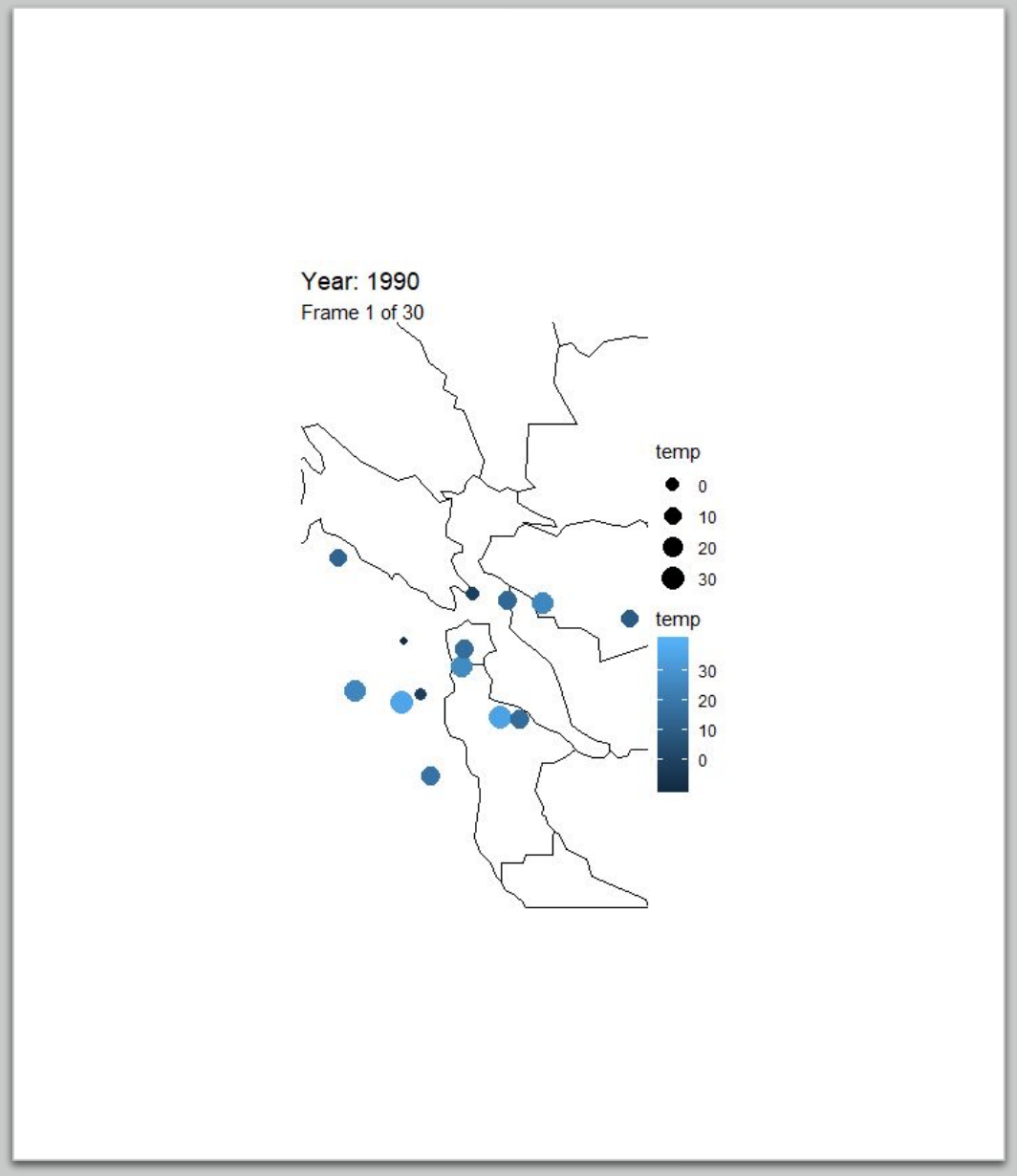
Lab 7: Plotting Data on a Map

- Learn to use ggplot2 and gganimate packages to plot data on simple maps.



