Figures

Maddy Kline

2023-06-06

Read in data

```
dat <- read_csv("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/output/
```

```
## Rows: 88128 Columns: 7
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): STATE, AGEGRP, PRIMARYCOND
## dbl (4): MONTH, SEX, YEAR, NVISITS
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
coh <- read_csv("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/output/
```

```
## Rows: 7344 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): STATE, AGEGRP
## dbl (3): SEX, NMEMB, YEAR
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#set color palette for regions
colors_df <- data.frame(Region = c("South", "Midwest", "Northeast", "West"),
                        region_color = c("#DE6C83", "#43AA8B", "#2274A5", "#EDDEA4"))

#set color palette for subregions (subdivisions of regions)
pal <- brewer.pal(n= 9, name = "Paired")
subcolors_df <- data.frame(Region = c("East South Central", "West South Central",
                                      "West North Central", "South Atlantic", "Middle Atlantic",
                                      "Mountain West", "East North Central", "New England", "Pacific We
                           region_color = c(pal[6], pal[5], pal[4], pal[9], pal[2], pal[8], pal[3], pal
```

Read in helper files

```
source("get_popsizes.R")
```

```
## Getting data from the 2011-2015 5-year ACS
```

```
## Warning: * You have not set a Census API key. Users without a key are limited to 500
## queries per day and may experience performance limitations.
## i For best results, get a Census API key at
## http://api.census.gov/data/key_signup.html and then supply the key to the
## `census_api_key()` function to use it throughout your tidycensus session.
## This warning is displayed once per session.
```

```
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
## Getting data from the 2011-2015 5-year ACS
```

```r
source("get_regions.R")
source("get_subregions.R")
source("yearly_average_functions.R")
source("monthly_average_functions.R")
source("get_state_sinusoids.R")
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## `summarise()` has grouped output by 'STATE', 'MONTH'. You can override using
## the `.groups` argument.
## Joining with `by = join_by(STATE)`
## `summarise()` has grouped output by 'STATE', 'MONTH'. You can override using
## the `.groups` argument.
## `summarise()` has grouped output by 'STATE'. You can override using the
## `.groups` argument.
```

```
## Warning: `as.tibble()` was deprecated in tibble 2.0.0.
## i Please use `as_tibble()` instead.
## i The signature and semantics have changed, see `?as_tibble`.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning: Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
```

```
## Joining with `by = join_by(STATE, MONTH)`
## Joining with `by = join_by(Region)`
## Joining with `by = join_by(STATE)`
```

```r
source("region_subregion_sinusoid_functions.R")
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
```

Calculate the proportion of the population represented in the dataset for calculation in line 110

```
total_us_pop_years_df <- data.frame(YEAR = c(2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018),
                                   pop = c(309414110, 311580009, 313874218, 316057727, 318386421,
                                           320742673, 323071342, 325147121, 327167434))
yearly_memb <- coh |> group_by(YEAR) |>
  summarize(yearly_memb = sum(NMEMB))
#percentage of population represented by the database
rep <- left_join(yearly_memb, total_us_pop_years_df) |> mutate(perc = yearly_memb / pop)
```

```
## Joining with `by = join_by(YEAR)`
```

```
rep |> arrange(yearly_memb)
```

```
## # A tibble: 9 x 4
##    YEAR yearly_memb       pop   perc
##   <dbl>       <dbl>     <dbl>  <dbl>
## 1  2017    16600062 325147121 0.0511
## 2  2018    17445536 327167434 0.0533
## 3  2016    20178001 323071342 0.0625
## 4  2015    20334167 320742673 0.0634
## 5  2013    29699593 316057727 0.0940
## 6  2014    31583109 318386421 0.0992
## 7  2010    32282742 309414110 0.104
## 8  2011    35733696 311580009 0.115
## 9  2012    36423161 313874218 0.116
```

Summarize characteristics of the cohort

Average membership by sex, age group, region, and subregion. Becomes Table 1.

```
#total
total_range <- coh |> group_by(YEAR) |>
  summarize(membership = sum(NMEMB)) |>
  summarize(avg_membership = mean(membership), min = min(membership), max = max(membership))

total_df <- data.frame(Category = "Total", avg_membership = total_range$avg_membership,
                       perc = 1, min = total_range$min, max = total_range$max)

total_avg_memb <- total_df$avg_membership

#sex membership
sex_membership <- coh |> group_by(SEX, YEAR) |>
  summarize(membership = sum(NMEMB)) |>
  group_by(SEX) |>
  summarize(avg_membership = mean(membership), min = min(membership), max = max(membership)) |>
  mutate(perc = avg_membership / total_avg_memb) |>
  mutate(Category = SEX) |>
  mutate(Category = c("Male", "Female")) |>
  select(Category, avg_membership, perc, min,max)
```

```
## `summarise()` has grouped output by 'SEX'. You can override using the `.groups`
## argument.
```

```
#age membership
age_membership <- coh |> group_by(AGEGRP, YEAR) |>
  summarize(membership = sum(NMEMB)) |>
  group_by(AGEGRP) |>
  summarize(avg_membership = mean(membership), min = min(membership), max = max(membership)) |>
  mutate(perc = avg_membership / total_avg_memb) |>
  mutate(Category = AGEGRP) |>
  select(Category, avg_membership, perc, min, max)
```

```
## `summarise()` has grouped output by 'AGEGRP'. You can override using the
## `.groups` argument.
```

```
#region membership
regions_membership <- coh |>
  left_join(regions) |> group_by(part, YEAR) |>
  summarize(membership = sum(NMEMB)) |>
  group_by(part) |>
  summarize(avg_membership = mean(membership), min = min(membership), max = max(membership)) |> mutate(
  select(Category, avg_membership, min, max) |>
  mutate(perc = avg_membership / total_avg_memb) |>
  select(Category, avg_membership, perc, min, max)
```

```
## Joining with `by = join_by(STATE)`
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
#subregion membership
subregions_membership <- coh |> left_join(subregions) |> group_by(part, YEAR) |>
  summarize(membership = sum(NMEMB)) |>
  group_by(part) |>
  summarize(avg_membership = mean(membership), min = min(membership), max = max(membership))
```

```
## Joining with `by = join_by(STATE)`
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
subregions_membership <- subregions_membership |> mutate(Category = gsub("_", " ", subregions_membershi
  select(Category, avg_membership, min, max) |>
  mutate(perc = avg_membership / total_avg_memb) |>
  select(Category, avg_membership, perc, min, max)

sex_df <- data.frame(Category = "Sex", avg_membership = NA,
                     perc = NA, min = NA, max = NA)

age_df <- data.frame(Category = "Age Group", avg_membership = NA,
                     perc = NA, min = NA, max = NA)
subregion_df <- data.frame(Category = "Subregion", avg_membership = NA,
                        perc = NA, min = NA, max = NA)
region_df <- data.frame(Category = "Region", avg_membership = NA,
                        perc = NA, min = NA, max = NA)

table1 <- rbind(total_df, sex_df, sex_membership, age_df, age_membership, region_df, regions_membership
  mutate(Average_Membership = format(avg_membership, scientific = TRUE, digits = 3),
         min = format(min, scientific = TRUE, digits = 3),
         max = format(max, scientific = TRUE, digits = 3)) |>
  mutate("%" = round(perc*100, 2),
        Average_Membership = paste0(Average_Membership," ","(", min, "-", max, ")")) |>
  select(Category, Average_Membership, `%`)
```

Calculate average membership by state

For quality control, threshold was set at an average of >5,000 members per year. Becomes Supplementary Figure 1.

```
continental_state_membership_noSC <- coh |>
  filter(!STATE %in% c('Hawaii', 'Alaska', 'South Carolina')) |>
  group_by(STATE, YEAR) |>
  summarize(total_memb = sum(NMEMB)) |>
  group_by(STATE) |>
  summarize(NMEMB = mean(total_memb)) |>
  ggplot(aes(x = reorder(STATE, -NMEMB), y = log(NMEMB))) +
  geom_bar(stat = "identity") +
  geom_hline(yintercept = log(5000), linetype = 2, color = "red") +
  theme_classic() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  # ggtitle("Average Yearly Membership by State, 2010-2018") +
  ylab("Log Average Yearly Membership") +
  xlab("State")
```

```
## `summarise()` has grouped output by 'STATE'. You can override using the
## `.groups` argument.
```

Yearly regional and subregional differences in disease burden

Calculate the total visits per 1000 members across all years by region. Becomes figure S2a.

```
vis_year <- dat |> group_by(STATE, SEX, AGEGRP, YEAR) |> summarize(yearly_visits = sum(NVISITS))
```

```
## `summarise()` has grouped output by 'STATE', 'SEX', 'AGEGRP'. You can override
## using the `.groups` argument.
```

```
regions_weighted_allyears <- get_yearly_differences(regions) |> mutate(Region = str_to_title(part)) |>
  left_join(colors_df) |> mutate(Region = factor(Region, levels = c("South", "Midwest", "Northeast", "We
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(Region)`
```

```
#plot
region_differences_byyear <- regions_weighted_allyears |> ggplot(aes(YEAR, sum_region_cases_per_thousand
  geom_line() + scale_color_manual(values= colors_df$region_color) + theme_minimal() +
  ylab("Visits per 1000") + xlab("Year") + labs(color = "Region") +
  # ggtitle("Streptococcal Pharyngitis Visits by Region over Time") +
  theme(plot.title = element_text(hjust = 0.5, size = 20))
```

Calculate average visits per year in each region and statistically compare

```
avg_yearly_regions <- regions_weighted_allyears |>
  group_by(Region) |>
  summarize(avg_vis_thous = mean(sum_region_cases_per_thousand), sd = sd(sum_region_cases_per_thousand)
  mutate(lower = avg_vis_thous - 1.96*sd/sqrt(9), upper = avg_vis_thous + 1.96*sd/sqrt(9)) |>
  mutate(Region = factor(Region, levels = c("South", "Midwest", "Northeast", "West"))) |>
  arrange(Region)

region_table <- avg_yearly_regions |>
  mutate("95%CI" = paste0("(", round(lower,2), "-", round(upper,2), ")")) |>
  select(Region, avg_vis_thous, `95%CI`)

year_averages_plot <- avg_yearly_regions |>
  ggplot(aes(Region, avg_vis_thous, color = Region)) +
  geom_point() +
  geom_errorbar(aes(ymin = lower, ymax = upper), width = 0.2) +
  scale_color_manual(values= colors_df$region_color, labels = colors_df$Region) +
  ylab("Visits per 1000") +
  # ggtitle("Streptococcal Pharyngitis Average Visits by Region") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, size = 20))


#make dataframe and perform each t-test
ttest_table_reg <- expand_grid(region1 = c("South", "Midwest", "Northeast", "West"),
            region2 = c("South", "Midwest", "Northeast", "West")) |>
  filter(region1 != region2) |>
  mutate(Comparison = paste0(region1, "-", region2)) |>
  filter(!Comparison %in% c("Midwest-South", "Northeast-South", "Northeast-Midwest",
                              "West-South", "West-Midwest", "West-Northeast"))

ttest_table_reg <- fill_table(ttest_table_reg, regions_weighted_allyears, avg_yearly_regions)

region_comparisons <- ttest_table_reg |>
  ggplot(aes(x = -log(pvals), y = diff, shape = Comparison)) +
  geom_point() +
  geom_vline(xintercept = -log(0.05/6), linetype = 2, color = "red") +
  xlab("Negative Log P-value") +
  ylab("Difference in Visits per 1000") +
  # ggtitle("Streptococcal Pharyngitis Average Yearly Region Comparisons") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, size = 18))
```

