep(start_dates, 51)

#Create a vector of the end dates
end_dates <- as.Date(c('2020-05-26', '2020-06-09', '2020-06-23', '2020-07-
07', '2020-07-21', '2020-08-31',
                       '2020-09-14', '2020-09-28', '2020-10-12', '2020-10-
26', '2020-11-09', '2020-11-23',
                       '2020-12-07', '2020-12-21', '2021-01-18', '2021-02-
01', '2021-02-15', '2021-03-01',
                       '2021-03-15'))
end_date <- rep(end_dates, 51)
```

```r
#Create data frame
df_MZ <- data.frame(state, start_date, end_date)

#Create new variable for covid cases
#I do this by creating a variable for what the total cases was on the start
date,...
#...and then a variable for total cases on the end date.
#The cases variable will the difference of these two values.
df_MZ$start_cases <- length(df_MZ$state)
for (i in seq_along(df_MZ$state)) {
  df_MZ$start_cases[i] <- covid$tot_cases[covid$submission_date ==
df_MZ$start_date[i]
                                  & covid$state == df_MZ$state[i]]
}
df_MZ$end_cases <- length(df_MZ$state)
for (i in seq_along(df_MZ$state)) {
  df_MZ$end_cases[i] <- covid$tot_cases[covid$submission_date ==
df_MZ$end_date[i]
                              & covid$state == df_MZ$state[i]]
}
df_MZ$cases <- length(df_MZ$state)
for (i in seq_along(df_MZ$state)) {
  df_MZ$cases[i] <- df_MZ$end_cases[i] - df_MZ$start_cases[i]
}

#Now we have cases. If we want to do a covid rate, like covid cases per
100,000 people...
#...then I need to upload a dataset of state populations
#First dataset under tables: https://www.census.gov/data/tables/time-
series/demo/popest/2010s-state-total.html
#I use the Census' 2019 estimate
populations <- c(731545,    4903185,    3017804,    7278717,    39512223,
5758736,    3565287,    705749, 973764,
                 21477737,  10617423,   1415872,    3155070,    1787065,
12671821,   6732219,    2913314,    4467673,
                 4648794,   6892503,    6045680,    1344212,    9986857,
5639632,    6137428,    2976149,    1068778,
                 10488084,  762062, 1934408,    1359711,    8882190,
2096829,    3080156,    19453561,
                 11689100,  3956971,    4217737,    12801989,   1059361,
5148714,    884659, 6829174,    28995881,
                 3205958,   8535519,    623989, 7614893,    5822434,
1792147,    578759)
census <- data.frame(state_codes, populations)
#Add a column to my datafram for covid rate
df_MZ$covid_rate <- length(df_MZ$state)
for (i in seq_along(df_MZ$state)) {
  df_MZ$covid_rate[i] <- (df_MZ$cases[i] /
census$populations[census$state_codes == df_MZ$state[i]]) * 100000
}

#Export to Excel document
library("writexl")
write_xlsx(df_MZ, "cases.xlsx")
```

Creation of Policies

```
#' Policies

#' Load Data
#' e for events dataset
#' tp for time periods
rm(list=ls())                 #clear environment if needed
gc()                          #garbage collector
setwd("~/SDM Project")        #set working directory to where file is located
(CTRL+Shift+h)

library(rio)
e = import("Policies/COVID-19 US state policy database 4_9_2021.xlsx",sheet =
"State policy changes ")
e = e[5:55, ]                                        #drop meta data & empty rows
colnames(e) = tolower(make.names(colnames(e)))    #set column names to
lowercase
e = e[, c(1:3,7:8,13,15,32:33,46,50,36:37,42:43
          ,49,52,54,127:128,10:12,17:18,22:24,29:31,47:48,53,55,199)]  #keep
only the columns we're interested in

tp = import("Y-Variable/WeekSplit.xlsx", sheet = "After")
tp$start = as.Date(tp$start)
tp$end = as.Date(tp$end)

#' Data Types
e$cldaycr = ifelse(e$cldaycr=="0",NA,e$cldaycr)      #set 0 date values to
null
e$opncldcr = ifelse(e$opncldcr=="0",NA,e$opncldcr)
e$clbsns = ifelse(e$clbsns=="0",NA,e$clbsns)
e$end_bsns = ifelse(e$end_bsns=="0",NA,e$end_bsns)
e$clgym = ifelse(e$clgym=="0",NA,e$clgym)
e$endgym = ifelse(e$endgym=="0",NA,e$endgym)
e$clgym2 = ifelse(e$clgym2=="0",NA,e$clgym2)
e$end_clgym2 = ifelse(e$end_clgym2=="0",NA,e$end_clgym2)
e$closebar = ifelse(e$closebar=="0",NA,e$closebar)
e$end_brs = ifelse(e$end_brs=="0",NA,e$end_brs)
e$bclbar2 = ifelse(e$bclbar2=="0",NA,e$bclbar2)
e$clbar2 = ifelse(e$clbar2=="0",NA,e$clbar2)
e$end_brs2 = ifelse(e$end_brs2=="0",NA,e$end_brs2)
e$clbar3 = ifelse(e$clbar3=="0",NA,e$clbar3)
e$end_clbar3 = ifelse(e$end_clbar3=="0",NA,e$end_clbar3)
e$tlhlmed = ifelse(e$tlhlmed=="0",NA,e$tlhlmed)
e$tlhlaud = ifelse(e$tlhlaud=="0",NA,e$tlhlaud)
e$stayhome = ifelse(e$stayhome=="0",NA,e$stayhome)
e$stayhomenogp = ifelse(e$stayhomenogp=="0",NA,e$stayhomenogp)
e$end_sthm = ifelse(e$end_sthm=="0",NA,e$end_sthm)
e$fm_all = ifelse(e$fm_all=="0",NA,e$fm_all)
e$fm_all2 = ifelse(e$fm_all2=="0",NA,e$fm_all2)
e$fm_emp = ifelse(e$fm_emp=="0",NA,e$fm_emp)
e$fm_end = ifelse(e$fm_end=="0",NA,e$fm_end)
e$fm_stp = ifelse(e$fm_stp=="0",NA,e$fm_stp)
e$clrest = ifelse(e$clrest=="0",NA,e$clrest)
e$endrest = ifelse(e$endrest=="0",NA,e$endrest)
e$clrst2 = ifelse(e$clrst2=="0",NA,e$clrst2)
e$endrest2 = ifelse(e$endrest2=="0",NA,e$endrest2)
```

