

Stat405

Advanced layering

Hadley Wickham

1. Motivation
2. Geom review
3. Symmetrical diamonds?
4. More maps
5. Napoleon's march

# Motivation

# Layering

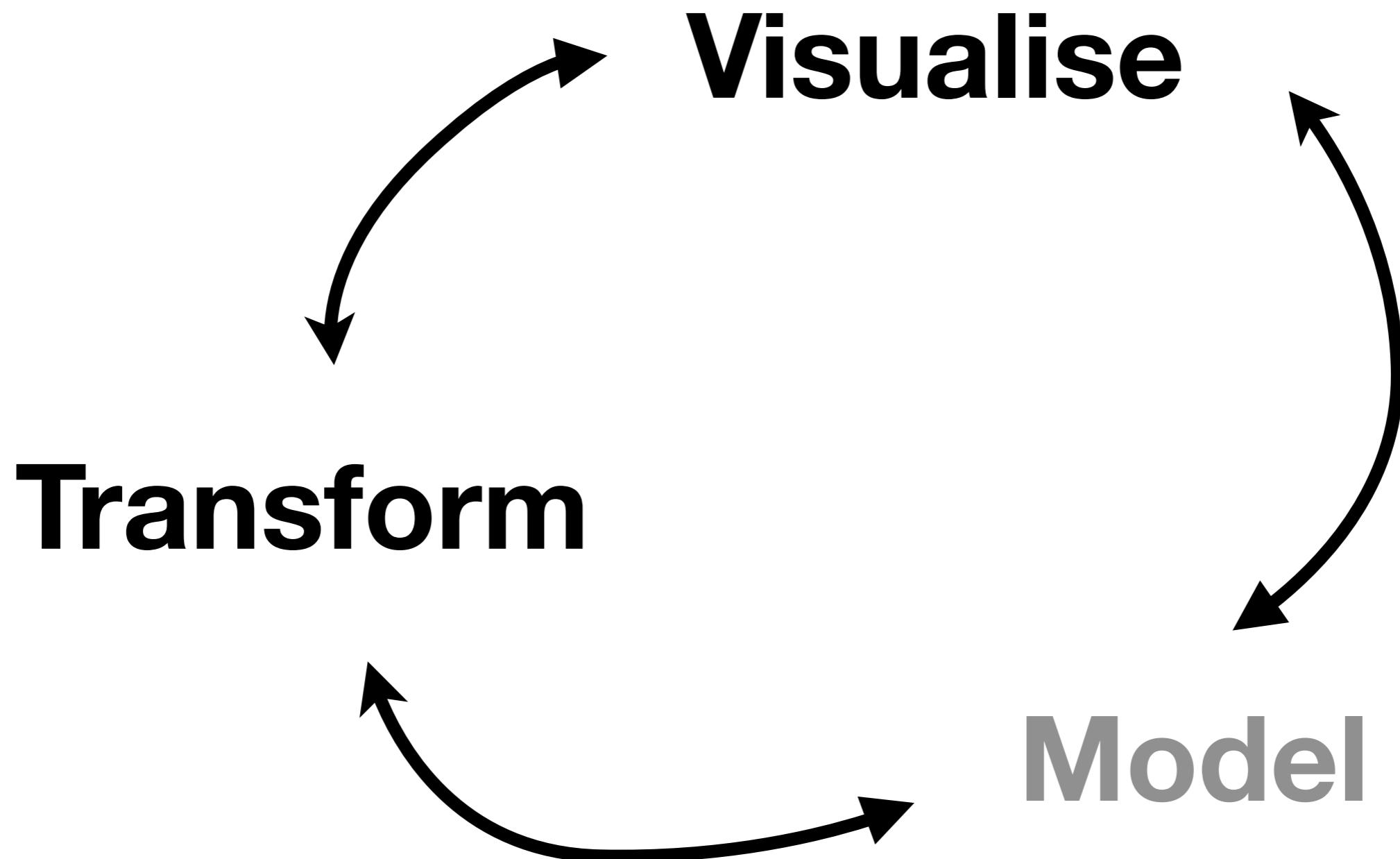
Key to rich graphics is taking advantage of layering.

Three types of layers: context, raw data, and summarised data

Each can come from a different dataset.

# Iteration

- First plot is never the best. Have to keep iterating to understand what's going on.
- Don't try and do too much in one plot.
- Best data analyses tell a story, with a natural flow from beginning to end.



# **Geom** **review**

# Lines

- `geom_path`: join points with a “line”
- `geom_line`: “functional” line, draws ordered by lowest to highest x
- `geom_segment`: line segments specified by start and end locations
- `geom_vline`, `geom_hline`, `geom_abline`: lines that span the whole plot

# Variability

- `geom_errorbar`, `geom_crossbar`,  
`geom_pointrange`: show lower, upper  
and mid point of y for fixed x
- `geom_linerange`: lower and upper
- `geom_ribbon`: continuous x

# Distribution

- `geom_boxplot`: show summary of y for fixed x
- `geom_dotplot`: show individual points, adjusted so they don't overlap
- `geom_violin`: sideways density plots

# Rectangles

- geom\_bar: base always on x axis
- geom\_rect: arbitrary rectangles  
(geom\_bar is a special case)
- geom\_tile/geom\_raster: tile the plane  
with equally sized rectangles

# Symmetric diamonds?

```
qplot(x, y, data = diamonds)
diamonds$x[diamonds$x == 0] <- NA
diamonds$x[diamonds$x > 10] <- NA
diamonds$y[diamonds$y == 0] <- NA
diamonds$y[diamonds$y > 10] <- NA
qplot(x, y, data = diamonds)

diamonds <- mutate(diamonds,
  area = x * y,
  lratio = log10(x / y))

qplot(area, lratio, data = diamonds)
diamonds$lratio[abs(diamonds$lratio) > 0.02] <- NA
```

```
ggplot(diamonds, aes(area, lratio)) +  
  geom_point()
```

```
ggplot(diamonds, aes(area, lratio)) +  
  geom_hline(yintercept = 0, size = 2, colour = "white") +  
  geom_point() +  
  geom_smooth(method = lm, se = F, size = 2)
```

```
ggplot(diamonds, aes(pr, abs(lratio))) +  
  geom_hline(yintercept = 0, size = 2, colour = "white") +  
  geom_point() +  
  geom_smooth(se = F, size = 2)
```

