Field Guide of Relational and Contextual Epidemiology

Jon Zelner

2023-09-08

Table of contents

1 Inspirations and Influences

2 Inspirations and Influences

What - if any - kind of book needs to be written about relational and contextual epidemiology?

2.1 Why relational and contextual epidemiology?

There is a hole in the epidemiological literature that limits our ability to come to grips with the *relational* aspects of health and illness. For the purposes of this book, relationships are sources of non-independence. These could be *social* influences, e.g. from our friends and family. But they could also be *spatial* in nature, e.g. a result of some physically proximate environmental exposure. Often, the relationships we care about are *temporal* in nature, as is evident in the lifecourse perspective in which early-life exposures are understood to impact later-life outcomes.

There are many good books and papers out there about these topics as well as some of their conjunctions with each other (e.g. spatiotemporal analysis). But there is less written about their overlaps and commonalities, why knowing one teaches you so much about the others, and what having access to this set of tools might mean for a working epidemiologist.

In many ways, my goal in writing this is selfish, to satisfy a personal curiosity I suspect is of some greater import to at least a few people. Specifically: What does it mean to do *contextual* or *relational* epidemiology? Are these meaningless terms that just appeal to the social and natural science centers of my brain, or does this tap into something more meaningful?

2.1.1 More than methods

Part of my inspiration is a reaction to the technocratic impulse and imperative in modern epidemiology. In particular, I find myself a bit paralyzed by the worry about what happens when we operate under the assumption that escalating methodological complexity is an imperative and that the road out of socio-epidemiological problems is paved with technological solutions.

2.1.2 But also, methods...

On the other side, simply put, I love the methodological tools of spatial epidemiology, Bayesian hierarchical analysis, and systems modeling. I have learned more than I ever could have hoped through learning, tinkering with, and applying these tools to problems in the real world and in my own head. But for me, the large majority of lessons learned have been from their conceptual isomorphisms (or conflicts) with the world as it appears to us through qualitative and quantitative data, rather than in the exact values of their parameter values and quantitative predictions.

For me, this book is about resolving this cognitive dissonance while providing useful 'how-to' pointers along the way. I hope to articulate the affirmative case for a systems-based, contextually-sensitive, justice-oriented, morally and ethically opinionated, and theoretically driven approach to epidemiology. Along the way, I hope to show why the tools of such an approach are necessarily heterogeneous in nature and require us to accept uncertaintities quantitative and epistemic.

2.2 Influences

There are any number of books and papers out there that have articulated a similar perspective on the tools of quantitative analysis. The following have been particularly important for my own thinking and their influences will be felt throughout this work:

- Statistical Rethinking (1)
- The Ecological Detective (2)
- ARM/Regression and Other Stories (3)

What makes these works so useful, strong, and enduring is the way that they articulate a coherent, opinionated perspective on the meaning and use of a set of methodological tools. On top of that, they are engaging and fun to read - the sort of thing you return to over time not just to get specific methodological tools, but to be exposed to their perspective.

References

Part I

Introduction

3 An invitation

One theme that comes up consistently when I'm teaching and talking to students and others about topics in relational epidemiology - particularly through my courses on spatial analysis - is that it has an aura of inaccessibility.

This makes a certain kind of sense: Making maps and measuring and modeling spatial relationships might seem like it is outside of the classic analytic toolkit of working epidemiologists, not to mention other fields in public health, medicine, and the social sciences. In fact, this was my take on it as well, until I realized space has always been at the heart of my research, even when I didn't realize it!

The diagram in Figure 3.1 reflects my interpretation of the way someone coming to this area of research and practice for the first time might see it:

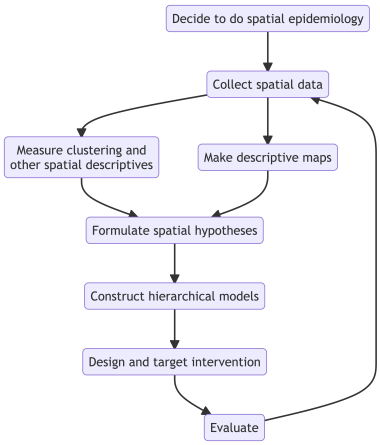


Figure 3.1: A textbook version of the spatial epidemiology workflow, without all the twists and turns that characterize actual research.

In this view, spatial epidemiology or analysis is taken to be a set of relatively fixed and well-described ideas and procedures connected to a highly technical set of methods. When we come from this perspective, starting a spatial project requires us to embark with a backpack that is already filled with specialized spatial tools.

3.1 You are (almost certainly) a relational analyst already

My goal in this short essay is to chip away a bit at the idea of spatial/relational epidemiology as something separate and apart from mainstream epidemiology and public health. Instead, I argue that these are better understood as a loose wrapper around a core set of ideas and tools that are part of the working arsenal of most professionals, students, and researchers in public health.

To explain what I mean, and why I think it's important, I'm going to subject you to a bit of autobiography about my own intellectual and professional trajectory.