Total visits per 1000 members across all years by subregion as well as calculation of average visits per year in each region and statistical comparison. Becomes Supplementary Figure 3.

```
subregions_weighted_allyears <- get_yearly_differences(subregions) |> mutate(Region = str_to_title(part
  left_join(subcolors_df) |> mutate(Region = factor(Region, levels = subcolors_df$Region))
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(Region)`
```

```
subregion_differences_byyear <- subregions_weighted_allyears |> ggplot(aes(YEAR, sum_region_cases_per_t
  geom_line() + scale_color_manual(values= subcolors_df$region_color) + theme_minimal() +
```

```r
  ylab("Visits per 1000") + xlab("Year") + labs(color = "Subegion") +
  # ggtitle("Streptococcal Pharyngitis Visits by Subregion over Time") +
  theme(plot.title = element_text(hjust = 0.5, size = 20))

avg_yearly_subregions <- subregions_weighted_allyears |>
  group_by(Region) |>
  summarize(avg_vis_thous = mean(sum_region_cases_per_thousand), sd = sd(sum_region_cases_per_thousand)
  mutate(lower = avg_vis_thous - 1.96*sd/3, upper = avg_vis_thous + 1.96*sd/3) |>
  arrange(-avg_vis_thous) |>
  left_join(subcolors_df)
```

```
## Joining with `by = join_by(Region)`
```

```r
order <- avg_yearly_subregions |> pull(Region)
avg_yearly_subregions <- avg_yearly_subregions |>
  mutate(Region = factor(Region, levels = order))

#write_csv(avg_yearly_subregions, "year_subregions_sd.csv")

subregion_table <- avg_yearly_subregions |>
  mutate("95%CI" = paste0("(", round(lower,2), "-", round(upper,2), ")")) |>
  select(Region, avg_vis_thous, `95%CI`)

year_averages_plot_subregions <- avg_yearly_subregions |>
  ggplot(aes(reorder(Region, -avg_vis_thous), avg_vis_thous, color = Region)) +
  geom_point() +
  geom_errorbar(aes(ymin = lower, ymax = upper), width = 0.2) +
  scale_color_manual(values= avg_yearly_subregions$region_color, labels = avg_yearly_subregions$Region)
  ylab("Visits per 1000") +
  # ggtitle("Streptococcal Pharyngitis Average Visits by Region") +
  theme_minimal() +
  xlab("Region") +
  theme(axis.text.x = element_text(angle = 60, vjust = 0.5, hjust=0.5),
        plot.title = element_text(hjust = 0.5, size = 20))

ttest_table_sub <- expand_grid(region1 = subregion_table$Region,
                               region2 = subregion_table$Region) |>
  filter(region1 != region2) |>
  mutate(Comparison = paste0(region1, "-", region2)) |>
  filter(!Comparison %in% c("West South Central-East South Central",
                            "West North Central-East South Central",
                            "West North Central-West South Central",
                            "South Atlantic-East South Central",
                            "South Atlantic-West South Central",
                            "South Atlantic-West North Central",
                            "Middle Atlantic-East South Central",
                            "Middle Atlantic-West South Central",
                            "Middle Atlantic-West North Central",
                            "Middle Atlantic-South Atlantic",
                            "Mountain West-East South Central",
                            "Mountain West-West South Central",
                            "Mountain West-West North Central",
                            "Mountain West-South Atlantic",
                            "Mountain West-Middle Atlantic",
                            "East North Central-East South Central",
                            "East North Central-West South Central",
                            "East North Central-West North Central",
                            "East North Central-South Atlantic",
                            "East North Central-Middle Atlantic",
                            "East North Central-Mountain West",
                            "New England-East South Central",
                            "New England-West South Central",
                            "New England-West North Central",
                            "New England-South Atlantic",
                            "New England-Middle Atlantic",
                            "New England-Mountain West",
                            "New England-East North Central",
                            "Pacific West-East South Central",
                            "Pacific West-West South Central",
                            "Pacific West-West North Central",
                            "Pacific West-South Atlantic",
                            "Pacific West-Middle Atlantic",
                            "Pacific West-Mountain West",
                            "Pacific West-East North Central",
                            "Pacific West-New England"

                            ))

ttest_table_sub <- fill_table(ttest_table_sub, subregions_weighted_allyears,avg_yearly_subregions)
subregion_comparisons <- ttest_table_sub |>
  ggplot(aes(x = -log(pvals), y = diff, color = Comparison)) +
  geom_point() + geom_text_repel(aes(label = Comparison)) +
  geom_vline(xintercept = -log(0.05/36), linetype = 2, color = "red") +
  xlab("Negative Log P-value") +
  ylab("Difference in Visits per 1000") +
  # ggtitle("Streptococcal Pharyngitis Average Yearly Subregion Comparisons") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5),
        legend.position = "none")
```

Average monthly regional and subregional differences

Calculate average monthly visits by region, becomes Figure 1.

```
regions_weighted_allyears <- calculate_averages(c("Hawaii", "Alaska"), regions)[[2]] |>
  mutate(Region = str_to_title(part)) |>
  left_join(colors_df) |> mutate(Region = factor(Region, levels = c("South", "Midwest", "Northeast", "W
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(Region)`
```

```
regions_weighted <- calculate_averages(c("Hawaii", "Alaska"), regions)[[3]] |>
  mutate(Region = str_to_title(part)) |>
  left_join(colors_df) |> mutate(Region = factor(Region, levels = c("South", "Midwest", "Northeast", "W
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(Region)`
```

```
points_region_plot <- plot_averages_CIs_points(regions_weighted,
                         regions_weighted_allyears,
                         colors_df) + labs(title = NULL)

#corresponding table of values
region_table_month <- calculate_averages(c("Hawaii", "Alaska"), regions)[[3]] |>
  mutate(CI = paste0("(", round(yearly_average_sum_region_cases_per_thousand - 1.96*sd/3, 2), "-",
                    round(yearly_average_sum_region_cases_per_thousand + 1.96*sd/3, 2), ")")) |>
  mutate(Region = str_to_title(part),
         Month = MONTH,
         Visits_per_1000 = yearly_average_sum_region_cases_per_thousand) |>
  mutate('Visits(CI)' = paste(round(Visits_per_1000,2), CI)) |>
  select(Region, Month, `Visits(CI)`)
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
## Adding missing grouping variables: `part`
```

```
region_table_month <- region_table_month[,2:4] |> pivot_wider(names_from = Region, values_from = `Visits
```

Average subregional differences over the course of the year. Becomes Supplementary Figure 4.

```
subregions_weighted_allyears <- calculate_averages(c("Hawaii", "Alaska"), subregions)[[2]] |>
  mutate(Region = factor(str_to_title(part), levels = c("East South Central", "West South Central",
                                      "West North Central", "South Atlantic", "Middle Atlantic",
                                      "Mountain West", "East North Central", "New England", "Pacific Wes
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
subregions_weighted <- calculate_averages(c("Hawaii", "Alaska"), subregions)[[3]] |>
  mutate(Region = factor(str_to_title(part), levels = c("East South Central", "West South Central",
                                      "West North Central", "South Atlantic", "Middle Atlantic",
                                      "Mountain West", "East North Central", "New England", "Pacific Wes
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
points_subregion_plot <- plot_averages_CIs_points(subregions_weighted,
                          subregions_weighted_allyears,
                          subcolors_df) + labs(title = NULL)

#tables with corresponding values
subregion_table_month <- calculate_averages(c("Hawaii", "Alaska"), subregions)[[3]] |>
  mutate(CI = paste0("(", round(yearly_average_sum_region_cases_per_thousand - 1.96*sd/3, 2), "-",
                     round(yearly_average_sum_region_cases_per_thousand + 1.96*sd/3, 2), ")")) |>
  mutate(Region = str_to_title(part),
         Month = MONTH,
         Visits_per_1000 = yearly_average_sum_region_cases_per_thousand) |>
  mutate('Visits(CI)' = paste(round(Visits_per_1000,2), CI)) |>
  select(Region, Month, `Visits(CI)`)
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
## Adding missing grouping variables: `part`
```

```
subregion_table_month <- subregion_table_month[,2:4] |>
  mutate(Region = gsub("_", " ",subregion_table_month$Region , fixed=TRUE)) |>
  mutate(Region = str_to_title(Region)) |>
  pivot_wider(names_from = Region, values_from = `Visits(CI)`)
```

Average quarterly visits between regions and subregions

Calculate average quarterly visits over 3 month chunks, becomes Figure S5a

```
quarter_df <- data.frame(MONTH = c(1:12), quarter = c(1,1,1,2,2,2,3,3,3, 4,4,4))
quarters <- left_join(regions_weighted_allyears, quarter_df) |> group_by(part, YEAR, quarter) |> summar
  mutate(part = factor(part, levels = c("South", "Midwest", "Northeast", "West")))
```

```
## Joining with `by = join_by(MONTH)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
```

```
region_quarters_plot <- quarters |> group_by(part, quarter) |> summarize(avg_quart_vis = mean(quarter_vi
  mutate(lower = avg_quart_vis - 1.96*sd/3, upper = avg_quart_vis + 1.96*sd/3) |>
  ggplot(aes(quarter, avg_quart_vis, fill = part)) + geom_bar(position = 'dodge', stat = 'identity') +
  geom_errorbar(aes(x =quarter, ymin = lower, ymax = upper), position = 'dodge',  alpha = 0.2)+ scale_f
```

```
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
#save table of average quarterly values
region_quarters_table <- quarters |> group_by(part, quarter) |> summarize(avg_quart_vis = mean(quarter_
  mutate(lower = avg_quart_vis - 1.96*sd/3, upper = avg_quart_vis + 1.96*sd/3) |>
  mutate("Average_Visits(95%CI)" = paste0(round(avg_quart_vis, 2), " ", "(", round(lower,2), "-", round
  select(part, quarter, `Average_Visits(95%CI)`)
```

```
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

Calculate average subregional visits over 3 month chunks, becomes Figure S5b

