SMI 606: Discovery Using Spatial Data Analysis

Dr. Todd Hartman
Sheffield Methods Institute

Chipotle

Visualising the growth of Chipotle restaurants in space and time
Prompted by an article in *Bloomberg Business* article on the 'oral history' of the company
LINK

https://www.bloomberg.com/graphics/2015-chipotle-oral-history/

Plan

Take data on store openings and locations
Visualization:

- Plot and animate this data in multiple ways on a map of the US

Chipotle

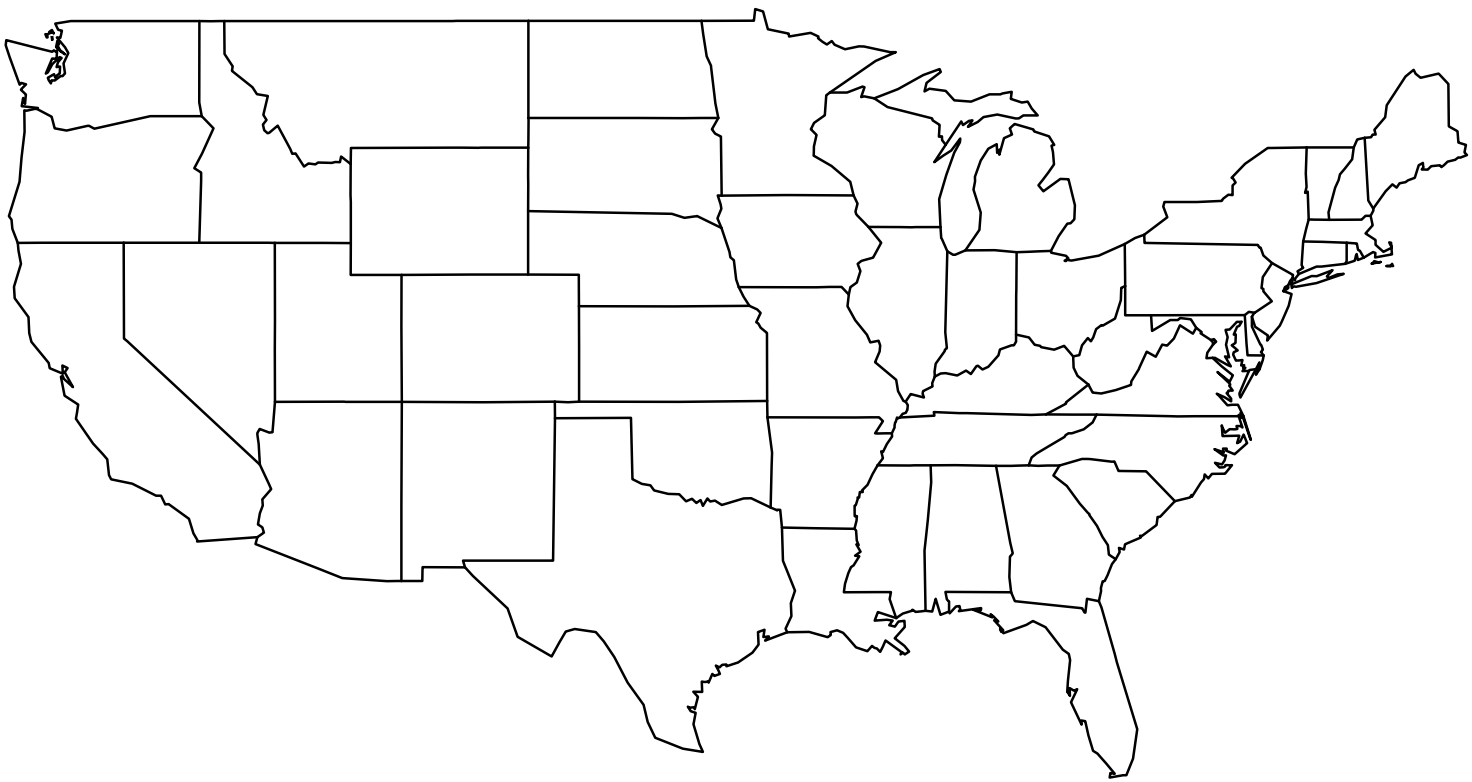
```
load('data/chipotle.RData')
colnames(chipotle)
```

```
 [1] "state"       "region"
 [3] "market"      "restaurant"
 [5] "id"          "address"
 [7] "city"        "zip"
 [9] "latitude"    "longitude"
[11] "zipcity"     "zipstate"
[13] "date"        "state.name"
[15] "old.date"
```