3.2 Maps: Gateway or destination?

I began working as an epidemiologist or epidemology-adjacent type as a PhD student at the University of Michigan some time around 2006. As part of my dissertation research, I worked with Joe Eisenberg on an analysis ofthe role of social networks as sources of risk and protection against diarrheal disease and other infections in a group of villages in an area of rural Ecuador:

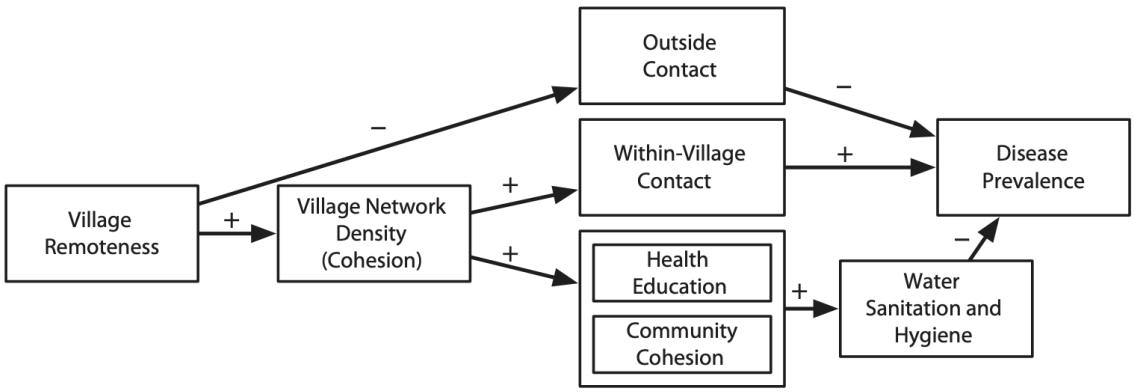


Figure 3.2: A figure from my dissertation research representing hypothesized relationships between village context (represented by inaccessibility or 'remoteness') and variation in disease outcomes within and between villages. (From (4))

Looking back, this was indisputably a spatial analysis: We were interested in how local social contexts impacted variation in health outcomes *across* a set of 20+ villages and also how within-village variation in social connectivity impacted risk *within* villages.

We employed multi-level data about common characteristics of individual villages, households, and the individuals within them. But at this time, I thought of myself as doing a few things, but none of them were spatial:

1. Infectious disease epidemiology: Why and how do people become *infected* with various pathogens?
2. Social epidemiology: How do social *relationships* impact disease outcomes?
3. Network analysis: How does the *structure* of relationships impact individual and community health?

As it happened, all of these things were correct. But what I didn't really understand at the time was that the collection of these different approaches into a single analysis made it spatial or geographic in nature, even if I didn't realize it

3.3 A process of progressive revelation

As a postdoc, working with Ted Cohen, I began analyzing data from a large study of household-level tuberculosis transmission in Lima, Peru. Figure 3.3 illustrates the model we developed to characterize household-level differences in transmission rates as a function of exposure type:

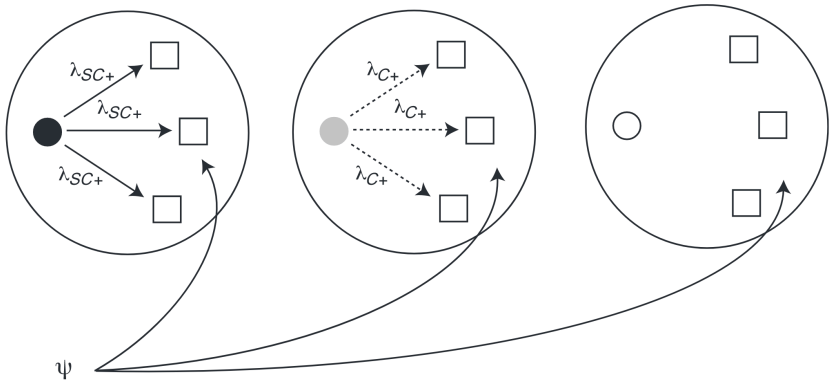


Figure 1. The figure illustrates the different sources of tuberculosis infection in the infection risk model. Smear-positive/culture-positive index cases (black circle) are hypothesized to be the most infectious, followed by smear-negative/culture-positive index cases (gray circle) and then smear-negative/culture-negative index cases (white circle). λ_{SC+} and λ_{C+} indicate the risk of infection for an uninfected household contact (white square) exposed to a smear-positive/culture-positive index case or smear-negative/culture-positive index case, respectively. ψ is the community risk of infection to which all households are subject. The solid arrows indicate a higher hypothesized level of infectiousness than values represented by the dashed arrows. Random intercepts are included in the model at the household and health center level to account for correlated responses within these units.

Figure 3.3: Characterizing household-level variation in risks of infection from community vs. household exposures (Figure from (5))

At the time, I knew that these households were distributed across neighborhoods of Lima, but I didn't give it much thought. I was more interested in risks experienced by an average

household. And to be honest, I didn't know that spatial metadata were available on each of the households, since I wasn't involved with the data collection!

In the interim, I got the chance to work on some collaborative projects with an explicitly spatial focus. In one, we reconstructed an outbreak of morbillivirus (think: measles) among a herd of migratory dolphins (Figure 3.4).