```
sub_quarters <- left_join(subregions_weighted_allyears, quarter_df) |> group_by(part, YEAR, quarter) |>
                                      "West North Central", "South Atlantic", "Middle Atlantic",
                                      "Mountain West", "East North Central", "New England", "Pacific We
```

```
## Joining with `by = join_by(MONTH)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
```

```
subregion_quarters_plot <- sub_quarters |> group_by(part, quarter) |> summarize(avg_quart_vis = mean(qua
  mutate(lower = avg_quart_vis - 1.96*sd/3, upper = avg_quart_vis + 1.96*sd/3) |>
  ggplot(aes(quarter, avg_quart_vis, fill = part)) + geom_bar(position = 'dodge', stat = 'identity') +
  geom_errorbar(aes(x =quarter, ymin = lower, ymax = upper), position = 'dodge',  alpha = 0.2)+ scale_fi
```

```
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
subregion_quarters_table <- sub_quarters |> group_by(part, quarter) |> summarize(avg_quart_vis = mean(qu
  mutate(lower = avg_quart_vis - 1.96*sd/3, upper = avg_quart_vis + 1.96*sd/3) |>
  mutate("Average_Visits(95%CI)" = paste0(round(avg_quart_vis, 2), " ", "(", round(lower,2), "-", round
  select(part, quarter, `Average_Visits(95%CI)`)
```

```
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

Calculate average differences between regions, becomes Supplementary Table 4

```
#initialize df
quarter_diff_allyears <- expand_grid(Comparison = c("West-South", "West-Northeast", "West-Midwest", "Sou

#loop through rows in df and fill
for(i in 1:nrow(quarter_diff_allyears)){
  R1 = quarters |> filter(part == strsplit(quarter_diff_allyears$Comparison[i], "-")[[1]][1],
                          YEAR == quarter_diff_allyears$Year[i],
                          quarter == quarter_diff_allyears$Quarter[i]) |>
    pull(quarter_vis)
  R2 = quarters |> filter(part == strsplit(quarter_diff_allyears$Comparison[i], "-")[[1]][2],
                          YEAR == quarter_diff_allyears$Year[i],
                          quarter == quarter_diff_allyears$Quarter[i]) |>
    pull(quarter_vis)
  quarter_diff_allyears$Diff[i] <- R1-R2
}

quarter_diffs_table <- quarter_diff_allyears |> group_by(Comparison, Quarter) |> summarize(avg_diff = me
  mutate(abs_avg_diff = abs(avg_diff),
         lower = abs_avg_diff - 1.96*sd/3,
         upper = abs_avg_diff + 1.96*sd/3) |>
  mutate('Abs_Avg_diff(CI)' = paste0(round(abs_avg_diff, 2), " ", "(", round(lower, 2), "-", round(upper
  select(Comparison, Quarter, `Abs_Avg_diff(CI)`)
```

```
## `summarise()` has grouped output by 'Comparison'. You can override using the
## `.groups` argument.
```

Statistically compare each month and region or subregion

T-tests comparing each region-month, becomes Supplementary Figure 6a

```
#create table of regional differences and p-values from T-test
#using helper function "get_diff_df" from "monthly_average_functions.R"
wide_table_regions <- get_diff_df(c("Hawaii", "Alaska"), regions) |>
  filter(region1 != region2) |>
  mutate("Diff(pval)" = paste0(round(diff,3), "(",round(pval,8), ')')) |>
  select(-region1, -region2, -diff, -pval) |>
  pivot_wider(names_from = Comparison, values_from = `Diff(pval)`)
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```r
#build heat map
#calculate regional differences
region_US_cont <- get_diff_df(c("Hawaii", "Alaska"), regions)
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
#calculate bonferroni-corrected p-value threshold
val <- 0.05 / nrow(region_US_cont)

#remove self and redundant comparisons
region_heatmap_significance_noredund <- region_US_cont |> mutate(sig = ifelse(pval < val, 0,1)) |>
  filter(region1 != region2) |>
  filter(!Comparison %in% c("South-West", "Northeast-West",
                            "Northeast-South", "Midwest-West",
                            "Midwest-Northeast", "Midwest-South")) |>
  ggplot(aes(x = MONTH, y = Comparison, fill = factor(sig))) + geom_tile() +
  xlab("Month") +
  scale_fill_manual(values = c('#0096FF','grey'),  labels = c("Yes", "No")) + theme_minimal() +
  labs(fill = "Significant") +
  scale_x_discrete(limits = c("JAN", "FEB", "MAR", "APR", "MAY", "JUN", "JUL", "AUG", "SEPT","OCT", "NOV
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
        plot.title = element_text(hjust = 0.5, size = 20))
```

T-tests comparing each subregion-month, becomes Supplementary Figure 6a

```
#create table of subregional differences and p-values from T-test
wide_table_subregions <- get_diff_df(c("Hawaii", "Alaska"), subregions) |>
  filter(region1 != region2) |>
  mutate("Diff(pval)" = paste0(round(diff,3), "(",round(pval,8), ')')) |>
  select(-region1, -region2, -diff, -pval) |>
  pivot_wider(names_from = Comparison, values_from = `Diff(pval)`)
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
#subregion heat map
subregion_order_heatmap <- data.frame(region2 = c("Pacific West", "Mountain West", "West North Central"
subregion_US_cont <- get_diff_df(c("Hawaii", "Alaska"), subregions)
```

```
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE, SEX, AGEGRP, YEAR)`
## Joining with `by = join_by(STATE, SEX, AGEGRP)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
val2 <- 0.05 / nrow(subregion_US_cont)
subregion_US_cont <- subregion_US_cont |>
  mutate(sig = ifelse(pval < val2, 0,1)) |>
  filter(region1 != region2)

#remove redundancies
subregion_US_cont2 <- subregion_US_cont |>
  filter(!Comparison %in% c("Pacific West-Mountain West",
                            "Pacific West-West North Central",
                            "Mountain West-West North Central",
                            "Pacific West-West South Central",
                            "Mountain West-West South Central",
                            "West North Central-West South Central",
                            "Pacific West-East North Central",
                            "Mountain West-East North Central",
                            "West North Central-East North Central",
                            "West South Central-East North Central",
                            "Pacific West-East South Central",
                            "Mountain West-East South Central",
                            "West North Central-East South Central",
                            "West South Central-East South Central",
                            "East North Central-East South Central",
                            "Pacific West-Middle Atlantic",
                            "Mountain West-Middle Atlantic",
                            "West North Central-Middle Atlantic",
                            "West South Central-Middle Atlantic",
                            "East North Central-Middle Atlantic",
                            "East South Central-Middle Atlantic",
                            "Pacific West-South Atlantic",
                            "Mountain West-South Atlantic",
                            "West North Central-South Atlantic",
                            "West South Central-South Atlantic",
                            "East North Central-South Atlantic",
                            "East South Central-South Atlantic",
                            "Middle Atlantic-South Atlantic",
                            "Pacific West-New England",
                            "Mountain West-New England",
                            "West North Central-New England",
                            "West South Central-New England",
                            "East North Central-New England",
                            "East South Central-New England",
                            "Middle Atlantic-New England",
                            "South Atlantic-New England")) |>
  left_join(subregion_order_heatmap) |>
  group_by(MONTH) |>
  arrange(order) |>
  mutate(Comparison2 = str_to_title(gsub("_", " ", Comparison, fixed=TRUE))) |>
  select(-Comparison) |>
  mutate(Comparison = Comparison2) |>
  select(-Comparison2)
```

```
## Joining with `by = join_by(region2)`
```

```
subregion_heatmap_significance_noredund <- subregion_US_cont2 |>
  ggplot(aes(x = MONTH, y = reorder(Comparison, -order), fill = factor(sig))) + geom_tile() +
  scale_fill_manual(values = c('#0096FF','grey'),  labels = c("Yes", "No")) + theme_minimal() +
  ylab("Comparison") +
  xlab("Month") +
  labs(fill = "Significant") +
  scale_x_discrete(limits = c("JAN", "FEB", "MAR", "APR", "MAY", "JUN", "JUL", "AUG", "SEPT","OCT", "NOV
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
        plot.title = element_text(hjust = 0.5, size = 25))
```

Sinusoidal Fitting

Fitting state visit trends to sinusoids

```
#state_sinusoids_parameters and state_sinusoids called from get_state_sinusoids.R helper file
#make a table of all state phases (Supplementary Table 5)
phase_table <- state_sinusoids_parameters |> mutate(Phase_corrected = Phase %% 12) |>
  mutate(Phase_lower = Phase_corrected - 1.96*Phase_SE,
         Phase_upper = Phase_corrected + 1.96*Phase_SE) |>
  mutate(Final_phase = paste0(round(Phase_corrected,2), " ", "(", round(Phase_lower,2), "-", round(Phase
  filter(!State %in% c("South Carolina", "Hawaii", "Alaska")) |>
  select(State, Final_phase) |>
  mutate("Phase(95% CI)" = Final_phase) |>
  select(-Final_phase)
```

