Climate Analysis with ClimateMosaic

Jasmine Wang, Flavia Jiang

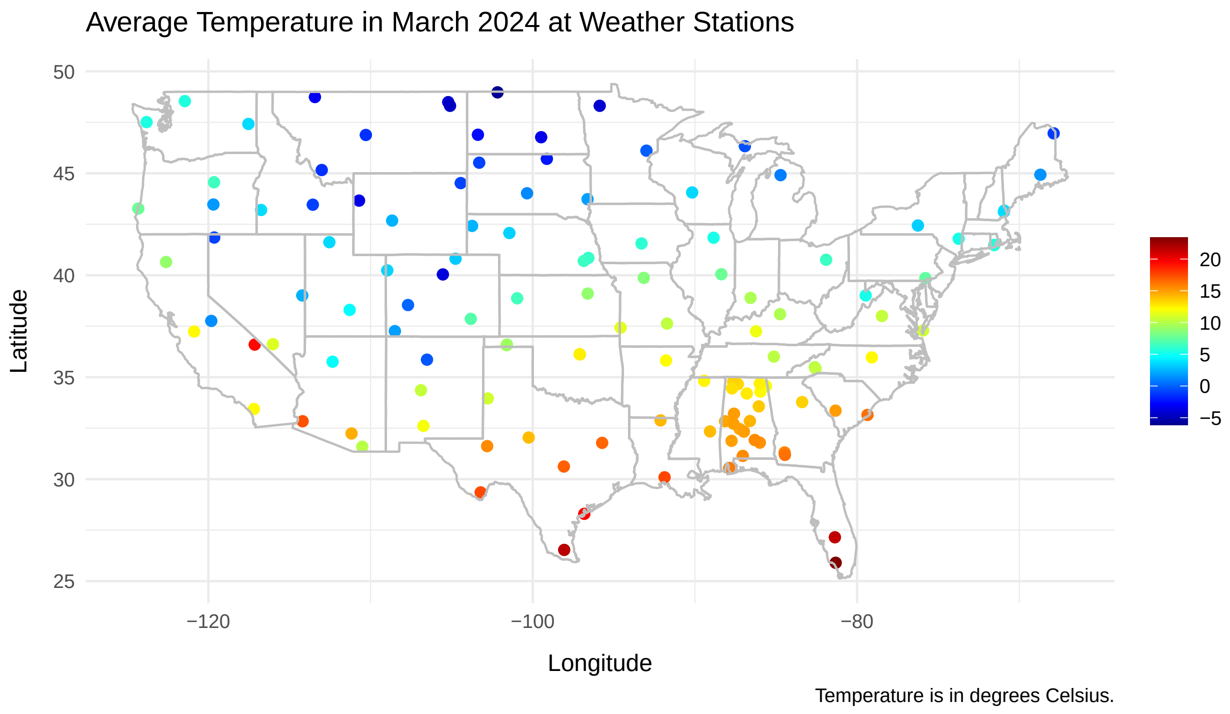
```r
# Import necessary packages
#' @import elevatr
#' @import ClimateMosaic
#' @import ggplot2
#' @import dplyr

library(ggplot2)
library(elevatr)
#> elevatr v0.99.0 NOTE: Version 0.99.0 of 'elevatr' uses 'sf' and 'terra'.  Use
#> of the 'sp', 'raster', and underlying 'rgdal' packages by 'elevatr' is being
#> deprecated; however, get_elev_raster continues to return a RasterLayer.  This
#> will be dropped in future versions, so please plan accordingly.
library(dplyr)
#>
#> Attaching package: 'dplyr'
#> The following objects are masked from 'package:stats':
#>
#>     filter, lag
#> The following objects are masked from 'package:base':
#>
#>     intersect, setdiff, setequal, union
library(ClimateMosaic)
```

```r
# prepare data and visualization elements
data("climate_df")
data("station_df")
climate_df_temp <- climate_df[,-c(2,3,5)]
contiguous_station_df <- station_df[station_df$contiguous, ]
us_map <- subset(map_data("state"), !region %in% c("alaska", "hawaii", "puerto rico"))
colors_vector <- c("#00008F", "#000099", "#0000A3", "#0000AE", "#0000B8", "#0000C2",
                   "#0000CC", "#0000D6", "#0000E0", "#0000EA", "#0000F6", "#0000FF",
                   "#0009FF", "#0015FF", "#001FFF", "#0029FF", "#0033FF", "#003DFF",
                   "#0047FF", "#0051FF", "#005CFF", "#0066FF", "#0070FF", "#007AFF",
                   "#0084FF", "#008EFF", "#0098FF", "#00A2FF", "#00ACFF", "#00B6FF",
                   "#00C0FF", "#00CBFF", "#00D5FF", "#00DFFF", "#00E9FF", "#00F4FF",
                   "#00FEFF", "#08FFF7", "#13FFEC", "#1DFFE2", "#27FFD8", "#31FFCE",
                   "#3CFFC3", "#46FFB9", "#50FFAF", "#5AFFA5", "#64FF9B", "#6FFF90",
                   "#79FF87", "#83FF7D", "#8CFF73", "#96FF69", "#A0FF5F", "#ABFF54",
                   "#B5FF4A", "#BFFF40", "#C9FF36", "#D3FF2C", "#DEFF21", "#E7FF18",
                   "#F2FF0D", "#FDFF02", "#FFF900", "#FFED00", "#FFE300", "#FFD900",
                   "#FFCF00", "#FFC500", "#FFBB00", "#FFB000", "#FFA600", "#FF9C00",
                   "#FF9200", "#FF8800", "#FF7F00", "#FF7400", "#FF6A00", "#FF6000",
                   "#FF5600", "#FF4C00", "#FF4100", "#FF3700", "#FF2D00", "#FF2300",
                   "#FF1900", "#FF0E00", "#FF0300", "#FB0000", "#EF0000", "#E50000",
                   "#DB0000", "#D00000", "#C60000", "#BC0000", "#B20000", "#A80000",
                   "#9E0000", "#930000", "#890000", "#800000")
```

Q1: Average Temperature in March 2024

```r
# extract data for plotting
dataframes <- lapply(contiguous_station_df$station_id, function(id) {
  extract_data(climate_df_temp, id, "t_daily_avg",
               starting_date = "2024-03-01",
               ending_date = "2024-03-31")
})

combined_data <- bind_rows(dataframes)

avg_temp <- combined_data %>%
  group_by(station_id, longitude, latitude) %>%
  summarise(avg_temp_2024_03 = mean(t_daily_avg, na.rm = TRUE))%>%
  ungroup()
#> `summarise()` has grouped output by 'station_id', 'longitude'. You can override
#> using the `.groups` argument.
```