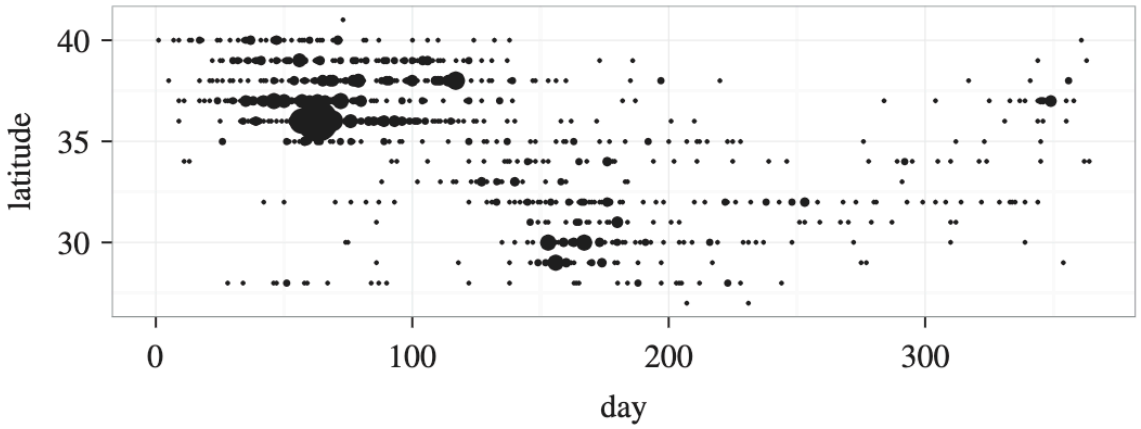


Figure 3.4: Locations of dolphin strandings during a morbillivirus outbreak in the North Atlantic (dot size indicates a greater number of strandings; Figure from (6))

In another, we looked at the relationship between environmental risks, such as neighborhood-level flooding, on the rate of pediatric diarrheal disease in Ho Chi Minh City in Vietnam (Figure 3.5).

These were the first experiences I had explicitly looking at these outcomes as a function of geographic space. While I had previously thought that mapping and spatial analysis and health geography were big scary things I couldn't do, I started to realize something important: These projects were not substantively very distinct from ones I had done before. The difference was that we were explicitly talking about spatial relationships and making maps (or simple one-dimensional diagrams as in Figure 3.4), instead of implicitly as in Figure 3.2 or Figure 3.3.

After completing these other projects, I dove back in to the Lima TB data to look at the drivers of multi-drug resistant TB (MDR-TB) risk. This was when I finally found out (some 2 years after I had started working with these data!) that spatial information on each household was available. So, with great trepidation, for the first time I made a map to explore spatial variability in MDR-TB outcomes.

And when I did this, we instantly saw that there were seemingly meaningful differences in the rate of TB overall, and MDR-TB in particular, across different health center catchment areas:

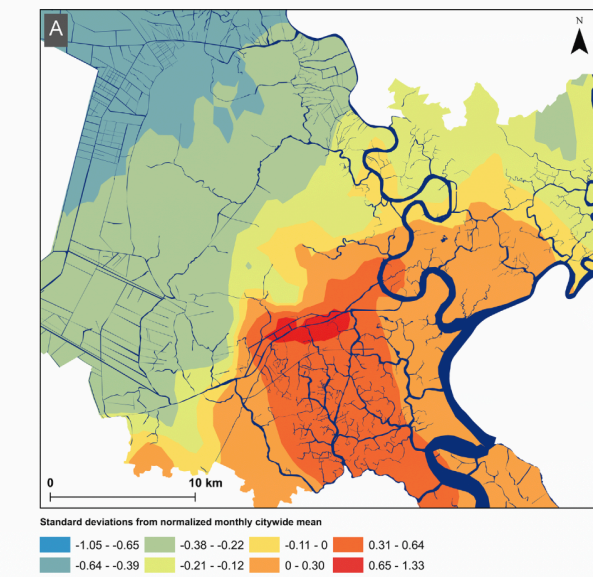


Figure 3.5: Incidence of pediatric diarrhea across neighborhoods of Ho Chi Minh City, Vietnam (Figure from (7))

This was the moment, some 10 years after I dipped my toes into the world of infectious disease epidemiology, where I realized I had been doing spatial work all along.

3.4 So what?

Why am I bothering you with this tedious and indulgent bit of personal history? *It's because it took me way too long to recognize that spatial epidemiology was a wrapper around a set of skills and ideas I had been working with for many years before I recognized what I was doing.* I was intimidated by anything preceded by 'spatial-': it sounded like a bunch of skills I didn't have and couldn't acquire.

My belated realization about the emergent quality of spatial epidemiology, and its broader connection to relational thinking in public health and the social sciences, has been crucially important for me. It made me realize that when I push into new areas - in life as much as research - that I probably have more of the tools I need than I realized in advance.