Plot state phases on a map of the U.S., becomes Figure 2.

```
#load US shapefile
my_spdf <- readOGR(dsn = "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/Strep_project/R
                   layer = "cb_2018_us_state_500k")
```

```
## Warning: OGR support is provided by the sf and terra packages among others

## Warning: OGR support is provided by the sf and terra packages among others

## Warning: OGR support is provided by the sf and terra packages among others

## Warning: OGR support is provided by the sf and terra packages among others

## Warning: OGR support is provided by the sf and terra packages among others

## Warning: OGR support is provided by the sf and terra packages among others

## OGR data source with driver: ESRI Shapefile
## Source: "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/Strep_project/R code/plots/Mo
## with 56 features
## It has 9 fields
## Integer64 fields read as strings:  ALAND AWATER
```

```
# https://www.census.gov/geographies/mapping-files/time-series/geo/carto-boundary-file.html
spdf_fortified <- tidy(my_spdf)
```

```
## Warning: `tidy.SpatialPolygonsDataFrame()` was deprecated in broom 1.0.4.
## i Please use functions from the sf package, namely `sf::st_as_sf()`, in favor
##   of sp tidiers.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## Regions defined for each Polygons
```

```
#remove non-continental US states and territories
non_continental_ids <- which(!(my_spdf$NAME %in% c("American Samoa" ,
                                   "Commonwealth of the Northern Mariana Islands",
                                   "Guam",
                                   "Hawaii",
                                   "Alaska",
                                   "Puerto Rico",
                                   "United States Virgin Islands")))
continental <- my_spdf[non_continental_ids,]

#fortify shapefile
continental_fortified <- tidy(continental)
```

```
## Regions defined for each Polygons
```

```
#add back in state names which were lost in tidy()
temp_df <- data.frame(my_spdf@data$NAME)
names(temp_df) <- c("NAME")
temp_df$id <-as.character(seq(0, nrow(temp_df)-1))
state_names <- left_join(continental_fortified, temp_df) |> mutate(State = NAME) |> select(-NAME)
```

```
## Joining with `by = join_by(id)`
```

```
continental_fortified_tojoin <- left_join(continental_fortified, state_names) |> select(-id) |> mutate(
## Joining with `by = join_by(long, lat, order, hole, piece, group, id)`
```

```
#correct state phases for map
state_sinusoids_parameters_tojoin <- state_sinusoids_parameters |> mutate(Phase_corrected = Phase %% 12
                                                                          id = State) |> select(id, Pha
#add phase data to shapefile
continental_phases <- left_join(continental_fortified_tojoin, state_sinusoids_parameters_tojoin)
```

```
## Joining with `by = join_by(id)`
```

```
#remove South Carolina
continental_phases_noSC <- continental_phases
continental_phases_noSC[which(continental_phases_noSC$id == "South Carolina"),8] = NA
sc <- continental_phases[which(continental_phases$id == "South Carolina"),]
state_phases_map_noSC <- ggplot() +
  geom_polygon(data = continental_phases_noSC, aes(x = long, y = lat, group = group, fill = Phase_corre
  geom_polygon_pattern(data = sc, aes(x = long, y = lat, group= group), fill = "gray", pattern = 'strip
                       pattern_spacing = 0.005, pattern_alpha = 0.5) +
  theme_void() + ggtitle("") + scale_fill_gradient(low = "#30239B", high =  "#FFFFFF", breaks = c(0, 1,
                                                   labels = c("Dec", "Jan", "Feb", "Mar"),
                                                   limits = range(0,3)) +
  labs(fill = "Peak Month") +
  theme(plot.title = element_text(hjust = 0.5, size = 25),
        plot.subtitle = element_text(hjust = 0.5, size = 23),
        legend.position = 'bottom')
```

Plot state sinusoids with confidence intervals, becomes Supplementary Figure 9

```
state_sinusoids_subregions <- ggplot() +
  geom_point(aes(x = MONTH, y = mean_weighted_cases_per_thousand, color = Region), data = state_sinusoi
  geom_ribbon(aes(x = MONTH, ymin = lower, ymax = upper, fill = Region), data = state_sinusoids_withCIs_
  geom_line(aes(x = MONTH, y = prediction, color = Region), data = state_sinusoids_withCIs_cont_sub, li
  scale_color_manual(values= subcolors_df$region_color, labels=gsub("_", " ",subcolors_df$Region , fixe
  scale_fill_manual(values = subcolors_df$region_color, labels=gsub("_", " ",subcolors_df$Region , fixe
  facet_wrap(~STATE) +
  # ggtitle("Streptococcal Pharyngitis Visits State Sinusoidal Fits") +
  labs(color = "Subregion", fill = "Subregion") +
  xlab("Month") +
  ylab("Visits per 1,000 People") +
  scale_x_discrete(limits = c("JAN", "FEB", "MAR", "APR", "MAY", "JUN", "JUL", "AUG", "SEPT","OCT", "NO
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
        plot.title = element_text(hjust = 0.5, size = 20))
```

Fit sinusoids to region trends and plot. Becomes Supplementary Figure 8a.

```
#get_regions_sinusoids() from "region_subregion_sinusoid_functions.R"
region_sinusoids <- get_region_sinusoids(c("Hawaii", "Alaska"))[[1]] |>
   mutate(Region = factor(part, levels = c("South", "Midwest", "Northeast", "West")))
```

```
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
## Warning: Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
```

```
## Joining with `by = join_by(part, MONTH)`
```

```r
region_sinusoids_CIs_plot_points <- region_sinusoids |>
  ggplot(aes(x=MONTH)) +
  geom_line(aes(x= MONTH, y = prediction, color = Region), linetype = 2) +
  geom_point(aes(x= MONTH, y = yearly_average_sum_region_cases_per_thousand, color = Region), size = 1,
  geom_ribbon(aes(ymin = lower, ymax = upper, fill= Region), alpha = 0.2) +
  # ggtitle("Streptococcal Pharyngitis Visits Region Sinusoidal Fits") +
  xlab("Month") +
  labs(color = "Region", fill = "Region") +
  ylab("Cases per 1,000 People") +
  scale_color_manual(values= colors_df$region_color, labels = colors_df$Region) +
  scale_fill_manual(values = colors_df$region_color, labels = colors_df$Region) +
  scale_x_discrete(limits = c("JAN", "FEB", "MAR", "APR", "MAY", "JUN", "JUL", "AUG", "SEPT","OCT", "NO
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
        plot.title = element_text(hjust = 0.5, size = 20))
```

Plot regional sinusoid phases. Becomes Supplementary Figure 8b.

```r
region_sinusoids_parameters <- get_region_sinusoids(c("Hawaii", "Alaska"))[[2]]
```

```
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
## Warning: Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
```

```
## Joining with `by = join_by(part, MONTH)`
```

```r
order <- region_sinusoids_parameters |> arrange(-Phase) |> pull(Region)
order_df <- data.frame(Region = str_to_title(order), num = c(1:4))
regions_colors <- left_join(colors_df, order_df) |> arrange(num)
```

```
## Joining with `by = join_by(Region)`
```

```
region_sinusoids_parameters <- region_sinusoids_parameters |> mutate(Region = factor(Region, order))
region_phase_plot <- region_sinusoids_parameters |>
  mutate(lower = Phase - 1.96*Phase_SE, upper = Phase + 1.96*Phase_SE) |>
  ggplot(aes(x = Phase %% 12, y = Region, color = Region)) + geom_point() + geom_errorbar(aes(xmin = low
  theme_minimal() +
  # ggtitle(paste("Phase Estimates for Region Sinusoids")) +
  ylab("Region") + xlab("Month")+
  scale_x_discrete(limits = c("JAN", "FEB", "MAR", "APR", "MAY", "JUN", "JUL", "AUG", "SEPT","OCT", "NO
  scale_color_manual(values= regions_colors$region_color, labels=regions_colors$Region) +
  scale_y_discrete(labels=levels(region_sinusoids_parameters$Region))+
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
        plot.title = element_text(hjust = 0.5, size = 20))
```

Fit sinusoids to subregional data and plot. Becomes Supplementary Figure 7a.

```
subregion_sinusoids <- get_subregion_sinusoids(c("Hawaii", "Alaska"))[[1]] |>
  mutate(Region = factor(part, c("East South Central", "West South Central",
                                 "West North Central", "South Atlantic", "Middle Atlantic",
                                 "Mountain West", "East North Central", "New England", "Pacific We
```

```
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
## Warning: Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
```

```
## Joining with `by = join_by(part, MONTH)`
```

```
subregion_sinusoids_CIs_plot_points <- subregion_sinusoids |>
  ggplot(aes(x=MONTH)) +
  geom_line(aes(x= MONTH, y = prediction, color = Region), linetype = 2) +
  geom_point(aes(x= MONTH, y = yearly_average_sum_region_cases_per_thousand, color = Region), size = 1,
  geom_ribbon(aes(ymin = lower, ymax = upper, fill= Region), alpha = 0.2) +
  # ggtitle("Streptococcal Pharyngitis Visits Subregion Sinusoidal Fits") +
  labs(color = "Subregion", fill = "Subregion") +
  ylab("Visits per 1,000 People") +
  scale_x_discrete(limits = c("JAN", "FEB", "MAR", "APR", "MAY", "JUN", "JUL", "AUG", "SEPT","OCT", "NO
  scale_color_manual(values= subcolors_df$region_color, labels=subcolors_df$Region) +
  scale_fill_manual(values= subcolors_df$region_color, labels=subcolors_df$Region) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
        plot.title = element_text(hjust = 0.5, size = 20))
```

Plot subregion phases. Becomes Supplementary Figure 7b.

```
subregion_sinusoids_parameters <- get_subregion_sinusoids(c("Hawaii", "Alaska"))[[2]]
```

```
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## Joining with `by = join_by(STATE)`
## Joining with `by = join_by(part)`
## `summarise()` has grouped output by 'part', 'YEAR'. You can override using the
## `.groups` argument.
## `summarise()` has grouped output by 'part'. You can override using the
## `.groups` argument.
```

```
## Warning: Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
## Setting row names on a tibble is deprecated.
```

```
## Joining with `by = join_by(part, MONTH)`
```

```
order <- subregion_sinusoids_parameters |> arrange(-Phase) |> pull(Region)
order_df <- data.frame(Region = order, num = 1:9)
subregions_colors <- left_join(subcolors_df, order_df) |> arrange(num)
```