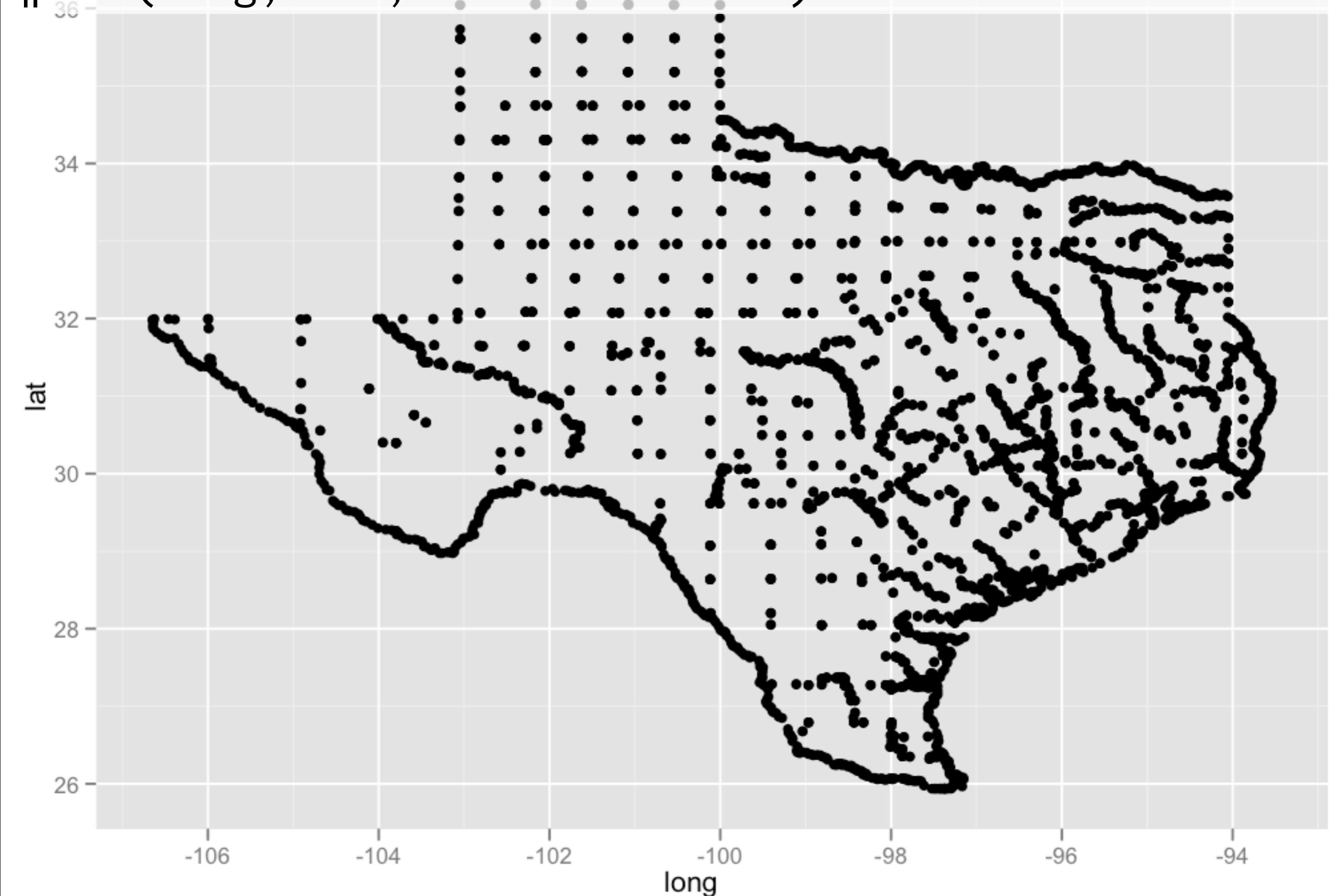
```
ggplot(diamonds, aes(area, abs(lratio))) +  
  geom_hline(yintercept = 0, size = 2, colour = "white") +  
  geom_boxplot(aes(group = round_any(area, 5))) +  
  geom_smooth(se = F, size = 2)
```

```
ggplot(diamonds, aes(area, abs(lratio))) +  
  geom_hline(yintercept = 0, size = 2, colour = "white") +  
  geom_boxplot(aes(group = round_any(area, 5)))
```