This means that you don't need to identify as a spatial or network or time series analyst to be one. And if you want to think of yourself as one, you should, because ultimately it is the intention to engage with the spatial, social and temporal relationships that drive health

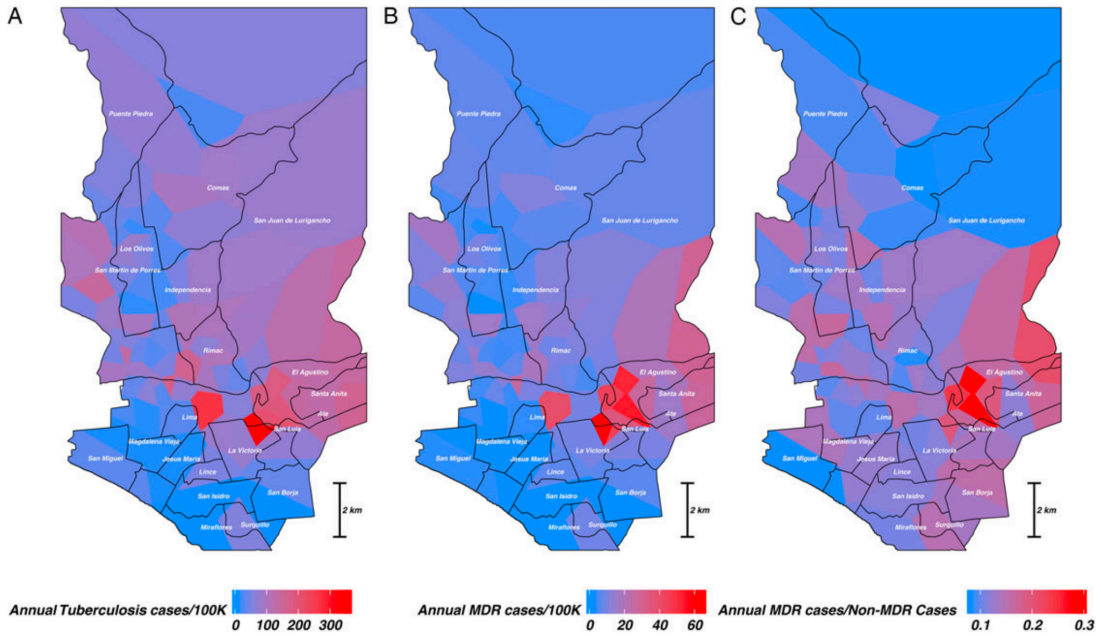


Figure 1. HC-level risks. Annual per-100 k rates of drug-sensitive and drug-resistant tuberculosis (*A*) and MDR tuberculosis (*B*), by HC catchment area. *C*, Ratio of the per-capita rate of MDR to non-MDR cases by HC. HC catchment areas are represented by polygons, with polygon fill color indicating the tuberculosis or MDR-tuberculosis rate in cases/100 K population. The boundaries of administrative districts of Lima are overlaid in black, and labeled in white**.** Abbreviations: HC, health center; MDR, multidrug-resistant.

Figure 3.6: The first map(s) I ever made, from (8), nearly 10 years after I started my research career.

outcomes that makes the difference. This is likely true for many if not most scientific subfields[1], but this one is mine and I'm glad I finally realized it!

References

[1]To be fair, you probably need to be working near-ish to a field for it to happen by chance: there is little chance of me taking on the characteristics a particle physicist or chemical engineer by chance, but I wouldn't rule out lepidopterist or archaeologist entirely!

4 What are the elements of relational epidemiology?

I arrived at the idea of relational epidemiology as an umbrella category from spatial and hierarchical analysis. The stereotypical challenge in this setting is to adjust away the impact of context to get at some more generally meaningful parameter estimate, e.g. a treatment effect. But another way of thinking about this is that the hard work is in characterizing not only the 'main' or fixed effects, but in capturing the drivers of variability in outcomes across locations.

But not every contextual question is strictly spatial: If we care about how the structure of social networks impacts individual and collective risks, we are talking about context once again. In the network example, the context is one's network 'neighborhood', the collection of individuals one interacts with. In reasonably homogeneous networks, groups of individuals may share very similar or almost identitical network contexts.

Ultimately, the key to understanding context is *relatedness*: Individuals sharing a context are likely to have similar relationships to their physical environment, the societies they are a part of, and potentially to each other, than those not sharing the same context. These relationships may be micro-level social relationships or macro-scale spatial ones, but they may also be temporal in nature. Temporal relationships may occur at a micro scale, e.g. the rise or fall in incidence of an infectious disease during a given week is likely to be a function of the prevalence of that same disease in the previous week. But these temporal relationships are also often more macro-scale and historical in nature: The long history of racial residential discrimination in the United States is undoubtedly a driver of many racial health dispartities we see today.

4.1 Problem-orientation is nonnegotiable

> "A common mistake that people make when trying to design something completely foolproof is to underestimate the ingenuity of complete fools." - Douglas Adams, Hitchhikers Guide to the Galaxy.

Sometimes, methods-y topics in epidemiology and public health are boiled down to a sequence of steps to be applied to each new problem. In the worst cases, they come to us as a series of

copy-paste, plug-and-chug pieces of code to be reused each time the same type of problem is encountered.

Beyond being a boring way to learn, this has the effect of putting the methods at the front of the train, with the question or problem implicitly assumed to be an opportunity or excuse to employ the model.