```
e$clrst3 = ifelse(e$clrst3=="0",NA,e$clrst3)
e$end_clrst3 = ifelse(e$end_clrst3=="0",NA,e$end_clrst3)

e$cldaycr = as.numeric(e$cldaycr)                    #convert dates to
numeric
e$opncldcr = as.numeric(e$opncldcr)
e$clbsns = as.numeric(e$clbsns)
e$end_bsns = as.numeric(e$end_bsns)
e$clgym = as.numeric(e$clgym)
e$endgym = as.numeric(e$endgym)
e$clgym2 = as.numeric(e$clgym2)
e$end_clgym2 = as.numeric(e$end_clgym2)
e$closebar = as.numeric(e$closebar)
e$end_brs = as.numeric(e$end_brs)
e$bclbar2 = as.numeric(e$bclbar2)
e$clbar2 = as.numeric(e$clbar2)
e$end_brs2 = as.numeric(e$end_brs2)
e$clbar3 = as.numeric(e$clbar3)
e$end_clbar3 = as.numeric(e$end_clbar3)
e$tlhlmed = as.numeric(e$tlhlmed)
e$tlhlaud = as.numeric(e$tlhlaud)
e$stayhome = as.numeric(e$stayhome)
e$stayhomenogp = as.numeric(e$stayhomenogp)
e$end_sthm = as.numeric(e$end_sthm)
e$fm_all = as.numeric(e$fm_all)
e$fm_all2 = as.numeric(e$fm_all2)
e$fm_emp = as.numeric(e$fm_emp)
e$fm_end = as.numeric(e$fm_end)
e$fm_stp = as.numeric(e$fm_stp)
e$clrest = as.numeric(e$clrest)
e$endrest = as.numeric(e$endrest)
e$clrst2 = as.numeric(e$clrst2)
e$endrest2 = as.numeric(e$endrest2)
e$clrst3 = as.numeric(e$clrst3)
e$end_clrst3 = as.numeric(e$end_clrst3)
e$mh19 = as.numeric(e$mh19)

e$cldaycr = as.Date(e$cldaycr, origin = "1899-12-30") #convert numeric dates
to datetype
e$opncldcr = as.Date(e$opncldcr, origin = "1899-12-30")
e$clbsns = as.Date(e$clbsns, origin = "1899-12-30")
e$end_bsns = as.Date(e$end_bsns, origin = "1899-12-30")
e$clgym = as.Date(e$clgym, origin = "1899-12-30")
e$endgym = as.Date(e$endgym, origin = "1899-12-30")
e$clgym2 = as.Date(e$clgym2, origin = "1899-12-30")
e$end_clgym2 = as.Date(e$end_clgym2, origin = "1899-12-30")
e$closebar = as.Date(e$closebar, origin = "1899-12-30")
e$end_brs = as.Date(e$end_brs, origin = "1899-12-30")
e$bclbar2 = as.Date(e$bclbar2, origin = "1899-12-30")
e$clbar2 = as.Date(e$clbar2, origin = "1899-12-30")
e$end_brs2 = as.Date(e$end_brs2, origin = "1899-12-30")
e$clbar3 = as.Date(e$clbar3, origin = "1899-12-30")
e$end_clbar3 = as.Date(e$end_clbar3, origin = "1899-12-30")
e$tlhlmed = as.Date(e$tlhlmed, origin = "1899-12-30")
e$tlhlaud = as.Date(e$tlhlaud, origin = "1899-12-30")
e$stayhome = as.Date(e$stayhome, origin = "1899-12-30")
e$stayhomenogp = as.Date(e$stayhomenogp, origin = "1899-12-30")
```

```r
e$end_sthm = as.Date(e$end_sthm, origin = "1899-12-30")
e$fm_all = as.Date(e$fm_all, origin = "1899-12-30")
e$fm_all2 = as.Date(e$fm_all2, origin = "1899-12-30")
e$fm_emp = as.Date(e$fm_emp, origin = "1899-12-30")
e$fm_end = as.Date(e$fm_end, origin = "1899-12-30")
e$fm_stp = as.Date(e$fm_stp, origin = "1899-12-30")
e$clrest = as.Date(e$clrest, origin = "1899-12-30")
e$endrest = as.Date(e$endrest, origin = "1899-12-30")
e$clrst2 = as.Date(e$clrst2, origin = "1899-12-30")
e$endrest2 = as.Date(e$endrest2, origin = "1899-12-30")
e$clrst3 = as.Date(e$clrst3, origin = "1899-12-30")
e$end_clrst3 = as.Date(e$end_clrst3, origin = "1899-12-30")
str(e)                                                  #confirm data type
changes

#' Data Issues
colSums(is.na(e))                                       #notice non-matching
NAs in date-pair columns
i = which(!is.na(e$clgym) & is.na(e$endgym))            #gyms has 1 row where
there is start date but no end
e$endgym[i] = tp$end[nrow(tp)]                          #for now, set as last
date in weeksplit
i = which(!is.na(e$clgym2) & is.na(e$end_clgym2))       #repeat for gyms2
e$end_clgym2[i] = tp$end[nrow(tp)]
i = which(!is.na(e$closebar) & is.na(e$end_brs))        #repeat for bars
for (j in i){
  e$end_brs[j] = tp$end[nrow(tp)]
}
i = which(!is.na(e$end_brs2) & is.na(e$clbar2) & !is.na(e$bclbar2)) #also
remove rows that have an end date for bclbar2 only
for (j in i){
  e$end_brs2[j] = NA
}
i = which(!is.na(e$clbar2) & is.na(e$end_brs2))         #repeat for bars2
for (j in i){
  e$end_brs2[j] = tp$end[nrow(tp)]
}
i = which((!is.na(e$stayhome)|!is.na(e$stayhomenogp))& is.na(e$end_sthm))
#repeat for stayhome
e$end_sthm[i] = tp$end[nrow(tp)]
i = which((!is.na(e$fm_all)|!is.na(e$fm_emp))& is.na(e$fm_end) &
is.na(e$fm_stp)) #repeat for face mask mandate
for (j in i){
  e$fm_end[j] = tp$end[nrow(tp)]
}
i = which((!is.na(e$fm_all)|!is.na(e$fm_emp))& is.na(e$fm_end) &
!is.na(e$fm_stp)) #check for states with no end date but an fm_stp value
for (j in i){
  e$fm_end[j] = e$fm_stp[j]             #ASSUMING all other states (GA) are
similar to FL in this regard, treat fm_stp as fm_end date
}
e$fm_end2 = NA
i = which(!is.na(e$fm_all2) & is.na(e$fm_end2))
for (j in i){
  e$fm_end2[j] = tp$end[nrow(tp)]
#repeat for fm_all2
}
```