US states on a map

```
library(maps)
map(database="state")
```

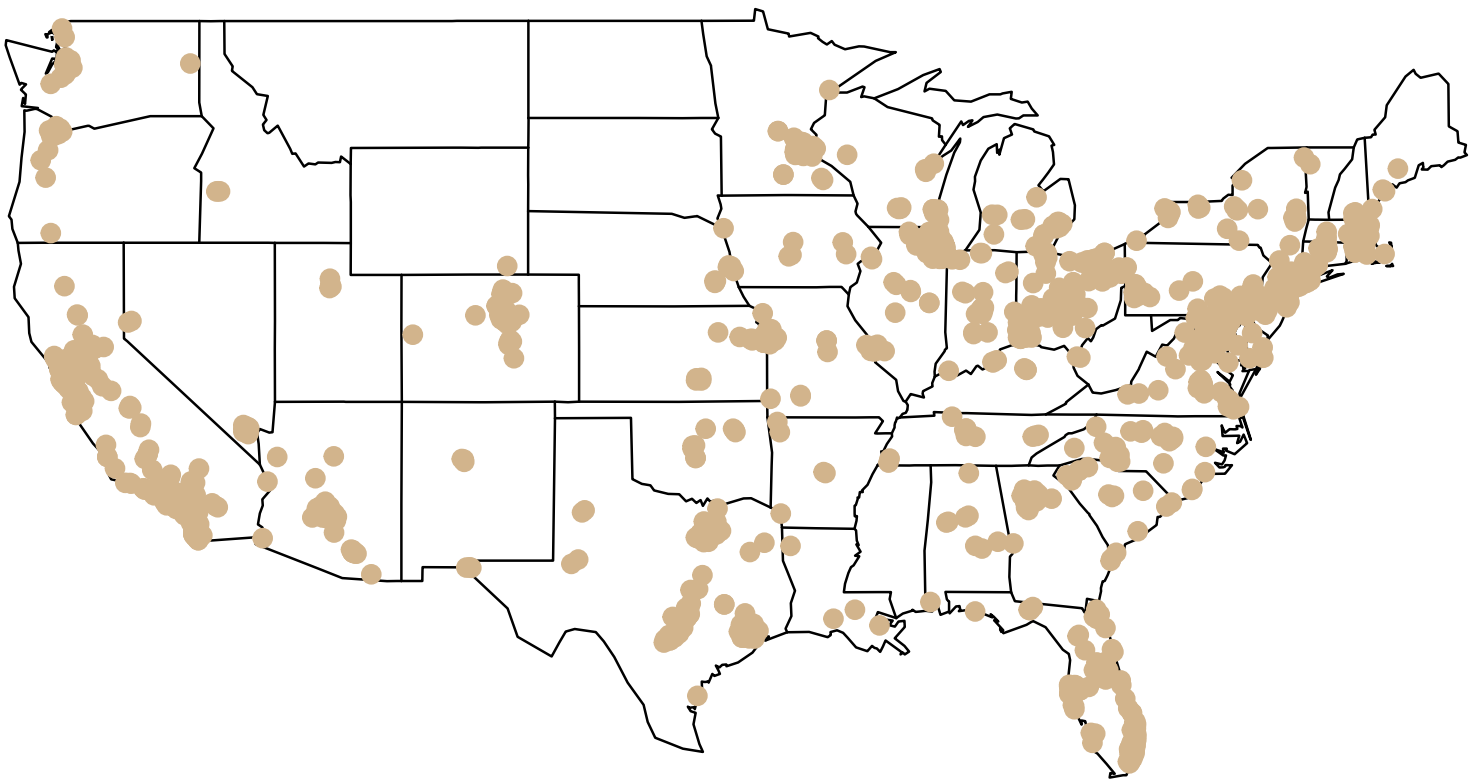
(Not quite all) US states



Chipotles in (not quite all) US states

```
points(chipotle$longitude, chipotle$latitude,
       pch=19, col='tan')
```

Chipotles in (not quite all) US states



Chipotles in (not quite all) US states

Tan? Seriously?

How to find out what colours R knows about

```
cols <- colors()
length(cols)

[1] 657

cols[1:5]

[1] "white"          "aliceblue"
[3] "antiquewhite"   "antiquewhite1"
[5] "antiquewhite2"
```

Colour me burritofull

'Overplotting' is messy and uninformative
Let's use some **transparency** to show Chipotle density