Data Format

- (x, y, t, z1, z2, …)
- x = longitude
- y = latitude
- t = optional time parameter
- z1, z2, … = optional values of some interesting characteristics of the data point

Example Test Data

```
 head(test_data)
      lat         long       year     temp
1  37.92775  -122.0816  2007   2.367641
2  37.73751  -122.5401  2017  28.682083
3  37.77283  -122.5401  2000  25.073657
4  37.18677  -122.3946  1999  -6.554683
5  37.93086  -121.9531  2017  27.434501
6  37.89064  -122.2742  2014  -2.052456
```

Basic Steps

- Generate base map
  - Use existing database of maps
  - Focus on region of interest
- Plot a dot on the map for each data point
  - Location specified by (x,y) coordinates
  - Optionally, color and size controlled by (z1, z2, …)
- Use time parameter t to show change vs time
- *Philosophy – add new parts to the base map as a series of smaller steps to layer on desired information*

Load the Packages Needed

- library(ggplot2)
  - Extensive package for making many different types of plots
- library(gganimate)
  - Package to provide simple animation
- library(gifski)
  - Package to generate GIFs

Get the Map Information

- From the “maps” package
  - Shapes of counties in U.S. states specified using (long,lat) of corners
- which_state <- "california"
- county_info <- map_data("county", region=which_state)

	long	lat	group	order	region	subregion
1	-121.4785	37.48290	1	1	california	alameda
2	-121.5129	37.48290	1	2	california	alameda
3	-121.8853	37.48290	1	3	california	alameda
4	-121.8968	37.46571	1	4	california	alameda
5	-121.9254	37.45998	1	5	california	alameda
6	-121.9483	37.47717	1	6	california	alameda

Create the Base Map

```
base_map <- ggplot(
    data = county_info,
    mapping = aes(x = long, y = lat, group = group)) +
    geom_polygon(color = "black", fill = "white") +
    coord_quickmap() +
    theme_void()
```

Create the Base Map

```
base_map <- ggplot(
    data = county_info,
    mapping = aes(x = long, y = lat, group = group)) +
    geom_polygon(color = "black", fill = "white") +
    coord_quickmap() +
    theme_void()
```

Tell ggplot which data frame to use

Create the Base Map

```
base_map <- ggplot(
    data = county_info,
    mapping = aes(x = long, y = lat, group = group)) +
    geom_polygon(color = "black", fill = "white") +
    coord_quickmap() +
    theme_void()
```

How to use the specific data in county_info

Create the Base Map

```
base_map <- ggplot(
    data = county_info,
    mapping = aes(x = long, y = lat, group = group)) +
    geom_polygon(color = "black", fill = "white") +
    coord_quickmap() +
    theme_void()
```

Specify the line color (black) and the color to fill the polygons (white)

Create the Base Map

```
base_map <- ggplot(
    data = county_info,
    mapping = aes(x = long, y = lat, group = group)) +
    geom_polygon(color = "black", fill = "white") +
    coord_quickmap() +
    theme_void()
```

Make the map shape look good

Create the Base Map

```
base_map <- ggplot(
    data = county_info,
    mapping = aes(x = long, y = lat, group = group)) +
    geom_polygon(color = "black", fill = "white") +
    coord_quickmap() +
    theme_void()
```

Set the background to nothing (white)

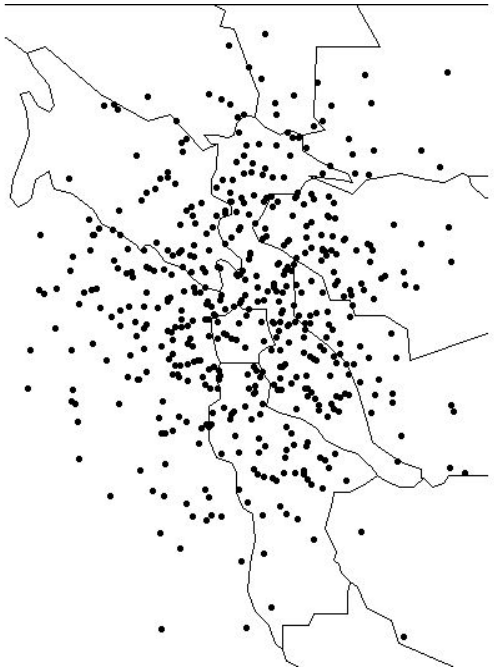
Type “base_map” at R prompt

Add Data Points to the Map

- map_with_data <- base_map +
  geom_point(data = test_data,
  aes(x = long,y = lat, group=year))
- map_with_data

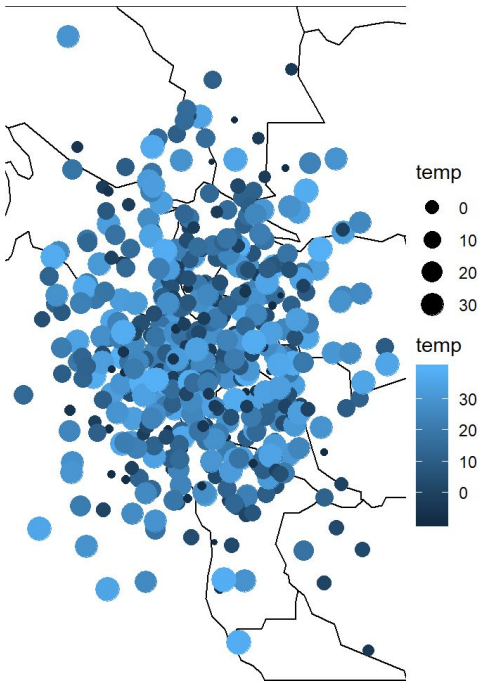
Zoom in to the Area with Data

- min_long <- min(test_data$long)
- max_long <- max(test_data$long)
- min_lat <- min(test_data$lat)
- max_lat <- max(test_data$lat)
- map_with_data <- map_with_data +
  coord_quickmap(xlim = c(min_long,
  max_long), ylim = c(min_lat, max_lat))
- map_with_data