This is an approach that can get you some publications and maybe a little bit of clout within the musty world of academia, but it doesn't do much to solve the types of problems working epidemiologists face. And in truth, it doesn't even work so well on the academic side of the fence.

Inside the universe of this book, the problem is the first and most important thing. The question is the fixed point against which our analytic approaches are chosen. *This means that there are no fixed methodological answers to applied questions: Our methodological approaches and tools must be as diverse and heterodox as the questions the world throws at us.*

Making sense of the types of patterns we see in the real world requires us to first identify:

1. A question we want or need to answer.
2. The most important types of relationships impacting our outcome of interest (time, space, individual-to-individual).
3. A methodological approach that will allow us to characterize the impact of those relationships on the outcome we care about.

4.1.1 A methodological caboose

It is important to note that the choice of method comes *last* here: A key motivation of this book is to sidestep the tendency to train ourselves into methodological hammers looking for data nails to whack away at.

In addition to this, there is no pre-supposition that the appropriate method is necessarily 'fancy' in the sense of being conceptually or mathematically complex, computationally intensive, or even primarily or at all quantitative in nature. Whether this is the case is entirely a function of what happens at the intersection between question, data, and theory.

5 Some laws of relational and contextual epidemiology

Part II

Relationships

Part III

Space, time, and network

References

6 Social

6.1 Social Stratification & Inequality

6.2 Social Networks

7 Spatial

Please read the following two pieces that discuss key ideas about geospatial relatedness:

Miller HJ. Tobler's First Law and Spatial Analysis. *Annals of the Association of American Geographers*. 2004;94(2):284-289. doi:10.1111/j.1467-8306.2004.09402005.x

Goodchild MF. The Validity and Usefulness of Laws in Geographic Information Science and Geography. *Annals of the Association of American Geographers*. 2004;94(2):300-303. doi:10.1111/j.1467-8306.2004.09402008.x

7.1 Physical Space

7.2 Social Space

8 Temporal

8.1 Short-term correlation and fluctuation

8.2 Time series

8.3 History

Part IV

Methods

9 Codifying Tobler's First Law using Locally Weighted Regression

In this brief tutorial, we will review some basic ideas about smoothing and start thinking through how we can express these ideas mathematically and in R.

Tobler's profound - but deceptively simple - first law states that:

> "Everything is related to everything else. But near things are more related than distant things." (9)

He applied this idea to his development of a dynamic model of urban growth in the Detroit region which assumed that rates of population growth were spatially similar:

In this post, we're going to start with a simpler problem - change in the value of a function in one dimension - to see how we can translate the concept of *distance decay* implied by Tobler's first law (TFL) into a useful model. To begin, we're going to keep it simple with no noise or observation error and just interpolate some values.

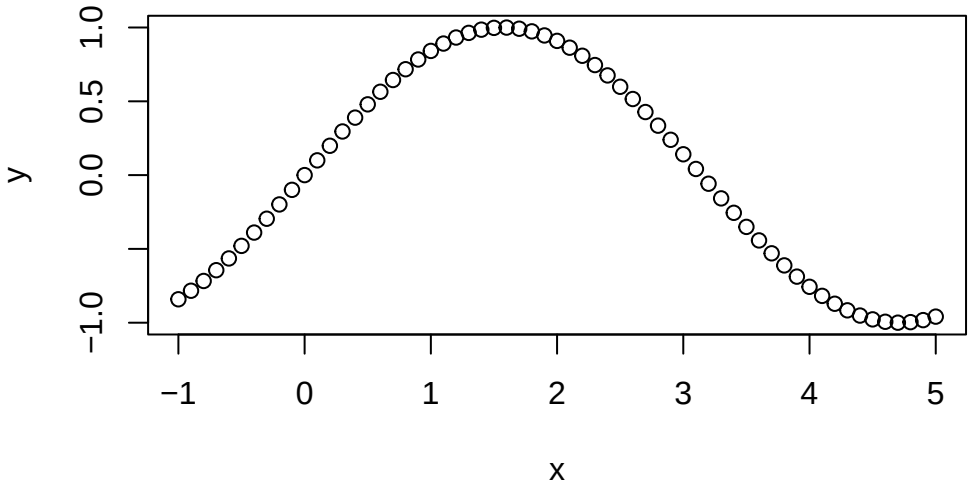
9.1 Notation and Terminology

In this example, we are interested in visualizing and predicting the values of a function $f(x_i)$ which outputs values y_i, the expected value of the output function.

$$y_i = f(x_i)$$

Lets start by getting the values of $f(x)$ for every input value x. For simplicity, we will assume that $f(x)$ is a sine function and that the values of x go from -1 to +5, allowing us to observe one half cycle of the sine function:

```
x <- seq(-1, 5, by = 0.1)
y <- sin(x)
plot(x,y)
```



You can see right away that this simple curve pretty neatly expresses Tobler's first law: $f(x)$ values of each point are in general more similar to each other for nearby values of x. If we want to press this idea into real-world practice, we need a *model* that can translate TFL into quantitative estimates and qualitative visualizations. There are lots of ways to do this, but we'll focus in on locally-weighted regression, also known LOWESS.