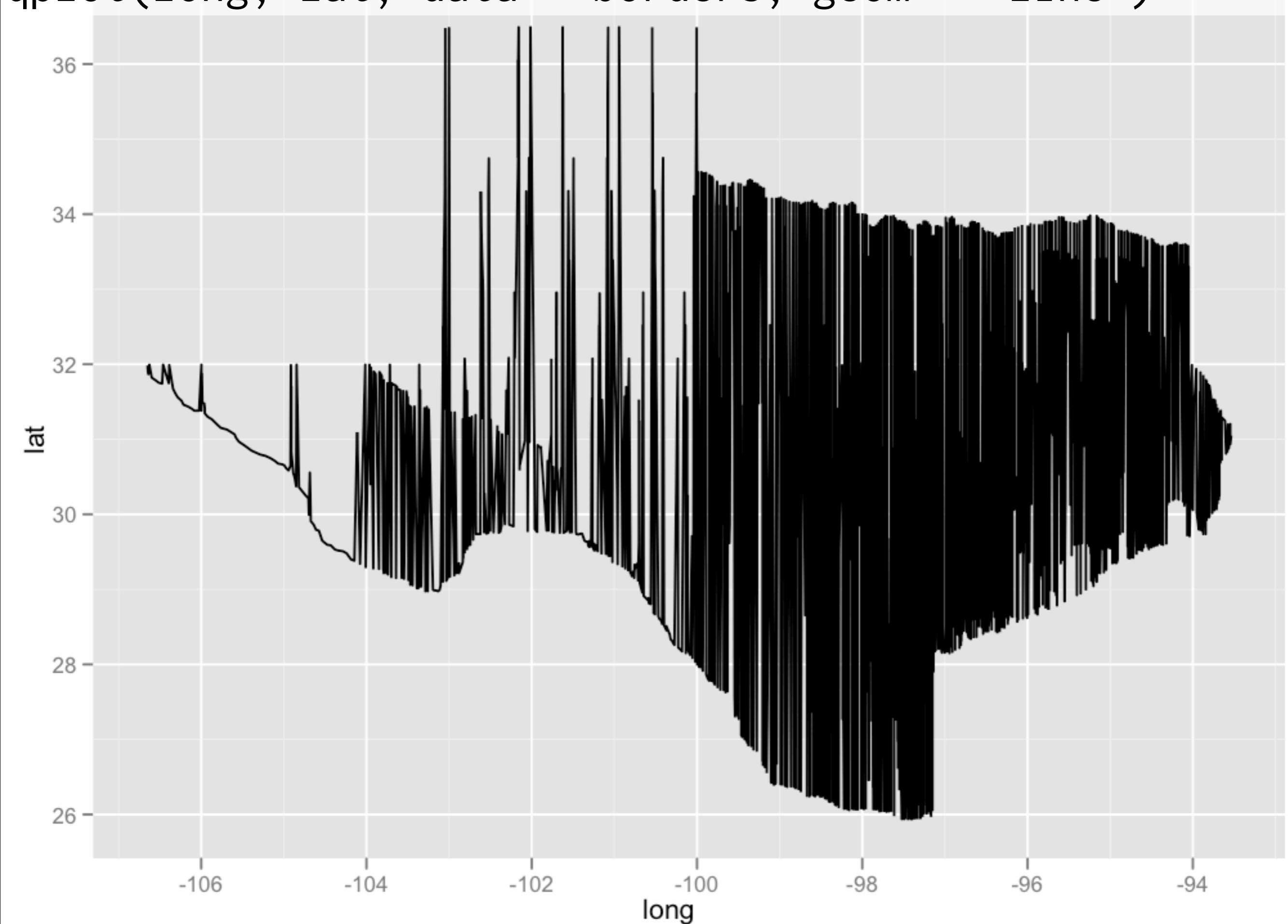
```
ggplot(diamonds, aes(area, lratio)) +  
  geom_hline(yintercept = 0, size = 2, colour = "white") +  
  geom_boxplot(aes(group = interaction(sign(lratio),  
    round_any(area, 5))), position = "identity")
```

# More maps

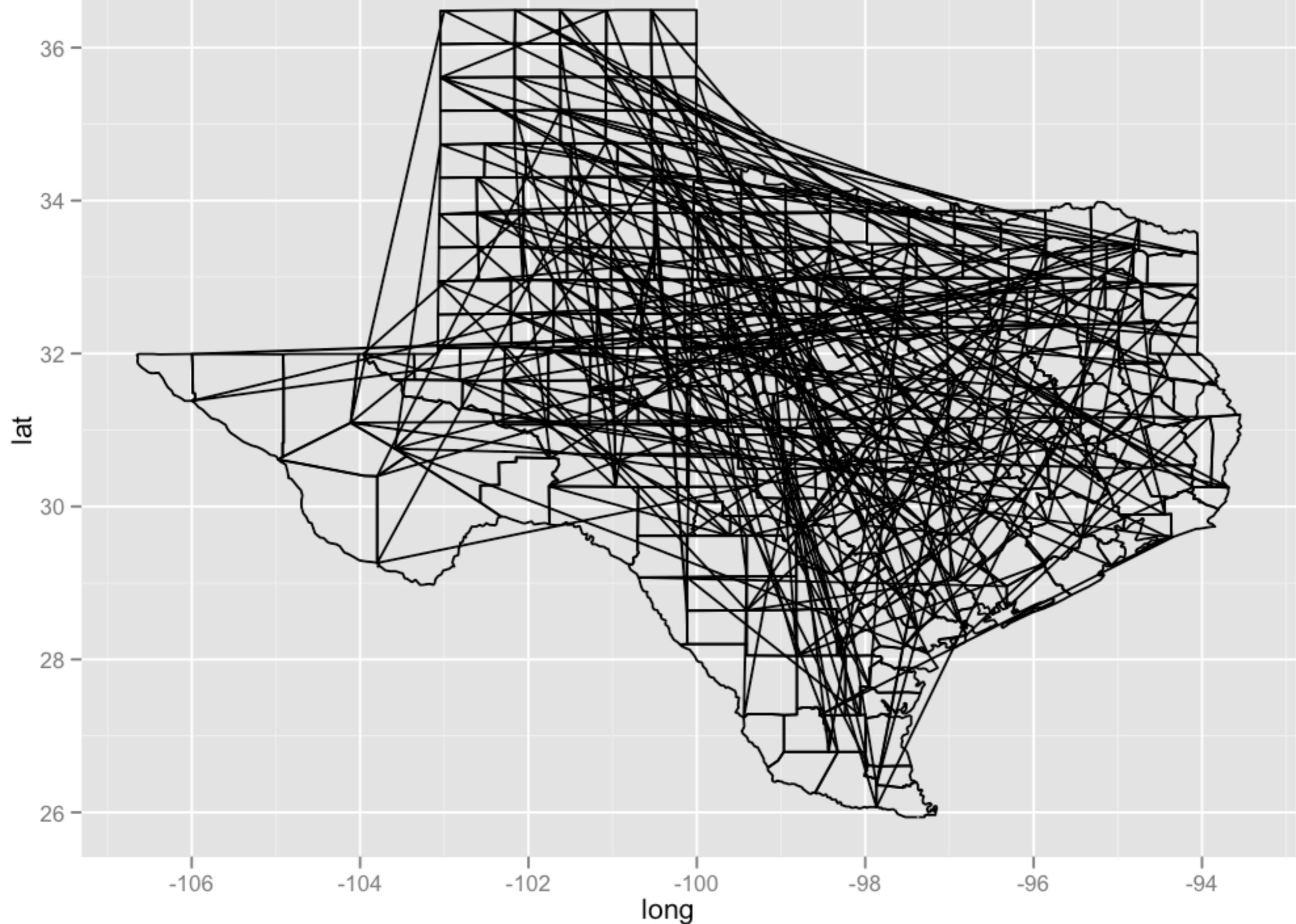
```
borders <- read.csv("tx-borders.csv")
qplot(long, lat, data = borders)
```