- alpha transparency

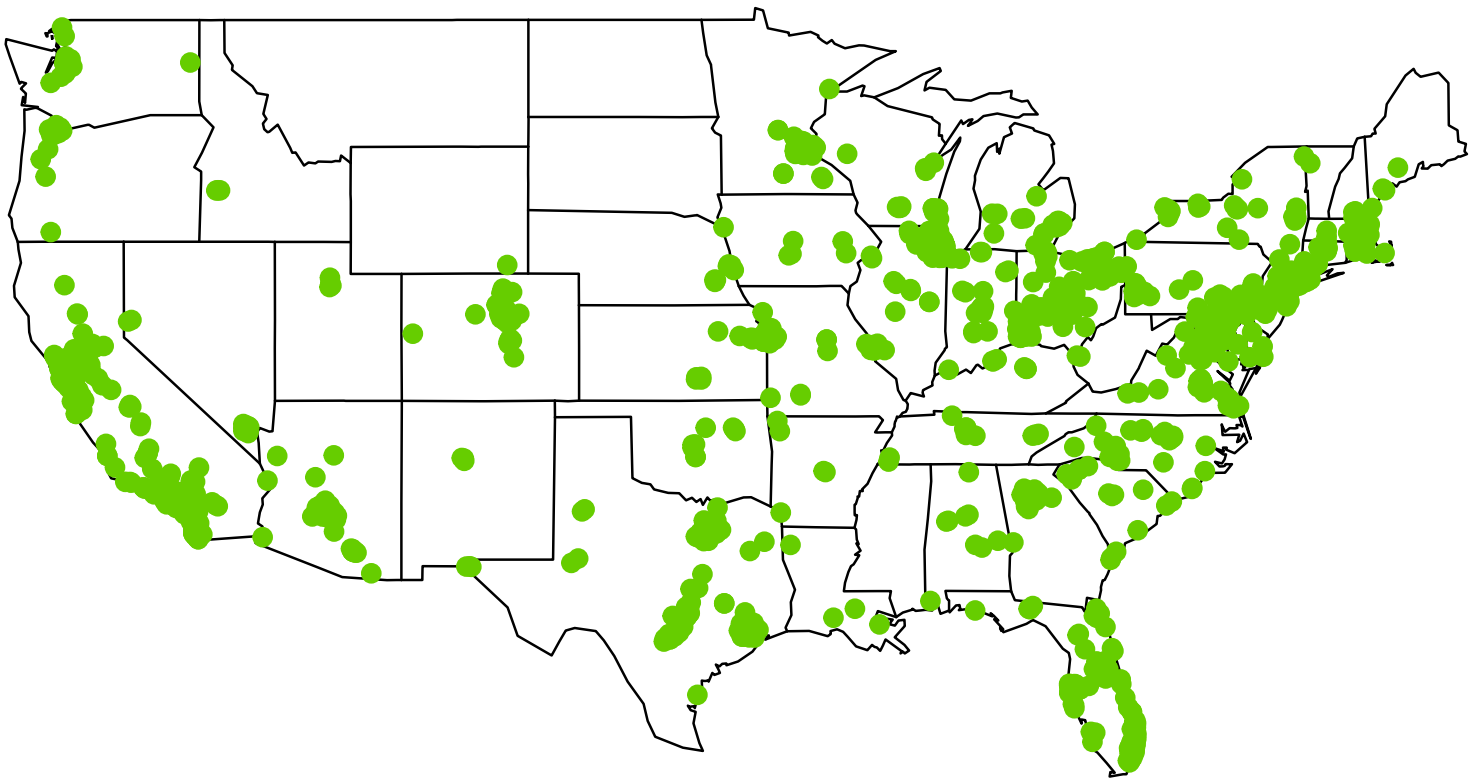
For this we need to *build our own colour*
It's going to cost us extra (extra R) but it's worth it

Colour me burritofull

Let's start with a nice avocado-ish green called 'chartreuse3'

```
map(database="state")
points(chipotle$longitude, chipotle$latitude,
       pch=19, col='chartreuse3')
```

Colour me burritofull



Chartreuse as RGB

There are a lot of ways to specify a colour numerically

- hue, chroma, and luminance: `hcl`
- hue, saturation and value: `hsv`
- this one:

```
grey(1:50 / 50)
```

But we'll use red, green, and blue: `rgb`

Chartreuse as RGB

```
rgb(1, 0, 0, 1) ## really really red
```

```
[1] "#FF0000FF"
```

100% red, 0% green, 0% blue, 100% opaque
Let's find out what `chartreuse3` really is

```
col2rgb('chartreuse3') ## careful: these are out of 255
```

```
      [,1]
red    102
green  205
blue     0
```

Let's make a colour quite like this, but a bit transparent

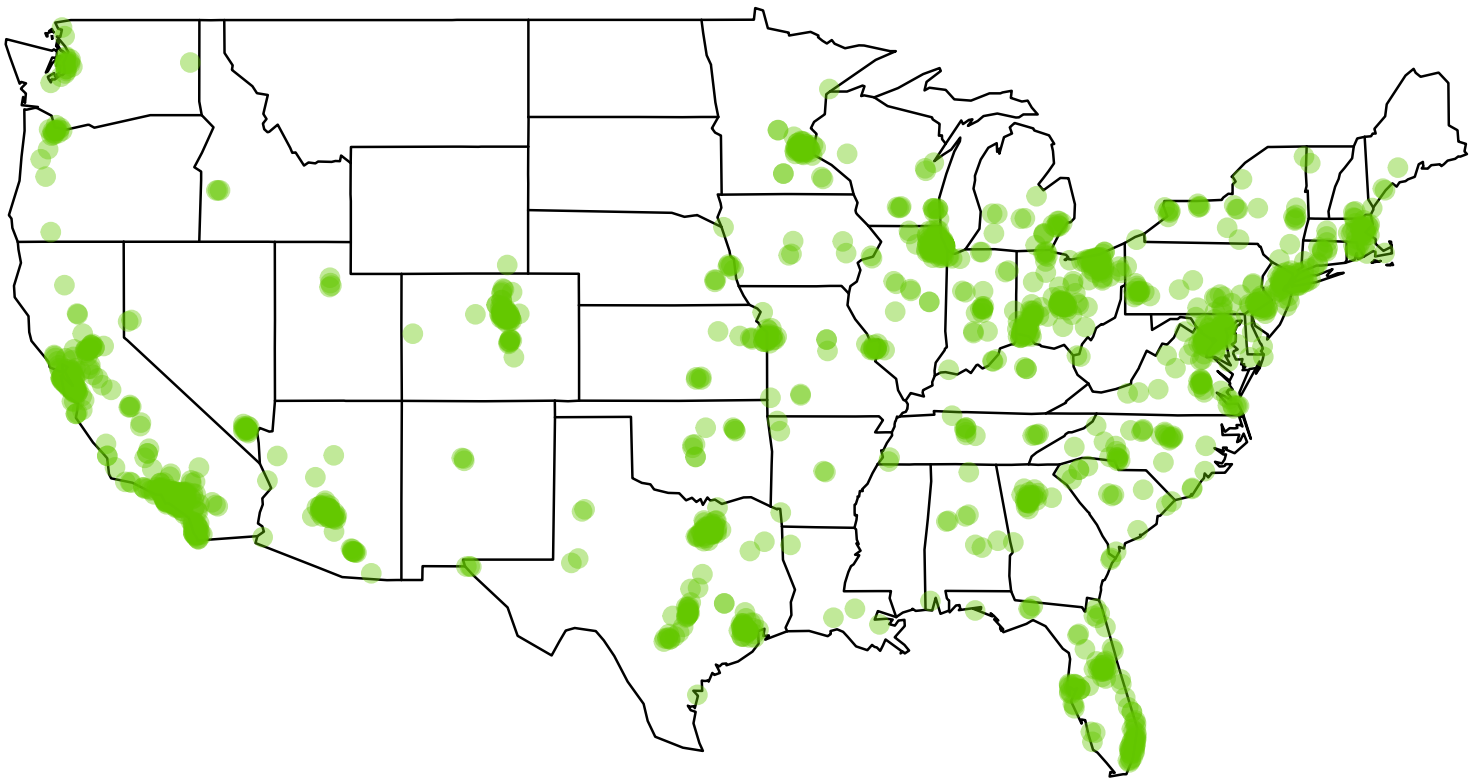
```
guac <- rgb(100/255, 200/255, 0, 0.4)
```