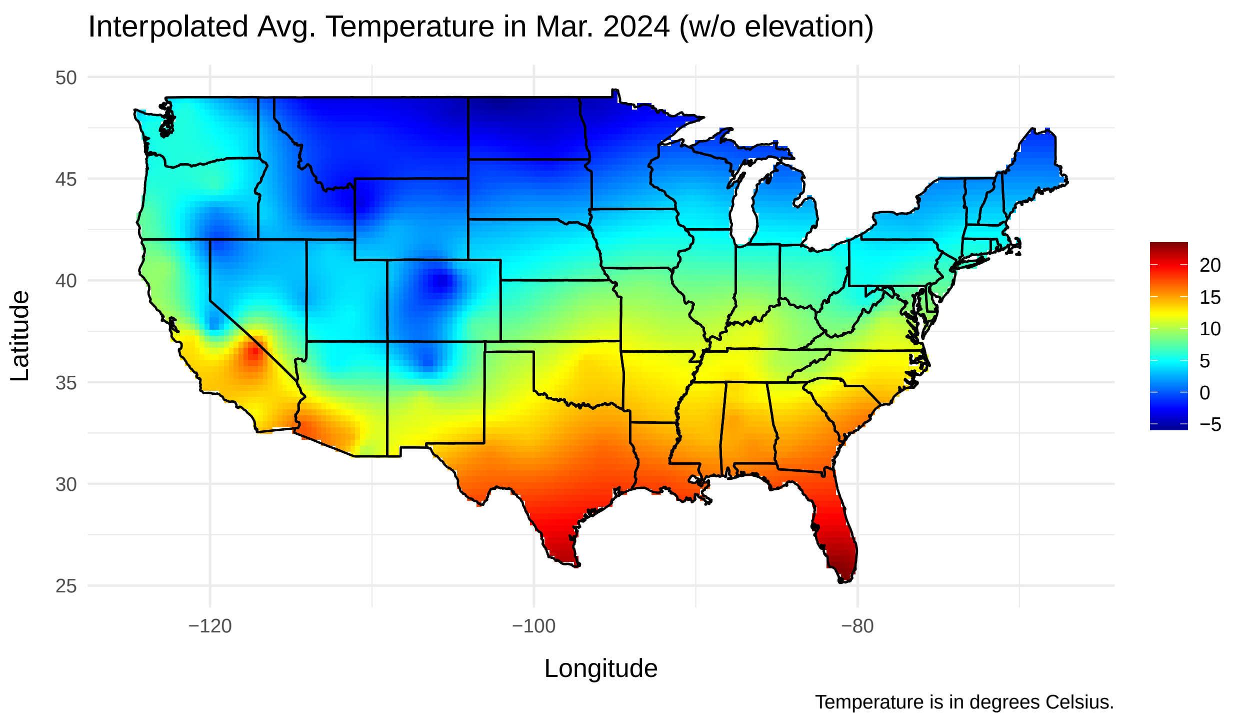
```r
ggplot() +
  geom_point(data = avg_temp,
                      aes(x = longitude, y = latitude,  color = avg_temp_2024_03),
                      size = 2)+
  geom_polygon(data = us_map,
                          aes(x = long, y = lat, group = group),
                          fill = NA,
                          color = "grey") +
  labs(title = "Average Temperature in March 2024 at Weather Stations",
           x = "Longitude", y = "Latitude", color = NULL,
           caption = "Temperature is in degrees Celsius.")  +
  theme_minimal() +
  scale_color_gradientn(colors = colors_vector) +
  coord_quickmap() +  # for aspect ratio adjustment
  theme(
      axis.title.x = element_text(margin = margin(t = 10)),
      axis.title.y = element_text(margin = margin(r = 10)),
      plot.title = element_text(margin = margin(b = 10))
  )
```


Average Temperature in March 2024 at Weather Stations
Latitude
Longitude
50
45
40
35
30
25
−120
−100
−80
20
15
10
5
0
−5
Temperature is in degrees Celsius.

Observations: The average daily temperature in March 2024 varies greatly across different weather stations, with the lowest being -6 degrees Celsius and the highest being 23.3 degrees Celcius. It is generally colder at stations at higher latitudes.

Q2: Interpolated Avg Temperature in March 2024

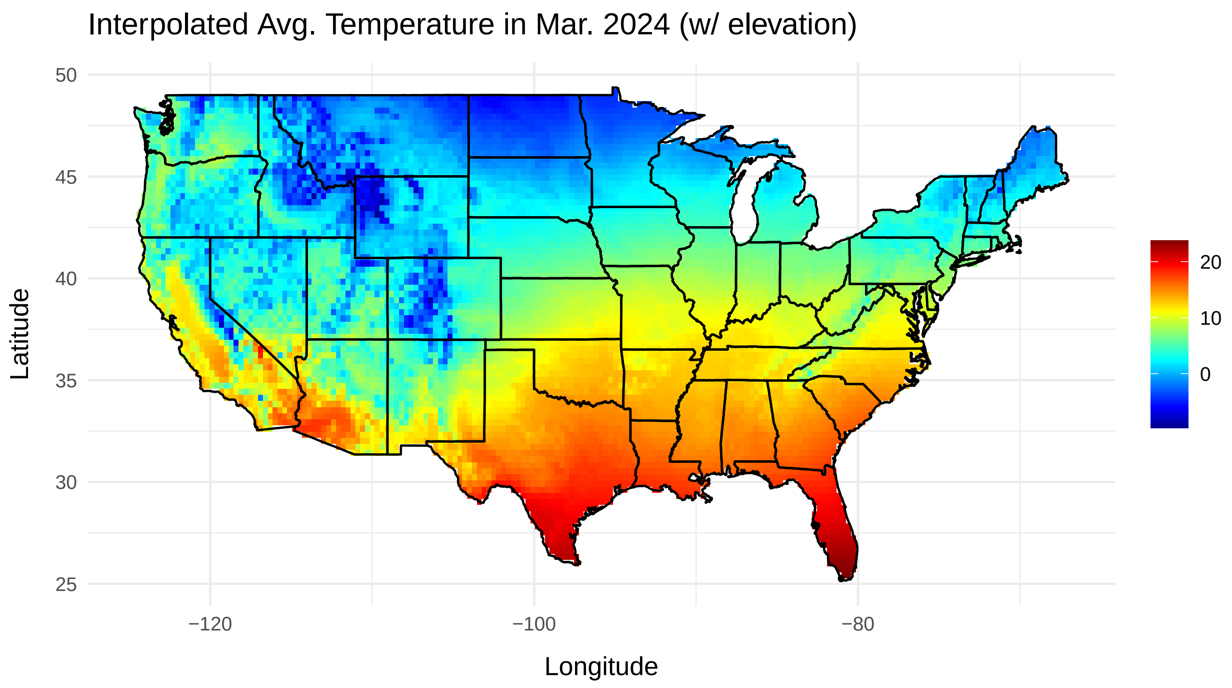
```
# Using just longitude and latitude as covariates

# we used dense grid for visualization purposes but this is more time-consuming
grid_points <- create_grid(0.3)
interpolations_2024_03 <- interpolate_data(avg_temp[, 2:4], grid_points)
#> Assuming columns 1 and 2 of locs are (longitude,latidue) in degrees
plot_interpolations(interpolations_2024_03,
                    title = "Interpolated Avg. Temperature in Mar. 2024 (w/o elevation)",
                    caption = "Temperature is in degrees Celsius.")
```