The basic idea of a LOWESS regression is to define a window of size k points around each value one wishes to estimate, and calculate a weighted average of the value of those points, which can then be used as the estimated value $\hat{y_j} \approx f(x_j)$. We then run the values of these nearest neighbors through a weight function $w(x)$.

These weight functions can take a lot of different forms, but we'll start simple with a uniform one, i.e. just taking the average of the k nearest neighbors, so that $\hat{y} = sum(z(x_i, k))/k$, where $KNNz$ is a function returning the y values of the k nearest observations to x_i. The value of k is sometimes referred to as the *bandwidth* of the smoothing function: Larger bandwidths use more data to estimate values at each point, smaller ones use less.

9.1.1 Making some locally weighted estimates

Using the `fnn` package for R, we can find the indices of the k nearest neighbors of each point we want to make an estimate at:

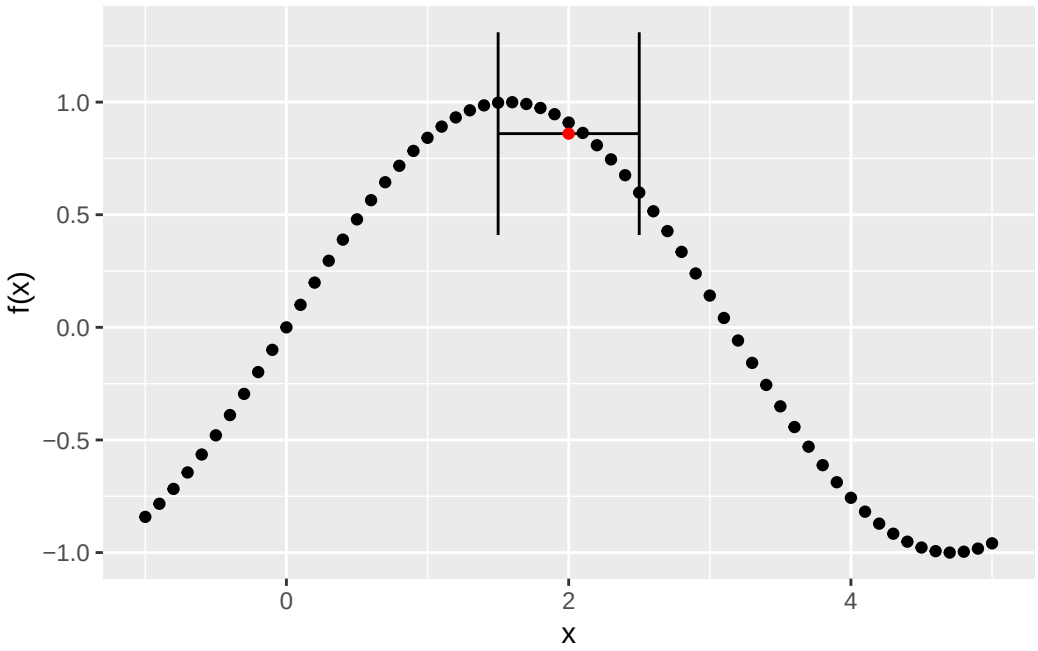
```
library(FNN)
k <- 10
z <- knn.index(x, k=k)
```

You can read the output of this function, below, as indicating the indices (i) of the 10 nearest points to each of the values of x.

```
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    2    3    4    5    6    7    8    9   10    11
 [2,]    1    3    4    5    6    7    8    9   10    11
 [3,]    2    4    1    5    6    7    8    9   10    11
 [4,]    5    3    6    2    1    7    8    9   10    11
 [5,]    6    4    7    3    8    2    1    9   10    11
 [6,]    5    7    4    8    3    9    2   10    1    11
 [7,]    8    6    9    5   10    4   11    3   12     2
 [8,]    7    9   10    6   11    5    4   12    3    13
 [9,]   10    8   11    7   12    6    5   13   14     4
[10,]    9   11    8   12    7   13   14    6    5    15
```

