Coronavirus Second Waves

Tyler DeGroff

Import New York Times Coronavirus Data

```
setwd("/Users/tylerdegroff/Documents/Github/NYTimes\ COVID-19\ Data")
nytimes <- read.csv("us-counties.csv")
setwd("/Users/tylerdegroff/Documents/Github/Coronavirus")
```

Source: New York Times, The, Smith, M., Yourish, K., Almukhtar, S., Collins, K., Ivory, D., & Harmon, A. (2020, January–July). *Coronavirus (Covid-19) Data in the United States* [Cumulative counts of coronavirus cases in the United States, at the county level, over time (daily frequency).]. Github. https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-counties.csv

```
data <- nytimes %>%
  filter(county != "Unknown") %>%
  mutate(date = as.Date(date)) %>%
  within(., fips[county == "New York City"] <- 36999) # treats NYC as a county
```

Import Census Bureau FIPS Code Metadata

```
fips.state <- read_excel("fips.state.xlsx", skip = 4)
```

Source: United States Census Bureau. (2020, May 20). *2019 Census Bureau Region and Division Codes and State FIPS Codes* [Reference file for vintage 2019 Census Bureau state-level FIPS codes.]. United States Department of Commerce. https://www2.census.gov/programs-surveys/popest/geographies/2019/state-geocodes-v2019.xlsx

```
fips.state <- fips.state %>%
  rename(fips.state = "State (FIPS)", state = "Name") %>%
  select(fips.state, state) %>%
  filter(fips.state != "00") %>%
  arrange(fips.state)
```

```
fips.granular <- read_excel("fips.granular.xlsx", skip = 4)
```

Source United States Census Bureau. (2020, May 20). *2019 State, County, Minor Civil Division, and Incorporated Place FIPS Codes* [Reference file for vintage 2019 Census Bureau county-level FIPS codes.]. United States Department of Commerce. https://www2.census.gov/programs-surveys/popest/geographies/2019/all-geocodes-v2019.xlsx

```
fips.county <- fips.granular %>%
  rename(
    fips.state = "State Code (FIPS)",
    fips.county = "County Code (FIPS)",
    county = "Area Name (including legal/statistical area description)"
  ) %>%
```

```r
  mutate(fips = as.numeric(paste0(
    as.character(fips.state),
    as.character(fips.county)
  ))) %>%
  filter(fips.county != "000") %>%
  select(fips, fips.state, fips.county, county)
```

```r
fips <- merge(x = fips.county, y = fips.state, by = "fips.state", all.x = TRUE)
fips <- fips[complete.cases(fips[, "state"]), ]
```

Import Census Bureau Population Estimates

```r
pop <- read_excel("pop.xlsx", skip = 3)
```

```
## New names:
## * `` -> ...1
```

Source: United States Census Bureau. (2010–2019, April 1–July 1). *County Population Totals: 2010-2019* [Annual estimates of the county-level resident population, over time (annual frequency).]. United States Department of Commerce. https://www2.census.gov/programs-surveys/popest/tables/2010-2019/counties/totals/co-est2019-annres.xlsx

```r
pop <- pop %>%
  rename(countyState = "...1", population = "2019") %>%
  filter(countyState != "United States") %>%
  select(countyState, population)
```

```r
fips <- fips %>% mutate(countyState = paste0(county, ", ", state))
pop <- merge(x = pop, y = fips, by = "countyState", all.x = TRUE)
```