```r
e$fm_end2 = as.Date(e$fm_end2, origin = "1970-01-01")
i = which(!is.na(e$clrst2) & is.na(e$endrest2))
for (j in i){
  e$endrest2[j] = tp$end[nrow(tp)]
#repeat for restaurants
}

#' New Dataframe
#' create new dataframe to store values
df = data.frame(seq(1,(nrow(e)*nrow(tp))))
colnames(df) = c("state")                #rename first column, as values will
be replaced later
df$state = NA                            #set column values as null for now
df$postcode = NA
df$fips = NA
df$time.period = NA
df$start.date = tp$start[1]              #set column values in dateformat for
now
df$end.date = tp$end[1]

df$cldaycr = NA                          #set column names for new binary
columns
df$clbsns = NA
df$clgym = NA
df$clbar = NA
df$tlhlaud = NA
df$tlhlmed = NA
df$stayhome = NA
df$fm_all = NA
df$fm_emp = NA
df$clinrst = NA
df$mh19 = NA

count = 0                                #populate with state and tp
for (i in 1:nrow(e)) {
  for (j in 1:nrow(tp)) {
    count = count + 1
    df$state[count] = e$state[i]
    df$postcode[count] = e$postcode[i]
    df$fips[count] = e$fips[i]
    df$time.period[count] = j
    df$start.date[count] = tp$start[j]
    df$end.date[count] = tp$end[j]
  }
}

#' Date Calculation
#' determine number of days event (ex. closed daycares) overlapped w/time
period
#' following loop is for those with only 2 columns (a single start and end
date)
#' daycare loop
for (i in 1:nrow(df)){
  s = df$state[i]
  r = which(e$state==s)
  event.sd = e$cldaycr[r]
  event.ed = e$opncldcr[r]
```

```
  df.sd = df$start.date[i]
  df.ed = df$end.date[i]
  #print(paste(event.sd,event.ed,df.sd,df.ed))
  if (is.na(event.sd)) {
    df$cldaycr[i] = 0
  }
  else if (event.ed <= df.sd) {
    df$cldaycr[i] = 0
  }
  else if (event.sd <= df.sd && event.ed >= df.ed) {
    df$cldaycr[i] = df.ed-df.sd
  } else if (event.sd <= df.sd && event.ed < df.ed && event.ed > df.sd){
    df$cldaycr[i] = event.ed-df.sd
  } else if (event.sd >= df.sd && event.sd < df.ed && event.ed >= df.ed) {
    df$cldaycr[i] = df.ed-event.sd
  } else if (event.sd >= df.sd && event.ed < df.ed) {
    df$cldaycr[i] = event.ed-event.sd
  }
}

colSums(is.na(df)) #check that all cases were covered (no NAs in event
column)
table(df$cldaycr)  #view # of days (max is 12)
df$cldaycr = ifelse(df$cldaycr>0,1,0) #' set arbitrary cutoff period for
dates (1/0)
# if even 1 day was included in the 12 day period, we'll change the factor to
1, else 0

#####################
#business loop
for (i in 1:nrow(df)){
  s = df$state[i]
  r = which(e$state==s)
  event.sd = e$clbsns[r]
  event.ed = e$end_bsns[r]
  df.sd = df$start.date[i]
  df.ed = df$end.date[i]
  #print(paste(event.sd,event.ed,df.sd,df.ed))
  if (is.na(event.sd)) {
    df$clbsns[i] = 0
  }
  else if ((event.ed <= df.sd)||(event.sd >= df.ed)) { #if event was
before/after time period
    df$clbsns[i] = 0
  }
  else if (event.sd <= df.sd && event.ed >= df.ed) { #case 1
    df$clbsns[i] = df.ed-df.sd
  } else if (event.sd <= df.sd && event.ed < df.ed && event.ed > df.sd){
#case 2
    df$clbsns[i] = event.ed-df.sd
  } else if (event.sd >= df.sd && event.sd < df.ed && event.ed >= df.ed) {
#case 3
    df$clbsns[i] = df.ed-event.sd
  } else if (event.sd >= df.sd && event.ed < df.ed) { #case 4
    df$clbsns[i] = event.ed-event.sd
  }
}
```