Interpolated Avg. Temperature in Mar. 2024 (w/o elevation)
50
45
40
35
30
25
Latitude
−120
−100
−80
Longitude
20
15
10
5
0
−5
Temperature is in degrees Celsius.

```r
# Using longitudes, latitudes, elevations as covariates
grid_points <- create_grid(0.3, elevation = T)
#> Mosaicing & Projecting
#> Note: Elevation units are in meters

locs <- data.frame(x = avg_temp$longitude, y = avg_temp$latitude)
elevation <- get_elev_point(locations = locs, prj = "+proj=longlat +datum=WGS84",
                                                src = "aws")
#> Mosaicing & Projecting
#> Note: Elevation units are in meters
avg_temp$elevation <- elevation$elevation
col_order <- c("station_id", "longitude", "latitude", "elevation", "avg_temp_2024_03")
avg_temp <- avg_temp[, col_order]
interpolations_2024_03 <- interpolate_data(avg_temp[, 2:5], grid_points)
#> Assuming columns 1 and 2 of locs are (longitude,latidue) in degrees
interpolations_2024_03$elevation <- NULL
```

```r
plot_interpolations(interpolations_2024_03,
                    title = "Interpolated Avg. Temperature in Mar. 2024 (w/ elevation)",
                    caption = "Temperature is in degrees Celsius.")
```



Temperature is in degrees Celsius.

Observations: The first plot looks very smooth because the only covariates are longitude and latitude, so grid points close to each other likely have similar values.
But after adding elevation as a covariate, we can see from the second plot that there is more variations in points close to each other, likely due to the differences in elevation.

Q3: Warmest and Coldest Day of Year

To estimate the warmest and coldest day of the year at each station, we used the `yearly_cycle` function to account for noise in average temperature across different years. The model used in `yearly_cycle` is a linear combination of first and second-order harmonics on the day of the year.

The equation for the harmonic fit is:

$$h(t) = A_0 + \sum_{k=1}^{M} [A_k \cos(\omega_k t) + B_k \sin(\omega_k t)]$$

where t is the day of the year from 1 to $N = 365$, $\omega_k = \frac{2\pi k}{N}$, M is the number of harmonics used, A_k and B_k are coefficients for the harmonic terms, A_0 is the intercept, and $h(t)$ is the expected average temperature at t. for Typically, $M = 2$ is the most suitable for daily temperature data, which was used here. Smaller M might be prone to underfitting, and larger M might result in overfitting. See this paper for details.

Assumptions for the model are:

- Linearity: The relationship between the predictor variables (harmonics - sine and cosine terms of different frequencies on t) and the response is linear. That is, the relationship between t and the response is curved in a way that can be captured by the harmonics.
- Independence: Observations should be independent of each other.

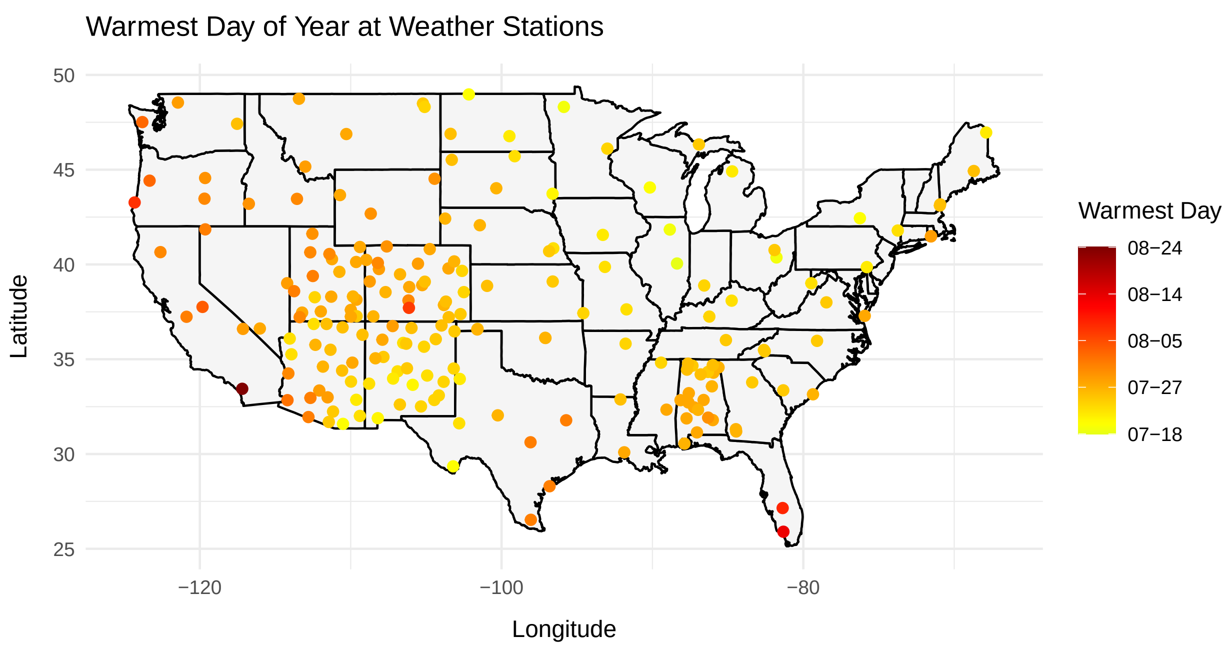
- Homoscedasticity: The variance of the errors (residuals) should be constant across all levels of the predictor variables.
- Normality of Errors: The errors (residuals) should be normally distributed.