and use it to plot Chipotle locations

Colour is extra

```
map(database="state")
points(chipotle$longitude, chipotle$latitude,
       pch=19, col=guac)
```

Colour is extra



Burritos in the space-time continuum

We have data about when each Chipotle opened. So let's make a map for *each year* of the company's existence and watch it grow
Steps:

- Identify the stores opened in a particular year
- and plot them on (top of the existing stores)
- repeat. . .

Burritos in the space-time continuum

We have data about when each Chipotle opened. So let's make a map for *each year* of the company's existence and watch it grow
Steps:

- Identify the stores opened in a particular year
- and plot them on (top of the existing stores)
- repeat...

In R:

- Subset the data
- Make a function to plot the stores
- Loop!

Make a function

Function strategy:

- Figure out how to do something for a *particular* set of data
- *Generalize* to all similar data

Subsetting

How to get the data for just 2011:

We have dates, but we want *years*

```
d <- chipotle$date[5] # store in 5th row of data
d
```

```
[1] "2013-04-09"
```

so let's make a variable `year` that we can subset on

First we get the year part of the date

```
format(d, "%Y") # %m for month
```

```
[1] "2013"
```

This is a string, so we'll convert it to a number for convenient indexing

```
as.numeric(format(d, "%Y"))
```

```
[1] 2013
```

Subsetting

Happily R will format and convert a vector of dates too, so let's use that fact and assign the result back to our data

```
chipotle$year <- as.numeric(format(chipotle$date, "%Y"))
```

So we can get 2011 data with a subset

```
one.year <- subset(chipotle, year==2011)
nrow(one.year)

[1] 147
```

Now we've got something to tell us what data to apply our function *to*

Make a function

We're going to *generalize* the `points` command we used earlier and make it *to go*
Generalize:
Before

```
points(chipotle$longitude, chipotle$latitude,
       pch=19, col=guac)
```

Make a function

We're going to *generalize* the `points` command we used earlier and make it *to go*
Generalize:
Before

```
points(chipotle$longitude, chipotle$latitude,
       pch=19, col=guac)
```

After

```
one.year <- subset(chipotle, year==2011)
points(one.year$longitude, one.year$latitude,
       pch=19, col=guac)
```

Make a function

Wrap:

```
plot.year <- function(y){
  one.year <- subset(chipotle, year==y)
  points(one.year$longitude, one.year$latitude,
         pch=19, col=guac)
}
```

Note: y is going to be any year that makes sense
so now we can say:

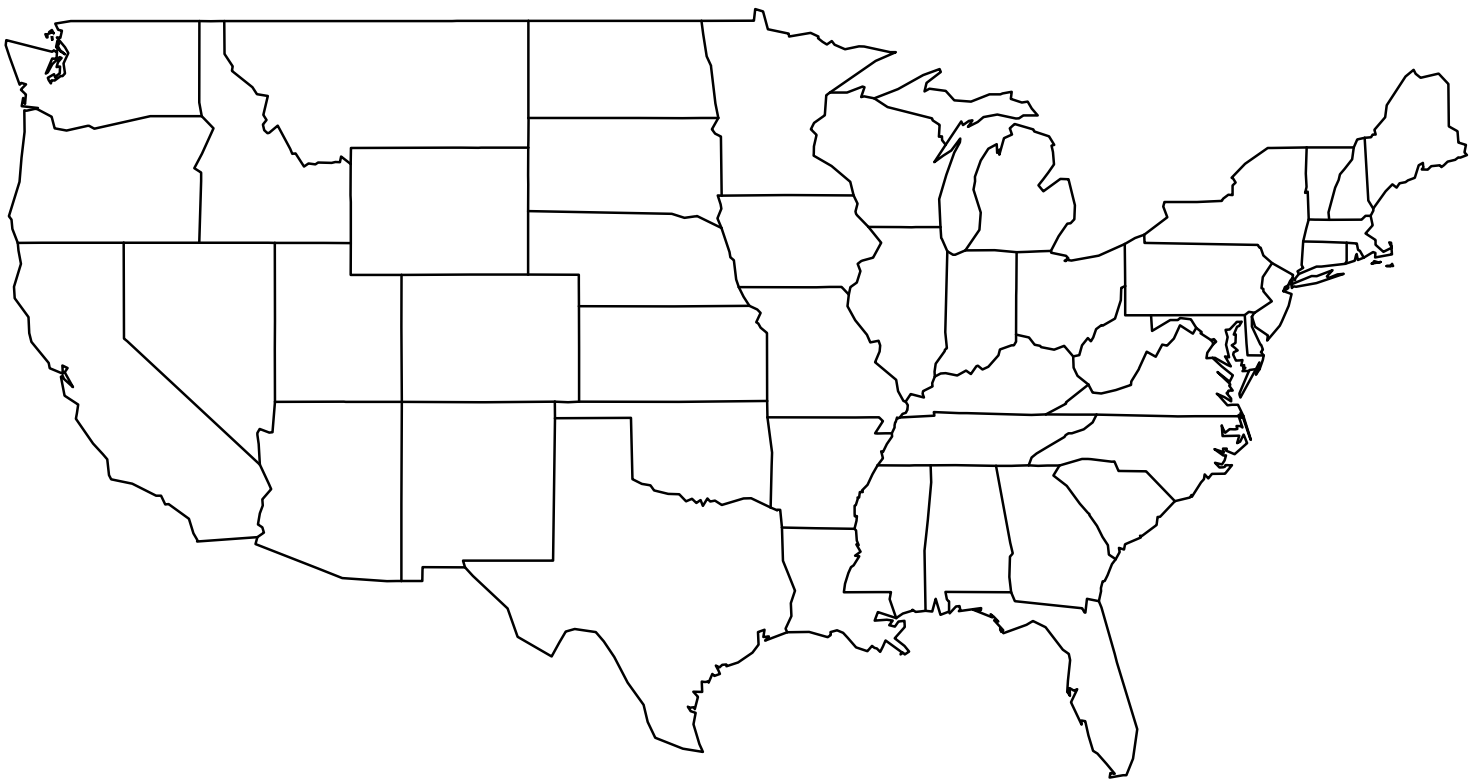
```
plot.year(2011)
```

Make a canvas to plot on

This bit is easy

```
map(database="state")
```

Make a canvas to plot on

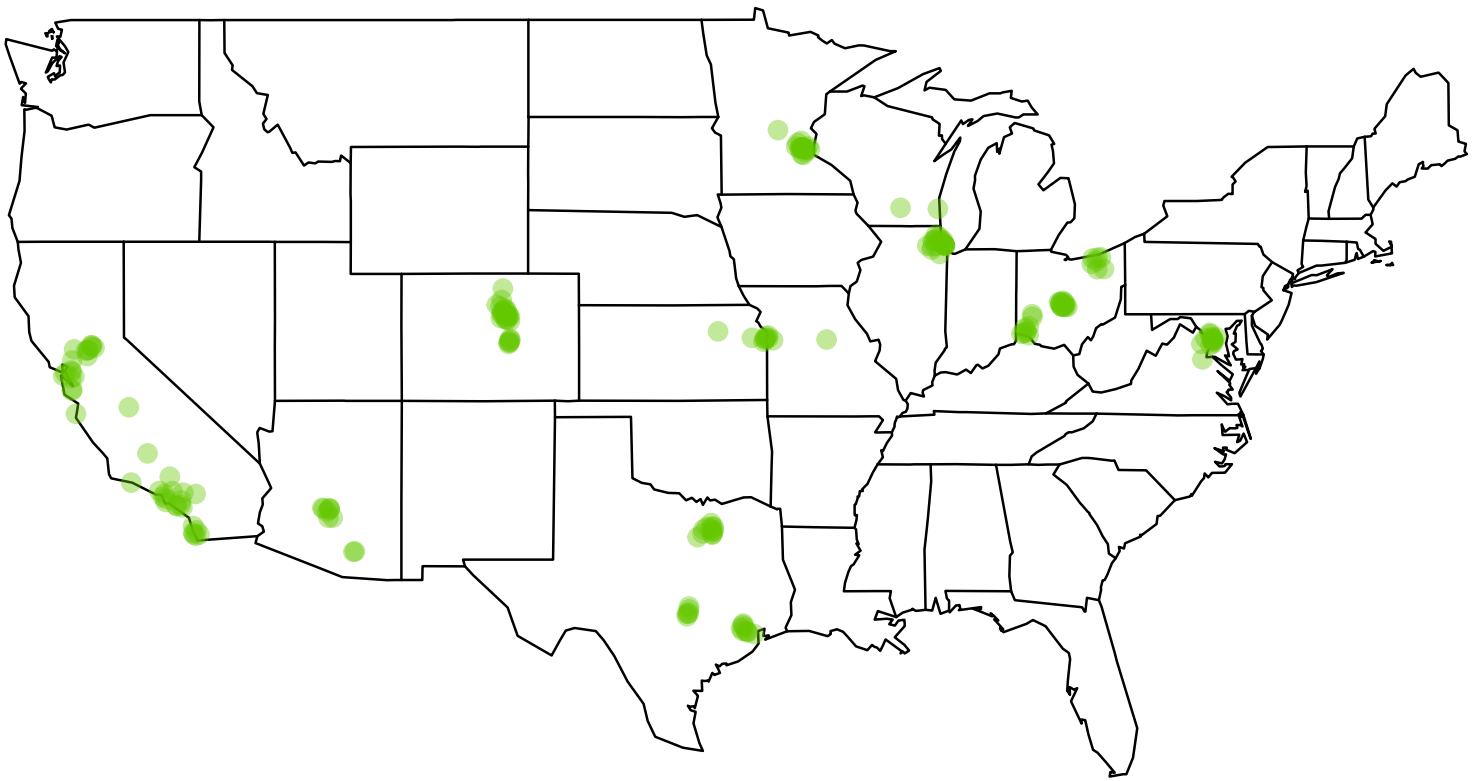


Loop!

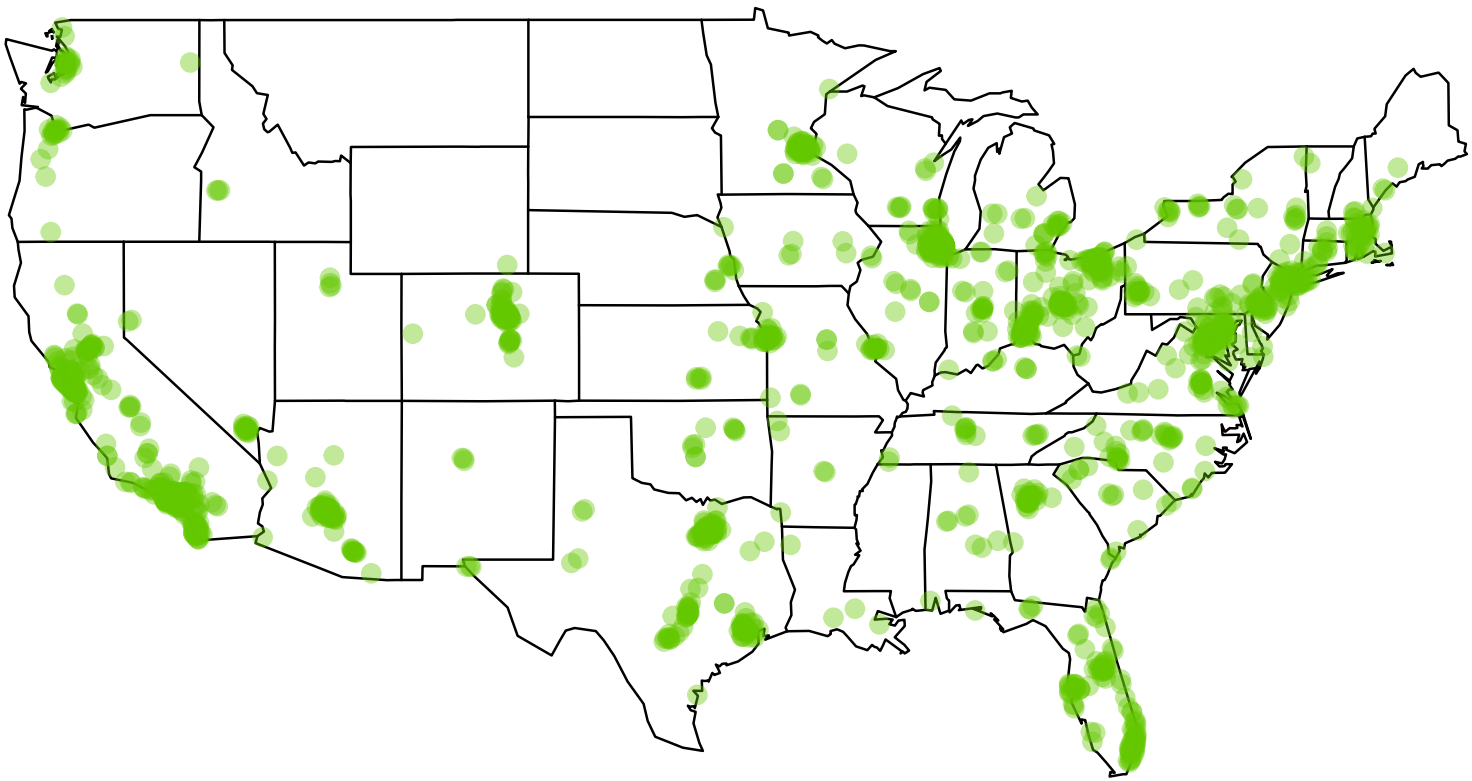
Let's plot *all store openings* through 2002

```
for (y in 1993:2002){
  plot.year(y)
}
```

Stores through 2002



Stores through 2014



Animation

We can use the `animation` library to animate.
Example: Make an offering to IMGUR

```
library(animation)
saveGIF({
    for (i in 1993:2014) {
      map(database="state")
      for (y in 1993:i){
        plot.year(y)
      }
      title(i)
    }

  }, movie.name="chip.gif",
     outdir=getwd(),
     interval=0.5)
```

The rise of Chipotle

PRESS ME

http://dl.conjugateprior.org/chip.gif

Statewide burritofest

We can also colour the *entire state* according to the number of Chipotles in it

- when regions are coloured according to a variable this is called a **chloropleth** map

e.g. colour a state more guac the more Chipotle's there are in it.