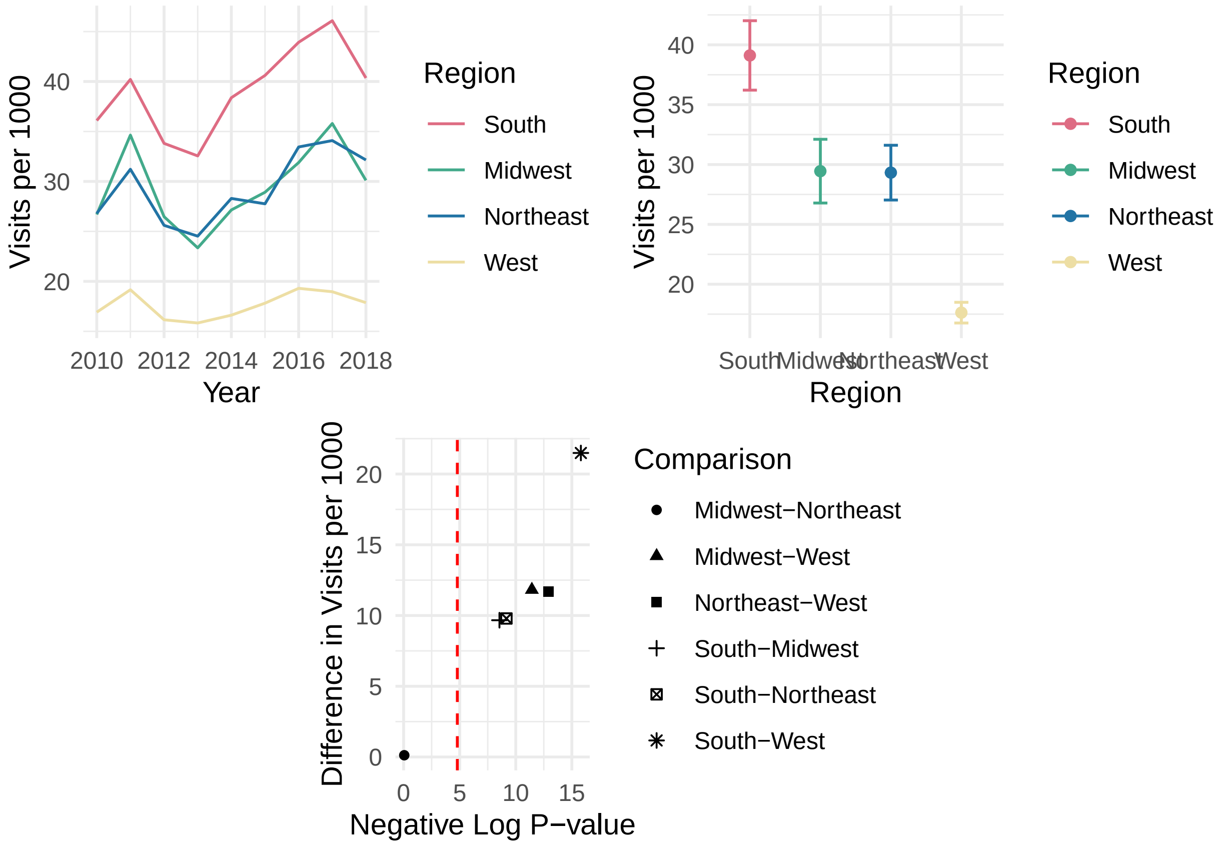
```
## Joining with `by = join_by(Region)`
```

```
subregion_sinusoids_parameters <- subregion_sinusoids_parameters |> mutate(Region = factor(Region, orde
subregion_phase_plot <- subregion_sinusoids_parameters |>
  mutate(lower = Phase - 1.96*Phase_SE, upper = Phase + 1.96*Phase_SE) |>
  ggplot(aes(x = Phase %% 12, y = Region, color = Region)) + geom_point() + geom_errorbar(aes(xmin = low
  theme_minimal() +
  # ggtitle(paste("Phase Estimates for Subregion Sinusoids")) +
  ylab("Subregion") + xlab("Month")+
  scale_x_discrete(limits = c("JAN", "FEB", "MAR", "APR", "MAY", "JUN", "JUL", "AUG", "SEPT","OCT", "NO
  scale_color_manual(values= subregions_colors$region_color, labels=gsub("_", " ",subregions_colors$Regi
  scale_y_discrete(labels=levels(subregion_sinusoids_parameters$Region)) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
        plot.title = element_text(hjust = 0.5, size = 20))
```

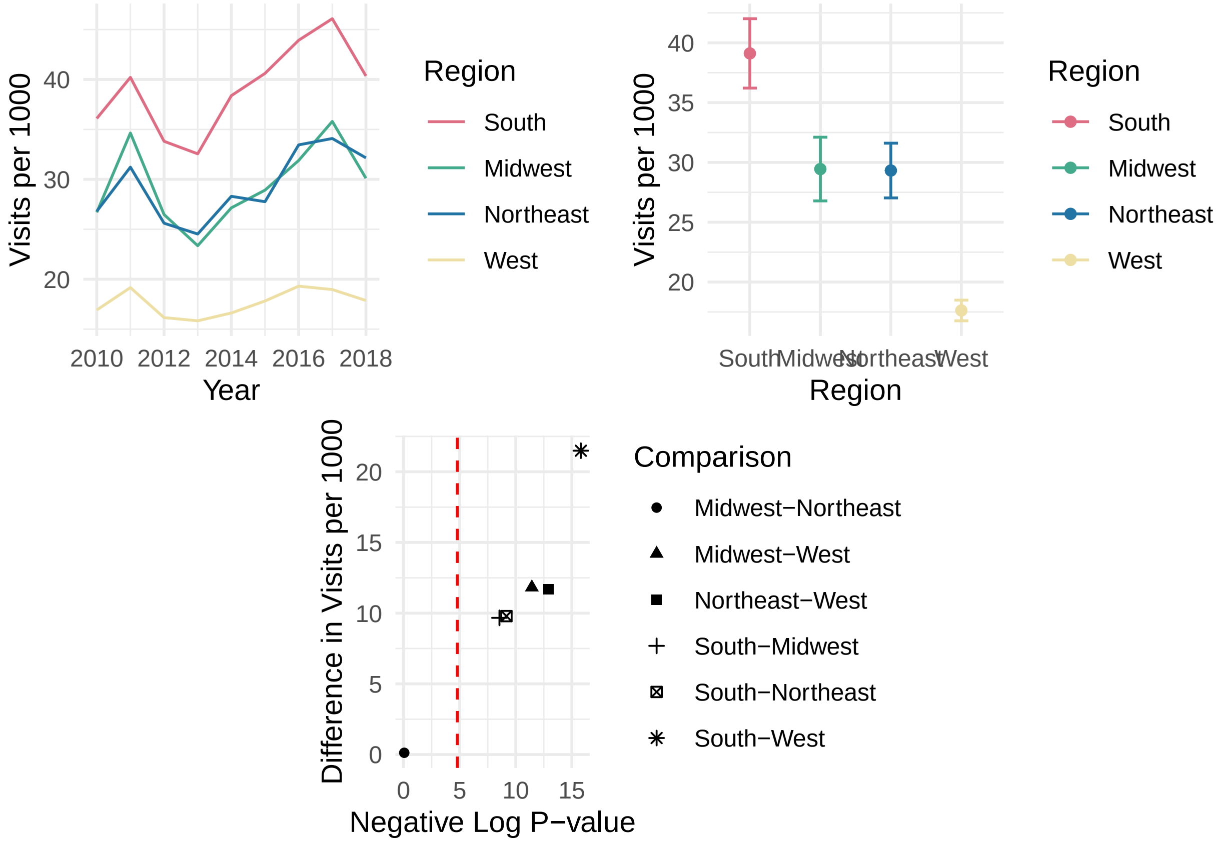
Save region and subregion sinusoid phases into tables

```
#Calculate subregion phases and 95% CIs
subregion_sinusoids_phase_table <- subregion_sinusoids_parameters |>
  arrange(Phase) |>
  mutate(lower = Phase - 1.96*Phase_SE, upper = Phase + 1.96*Phase_SE) |>
  mutate(CI = paste0("(", round(lower,2),"-", round(upper,2), ")")) |>
  mutate(Phase_CI = paste(round(Phase, 2), CI)) |>
  mutate(Subregion = Region) |>
  select(Subregion, Phase_CI)

#Calculate region phases and 95% CIs
region_sinusoids_phase_table <- region_sinusoids_parameters |>
  arrange(Phase) |>
  mutate(lower = Phase - 1.96*Phase_SE, upper = Phase + 1.96*Phase_SE) |>
  mutate(CI = paste0("(", round(lower,2),"-", round(upper,2), ")")) |>
  mutate(Phase_CI = paste(round(Phase, 2), CI)) |>
  mutate(Subregion = Region) |>
  select(Subregion, Phase_CI)
```

```
#write all tables
write_csv(table1, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/outpu
write_csv(region_table, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/
write_csv(subregion_table, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngi
write_csv(region_table_month, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPhary
write_csv(subregion_table_month, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPha
write_csv(region_quarters_table, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPha
write_csv(subregion_quarters_table, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/Strep
write_csv(quarter_diffs_table, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPhar
write_csv(wide_table_regions, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPhary
write_csv(wide_table_subregions, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPha
write_csv(phase_table, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/
write_csv(subregion_sinusoids_parameters, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLa
write_csv(region_sinusoids_parameters, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/S
write_csv(subregion_sinusoids_phase_table, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradL
write_csv(region_sinusoids_phase_table, "/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/

#save all figures as pngs and pdfs
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS1.png
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS1.pd
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS2a.p
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS2a.p
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS2b.p
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS2b.p
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS2c.p
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS2c.p

#put multi-panel figures together
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS2.pd
```



```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS2.png
```



```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS3a.p
```

```
## Saving 10 x 4.5 in image
```

```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS3a.p
```

```
## Saving 10 x 4.5 in image
```

```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS3b.p
```

```
## Saving 6.5 x 4.5 in image
```

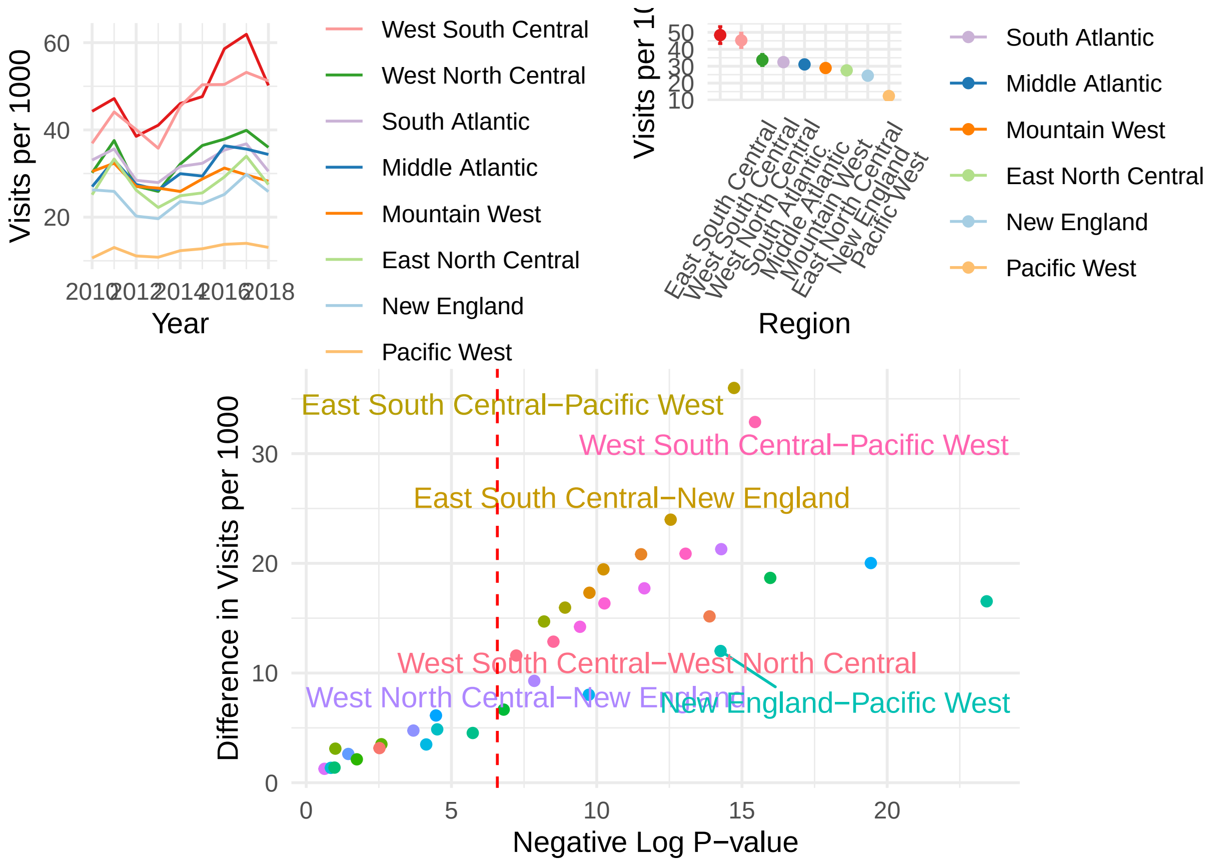
```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS3b.p
```

```
## Saving 6.5 x 4.5 in image
```

```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS3c.p
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS3c.p

ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS3.pd
```