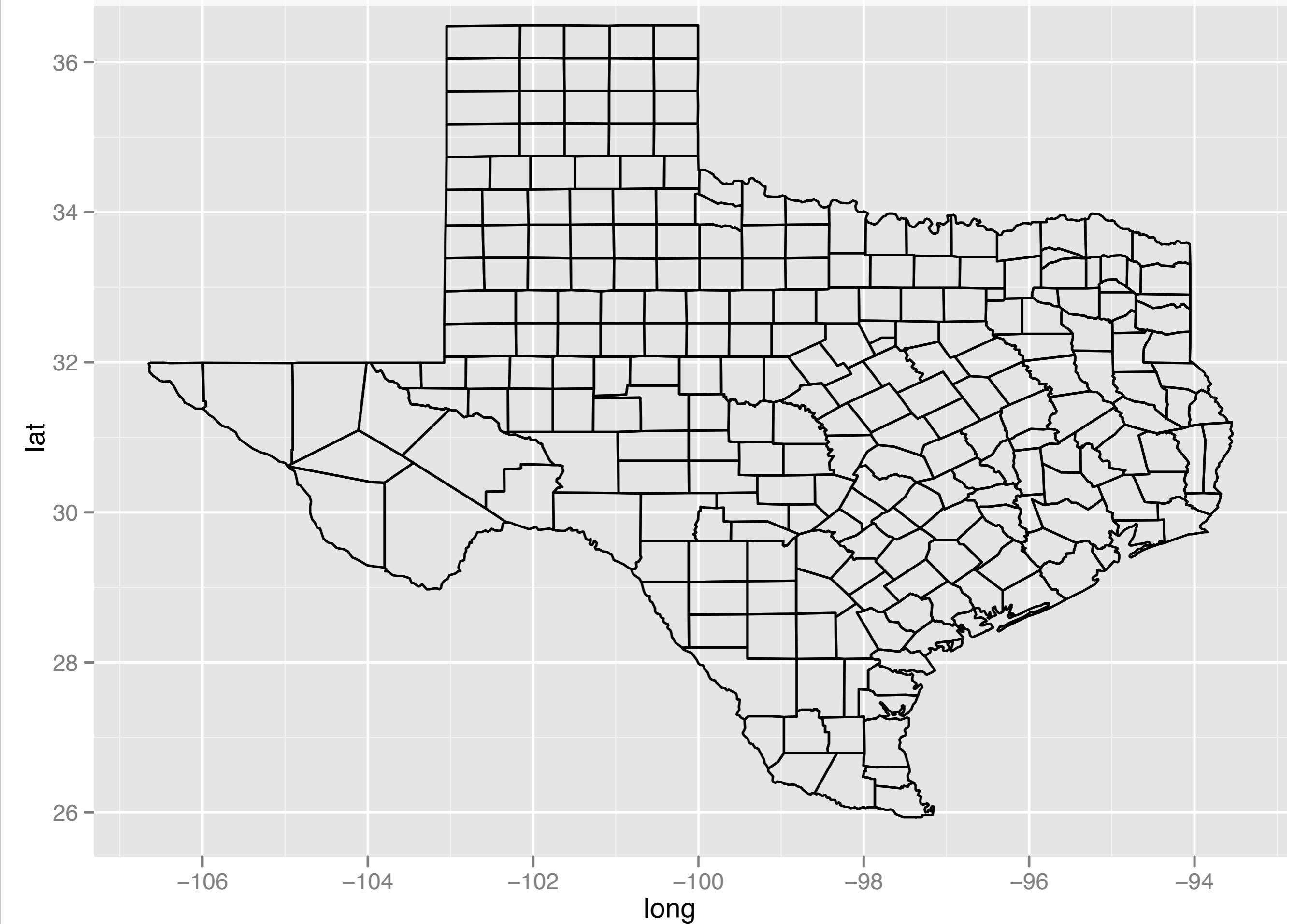
```
qplot(long, lat, data = borders, geom = "line")
```



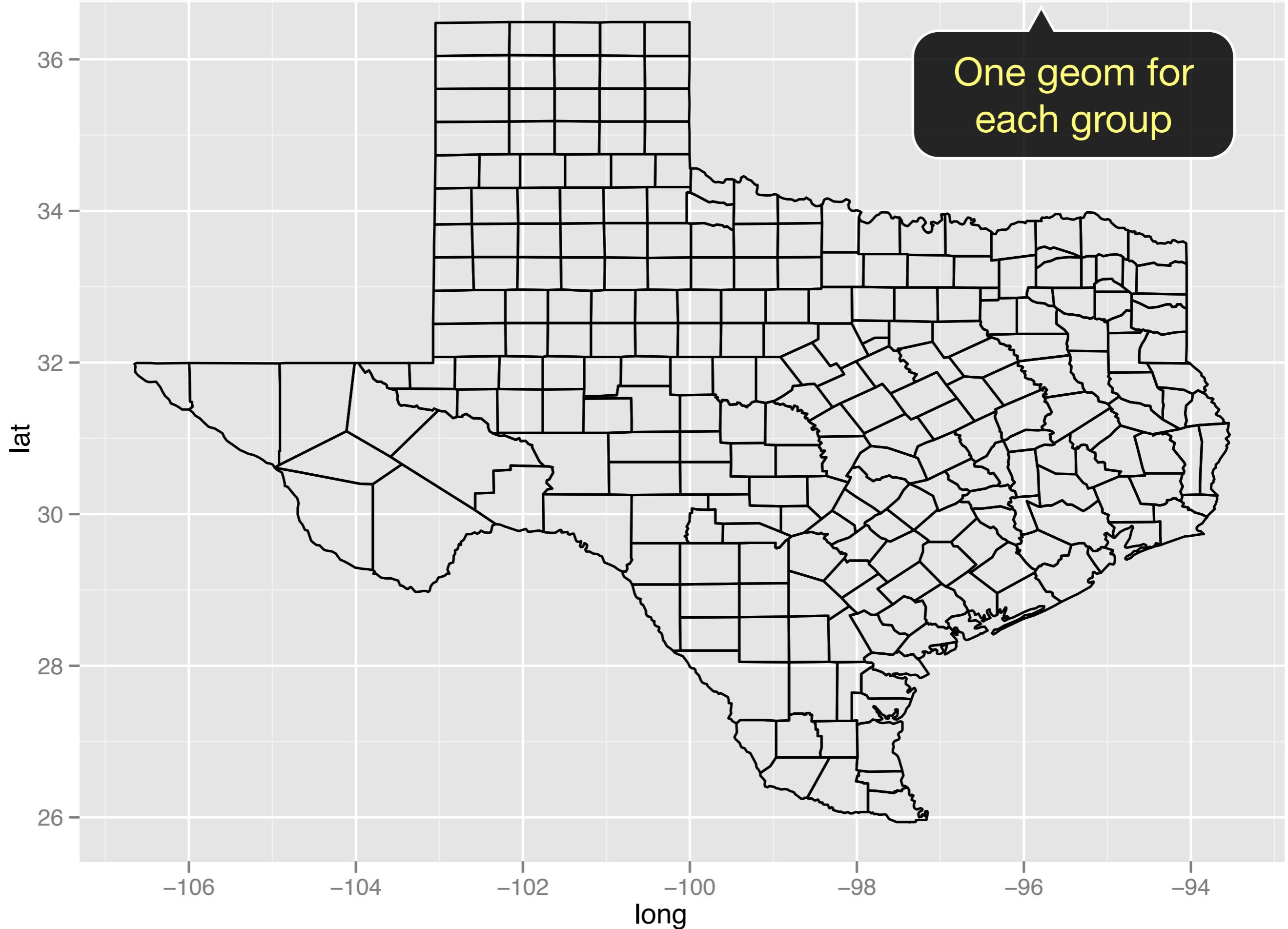
```
qplot(long, lat, data = borders, geom = "path")
```