We plotted daily average temperatures for some stations to check for the first assumption, which is reasonably satisfied. For the last three assumptions, they might not be well satisfied due to for some stations, but the model is still able to do a decent job in fitting the yearly cycle.

```r
# get yearly cycle at each station
init <- rep(NA, nrow(contiguous_station_df))
coldest_df <- data.frame(station_id = init, coldest_day = init)
warmest_df <- data.frame(station_id = init, warmest_day = init)

for (i in 1:nrow(contiguous_station_df)){
  id <- contiguous_station_df[i, 1]
  dat <- extract_data(climate_df_temp, id = id, var = "t_daily_avg")
  station_cycle <- yearly_cycle(dat[, 4:5], M = 2)
  coldest_df[i, 1] <- id
  coldest_df[i, 2] <- station_cycle[which.min(station_cycle[[2]]), 1]
  warmest_df[i, 1] <- id
  warmest_df[i, 2] <- station_cycle[which.max(station_cycle[[2]]), 1]
}

station_df_subset <- contiguous_station_df[, c(1, 2, 3)]
coldest_df <- merge(coldest_df, station_df_subset, by = "station_id", all.x = TRUE)
warmest_df <- merge(warmest_df, station_df_subset, by = "station_id", all.x = TRUE)
```

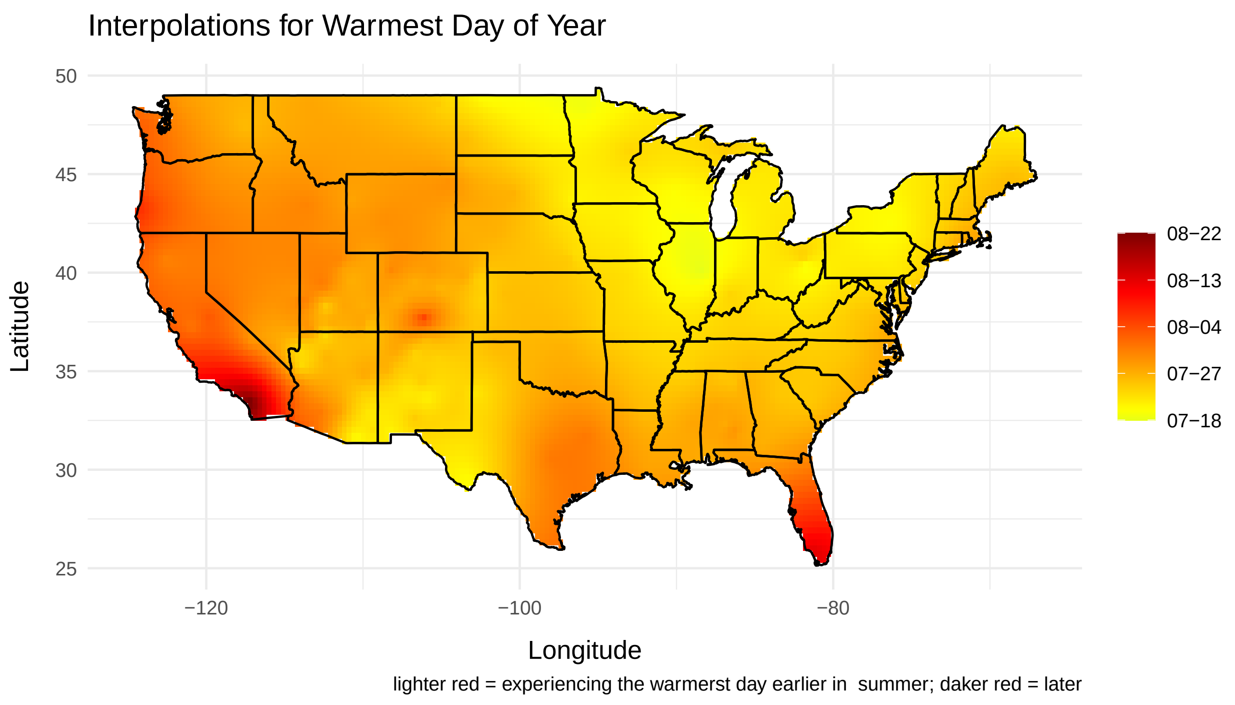
Instead of using day of year in plots, we use dates in the form mm-dd, which is more straightforward.

```r
breaks <- seq(min(warmest_df$warmest_day), max(warmest_df$warmest_day),
                       length.out = 5)
labels <- format(as.Date(breaks - 1, origin = "2021-01-01"), "%m-%d")

ggplot() +
  geom_polygon(data = us_map, aes(x = long, y = lat, group = group),
                         fill= "#f5f5f5", color = "black") +
  geom_point(data = warmest_df,
             aes(x = longitude, y = latitude,  color = warmest_day),
             size = 2)+
  labs(title = "Warmest Day of Year at Weather Stations",
       x = "Longitude", y = "Latitude",
       color = "Warmest Day",
       caption = "lighter red = experiencing the warmerst day earlier in summer; daker red = later")  +
   theme_minimal() +
   scale_color_gradientn(colors = colors_vector[60:100], breaks = breaks, labels = labels) +
   coord_quickmap() +
   theme(
      axis.title.x = element_text(margin = margin(t = 10)),
      axis.title.y = element_text(margin = margin(r = 10)),
      plot.title = element_text(margin = margin(b = 10)),
      legend.title = element_text(margin = margin(b = 12))
    )
```



lighter red = experiencing the warmerst day earlier in summer; daker red = later

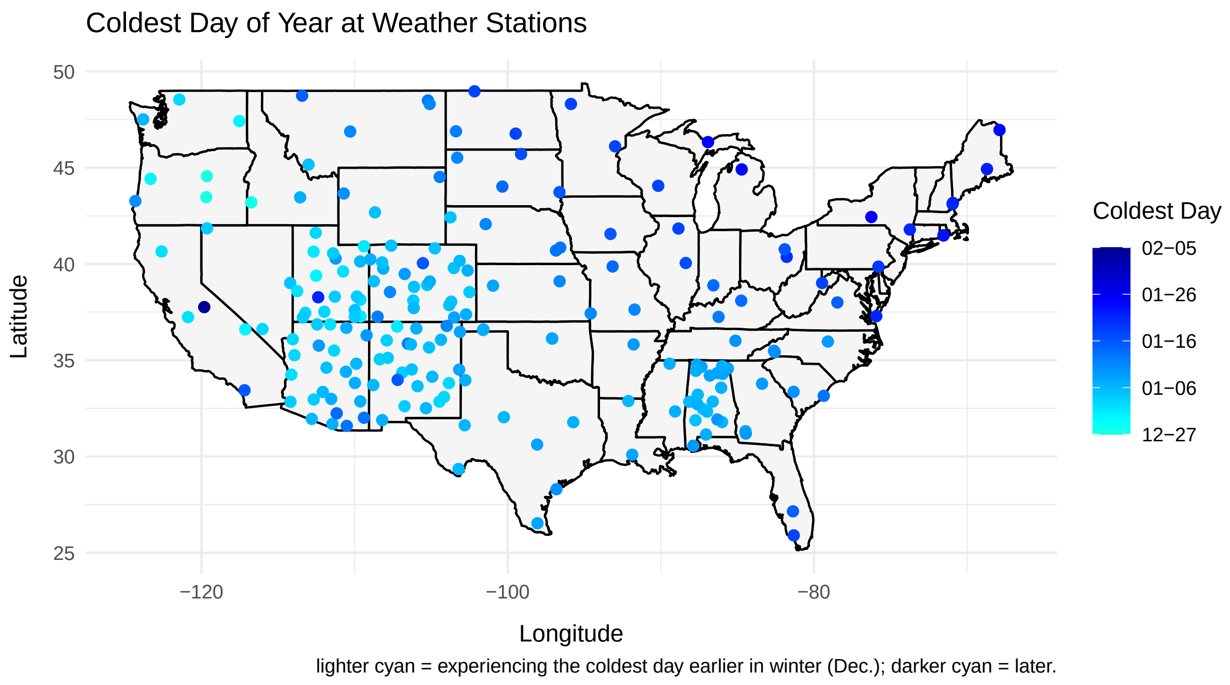
```
grid_points <- create_grid(0.3)
interpolations <- interpolate_data(warmest_df[, c(3,4,2)], grid_points)
#> Assuming columns 1 and 2 of locs are (longitude,latidue) in degrees
breaks <- seq(min(interpolations$warmest_day), max(interpolations$warmest_day),
                       length.out = 5)
labels <- format(as.Date(breaks - 1, origin = "2021-01-01"), "%m-%d")
plot_interpolations(interpolations, title ="Interpolations for Warmest Day of Year",
                              vibe = "hot", labels = labels, breaks = breaks,
        caption = "lighter red = experiencing the warmerst day earlier in  summer; daker red = later")
```