Aggregate and Treat NYC as Its Own County

```r
pop.nyc <- pop %>%
  filter(
    fips == 36005 | # Bronx County (Bronx)
    fips == 36047 | # Kings County (Brooklyn)
    fips == 36061 | # New York County (Manhattan)
    fips == 36081 | # Queens County (Queens)
    fips == 36085   # Richmond County (Staten Island)
  )

pop <- rbind(
  pop,
  data.frame(
    fips.state = "36",    # actual New York State state-level FIPS code
    fips.county = "999", # synthetic county-level FIPS code
    fips = "36999",       # synthetic FIPS code
    county = "New York City",
    state = "New York",
    countyState = "New York, New York",
    population = sum(pop.nyc$population)
  )
```

```r
)
```

```r
data <- merge(
  x = data,
  y = pop %>% select(fips, population),
  by = "fips",
  all.x = TRUE
)

data <- data %>%
  mutate(
    cases.percap = cases / population,
    deaths.percap = deaths / population
  )

data <- data[order(data$date, data$state, data$county), ]

data <- data %>%

  # mutate across dates by unique county/state combinations

  mutate(countyState = paste0(county, ", ", state)) %>%
  group_by(countyState) %>%

  mutate(

    cases.new  = c(cases[1], diff(cases)),
    deaths.new = c(deaths[1], diff(deaths)),

    cases.new.7dsma  = rollmean(cases.new, k = 7, fill = NA, align = "right"),
    deaths.new.7dsma = rollmean(deaths.new, k = 7, fill = NA, align = "right")

  )
```

```r
data.state <- data %>%

  # aggregate across counties by unique date/state combination

  mutate(dateState = paste0(date, ", ", state)) %>%
  group_by(dateState) %>%
  summarize(
    date = date[1],
    state = state[1],
    cases = sum(cases),
    deaths = sum(deaths),
    pop = sum(population)
  ) %>%
  mutate(
    cases.perCap = cases / pop,
    deaths.perCap = deaths / pop
  ) %>%

  # mutate across individual states, exclusively
```

```
  group_by(state) %>%

  mutate(
    cases.new = c(cases[1], diff(cases)),
    deaths.new = c(deaths[1], diff(deaths)),

    cases.new.perCap = c(cases.perCap[1], diff(cases.perCap)),
    deaths.new.perCap = c(deaths.perCap[1], diff(deaths.perCap)),

    cases.new.perM = cases.new.perCap * 1000000,
    deaths.new.perM = deaths.new.perCap * 1000000
  ) %>%

  mutate(
    cases.new.7dsma = rollmean(cases.new, k = 7, fill = 0, align = "right"),
    deaths.new.7dsma = rollmean(deaths.new, k = 7, fill = 0, align = "right"),

    cases.new.perCap.7dsma =
      rollmean(cases.new.perCap, k = 7, fill = 0, align = "right"),
    deaths.new.perCap.7dsma =
      rollmean(deaths.new.perCap, k = 7, fill = 0, align = "right"),

    cases.new.perM.7dsma =
      rollmean(cases.new.perM, k = 7, fill = 0, align = "right"),
    deaths.new.perM.7dsma =
      rollmean(deaths.new.perM, k = 7, fill = 0, align = "right"),
  ) %>%

  mutate(
    cases.active = rollsum(cases.new, k = 9, fill = 0, align = "right"),
    cases.active.perCap = rollsum(cases.new.perCap, k = 9, fill = 0, align = "right"),
    cases.active.perM = rollsum(cases.new.perM, k = 9, fill = 0, align = "right")
  )
```

```
write_csv(data, "data.csv")
write_csv(data.state, "data.state.csv")
```

Analysis

```
guests <- data.frame(
  state = c("Connecticut", "Wisconsin", "South Dakota", "Indiana", "Arizona", "Nebraska"),
  guests = c(1, 4, 8, 3, 4, 200 - 8 - 3 - (4 * 2) - 1)
)

wedding <- merge(
  x = guests,
  y = data.state %>%
    filter(
      date == max(date),
      state %in% unique(guests$state)
    ) %>%
  select(cases.active.perM),
```

```r
  by = "state"
) %>% arrange(desc(cases.active.perM))

kable(
  wedding,
  col.names = c("State", "Guests", "Active Cases per Million")
)
```

State	Guests	Active Cases per Million
South Dakota	8	8433.758
Wisconsin	4	7178.269
Nebraska	180	3910.240
Indiana	3	2820.912
Connecticut	1	1620.907
Arizona	4	1237.718

```r
summary <- data.frame(
  state = "Summary",
  guests = sum(wedding$guests),
  cases.active.perM = weighted.mean(
    x = wedding$cases.active.perM,
    w = wedding$guests
  )
)

kable(
  summary,
  col.names = c("", "Total Guests", "W.Avg. Active Cases/Mln.")
)
```