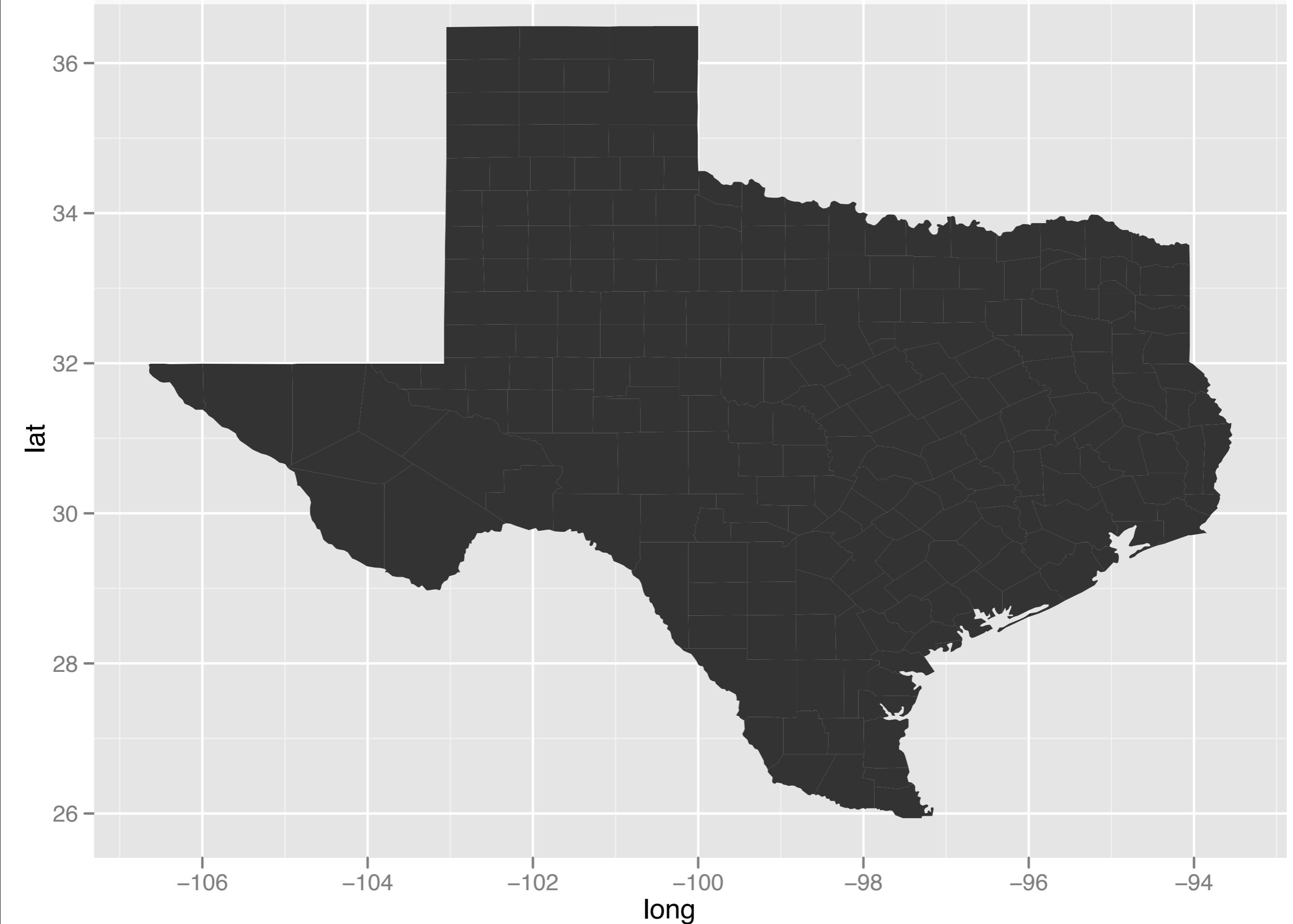
```
qplot(long, lat, data = borders, geom = "path", group = group)
```