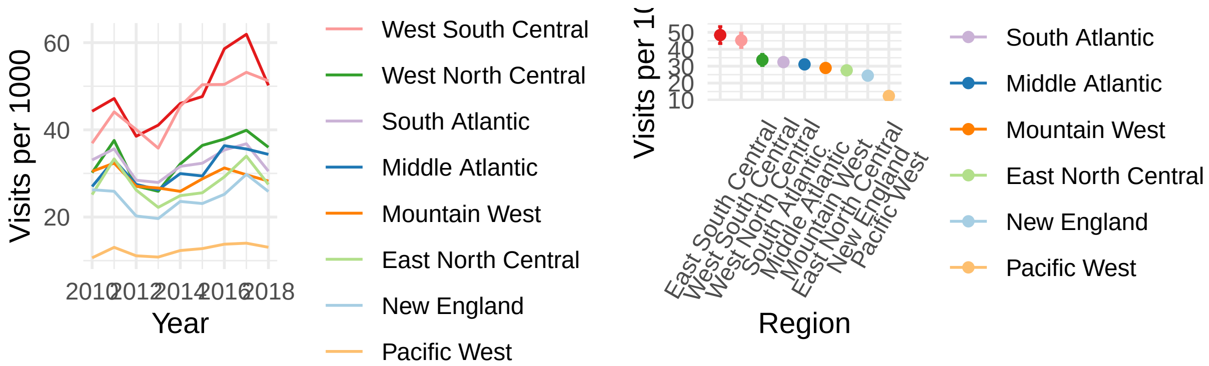
```
## Warning: ggrepel: 30 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

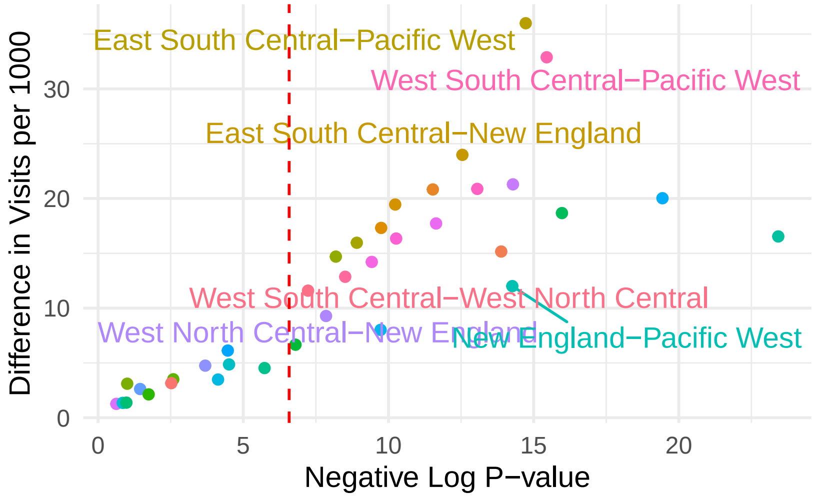
```
## Warning: ggrepel: 6 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



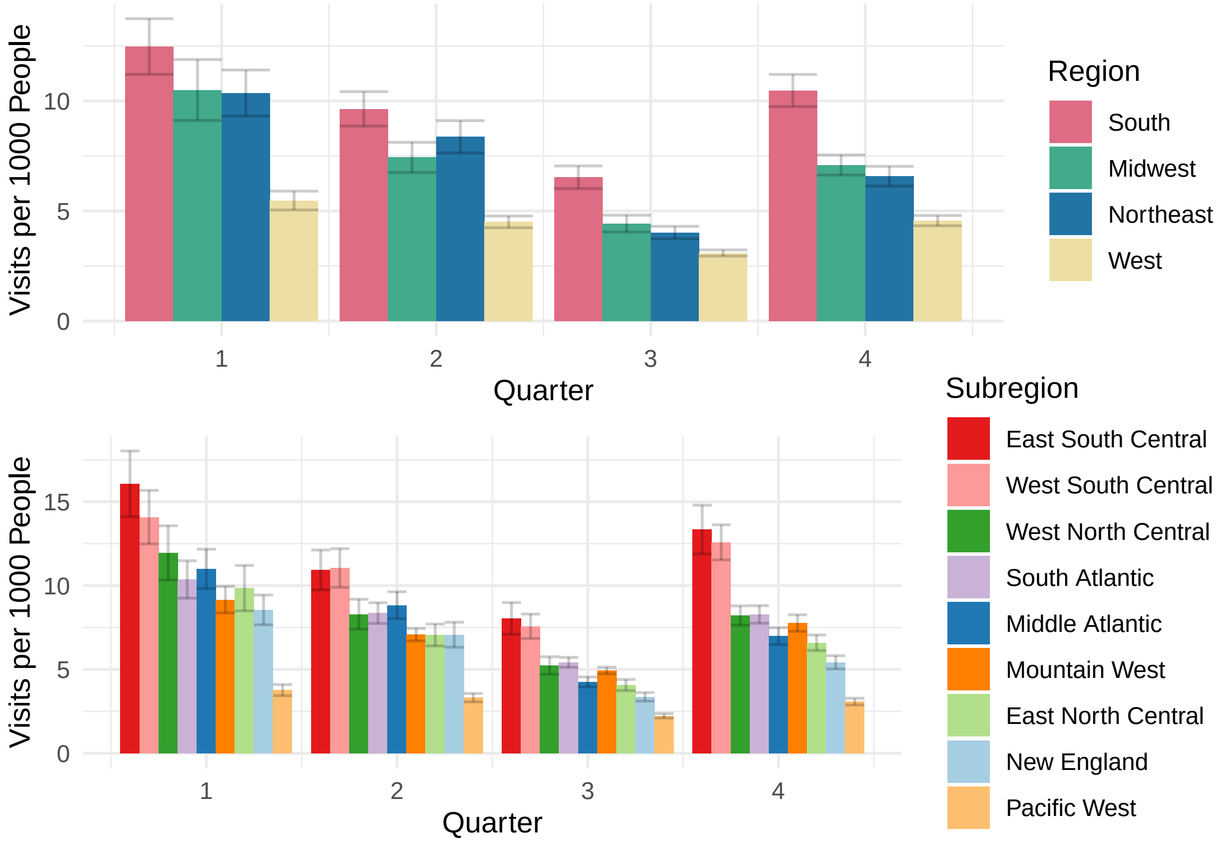
```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS3.png
```

```
## Warning: ggrepel: 30 unlabeled data points (too many overlaps). Consider increasing max.overlaps
## ggrepel: 6 unlabeled data points (too many overlaps). Consider increasing max.overlaps
```