```
colSums(is.na(df)) #check that all cases were covered (no NAs in event
column)
table(df$clbsns)  #view # of days (max is 12)
df$clbsns = ifelse(df$clbsns>0,1,0) #set arbitrary cutoff period for dates
(1/0)
# if even 1 day was included in the 12 day period, we'll change the factor to
1, else 0
################################
#gym loop - method 1 (single loop), alt method (multiple loops)
library(dplyr) # needed to preserve date class in if_else statement
for (i in 1:nrow(df)){
  s = df$state[i]
  r = which(e$state==s)
  event.sd = e$clgym[r]
  event.ed = e$endgym[r]
  event2.sd = if_else(is.na(e$clgym2[r]),tp$start[1]-2,e$clgym2[r])
  event2.ed = if_else(is.na(e$end_clgym2[r]),tp$start[1]-1,e$end_clgym2[r])
  df.sd = df$start.date[i]
  df.ed = df$end.date[i]
  e.end = if_else(((event.ed <= df.sd)||(event.sd >= df.ed)),TRUE,FALSE)
  e2.end = if_else(((event2.ed <= df.sd)||(event2.sd >= df.ed)),TRUE,FALSE)
  #print(paste(s,"event",event.sd,event.ed,"time
period",df.sd,df.ed,"second",event2.sd,event2.ed))
  if (is.na(event.sd)) {
    df$clgym[i] = 0 #empty case
  } else if (e.end && e2.end) {
    df$clgym[i] = 0 #if both events ended before/after time period
  } else if ((event.sd <= df.sd && event.ed >= df.ed)||(event2.sd <= df.sd &&
event2.ed >= df.ed)) {
    df$clgym[i] = df.ed-df.sd #case 1
  } else if (event.sd <= df.sd && event.ed < df.ed && event.ed > df.sd) {
    df$clgym[i] = event.ed-df.sd #case 2
  } else if (event2.sd <= df.sd && event2.ed < df.ed && event2.ed > df.sd) {
    df$clgym[i] = event2.ed-df.sd
  } else if (event.sd >= df.sd && event.sd < df.ed && event.ed >= df.ed) {
    df$clgym[i] = df.ed-event.sd #case 3
  } else if (event2.sd >= df.sd && event2.sd < df.ed && event2.ed >= df.ed) {
    df$clgym[i] = df.ed-event2.sd
  } else if (event.sd >= df.sd && event.ed < df.ed) {
    df$clgym[i] = event.ed-event.sd #case 4
  } else if (event2.sd >= df.sd && event2.ed < df.ed) {
    df$clgym[i] = event2.ed-event2.sd
  }
}
colSums(is.na(df)) #check that all cases were covered (no NAs in event
column)
table(df$clgym)  #view # of days (max is 12)
df$clgym = ifelse(df$clgym>0,1,0) #set arbitrary cutoff period for dates
(1/0)
############################################################
#bar loop
for (i in 1:nrow(df)){
  s = df$state[i]
  r = which(e$state==s)
  event.sd = e$closebar[r]
  event.ed = e$end_brs[r]
```

```
  event2.sd = if_else(is.na(e$clbar2[r]),tp$start[1]-2,e$clbar2[r])
  event2.ed = if_else(is.na(e$end_brs2[r]),tp$start[1]-1,e$end_brs2[r])
  event3.sd = if_else(is.na(e$clbar3[r]),tp$start[1]-2,e$clbar3[r])
  event3.ed = if_else(is.na(e$end_clbar3[r]),tp$start[1]-1,e$end_clbar3[r])
  df.sd = df$start.date[i]
  df.ed = df$end.date[i]
  e.end = if_else(((event.ed <= df.sd)||(event.sd >= df.ed)),TRUE,FALSE)
  e2.end = if_else(((event2.ed <= df.sd)||(event2.sd >= df.ed)),TRUE,FALSE)
  e3.end = if_else(((event3.ed <= df.sd)||(event3.sd >= df.ed)),TRUE,FALSE)
  #print(paste(s,"event",event.sd,event.ed,"time
period",df.sd,df.ed,"second",event2.sd,event2.ed))
  if (is.na(event.sd)) {
    df$clbar[i] = 0 #empty case
  } else if (e.end && e2.end && e3.end) {
    df$clbar[i] = 0 #if all events ended before/after time period
  } else if ((event.sd <= df.sd && event.ed >= df.ed)||(event2.sd <= df.sd &&
event2.ed >= df.ed)
             ||(event3.sd <= df.sd && event3.ed >= df.ed)) {
    df$clbar[i] = df.ed-df.sd #case 1
  } else if (event.sd <= df.sd && event.ed < df.ed && event.ed > df.sd) {
    df$clbar[i] = event.ed-df.sd #case 2a
  } else if (event2.sd <= df.sd && event2.ed < df.ed && event2.ed > df.sd) {
    df$clbar[i] = event2.ed-df.sd #case 2b
  } else if (event3.sd <= df.sd && event3.ed < df.ed && event3.ed > df.sd) {
    df$clbar[i] = event3.ed-df.sd #case 2c
  } else if (event.sd >= df.sd && event.sd < df.ed && event.ed >= df.ed) {
    df$clbar[i] = df.ed-event.sd #case 3a
  } else if (event2.sd >= df.sd && event2.sd < df.ed && event2.ed >= df.ed) {
    df$clbar[i] = df.ed-event2.sd #case 3b
  } else if (event3.sd >= df.sd && event3.sd < df.ed && event3.ed >= df.ed) {
    df$clbar[i] = df.ed-event3.sd #case 3c
  } else if (event.sd >= df.sd && event.ed < df.ed) {
    df$clbar[i] = event.ed-event.sd #case 4a
  } else if (event2.sd >= df.sd && event2.ed < df.ed) {
    df$clbar[i] = event2.ed-event2.sd #case 4b
  } else if (event3.sd >= df.sd && event3.ed < df.ed) {
    df$clbar[i] = event3.ed-event3.sd #case 4c
  }
}
colSums(is.na(df)) #check that all cases were covered (no NAs in event
column)
table(df$clbar)  #view # of days (max is 12)
df$clbar = ifelse(df$clbar>0,1,0) #set arbitrary cutoff period for dates
(1/0)
#####################################################
#telehealth expansion loop
#single date, assuming telehealth expansion is forever applicable after it
occurs
for (i in 1:nrow(df)){
  s = df$state[i]
  r = which(e$state==s)
  event = e$tlhlmed[r]
  df.sd = df$start.date[i]
  df.ed = df$end.date[i]
  #print(paste(s,"event",event,"time period",df.sd,df.ed))
  if (is.na(event)) {
    df$tlhlmed[i] = 0 #empty case
```