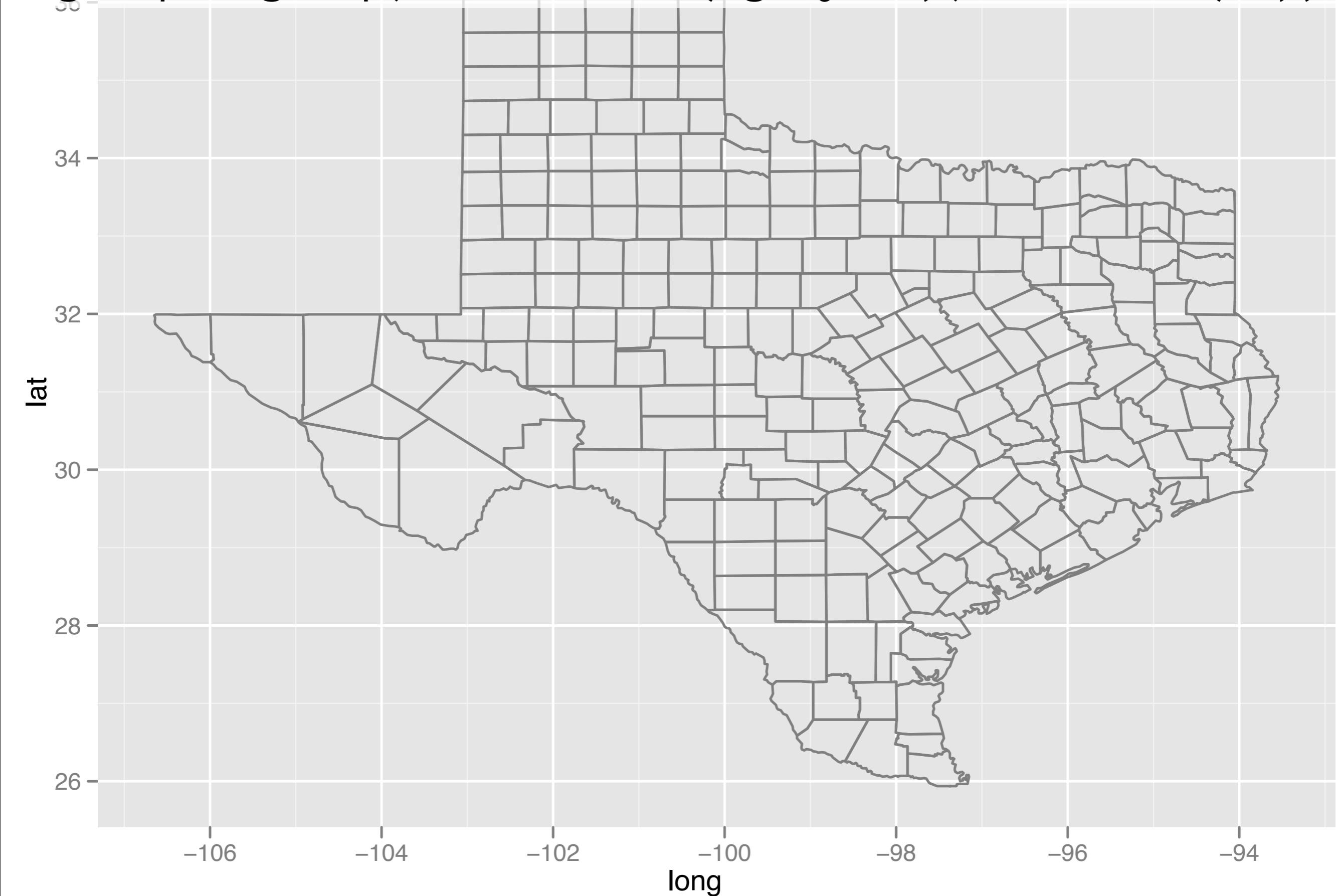
```
qplot(long, lat, data = borders, geom = "path", group = group)
```



```
qplot(long, lat, data = borders, geom = "polygon", group = group)
```



```
qplot(long, lat, data = borders, geom = "polygon",  
group = group, colour = I("grey50"), fill = I(NA))
```



```
qplot(long, lat, data = borders, geom = "polygon",  
group = group, colour = I("grey50"), fill = I(NA))
```