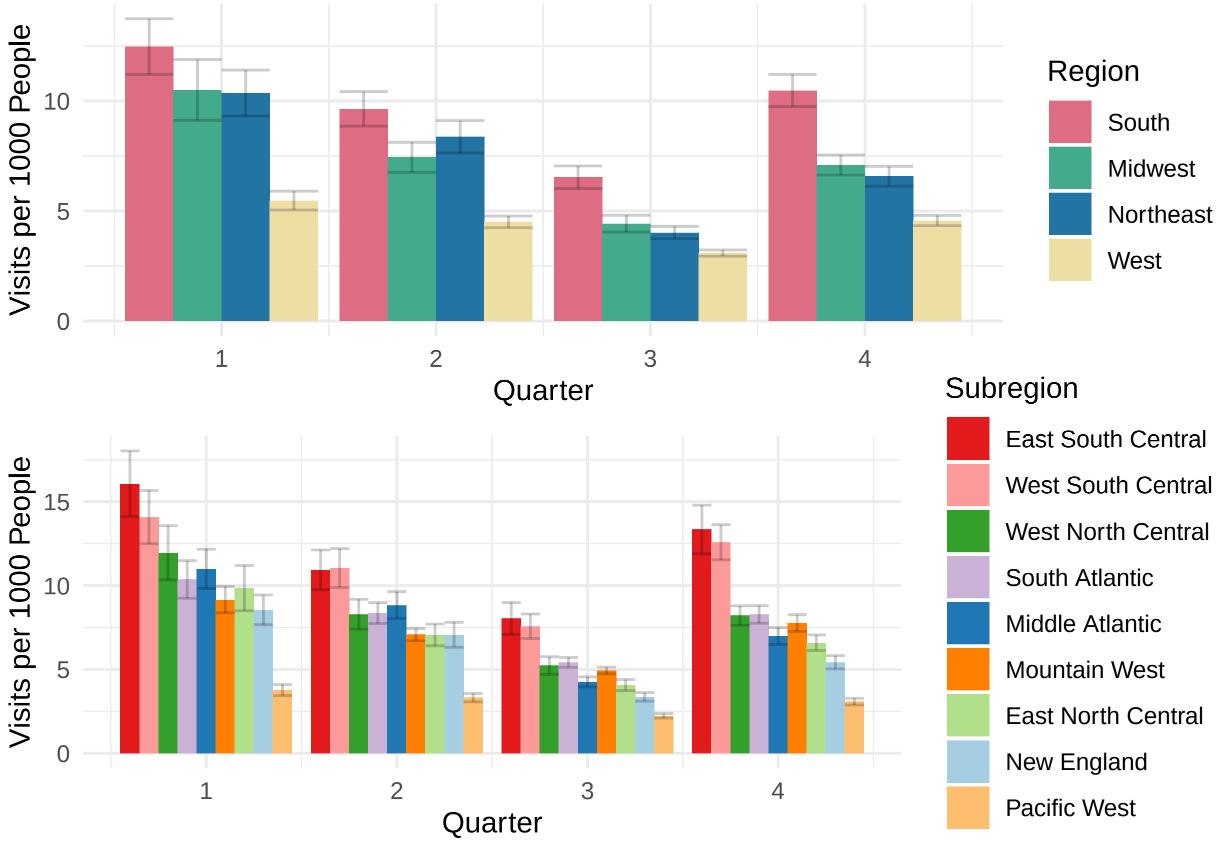




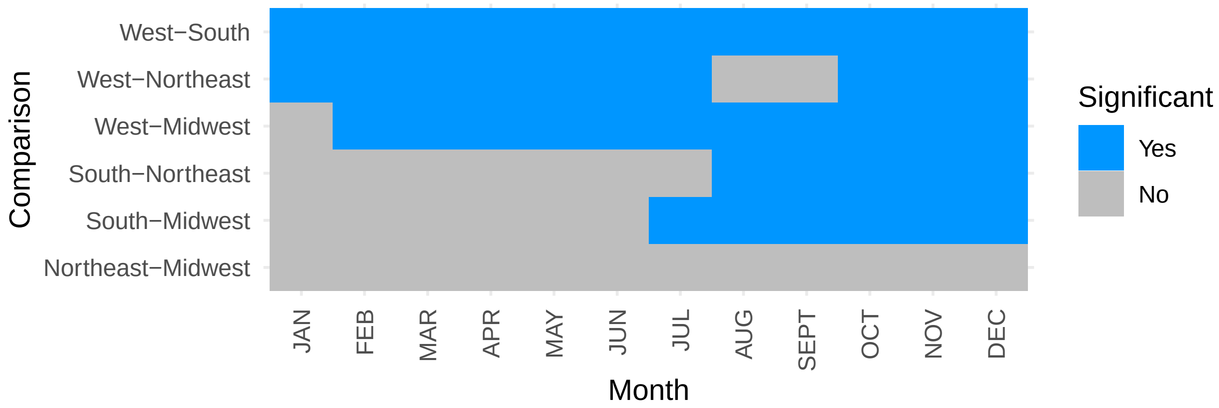
```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/fig1.png
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/fig1.pdf
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS4.png
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS4.pdf
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS5a.pn
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS5a.pd
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS5b.pn
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS5b.pd

#layout quarters plot for figure
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS5.pdf
```

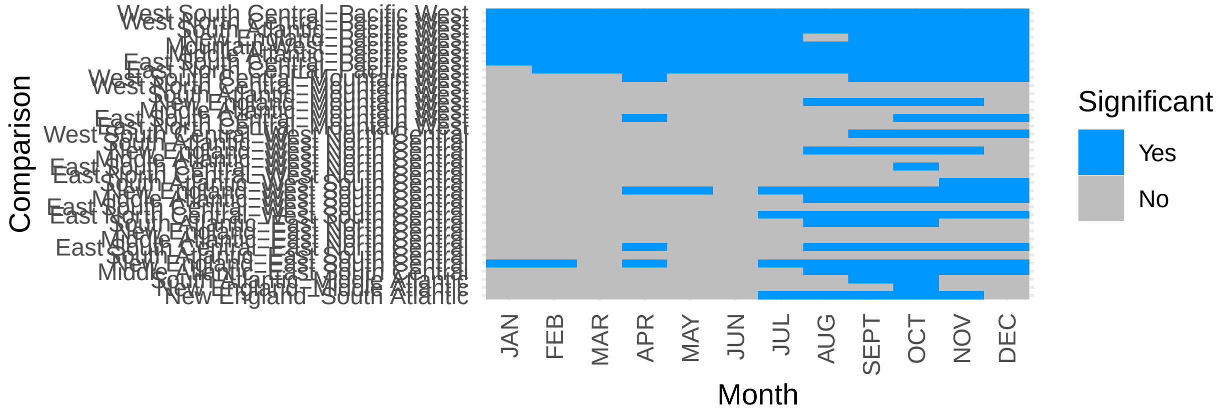


```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS5.png
```

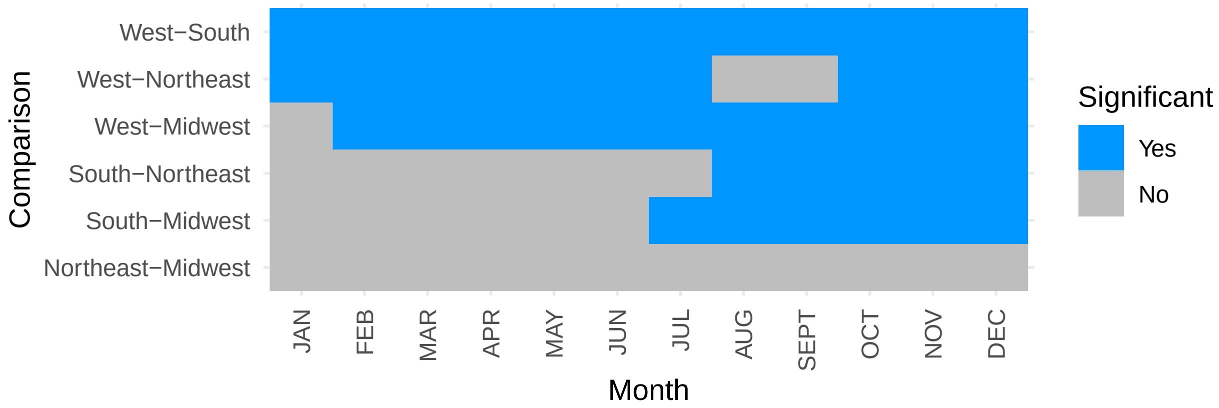


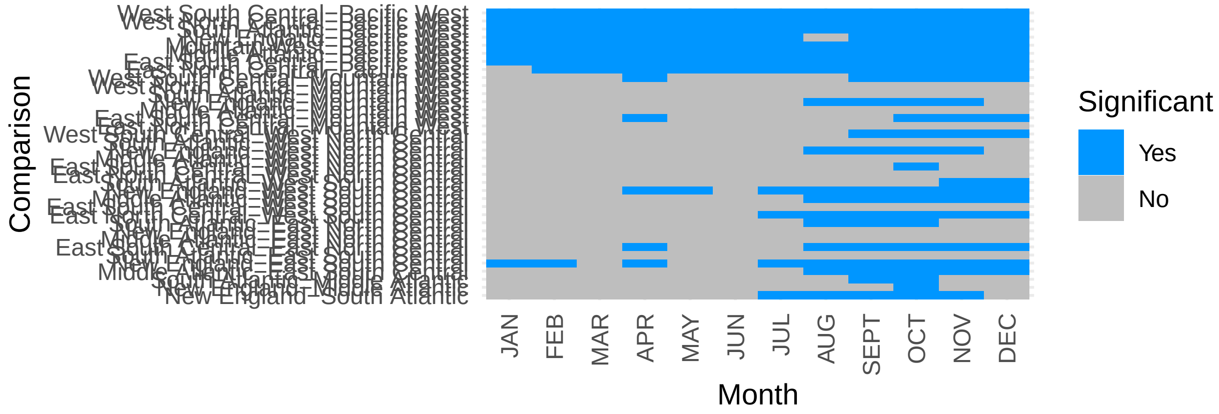
```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS6a.pn
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS6a.pd
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS6b.pn
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS6b.pd
#layout heatmaps for figure
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS6.pdf
```



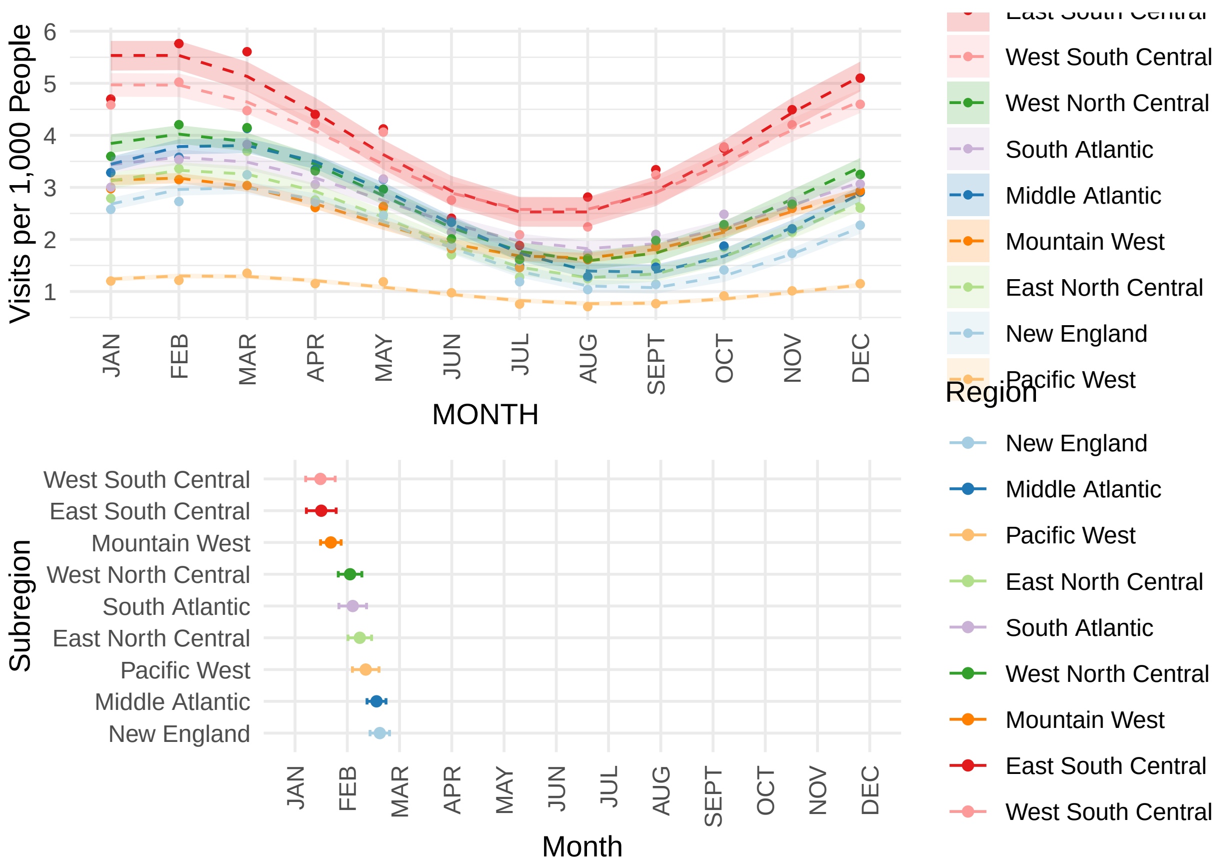


```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS6.png
```