```
  } else if (event >= df.ed) {
    df$tlhlmed[i] = 0 #if event started after time period
  } else if (event <= df.sd) {
    df$tlhlmed[i] = df.ed-df.sd #case 1
  } else if (event > df.sd && event < df.ed) {
    df$tlhlmed[i] = df.ed-event #case 2
  }
}
colSums(is.na(df)) #check that all cases were covered (no NAs in event
column)
table(df$tlhlmed)  #view # of days (max is 12)
df$tlhlmed = ifelse(df$tlhlmed>0,1,0) #set arbitrary cutoff period for dates
(1/0)
########################
#tlhl aud
#telehealth audio loop
#single date, assuming telehealth audio is forever applicable after it occurs
for (i in 1:nrow(df)){
  s = df$state[i]
  r = which(e$state==s)
  event = e$tlhlaud[r]
  df.sd = df$start.date[i]
  df.ed = df$end.date[i]
  #print(paste(s,"event",event,"time period",df.sd,df.ed))
  if (is.na(event)) {
    df$tlhlaud[i] = 0 #empty case
  } else if (event >= df.ed) {
    df$tlhlaud[i] = 0 #if event started after time period
  } else if (event <= df.sd) {
    df$tlhlaud[i] = df.ed-df.sd #case 1
  } else if (event > df.sd && event < df.ed) {
    df$tlhlaud[i] = df.ed-event #case 2
  }
}
colSums(is.na(df)) #check that all cases were covered (no NAs in event
column)
table(df$tlhlaud)  #view # of days (max is 12)
df$tlhlaud = ifelse(df$tlhlaud>0,1,0) #set arbitrary cutoff period for dates
(1/0)
############################
#stayhome
#method1
#first perform a loop with stayhome-end values
df$tempA = NA                 #create a temp column
for (i in 1:nrow(df)){
  s = df$state[i]
  r = which(e$state==s)
  event.sd = e$stayhome[r]
  event.ed = e$end_sthm[r]
  df.sd = df$start.date[i]
  df.ed = df$end.date[i]
  #print(paste(event.sd,event.ed,df.sd,df.ed))
  if (is.na(event.sd)) {
    df$tempA[i] = 0
  }
  else if ((event.ed <= df.sd)||(event.sd >= df.ed)) { #if event was
before/after time period
```

```
    df$tempA[i] = 0
  }
  else if (event.sd <= df.sd && event.ed >= df.ed) { #case 1
    df$tempA[i] = df.ed-df.sd
  } else if (event.sd <= df.sd && event.ed < df.ed && event.ed > df.sd){
#case 2
    df$tempA[i] = event.ed-df.sd
  } else if (event.sd >= df.sd && event.sd < df.ed && event.ed >= df.ed) {
#case 3
    df$tempA[i] = df.ed-event.sd
  } else if (event.sd >= df.sd && event.ed < df.ed) { #case 4
    df$tempA[i] = event.ed-event.sd
  }
}
#perform a second loop with nogp-end values
df$tempB = NA
for (i in 1:nrow(df)){
  s = df$state[i]
  r = which(e$state==s)
  event.sd = e$stayhomenogp[r]
  event.ed = e$end_sthm[r]
  df.sd = df$start.date[i]
  df.ed = df$end.date[i]
  #print(paste(event.sd,event.ed,df.sd,df.ed))
  if (is.na(event.sd)) {
    df$tempB[i] = 0
  }
  else if ((event.ed <= df.sd)||(event.sd >= df.ed)) { #if event was
before/after time period
    df$tempB[i] = 0
  }
  else if (event.sd <= df.sd && event.ed >= df.ed) { #case 1
    df$tempB[i] = df.ed-df.sd
  } else if (event.sd <= df.sd && event.ed < df.ed && event.ed > df.sd){
#case 2
    df$tempB[i] = event.ed-df.sd
  } else if (event.sd >= df.sd && event.sd < df.ed && event.ed >= df.ed) {
#case 3
    df$tempB[i] = df.ed-event.sd
  } else if (event.sd >= df.sd && event.ed < df.ed) { #case 4
    df$tempB[i] = event.ed-event.sd
  }
}
df$tempA = ifelse(df$tempA>0,1,0) #set arbitrary cutoff period for dates
(1/0)
df$tempB = ifelse(df$tempB>0,1,0)
table(df$tempA)  #view # of days (max is 12)
table(df$tempB)
df$stayhome = ifelse(df$tempA>0,2,(ifelse(df$tempB>0,1,(ifelse(df$tempA==0 ||
df$tempB==0,0,NA))))) #factor in order, prioritizing stayhome as lvl 2, then
nogp
table(df$stayhome)
colSums(is.na(df)) #check that all cases were covered (no NAs in event
column)


#####
```

```r
#method 2
#hawaii (single overlap case) has 2nd event date start before first time
period begins - does not apply
df$stayhomeA = NA #temp
df$tempC = 0
for (i in 1:nrow(df)){
  s = df$state[i]
  r = which(e$state==s)
  event.sd = if_else(!is.na(e$stayhome[r]),e$stayhome[r],e$stayhomenogp[r])
  event.ed = e$end_sthm[r]
  df.sd = df$start.date[i]
  df.ed = df$end.date[i]
  #print(paste(s,"event",event.sd,event.ed,"tp",df.sd,df.ed))
  if (is.na(event.sd)) {
    df$stayhomeA[i] = 0
  }
  else if ((event.ed <= df.sd)||(event.sd >= df.ed)) { #if event was
before/after time period
    df$stayhomeA[i] = 0
  }
  else if (event.sd <= df.sd && event.ed >= df.ed) { #case 1
    df$stayhomeA[i] = df.ed-df.sd
    if (!is.na(e$stayhome[r])) {df$tempC[i]=1}
  } else if (event.sd <= df.sd && event.ed < df.ed && event.ed > df.sd){
#case 2
    df$stayhomeA[i] = event.ed-df.sd
    if (!is.na(e$stayhome[r])) {df$tempC[i]=1}
  } else if (event.sd >= df.sd && event.sd < df.ed && event.ed >= df.ed) {
#case 3
    df$stayhome[i] = df.ed-event.sd
    if (!is.na(e$stayhome[r])) {df$tempC[i]=1}
  } else if (event.sd >= df.sd && event.ed < df.ed) { #case 4
    df$stayhomeA[i] = event.ed-event.sd
    if (!is.na(e$stayhome[r])) {df$tempC[i]=1}
  }
}
table(df$tempC)
table(df$stayhomeA)
df$stayhomeA = ifelse(df$stayhomeA>0,1,0) #cutoff value
df$stayhomeA = ifelse(df$tempC>0,2,df$stayhomeA)
table(df$stayhomeA) #notice method 1 and 2 have same distribution
table(df$stayhome)
t = as.numeric(df$stayhome == df$stayhomeA)          #are both columns same?
temp = as.data.frame(t,stringsAsFactors=F)
table(temp)                                           #yes, columns exactly
the same
colSums(is.na(df)) #check that all cases were covered (no NAs in event
column)
#remove temp columns and stayhomeA if using method 1
df = subset(df,select=-c(tempA,tempB,tempC,stayhomeA))

############################
#fm mandate
#all
for (i in 1:nrow(df)){
  s = df$state[i]
  r = which(e$state==s)
```