Statewide burritofest

In the book we colour states acording to their relative vote percentages, e.g.

- 45% red, 55% blue, 0% green

Here, we have just one colour: `guac`, and we will fade it in and out to represent number of stores

Alpha transparency again

Recall

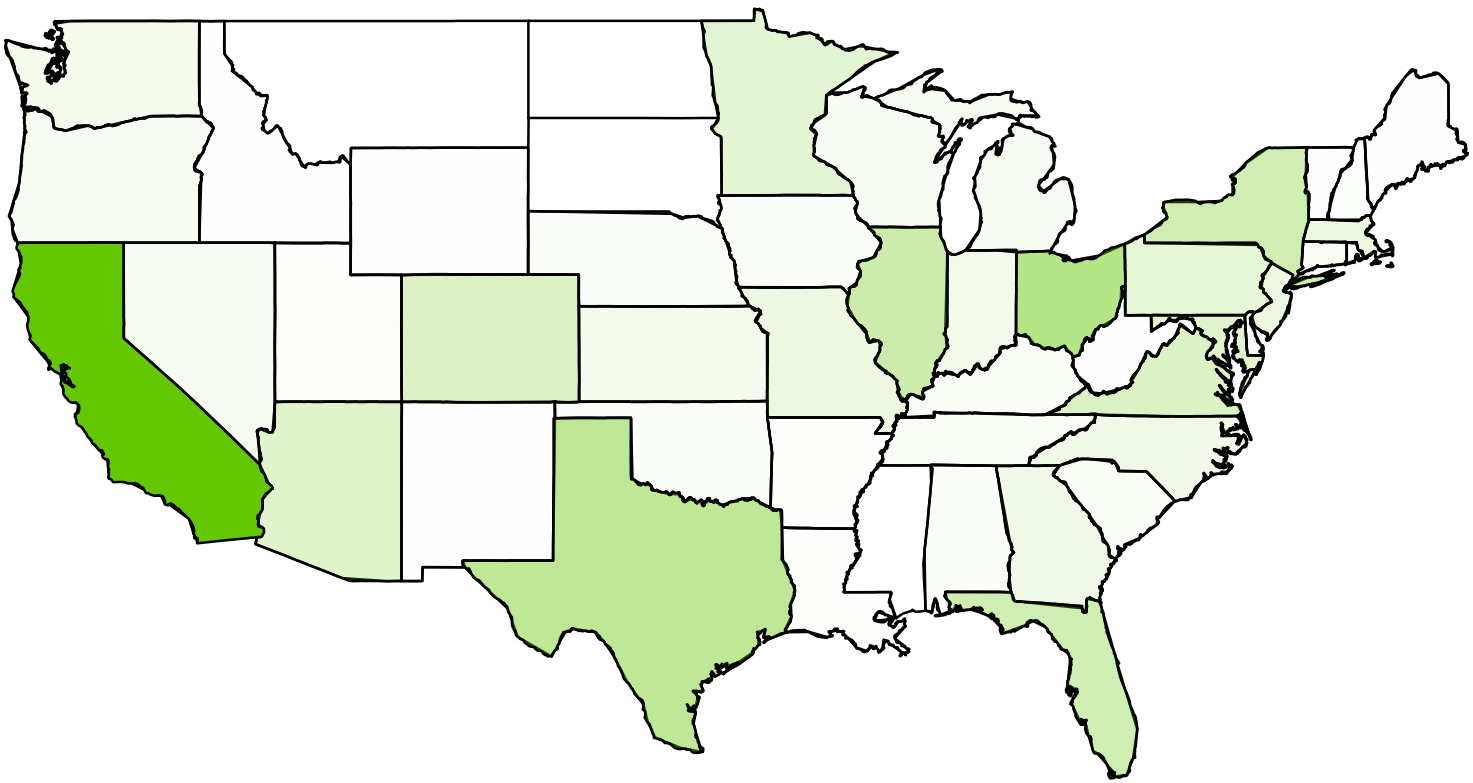
- Behind each point is the (white) background
- When a point has a colour with alpha transparency *less than 1* then some white background (or another point) *shows through*

```
pstate <- tapply(chipotle$id, chipotle$state.name, length)
prop.state <- pstate / max(pstate) # so it's from 0 to 1
prop.state[1:5]

   Alabama    Arizona   Arkansas
0.02795031 0.19875776 0.01242236
California   Colorado
1.00000000 0.22670807
```

Chipotles by US state

```
map(database = "state")
for (i in 1:length(prop.state)){
  state <- names(prop.state)[i]
  if (state != "Alaska" &
      state != "District of Colombia"){

    state.col <- rgb(100/255, 200/255, 0, prop.state[i])
    map(database="state", regions=state,
        col=state.col, fill=TRUE, add=TRUE)
  }
}
```