Interpolations for Warmest Day of Year
50
45
40
35
30
25
Latitude
−120
−100
−80
Longitude
08−22
08−13
08−04
07−27
07−18
lighter red = experiencing the warmerst day earlier in summer; daker red = later

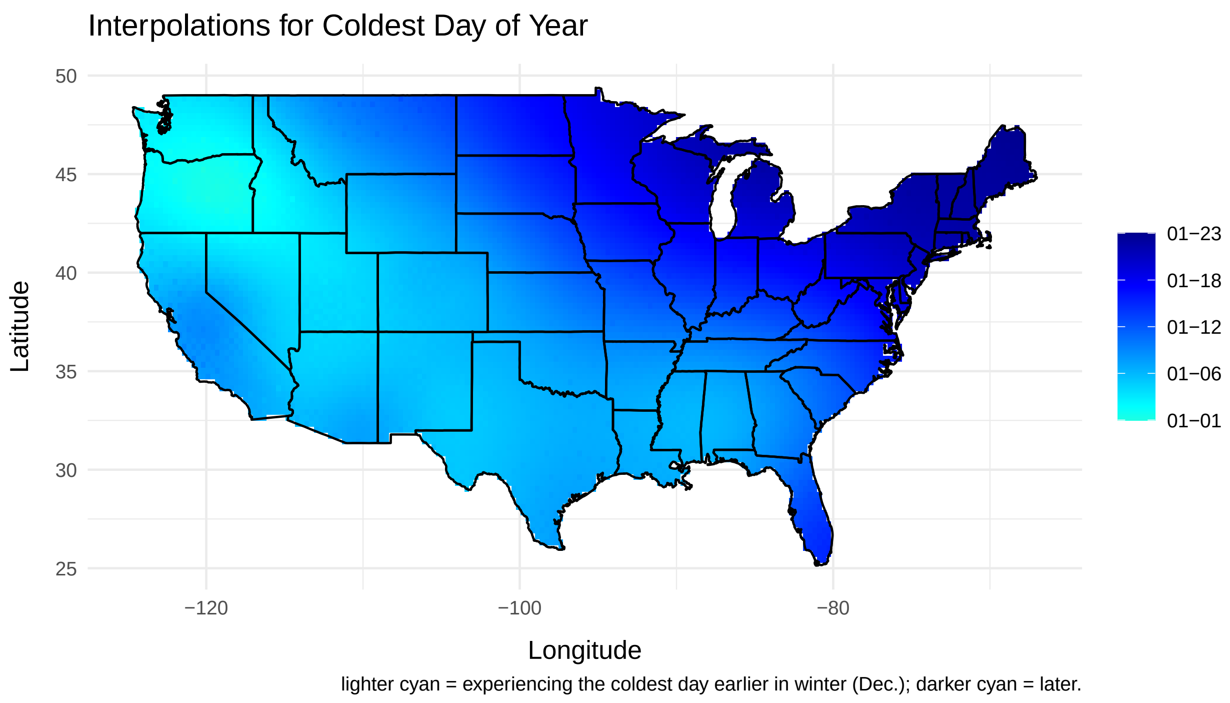
For coldest day of year, we used an adjusted version of day of year to plot and do interpolations. Since at some stations, the coldest days of year are in Dec. (i.e., day of year ~= 360). Logically, these stations experience colder days earlier in winter and should have lighter colors. So we adjusted the values.

```
coldest_df$adjusted_day <- coldest_df$coldest_day + 365*(coldest_df$coldest_day < 350)
breaks <- seq(min(coldest_df$adjusted_day), max(coldest_df$adjusted_day),
                       length.out = 5)
labels <- breaks - 365*(breaks > 365)
labels <- format(as.Date(labels - 1, origin = "2021-01-01"), "%m-%d")

ggplot() +
  geom_polygon(data = us_map,
               aes(x = long, y = lat, group = group), fill=  "#f5f5f5", color = "black") +
  geom_point(data = coldest_df,
             aes(x = longitude, y = latitude,  color = adjusted_day),
             size = 2)+
  labs(title = "Coldest Day of Year at Weather Stations",
       x = "Longitude", y = "Latitude",
       color = "Coldest Day",
caption = "lighter cyan = experiencing the coldest day earlier in winter (Dec.); darker cyan = later.")
  theme_minimal() +
  scale_color_gradientn(colors = rev(colors_vector[0:40]),
                                      breaks = breaks, labels = labels) +
  coord_quickmap() +
  theme(
      axis.title.x = element_text(margin = margin(t = 10)),
      axis.title.y = element_text(margin = margin(r = 10)),
      plot.title = element_text(margin = margin(b = 10)),
      legend.title = element_text(margin = margin(b = 12))
```

```
    )
```


Coldest Day of Year at Weather Stations
Latitude
Longitude
50
45
40
35
30
25
−120
−100
−80
Coldest Day
02−05
01−26
01−16
01−06
12−27
lighter cyan = experiencing the coldest day earlier in winter (Dec.); darker cyan = later.

```
# interpolate and plot
interpolations <- interpolate_data(coldest_df[, 3:5], grid_points)
#> Assuming columns 1 and 2 of locs are (longitude,latidue) in degrees
breaks <- seq(min(interpolations$adjusted_d), max(interpolations$adjusted_day),
              length.out = 5)
labels <- breaks - 365*(breaks > 365)
labels <- format(as.Date(labels - 1, origin = "2021-01-01"), "%m-%d")
plot_interpolations(interpolations, "Interpolations for Coldest Day of Year",
                    vibe = "cold", breaks = breaks, labels = labels,
caption = "lighter cyan = experiencing the coldest day earlier in winter (Dec.); darker cyan = later.")
```