Need `I()` for  
constants

lat

34

32

30

28

26

-106

-104

-102

-100

-98

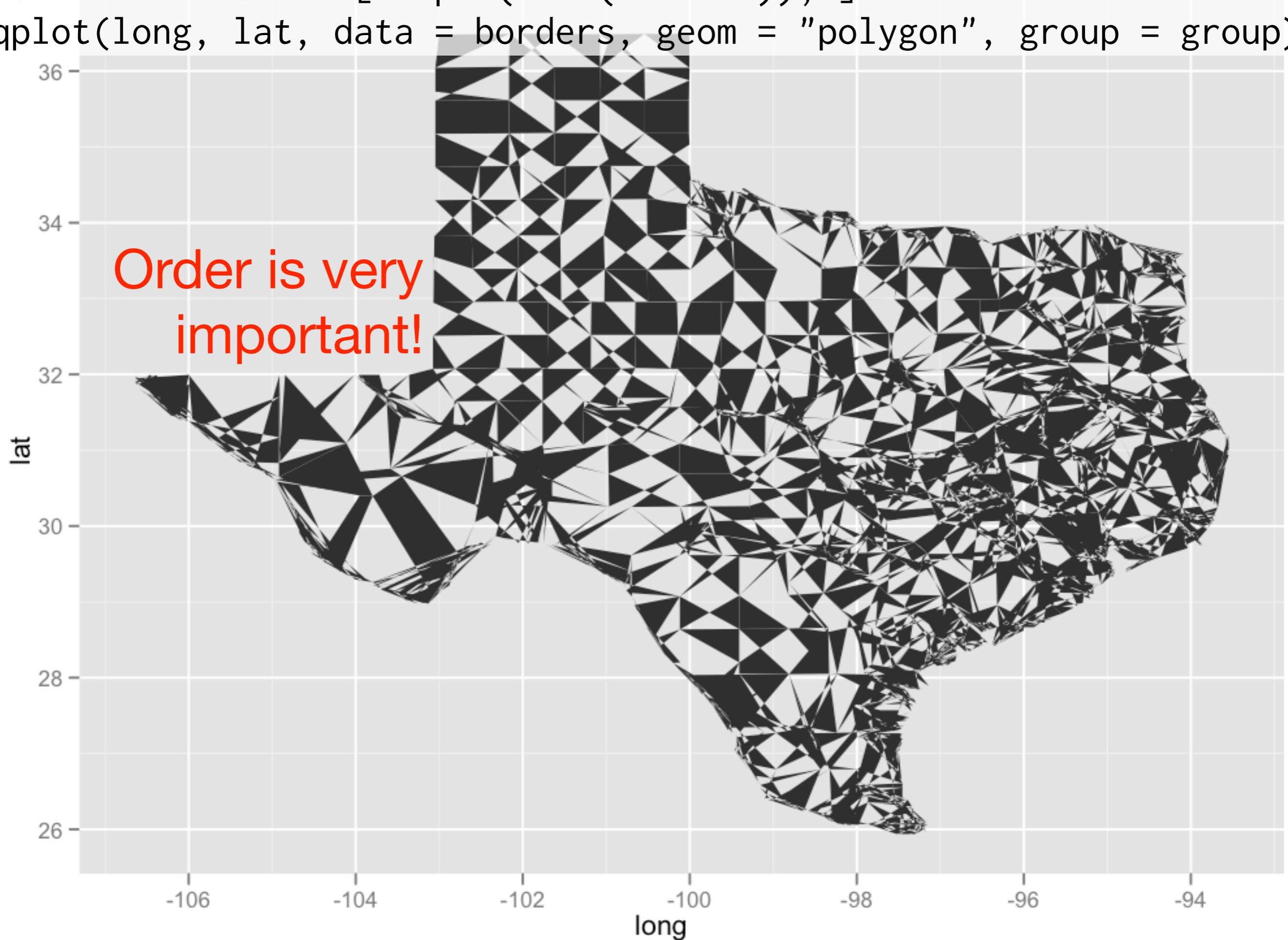
-96

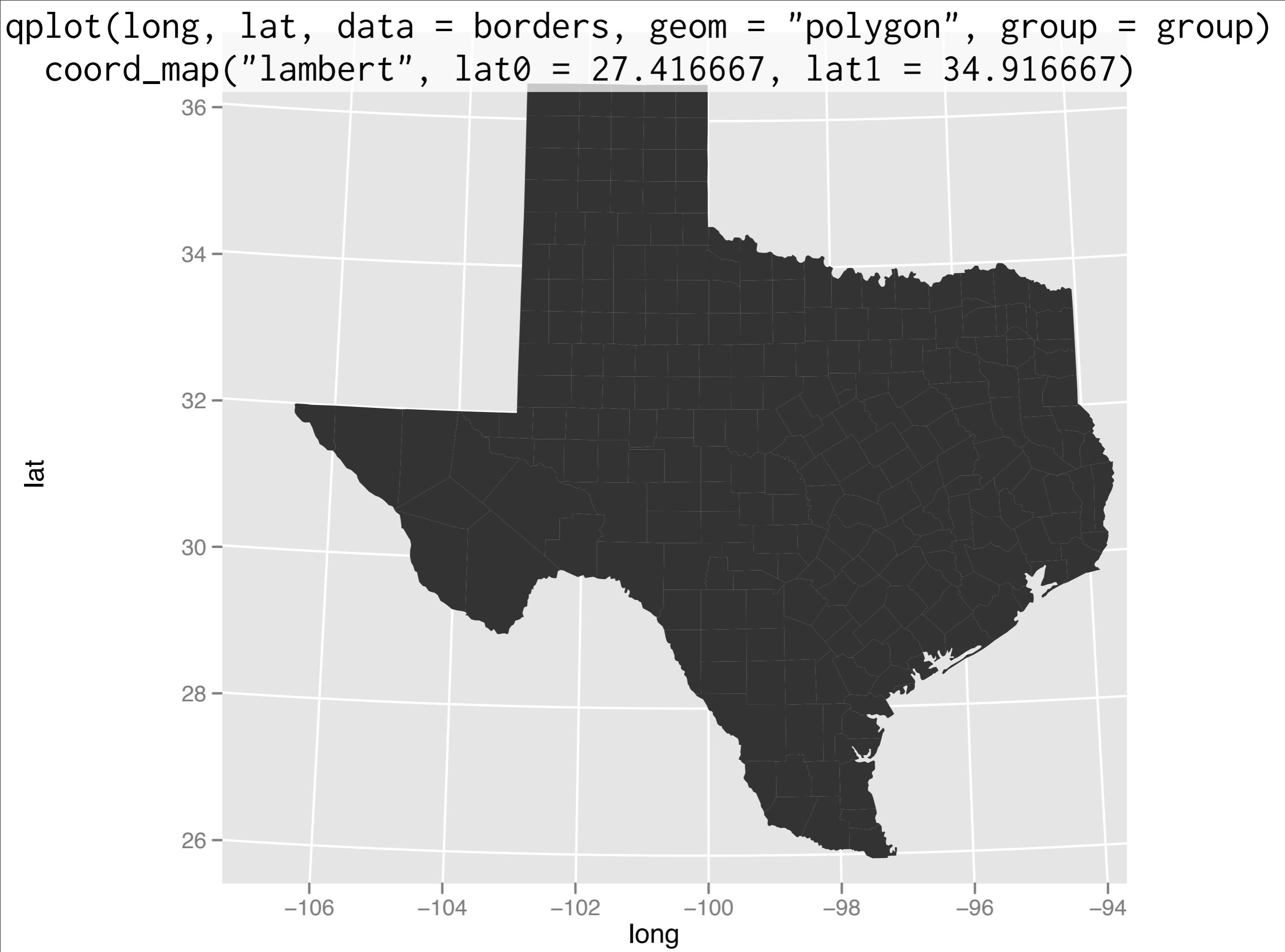
-94

long

```
borders <- borders[sample(nrow(borders)), ]  
qplot(long, lat, data = borders, geom = "polygon", group = group)
```

Order is very  
important!





# Your turn

How could you add county names to this plot? (Hint: think about where you'd want to position them, and how you might summarise the data)

```
library(ggplot2)
library(plyr)

tx <- read.csv("tx-borders.csv",
  stringsAsFactors = FALSE)

mid_range <- function(x) mean(range(x, na.rm = TRUE))
centers <- ddply(tx, "county", summarise,
  long = mid_range(long),
  lat = mid_range(lat))

ggplot(mapping = aes(long, lat)) +
  geom_path(aes(group = group), data = tx) +
  geom_text(aes(label = county), data = centers)
```

```
centroid <- function (x1, y1) {
  n <- length(x1)
  wrap <- c(n, 1:(n - 1))
  x2 <- x1[wrap]
  y2 <- y1[wrap]
  a <- x1 * y2 - x2 * y1
  s <- sum(a) * 3
  if (s < 1e-3) {
    c(mean(x1), mean(y1))
  } else {
    c(sum((x1 + x2) * a)/s, sum((y1 + y2) * a)/s)
  }
}

centers <- ddply(tx, "county", function(df) {
  c <- centroid(df$long, df$lat)
  data.frame(long = c[1], lat = c[2])
})
```

# Map data

Maps are a common type of layer. They are easily recognized and provide context for spatial data. Fairly low-res maps of states and countries can be had from the maps package

# Source of map data

- maps (states, counties)
- ggmap (google, osm)
- osmar (polygon data)
- <http://gadm.org>

```
# install.packages("sp")

library(sp)
load(url("http://gadm.org/data/rda/CHE_adm1.RData"))

head(as.data.frame(gadm))
ch <- fortify(gadm, region = "ID_1")
str(ch)

qplot(long, lat, group = group, data = ch,
      geom = "polygon", colour = I("white")) +
  coord_map()
```

```
# install.packages("maps")

library(maps)
states <- map_data("state")

ggplot(states, aes(long, lat)) +
  geom_path(aes(group = group))
```

# Napoleon's march

*Carte Figurative des pertes successives en hommes de l'Armée Française dans la campagne de Russie 1812-1813.*

Dessiné par M. Minard, Inspecteur Général des Ponts et Chaussées en retraite.