Chipotles by US state



Population

Let's work instead with *Chipotle's per million people* (CPM)
The (averaged) population data is called `mpop`

```
load('data/mpop.RData')
mpop[3:5,]
```

```
                state      pop
Arizona       Arizona  5498967
Arkansas     Arkansas  2727250
California California 35158366
             pop.prop
Arizona    0.15640566
Arkansas   0.07757043
California 1.00000000
```

Population

We'll choose the amount of colour for state i according to

$$CPM_i = \frac{\text{number of chipotles in } i}{\text{population size in } i} \times 1000000$$

```
served <- merge(mpop,
   data.frame(state=names(pstate), count=pstate))
served$cpm <- served$count / served$pop * 1000000
```

CPM

Which states are better served with Chipotle's?

```
sort(served$cpm, decreasing=TRUE)[1:4]

District of Colombia
            25.42889
            Colorado
            16.22812
                Ohio
            13.26741
             Arizona
            11.63855
```

CPM

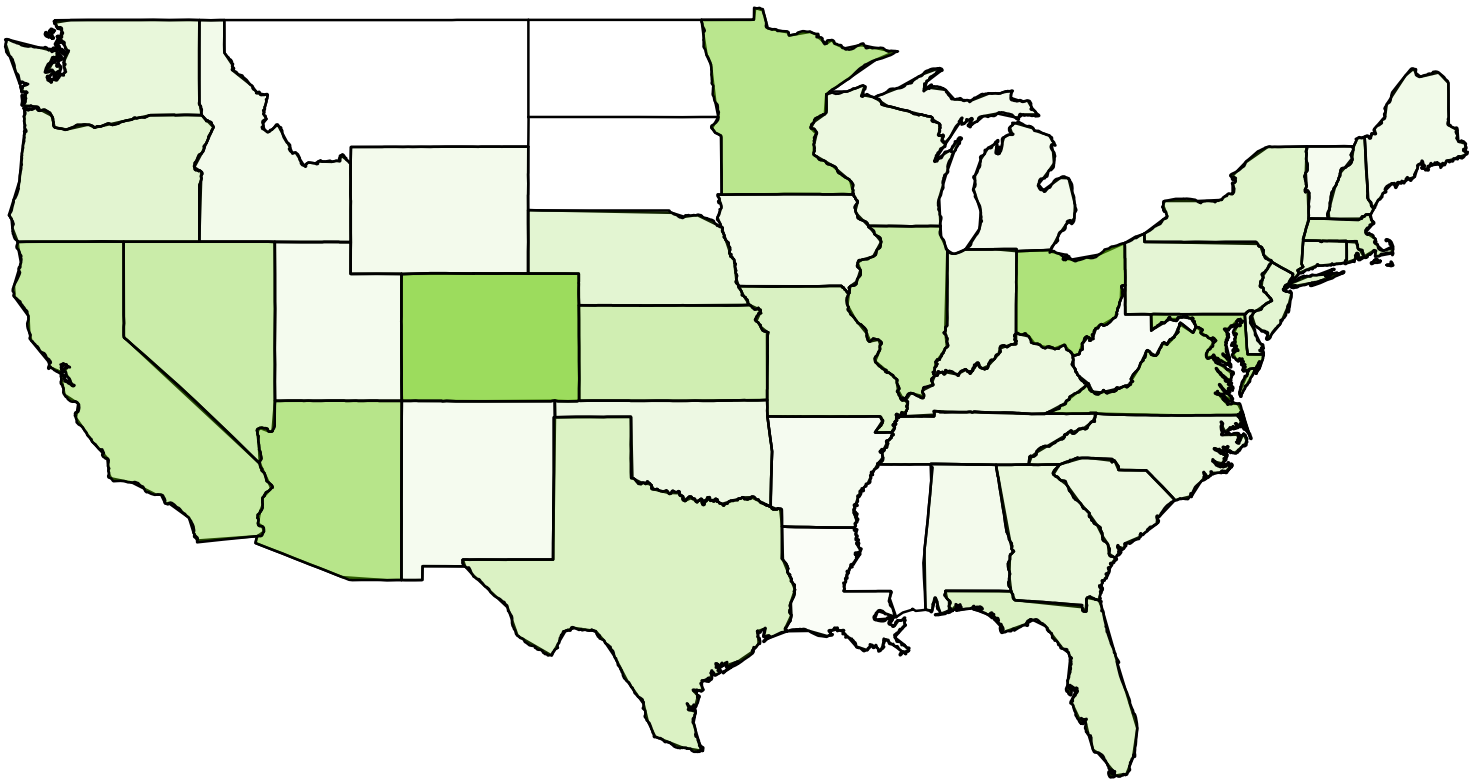
Normalise the measure so we can use it as an alpha transparency

```
served$norm.cpm <- served$cpm / max(served$cpm)
```

CPM

```
map(database = "state")
for (i in 1:nrow(served)){
  state <- served$state[i]
  if (state != "Alaska" &
      state != "District of Colombia"){

    state.col <- rgb(100/255, 200/255, 0, served$norm.cpm[i])
    map(database="state", regions=state,
        col=state.col, fill=TRUE, add=TRUE)
  }
}
```

CPM



Wrapping up

Maps are great!
but do think carefully about what you're showing

The other kind of wrapping up