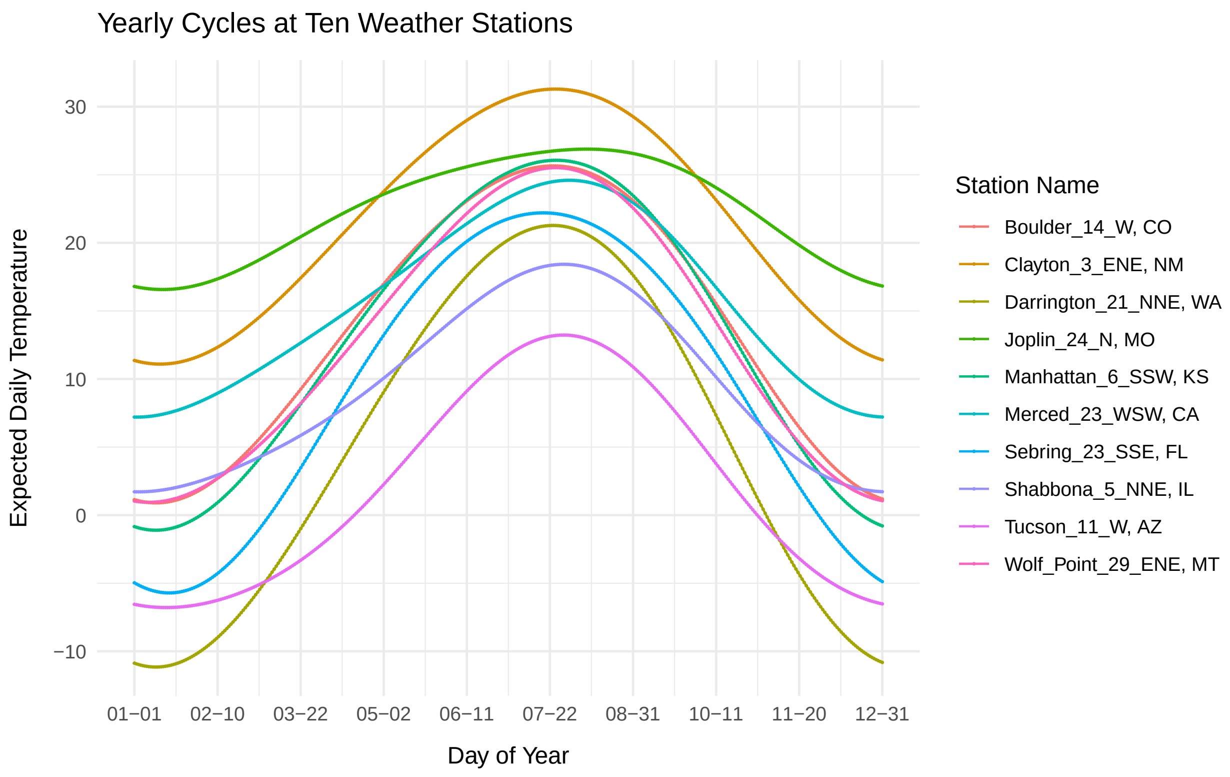
Interpolations for Coldest Day of Year
Latitude
50
45
40
35
30
25
−120
−100
−80
Longitude
01−23
01−18
01−12
01−06
01−01
lighter cyan = experiencing the coldest day earlier in winter (Dec.); darker cyan = later.

It's interesting that the interpolated data doesn't contain any days in December when the adjusted day of year was used to interpolate.

Q4: Yearly Cycles at 10 Stations

```r
# get yearly cycles at 10 stations within contiguous US
stations <- c("53131", "94060", "92827", "94075", "53974", "04223", "54811", "93243",
              "03094", "23908")
dataframes <- lapply(stations, function(id) {
  extract_data(climate_df_temp, id, "t_daily_avg")
})
cycles <- setNames(lapply(dataframes, function(dat) {
  yearly_cycle(dat[, 4:5])
}), stations)

combined_cycles <- bind_rows(cycles, .id = "station_id")

breaks <- seq(min(combined_cycles$day_of_year, na.rm = T),
                       max(combined_cycles$day_of_year, na.rm = T), length.out = 10)
labels <- format(as.Date(breaks - 1, origin = "2021-01-01"), "%m-%d")
station_info <- merge(combined_cycles, contiguous_station_df[, c(1,5,4)],
                                        by = "station_id", all.x = TRUE)[, 4:5]
combined_cycles$station_name <- paste0(station_info[[1]], ", ", station_info[[2]])
```

```r
ggplot(combined_cycles, aes(x = day_of_year, y = t_expected, color = station_name)) +
  geom_point(size = 0.1) +
  geom_line() +
  theme_minimal() +
  labs(title = "Yearly Cycles at Ten Weather Stations",
       x = "Day of Year",
       y = "Expected Daily Temperature",
       color = "Station Name") +
   scale_x_continuous(breaks = breaks, labels = labels) +
   theme(
      axis.title.x = element_text(margin = margin(t = 10)),
      axis.title.y = element_text(margin = margin(r = 10)),
      plot.title = element_text(margin = margin(b = 10))
    )
```



We selected weather stations in 10 different states, and from the plot we can see that they all follow a similar cycle, with the highest daily temperature occurring in around June and July. The yearly cycle of Joplin, MO is relatively flat compared to that of other locations we chose. We also observed that the range of temperatures vary greatly. For example, the highest daily temperature in Tucson, AZ is even lower than the lowest daily temperature in Joplin, MO.

Q5: Temperature Trend and Interpolation

To estimate temperature trends over years, we fitted a linear model of daily average temperature versus year and harmonics on day of year (first and second order) for each station.

The equation for the harmonic fit is:

$$h(t) = A_0 + Cy + \sum_{k=1}^{M} [A_k \cos(\omega_k t) + B_k \sin(\omega_k t)]$$

where t is the day of the year from 1 to $N = 365$, y is year, $\omega_k = \frac{2\pi k}{N}$, M is the number of harmonics used, C is the coefficient for year (trend), A_k and B_k are coefficients for the harmonic terms, A_0 is the intercept, and $h(t)$ is the expected average temperature given t and y. Here we used M = 2.