Paris, le 20 Novembre 1869.

Les nombres d'hommes présents sont représentés par les largeurs des zones colorées à raison d'un millimètre pour dix mille hommes; ils sont de plus écrits en lettres des zones. Le rouge désigne les hommes qui ont été en Russie, le noir ceux qui en sortent. — Les renseignements qui ont servi à dresser la carte ont été puisés dans les ouvrages de M. Chiers, de Séguir, de Fezensac, de Chambray et le journal inédit de Jacob, pharmacien de l'Armée depuis le 28 Octobre.

Pour mieux faire juger à l'œil la diminution de l'armée, j'ai supposé que les corps du Prince Jérôme et du Maréchal Davout qui avaient été détachés sur Minsk et Maliblow se sont rejoints vers Orysha et Witlobk, avant de toujours marcher avec l'armée.

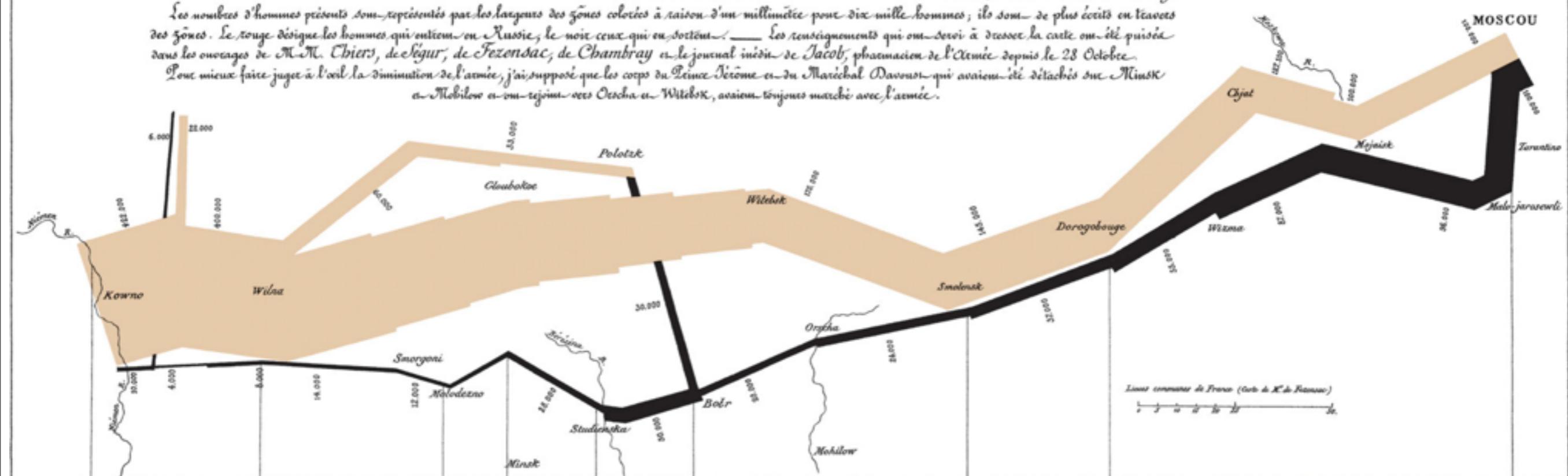


TABLEAU CRAPHIQUE de la température en degrés du thermomètre de Réaumur au dessous de zéro.

Les Cosaques passent au gelop  
le Nilmen gelé.

- 26° le 7 X<sup>bre</sup>

- 30° le 6 X<sup>bre</sup>

- 24° le 11 X<sup>bre</sup>

- 20° le 28 9<sup>bre</sup>

- 11°

- 21° le 14 9<sup>bre</sup>

- 8° le 9 9<sup>bre</sup>

Zéro le 18 8<sup>bre</sup>

Pluie 24 8<sup>bre</sup>

30 degré

Imp. Litt. Regnier et Bourdet.

Avec la Régie, E. Par. S<sup>e</sup> Maria S<sup>e</sup> G<sup>e</sup> à Paris.

# Your turn

What are the layers in Minard's plot?

What geom and aesthetics do each layer use?

Which layers show the main data and which show contextual information?

```
troops <- read.table("minard-troops.txt", header=T)
cities <- read.table("minard-cities.txt", header=T)

russia <- map_data("world", region = "USSR")

qplot(long, lat, group = group, data = russia,
      geom = "polygon")

russia[russia$long < 0, ]
russia <- subset(russia, group != 32)
qplot(long, lat, group = group, data = russia,
      geom = "polygon")
```

# Your turn

Attempt to recreate Napoleon's march by  
Minard as closely as possible

```
ggplot(troops, aes(long, lat)) +  
  geom_path(aes(size = survivors, colour = direction,  
    group = group), lineend = "round") +  
  geom_text(aes(label = city), size = 4, data = cities)
```

```
ussr <- geom_polygon(aes(long, lat, group = group),  
  data = russia, fill = "white")  
  
ggplot(troops, aes(long, lat)) +  
  ussr +  
  geom_path(aes(size = survivors, colour = direction,  
    group = group, lineend = "round")) +  
  geom_text(aes(label = city), size = 3,  
  data = cities)
```

```
# polishing plot
last_plot() +
  scale_size(range = c(1, 6),
             breaks = c(1, 2, 3) * 10^5, labels = comma(c(1, 2, 3) * 10^5)) +
  scale_colour_manual(values = c("bisque2", "grey10")) +
  xlab(NULL) +
  ylab(NULL) +
  coord_equal(xlim = c(20, 40), ylim = c(45, 65))
```