```
  event.sd = e$fm_all[r]
  event.ed = e$fm_end[r]
  event2.sd = if_else(is.na(e$fm_all2[r]),tp$start[1]-2,e$fm_all2[r]) #if
blank, say event started/ended before time periods
  event2.ed = if_else(is.na(e$fm_end2[r]),tp$start[1]-1,e$fm_end2[r])
  df.sd = df$start.date[i]
  df.ed = df$end.date[i]
  e.end = if_else(((event.ed <= df.sd)||(event.sd >= df.ed)),TRUE,FALSE)
  e2.end = if_else(((event2.ed <= df.sd)||(event2.sd >= df.ed)),TRUE,FALSE)
  #print(paste(s,"event",event.sd,event.ed,"time
period",df.sd,df.ed,"second",event2.sd,event2.ed))
  if (is.na(event.sd)) {
    df$fm_all[i] = 0 #empty case
  } else if (e.end && e2.end) {
    df$fm_all[i] = 0 #if both events ended before/after time period
  } else if ((event.sd <= df.sd && event.ed >= df.ed)||(event2.sd <= df.sd &&
event2.ed >= df.ed)) {
    df$fm_all[i] = df.ed-df.sd #case 1
  } else if (event.sd <= df.sd && event.ed < df.ed && event.ed > df.sd) {
    df$fm_all[i] = event.ed-df.sd #case 2
  } else if (event2.sd <= df.sd && event2.ed < df.ed && event2.ed > df.sd) {
    df$fm_all[i] = event2.ed-df.sd
  } else if (event.sd >= df.sd && event.sd < df.ed && event.ed >= df.ed) {
    df$fm_all[i] = df.ed-event.sd #case 3
  } else if (event2.sd >= df.sd && event2.sd < df.ed && event2.ed >= df.ed) {
    df$fm_all[i] = df.ed-event2.sd
  } else if (event.sd >= df.sd && event.ed < df.ed) {
    df$fm_all[i] = event.ed-event.sd #case 4
  } else if (event2.sd >= df.sd && event2.ed < df.ed) {
    df$fm_all[i] = event2.ed-event2.sd
  }
}
colSums(is.na(df)) #check that all cases were covered (no NAs in event
column)
table(df$fm_all)  #view # of days (max is 12)
df$fm_all = ifelse(df$fm_all>0,1,0) #set arbitrary cutoff period for dates
(1/0)

##emp
for (i in 1:nrow(df)){
  s = df$state[i]
  r = which(e$state==s)
  event.sd = e$fm_emp[r]
  event.ed = e$fm_end[r]
  df.sd = df$start.date[i]
  df.ed = df$end.date[i]
  #print(paste(event.sd,event.ed,df.sd,df.ed))
  if (is.na(event.sd)) {
    df$fm_emp[i] = 0
  }
  else if ((event.ed <= df.sd)||(event.sd >= df.ed)) {
    df$fm_emp[i] = 0
  }
  else if (event.sd <= df.sd && event.ed >= df.ed) {
    df$fm_emp[i] = df.ed-df.sd
  } else if (event.sd <= df.sd && event.ed < df.ed && event.ed > df.sd){
    df$fm_emp[i] = event.ed-df.sd
```

```
  } else if (event.sd >= df.sd && event.sd < df.ed && event.ed >= df.ed) {
    df$fm_emp[i] = df.ed-event.sd
  } else if (event.sd >= df.sd && event.ed < df.ed) {
    df$fm_emp[i] = event.ed-event.sd
  }
}

colSums(is.na(df)) #check that all cases were covered (no NAs in event
column)
table(df$fm_emp)  #view # of days (max is 12)
df$fm_emp = ifelse(df$fm_emp>0,1,0) #' set arbitrary cutoff period for dates
(1/0)
#######
#indoor dining closed
#bar loop
for (i in 1:nrow(df)){
  s = df$state[i]
  r = which(e$state==s)
  event.sd = e$clrest[r]
  event.ed = e$endrest[r]
  event2.sd = if_else(is.na(e$clrst2[r]),tp$start[1]-2,e$clrst2[r])
  event2.ed = if_else(is.na(e$endrest2[r]),tp$start[1]-1,e$endrest2[r])
  event3.sd = if_else(is.na(e$clrst3[r]),tp$start[1]-2,e$clrst3[r])
  event3.ed = if_else(is.na(e$end_clrst3[r]),tp$start[1]-1,e$end_clrst3[r])
  df.sd = df$start.date[i]
  df.ed = df$end.date[i]
  e.end = if_else(((event.ed <= df.sd)||(event.sd >= df.ed)),TRUE,FALSE)
  e2.end = if_else(((event2.ed <= df.sd)||(event2.sd >= df.ed)),TRUE,FALSE)
  e3.end = if_else(((event3.ed <= df.sd)||(event3.sd >= df.ed)),TRUE,FALSE)
  #print(paste(s,"event",event.sd,event.ed,"time
period",df.sd,df.ed,"second",event2.sd,event2.ed))
  if (is.na(event.sd)) {
    df$clinrst[i] = 0 #empty case
  } else if (e.end && e2.end && e3.end) {
    df$clinrst[i] = 0 #if all events ended before/after time period
  } else if ((event.sd <= df.sd && event.ed >= df.ed)||(event2.sd <= df.sd &&
event2.ed >= df.ed)
             ||(event3.sd <= df.sd && event3.ed >= df.ed)) {
    df$clinrst[i] = df.ed-df.sd #case 1
  } else if (event.sd <= df.sd && event.ed < df.ed && event.ed > df.sd) {
    df$clinrst[i] = event.ed-df.sd #case 2a
  } else if (event2.sd <= df.sd && event2.ed < df.ed && event2.ed > df.sd) {
    df$clinrst[i] = event2.ed-df.sd #case 2b
  } else if (event3.sd <= df.sd && event3.ed < df.ed && event3.ed > df.sd) {
    df$clinrst[i] = event3.ed-df.sd #case 2c
  } else if (event.sd >= df.sd && event.sd < df.ed && event.ed >= df.ed) {
    df$clinrst[i] = df.ed-event.sd #case 3a
  } else if (event2.sd >= df.sd && event2.sd < df.ed && event2.ed >= df.ed) {
    df$clinrst[i] = df.ed-event2.sd #case 3b
  } else if (event3.sd >= df.sd && event3.sd < df.ed && event3.ed >= df.ed) {
    df$clinrst[i] = df.ed-event3.sd #case 3c
  } else if (event.sd >= df.sd && event.ed < df.ed) {
    df$clinrst[i] = event.ed-event.sd #case 4a
  } else if (event2.sd >= df.sd && event2.ed < df.ed) {
    df$clinrst[i] = event2.ed-event2.sd #case 4b
  } else if (event3.sd >= df.sd && event3.ed < df.ed) {
    df$clinrst[i] = event3.ed-event3.sd #case 4c
```