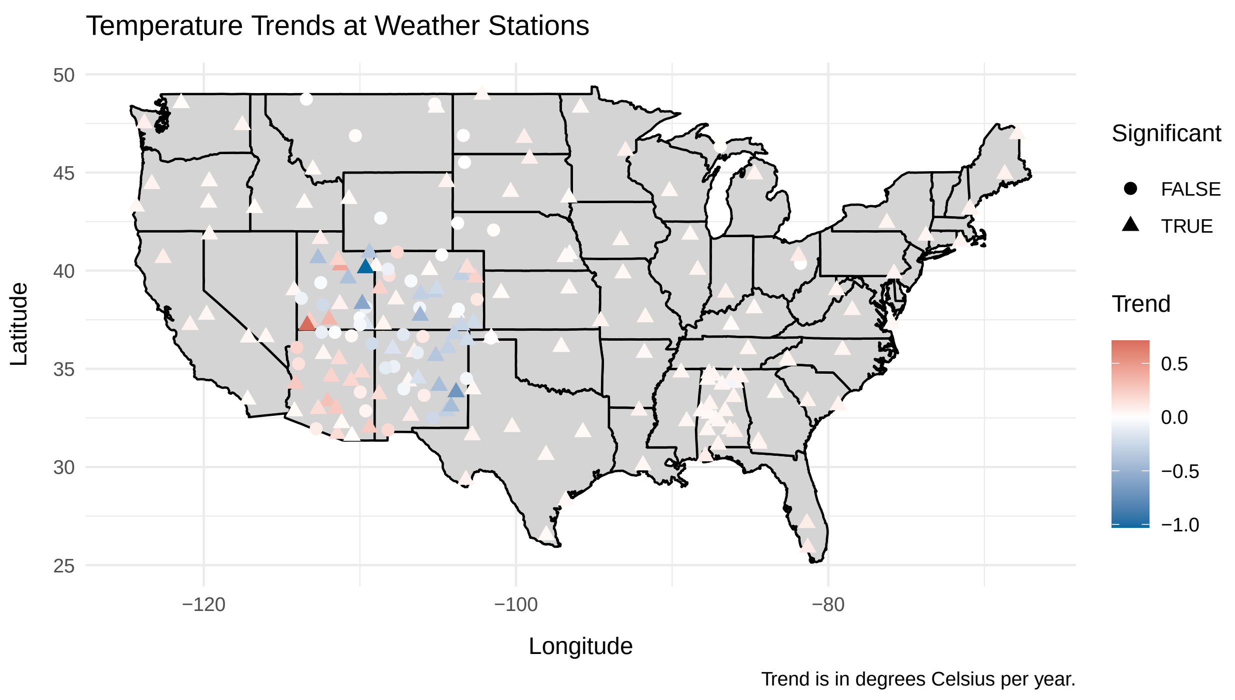
Assumptions for the model are:

- Linearity: The relationship between year and the response is linear when holding day of year constant. Also, there is an annual cycle within each year that can be captured by harmonics on day of year.
- Independence: Observations should be independent of each other.
- Homoscedasticity: The variance of the errors (residuals) should be constant across all levels of the predictor variables.
- Normality of Errors: The errors (residuals) should be normally distributed.

Generally the model provides a good fit even though the assumptions might not be well satisfied for some stations.

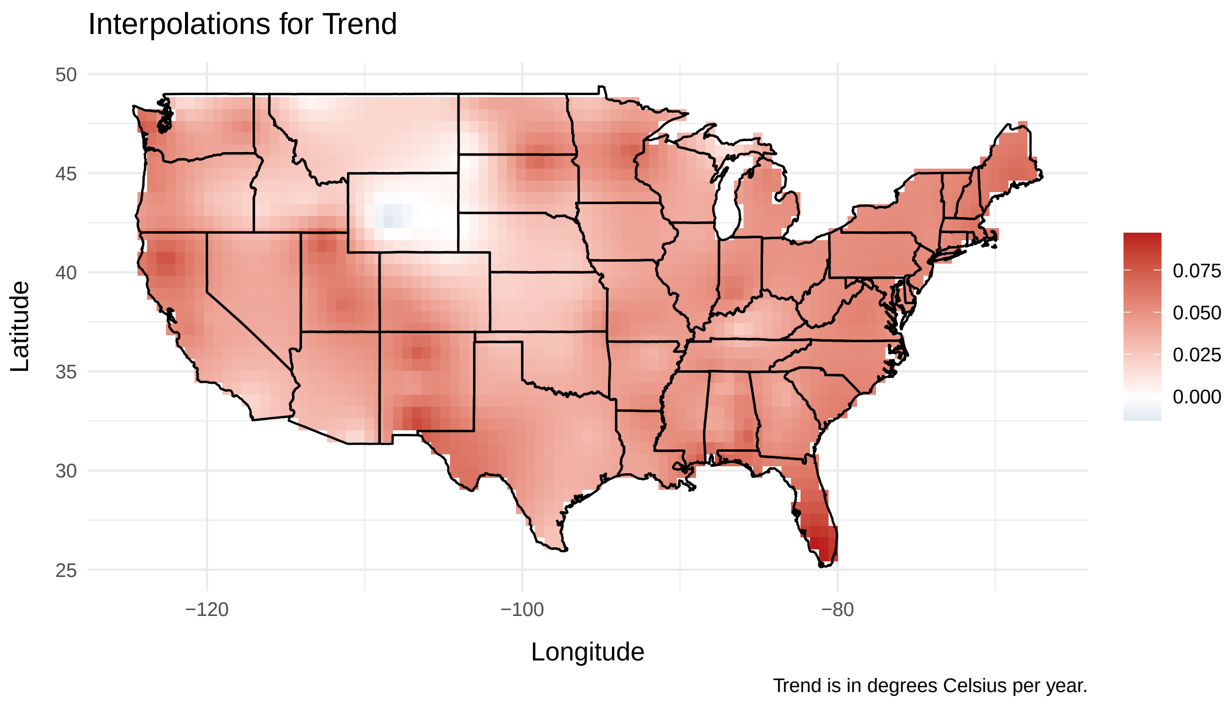
```r
# get data
init <- rep(NA, nrow(contiguous_station_df))
trend_df <- data.frame(station_id = init, trend = init, se = init, p_value = init)

for (i in 1:nrow(contiguous_station_df)){
  id <- contiguous_station_df[i, 1]
  dat <- extract_data(climate_df_temp, id = id, var = "t_daily_avg")
  stats <- overall_trend(dat[, 4:5], M = 2)
  trend_df[i, 1] <- id
  trend_df[i, 2] <- stats$trend
  trend_df[i, 3] <- stats$se
  trend_df[i, 4] <- stats$p_value
}

station_df_subset <- contiguous_station_df[, c(1, 2, 3)]
trend_df <- merge(trend_df, station_df_subset, by = "station_id", all.x = TRUE)
# use alpha = 0.1
trend_df$significant <- trend_df$p_value < 0.1
```

```r
ggplot() +
  geom_polygon(data = us_map,
               aes(x = long, y = lat, group = group), fill = "#d4d4d4", color = "black") +
  geom_point(data = trend_df,
             aes(x = longitude, y = latitude,  color = trend, shape = significant),
             size = 2.5)+
  labs(title = "Temperature Trends at Weather Stations",
       x = "Longitude", y = "Latitude",
       color = "Trend", shape = "Significant",
       caption = "Trend is in degrees Celsius per year.") +
  theme_minimal() +
   scale_color_gradient2(midpoint = 0, mid = "white", low = "#0369a1" ,high = "#b91c1c")+
  coord_quickmap() +
  theme(
    axis.title.x = element_text(margin = margin(t = 10)),
    axis.title.y = element_text(margin = margin(r = 10)),
    plot.title = element_text(margin = margin(b = 10)),
    legend.title = element_text(margin = margin(b = 12))
   )
```



```
# interpolate
grid_points <- create_grid(0.6)
# remove trends with high standard error (keep the first 60% quantile)
trend_df_se <- trend_df[trend_df$se <= quantile(trend_df$se, 0.6), ]
interpolations <- interpolate_data(trend_df_se[, c(5, 6, 2)], grid_points)
#> Assuming columns 1 and 2 of locs are (longitude,latidue) in degrees
```

```
plot_interpolations(interpolations, title = "Interpolations for Trend", vibe = "split_zero",
                    caption = "Trend is in degrees Celsius per year.")
```