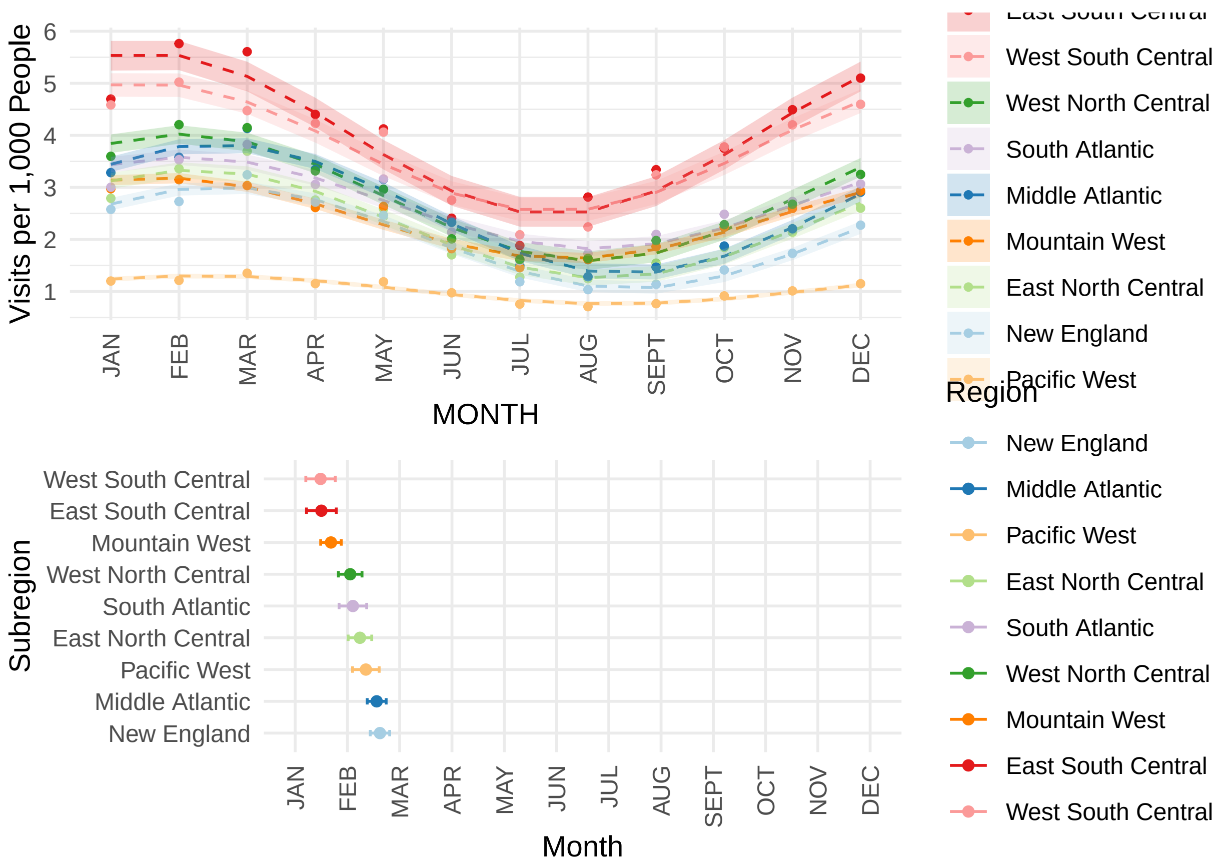




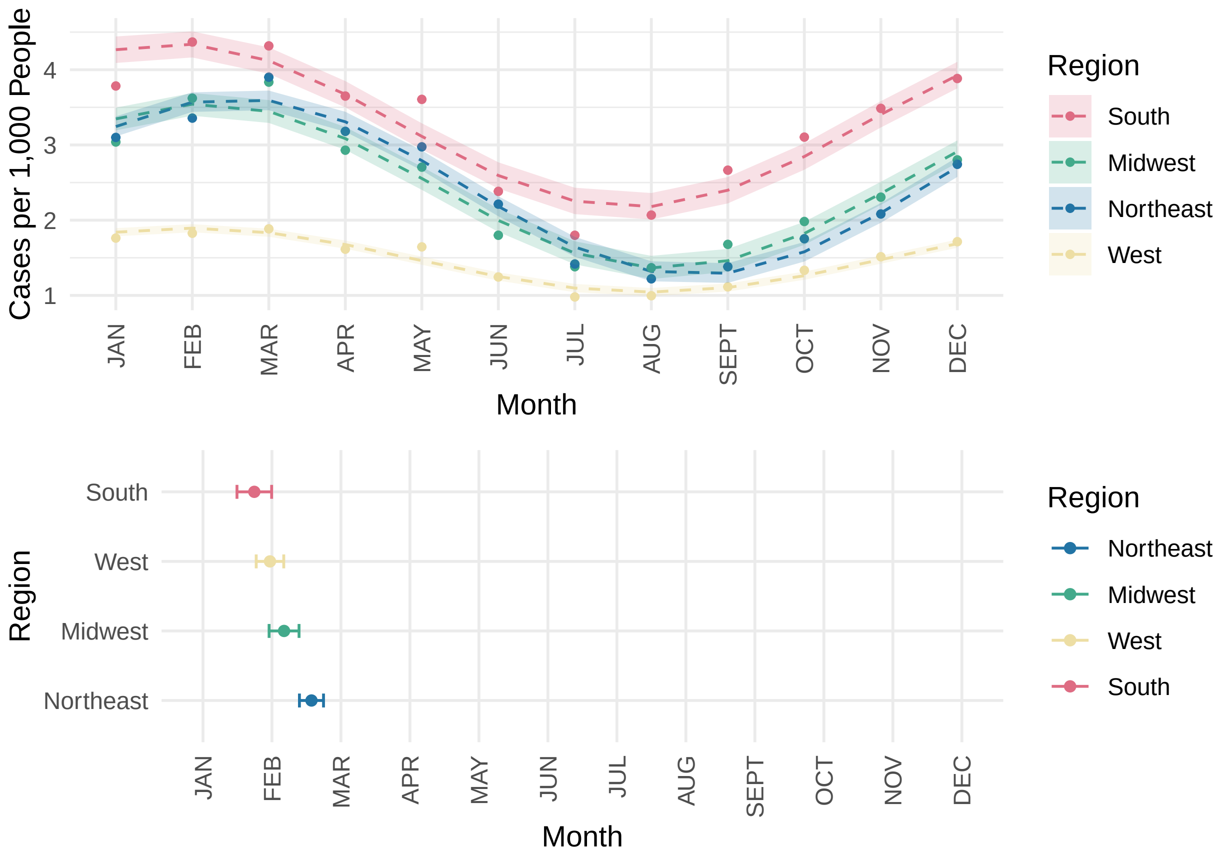
```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/fig2.png
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/fig2.pdf
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS7a.pn
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS7a.pd
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS7b.pn
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS7b.pd

#layout sinusoid plots and phases
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS8.pdf
```



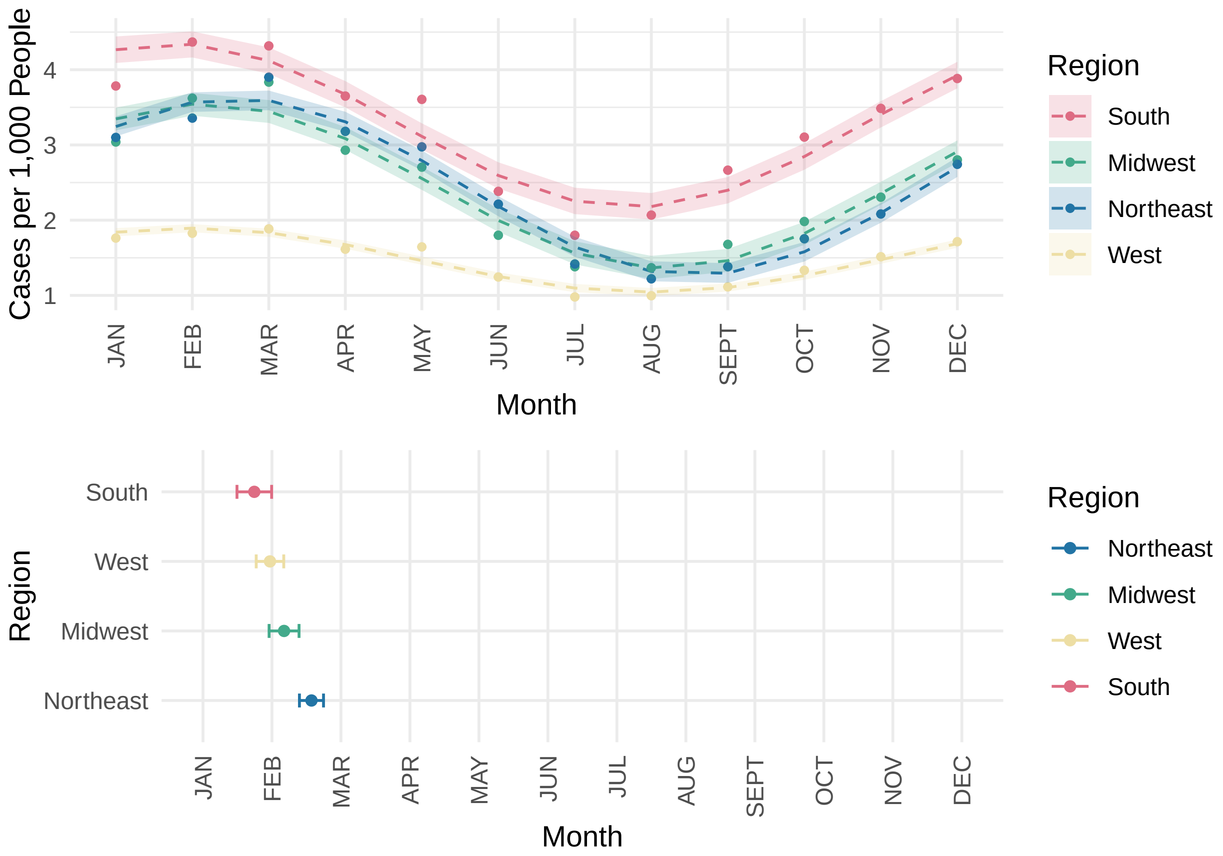
```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS8.png
```



```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS8a.p
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS8a.p
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS8b.p
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS8b.p

#layout region sinusoids
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS9.pd
```



```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS9.png
```



```
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS9.png
ggsave("/Users/madeleinekline/Dropbox (Harvard University)/G1/GradLab/StrepPharyngitis/figures/figS9.pdf
```