```
  }
}
colSums(is.na(df)) #check that all cases were covered (no NAs in event
column)
table(df$clinrst)  #view # of days (max is 12)
df$clinrst = ifelse(df$clinrst>0,1,0) #set arbitrary cutoff period for dates
(1/0)
#######################33
#healthcare loop
for (i in 1:nrow(df)) {
  s = df$state[i]
  r = which(e$state==s)
  df$mh19[i] = e$mh19[r]
}
colSums(is.na(df)) #check that all cases were covered (no NAs in event
column)
library("writexl")
write_xlsx(df,"Policies/Policies.xlsx")
```

Final Merged Dataset

```
#' Merging Datasets
#' Load Data
rm(list=ls())                 #clear environment if needed
gc()                          #garbage collector
setwd("~/SDM Project")        #set working directory to where file is located
(CTRL+Shift+h)

library(rio)
df = import("Policies/Policies.xlsx", sheet = "Sheet1")
c = import("Covid/cases.xlsx", sheet = "Sheet1")
y = import("Y-Variable/Y-Var.xlsx", sheet = "Sheet1")
df$cases = NA
df$covid.rate = NA
df$sad = NA   #percent with [S]ymptoms of [A]nxiety and [D]epression

for (i in 1:nrow(df)) {
  r = which(c$state==df$postcode[i]&c$start_date==df$start.date[i])
  df$cases[i] = c$cases[r]
  df$covid.rate[i] = c$covid_rate[r]
}

for (i in 1:nrow(df)) {
  #print(paste(df$state[i],df$start.date[i],df$end.date[i]))
  r = which(y$state==df$state[i]&y$start.date==df$start.date[i])
  df$sad[i] = y$value[r]
}

library("writexl")
write_xlsx(df,"MergedDataset.xlsx")
```

Visualizations, Models, and Assumptions

```
#Load the final dataset
rm(list=ls())
library(rio)
df <- import("MergedDataset.xlsx")
```

```r
#Variable creation
#Add seasonality by month
install.packages("lubridate")
library(lubridate)
df$month <- length(df$end.date)
for (i in seq_along(df$end.date)) {
  df$month[i] <- month(df$end.date[i])
  if (df$time.period[i] == 16) {
    df$month[i] <- 1
  }
  if (df$time.period[i] == 18) {
    df$month[i] <- 2
  }
}
df$month <- as.factor(df$month)

#Export to Excel. The final dataset has months and lags.
library(writexl)
write_xlsx(dlag, "Final_Merged_Dataset.xlsx")

#Attach
attach(df)
hist(log(mh19))
#------------------------------------------------------------------------------
------------------------
#Visualizations
library(lattice)
#Histograms
plot(time.period, sad)
hist(sad, main = "Distribution of Y-Var", xlab = "Y-Variable")
#Actually looks nicely normal
hist(log(sad), main = "Distribution of Y-Var")  #Still looks normal
densityplot(~sad)
#Covid cases are a count data, so covid rate is likely Poisson
hist(covid.rate)
plot(density(covid.rate), main = "Distribution of Covid Rate", lwd = 2)
plot(density(log(covid.rate)), main = "Distribution of Log Covid Rate", lwd =
2)
df$log.covid <- log(df$covid.rate)

#Plots of y-var by time period
boxplot(sad~start.date, main = "Y-Var by Time Period", xlab = "Date", ylab =
"Y-Var")
abline(35, 0)
abline(45, 0)
#Certainly no linear trend but there are differences
#How would this be handled as a time series?
#Do most states follow this trend?
plot(sad~start.date, data = df[ which(fips == 1), ], main = "Alabama",
     ylim=c(25,50), type = "o", pch=20)
plot(sad~time.period, data = df[ which(fips == 6), ], main = "California",
ylim=c(25,45))
plot(sad~time.period, data = df[ which(fips == 12), ], main = "Florida",
ylim=c(25,45))
plot(sad~time.period, data = df[ which(fips == 48), ], main = "Texas",
ylim=c(25,45))
```

```r
#Plots of y-var by time period, split up by state
library(ggplot2)
ggplot(df, aes(x = start.date, y = sad, colour = postcode)) +
  geom_line() +
  ggtitle("Y-Var Over Time by State")
random_states <- sample(postcode, 10)
random_states_df <- df[postcode %in% random_states, ]
ggplot(random_states_df, aes(x = start.date, y = sad, colour = postcode)) +
  geom_line() +
  ggtitle("Y-Var Over Time by State")

#Plots of y-var by state
boxplot(sad~postcode, main = "Y-var by State")
m <- by(sad, state, mean)    #Displays the mean sad for each state
order <- with(df[ , c(2,20)], reorder(postcode, sad, median, na.rm=T))
boxplot(sad~order, main = "Y-Var by State", xlab = "State", ylab = "Y-
Variable")
#Multi-level may make sense. State as a random or fixed effect?

#Most of our variables aren't continuous, but the ones that are don't seem to
be correlated.
#Matrix for binary variables
temp = subset(df, select=c(7:20))
t = as.data.frame(cor(temp))
t[t < 0.5 | t == 1] = ""

#Correlation matrix plot (with histograms and scatterplot) for continuous
variables
library(PerformanceAnalytics)
chart.Correlation(df[ , c(17, 22, 20)], histogram = TRUE)
plot(sad ~ log(covid.rate), data = df)

#-------------------------------------------------------------------------------
----------------------
#Models
unique(start.date)
colnames(df)

#plm package for Panel data
install.packages("plm")
library(plm)

#Panel model, fixed effect on state
panel <- plm(sad ~ month + cldaycr + clbsns + clgym + clbar + tlhlaud +
tlhlmed + stayhome + fm_all + fm_emp
             + clinrst + mh19 + log(covid.rate), data = df, index =
c("postcode", "time.period"),
             model = "within")
summary(panel)
summary(fixef(panel))   #intercept of each state?