	Total Guests	W.Avg. Active Cases/Mln.
Summary	200	4075.305

Binomial Probability Distriubtion: Cumulative Density Function

$$1 - P(q > x|n, p) = 1 - \binom{n}{x} p^x \, (1-p)^{(n-x)} \tag{1}$$

$$1 - P(q > x = 0|n = 200, p = 4075 \cdot 10^{-7}) = 1 - \left[\frac{200!}{0! \cdot (200-0)!}\right] \cdot \left[4075 \cdot 10^{-7}\right]^0 \cdot \left[1 - 4075 \cdot 10^{-7}\right]^{200-0} \tag{2}$$

```r
pbinom(
  q = 0,
  size = sum(wedding$guests),
  prob = weighted.mean(x = wedding$cases.active.perM, w = wedding$guests) / 1000000,
  lower.tail = FALSE
)
```

```
## [1] 0.5581241
```

Binomial Probability Distriubtion: Cumulative Density Function

```
eventSize <- 25000

caseDensity <- pull(data.state %>%
  ungroup() %>%
  filter(date == max(date), state == "Nebraska") %>%
  select(cases.active.perM)
)
```

$$1 - P(q > x|n, p) = 1 - \binom{n}{x} p^x (1-p)^{(n-x)} \tag{3}$$

$$1 - P(q > x|n = 2.5 \times 10^4, p = 3910 \cdot 10^{-7}) = 1 - \left[\frac{2.5 \times 10^4!}{x! \cdot (2.5 \times 10^4 - x)!}\right] \cdot \left[3910 \cdot 10^{-7}\right]^x \cdot \left[1 - 3910 \cdot 10^{-7}\right]^{2.5 \times 10^4 - x} \tag{4}$$

```
pbinom.iterate <- function(x = 0) {

  df <- data.frame()

  for (i in 1:x - 1) {
    df <- rbind(
      df,
      data.frame(
        infectiousPersons = paste("at least", i + 1),
        probabilityPercent = round(pbinom(
          q = i, size = eventSize, prob = caseDensity / 1000000, lower.tail = FALSE
        ) * 100, 1)
      )
    )
  }

  return(df)

}
```

```
kable(pbinom.iterate(100))
```

infectiousPersons	probabilityPercent
at least 1	100.0
at least 2	100.0
at least 3	100.0
at least 4	100.0
at least 5	100.0
at least 6	100.0
at least 7	100.0
at least 8	100.0
at least 9	100.0
at least 10	100.0
at least 11	100.0
at least 12	100.0
at least 13	100.0

infectiousPersons	probabilityPercent
at least 14	100.0
at least 15	100.0
at least 16	100.0
at least 17	100.0
at least 18	100.0
at least 19	100.0
at least 20	100.0
at least 21	100.0
at least 22	100.0
at least 23	100.0
at least 24	100.0
at least 25	100.0
at least 26	100.0
at least 27	100.0
at least 28	100.0
at least 29	100.0
at least 30	100.0
at least 31	100.0
at least 32	100.0
at least 33	100.0
at least 34	100.0
at least 35	100.0
at least 36	100.0
at least 37	100.0
at least 38	100.0
at least 39	100.0
at least 40	100.0
at least 41	100.0
at least 42	100.0
at least 43	100.0
at least 44	100.0
at least 45	100.0
at least 46	100.0
at least 47	100.0
at least 48	100.0
at least 49	100.0
at least 50	100.0
at least 51	100.0
at least 52	100.0
at least 53	100.0
at least 54	100.0
at least 55	100.0
at least 56	100.0
at least 57	100.0
at least 58	100.0
at least 59	100.0
at least 60	100.0
at least 61	100.0
at least 62	100.0
at least 63	100.0
at least 64	100.0
at least 65	100.0

infectiousPersons	probabilityPercent
at least 66	100.0
at least 67	100.0
at least 68	99.9
at least 69	99.9
at least 70	99.9
at least 71	99.8
at least 72	99.7
at least 73	99.6
at least 74	99.5
at least 75	99.3
at least 76	99.0
at least 77	98.7
at least 78	98.3
at least 79	97.7
at least 80	97.1
at least 81	96.3
at least 82	95.3
at least 83	94.2
at least 84	92.9
at least 85	91.3
at least 86	89.5
at least 87	87.4
at least 88	85.1
at least 89	82.5
at least 90	79.7
at least 91	76.7
at least 92	73.4
at least 93	69.9
at least 94	66.2
at least 95	62.4
at least 96	58.4
at least 97	54.4
at least 98	50.4
at least 99	46.3
at least 100	42.4