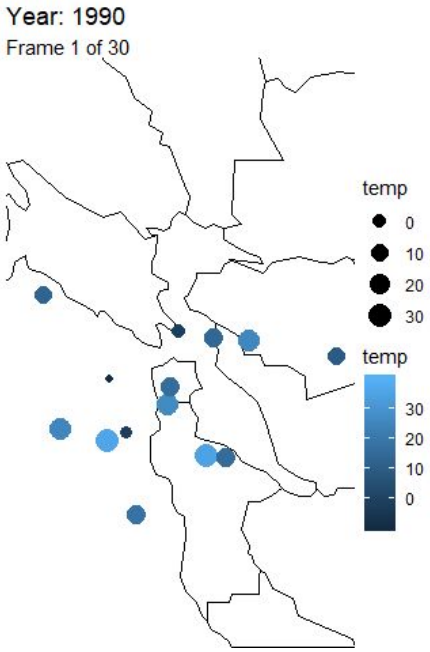
Set Color and Size of Data Points

```
map_with_data <- base_map + geom_point(data = test_data,
      aes(x = long, y = lat, color=temp, size=temp, group=year)) +
coord_quickmap(xlim = c(min_long, max_long),
      ylim = c(min_lat, max_lat))
map_with_data
```



Add Motion

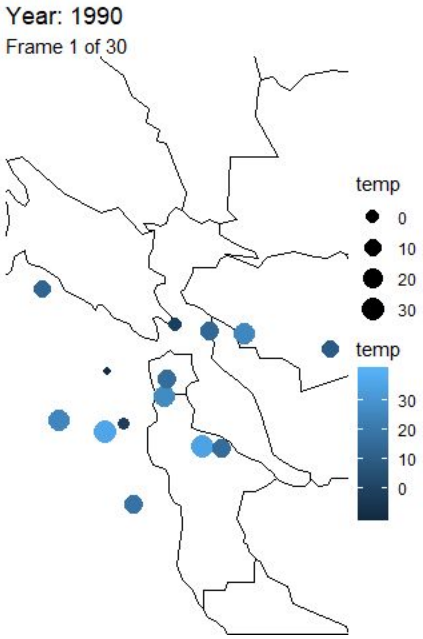
```
map_with_animation <- map_with_data +
      transition_time(year) +
      ggtitle('Year: {frame_time}',
      subtitle = 'Frame {frame} of {nframes}')
num_years < max(test_data$year) - min(test_data$year) + 1
animate(map_with_animation, nframes = num_years)
```



Keep the Old Data Points

map_with_shadow <- map_with_animation + shadow_mark()

animate(map_with_shadow, nframes = num_years)



Lab 7: Plot House Price Predictions

- Divide data into training and testing sets
- Training set
  - All inputs plus price
- Testing set
  - Predict the price using the model you developed with the training set
- Compute error between your predicted price and the actual priced provided in data set.
  - Use “percent error” this time

Add Your Predictions and Location

```
house_data <- read.csv("house_data.csv")
data_by_zipcode <- house_data %>%
  group_by(zipcode) %>%
  summarize(
    count = n(),
    med_price = median(price),
    mean_lat = mean(lat),
    mean_long = mean(long),
    med_yr_built = median(yr_built),
    percent_error = price_prediction_error(price, bedrooms,
sqft_living, …..)
)
```

head(data_by_zipcode)

Zipcode	count	med_price	mean_lat	mean_long	med_yr_built	Percent_error
98001	362	260000.0	47.30902	-122.2706	1981	38.2
98002	199	235000.0	47.30878	-122.2134	1966	96.7
98003	280	267475.0	47.31574	-122.3101	1975	124.4
98004	317	1150000.0	47.61618	-122.2052	1965	….
98005	168	765475.0	47.61153	-122.1673	1967	….
98006	498	760184.5	47.55802	-122.1468	1978	….

Now plot your prediction errors on an appropriate map.

To Do

- Download and complete Lab 7
- See the tutorial for detailed examples
  - z.umn.edu/mapsUsingR