#Panel model two-way fixed effect on state and time period
panel2 <- plm(sad ~ month + cldaycr + clbsns + clgym + clbar + tlhlaud +
tlhlmed + stayhome + fm_all + fm_emp
             + clinrst + mh19 + log(covid.rate), data = df, index =
c("postcode", "time.period"),
```

```
            model = "within", effect = "twoways")
summary(panel2)
summary(fixef(panel3))
phtest(panel, panel2)

#Fixed effect, same as first Panel model
fixed <- lm(sad ~ month + cldaycr + clbsns + clgym + clbar + tlhlaud +
tlhlmed + stayhome + fm_all + fm_emp
            + clinrst + mh19 + log(covid.rate) + as.factor(postcode), data =
df)
fixedLag <- lm(sad ~ month + cldaycr + clbsns + clgym + clbar + tlhlaud +
tlhlmed + stayhome + fm_all + fm_emp
            + clinrst + mh19 + log(covid.rate) + as.factor(postcode) + lag1 +
lag2, data = dlag)
fixedLag2 <- lm(sad ~ cldaycr + clgym + clbar + tlhlaud + tlhlmed + stayhome
+ fm_all + fm_emp
                + clinrst + mh19 + log(covid.rate) + as.factor(postcode) +
lag1 + lag2, data = dlag)
summary(fixed)
summary(fixedLag)

#Random effects model, state is a random effect
library(lme4)
random <- lmer(sad ~ month + cldaycr + clbsns + clgym + clbar + tlhlaud +
tlhlmed + stayhome + fm_all + fm_emp
               + clinrst + mh19 + log(covid.rate) + (1 | postcode), data =
df)
panelRandom <- plm(sad ~ month + cldaycr + clbsns + clgym + clbar + tlhlaud +
tlhlmed + stayhome + fm_all + fm_emp
             + clinrst + mh19 + log(covid.rate), data = df, index =
c("postcode", "time.period"),
             model = "random")
randomLag <- lmer(sad ~ cldaycr + clbsns + clgym + clbar + tlhlaud + tlhlmed
+ stayhome + fm_all + fm_emp
               + clinrst + mh19 + log(covid.rate) + lag1 + lag2 + (1 |
postcode), data = dlag)
summary(random)


#Introducing a lag variable to our panel models
#Creation of a lag variable with Dr. Bhattacherjee's code. Let's try this.
n <- length(unique(postcode)); n
state_list <- unique(postcode); state_list
dlag <- NULL
for(i in 1:n) {
  temp <- NULL
  temp <- df[df$postcode == state_list[i],]
  temp$lag1 <- c(NA, temp$sad[1:18])
  temp$lag2 <- c(NA, NA, temp$sad[1:17])
  dlag <- rbind(dlag,temp)
}
#Recreating our first two panel models, but now with lag variable
panel3 <- plm(sad ~ month + cldaycr + clbsns + clgym + clbar + tlhlaud +
tlhlmed + stayhome + fm_all + fm_emp
             + clinrst + mh19 + log(covid.rate) + lag1 + lag2,
             data = dlag, index = c("postcode", "time.period"), model =
"within")
```

```r
summary(panel3)

panel4 <- plm(sad ~ cldaycr + clbsns + clgym + clbar + tlhlaud + tlhlmed +
stayhome + fm_all + fm_emp
              + clinrst + mh19 + log(covid.rate) + lag1 + lag2,
              data = dlag, index = c("postcode", "time.period"),
              model = "within", effect = "twoways")
summary(panel4)

#Compare similarities with Hausman test
phtest(panel, panel3)
phtest(panel2, panel4)

#-------------------------------------------------------------------------------
-----------------------
#Assumptions
#Some functions don't work for plm() models, which is why I test them on the
lm() versions of the models
#By this I mean I use "fixed" instead of "panel"
library(lmtest)
library(car)
vif(fixed)  #Does not work because of alias'ed variables
alias(fixed) #Apparently West Virginia and Wyoming are perfectly collinear?

#Autocorrelation
pdwtest(panel)  #2.1013
pdwtest(panel3)
#Our original "panel" model (AKA "fixed") passes autocorrelation test.
#The lag variable values are very insignificant in "panel3" model.
#Should I choose the simpler "panel" over "panel3"?
#For the rest I'm going to compare "panel" (fixed effects) against "panel3"
(fixed effects with lag)

#Heteroskedasticity
bptest(panel) #heteroskedasticity, not passed. We should still look at it
visually
bptest(panel3) #Here, it IS passed. So lag variable model is good?
plot(fixed$residuals ~ fixed$fitted.values, ylab = "Residuals",
     xlab = "Fitted values", main = "Test for Constant Error Variance")

plot(fixedLag$residuals ~ fixedLag$fitted.values, ylab = "Residuals",
     xlab = "Fitted values", main = "Test for Constant Error Variance")

#normality/multivariate linearity
qqnorm(panel$residuals) #looks great
qqline(panel$residuals, col = "red")
hist(panel$residuals, xlab = "Panel Residuals", main = "Histogram of
Residuals") #looks good

qqnorm(panel3$residuals) #Also looks great
qqline(panel3$residuals, col = "red")
histogram(panel3$residuals, xlab = "Panel Residuals") #Also looks good

#linearity check
plot(sad ~ fixed$fitted.values, xlab = "Fitted values", ylab = "Actual
values", main = "Linearity Check")
abline(0,1,col="red",lwd=3)  #looks good
```

```r
plot(sad ~ fixedLag$fitted.values, xlab = "Fitted values", ylab = "Actual
values", main = "Linearity Check")
abline(0,1,col="red",lwd=3)  #This function fails because of lag. Can I get
it to work to plot Linearity?

plot(fixed)  #Is this linear (first plot)?

#-------------------------------------------------------------------------
---------------------------
#Final models: panel, panel2, panel3, random
summary(fixef(panel), order)
coef(panel)
stargazer::stargazer(panel, panel3, type = "text", single.row = TRUE)
```