Q6: Comparison between Sources

We found this plot from the U.S. Environmental Protection Agency (EPA) at https://www.epa.gov/climate-indicators/climate-change-indicators-us-and-global-temperature. See next page.

This plot is not perfectly comparable to our plot because (1) It showed the rate of change from 1901 to 2021; (2) Rate of change is in degrees Fahrenheit per century. But we thought this is still a valuable source since it shows changes in each relatively small region, similar to the grid in our interpolation plot.

Our interpolation shows similar trends as their plot except for the region around Wyoming and around Alabama. For Wyoming, their plot shows an increasing trend, but ours is a decreasing trend. Possible reason is that there are only about 3 weather stations in Wyoming, and maybe available data only covers limited time flame, causing bias. For Alabama, our interpolation indicates no trend or increasing trend, opposite to their plot. Notice that there are many weather stations around Alabama, so we should be confident in the interpolations, but the difference is possibly due to the difference in time frame covered.

In terms of magnitude of change, a 2 degrees F change per century is equivalent to a 0.11 degree C change per year. So our plot is underestimating the increasing trend at most regions compared to theirs. However, their plot covered one more century than ours, and humans did cause huge global warming in the century of 1900, which might explain the overestimation. Overall, both sources indicate that global warming is HAPPENING; we need to take efforts to reduce emissions!

Figure 3. Rate of Temperature Change in the United States, 1901–2021

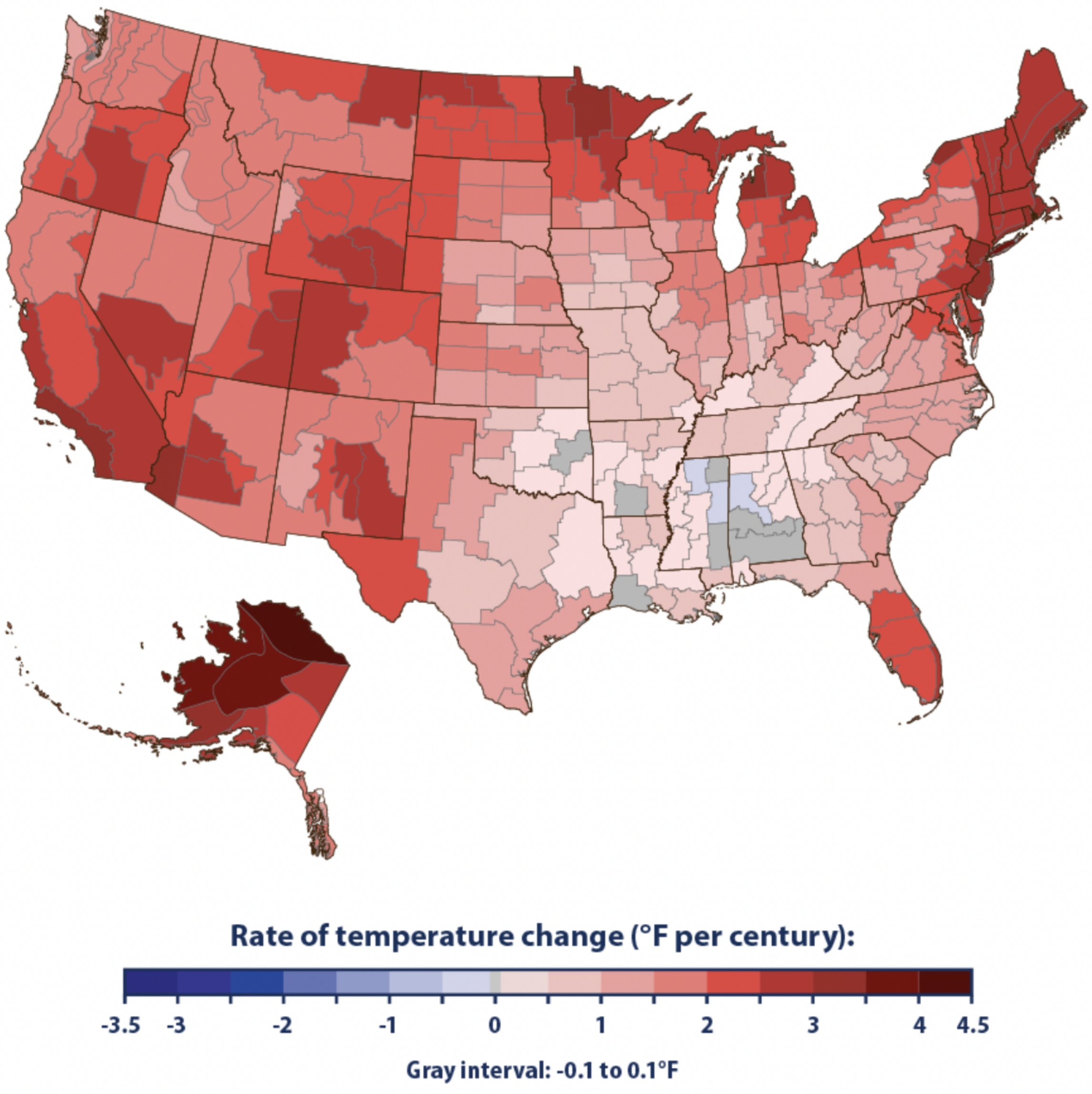

Rate of temperature change (°F per century):
-3.5
-3
-2
-1
0
1
2
3
4
4.5
Gray interval: -0.1 to 0.1°F

Figure 1: Plot by EPA