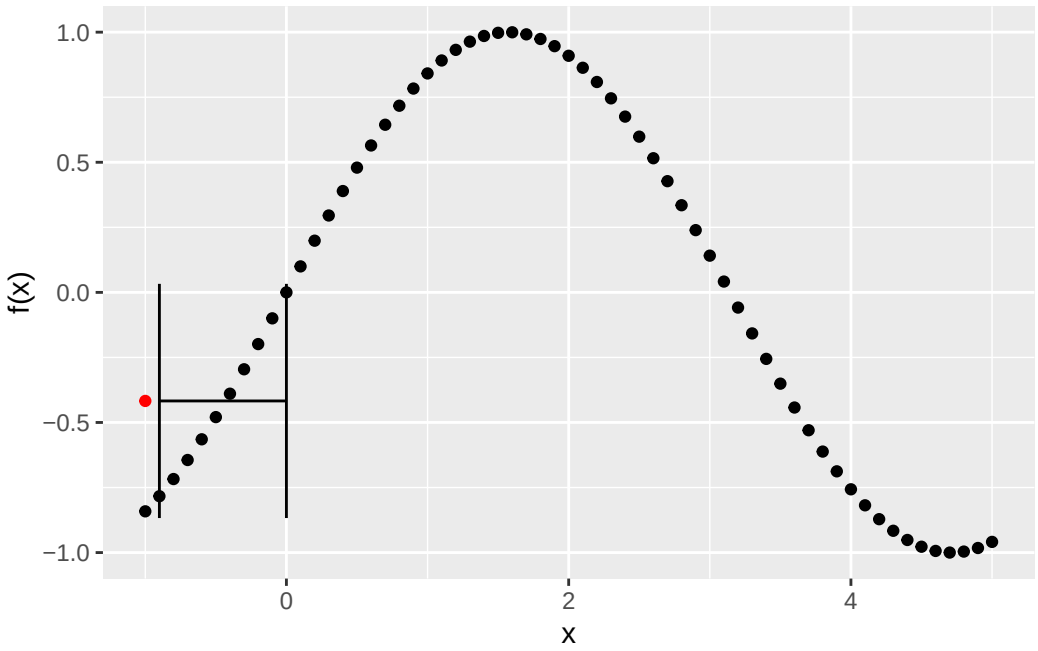
We can visualize this by picking a point in the middle of the series and its 10 nearest neighbors and the estimated value of $\hat{y}_i$ obtained by just taking the average of the k nearest points:

```
library(ggplot2)

## Plot the original data
g <- ggplot() + geom_point(aes(x=x, y = y)) +
  xlab("x") + ylab("f(x)")

## Now, get the index for x = 2
x_index <- which(x==2)

## Show the range of k nearest neighbors of this point
knn_low <- min(x[z[x_index, ]])
knn_high <- max(x[z[x_index, ]])
y_hat <- mean(y[z[x_index,]], )

## Add errorbars to the figure to show the 10 nearest values with the height of the point
g <- g + geom_errorbarh(aes(xmin = knn_low, xmax = knn_high, y = y_hat)) + geom_point(aes(

plot(g)
```



Notice that if the `knn` function is applied at the low end of the series, i.e. to the first value, it will use points to the right of that one instead of to either side:

```
library(ggplot2)

## Plot the original data
g <- ggplot() + geom_point(aes(x=x, y = y)) +
  xlab("x") + ylab("f(x)")

## Use the index for the lowest value
x_index <- 1

## Show the range of k nearest neighbors of this point
knn_low <- min(x[z[x_index, ]])
knn_high <- max(x[z[x_index, ]])
y_hat <- mean(y[z[x_index,]], )

## Add errorbars to the figure to show the 10 nearest values with the height of the point
g <- g + geom_errorbarh(aes(xmin = knn_low, xmax = knn_high, y = y_hat)) + geom_point(aes(

plot(g)
```

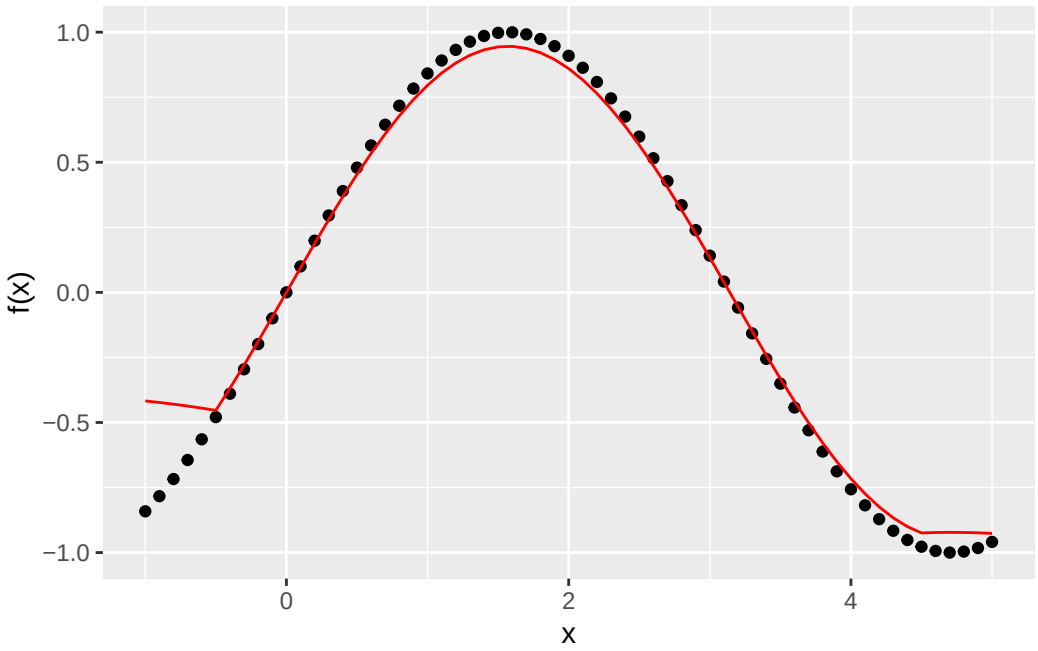


Now, lets see what happens if we run our smoother over the whole series and take the average of the 10 nearest points for each and compare them to the observed data:

```
y_hat <- rep(0, length(x))
for (i in 1:length(x) ) {
  y_hat[i] <- mean(y[z[i,]], )
}
```

Now plot the predicted vs. the observed values:

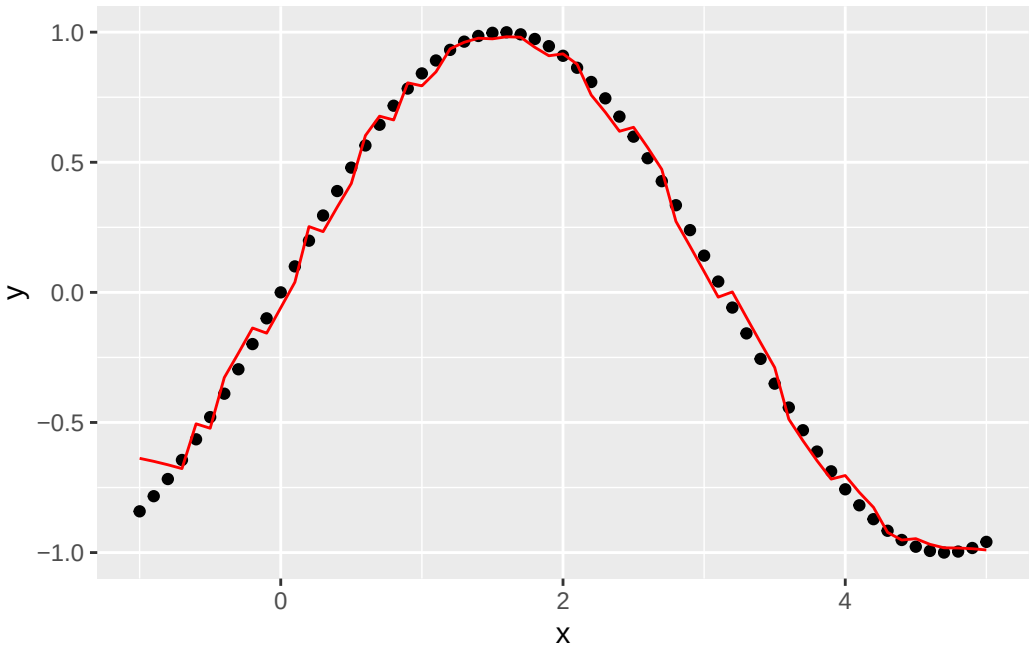
```
g <- ggplot() + geom_point(aes(x=x, y = y)) +
  xlab("x") + ylab("f(x)") + geom_line(aes(x=x, y=y_hat), colour = "red")
plot(g)
```



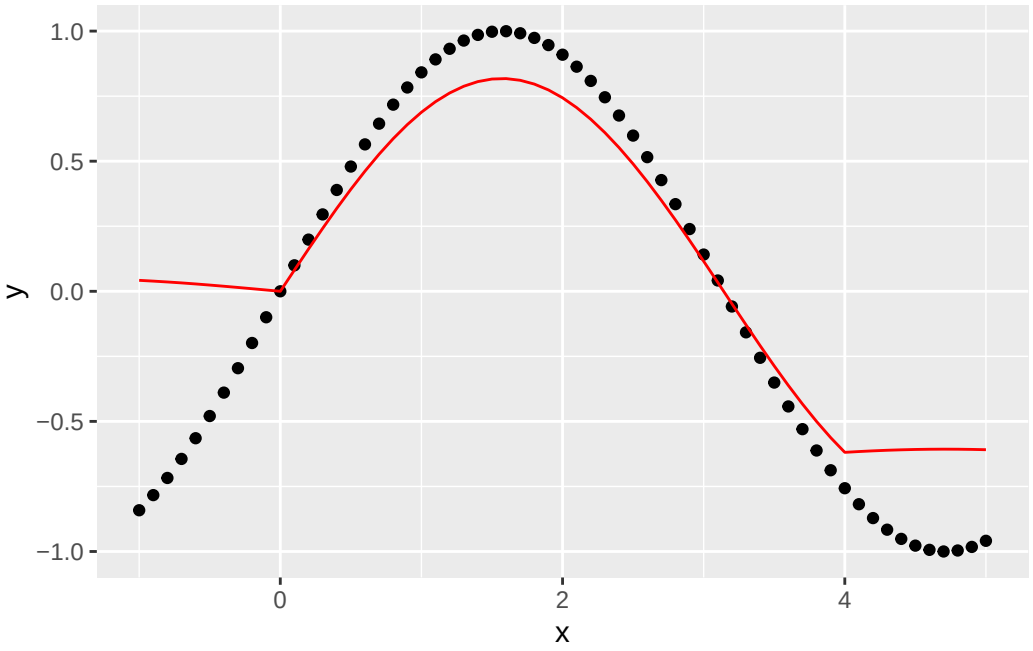
You can see this does a pretty good job all the way through, except at the edges. Lets try it again with a smaller window - or bandwidth - of 5 and see what happens. First, we'll write a function that will give us the predicted value of y at each point given a window of size k and an input value:

```
knn_est <- function(x, y, k) {
  z <- knn.index(x, k=k)
  y_hat <- rep(0, length(x))

for (i in 1:length(x) ) {
  y_hat[i] <- mean(y[z[i,]], )
}

  df <- data.frame(x=x, y = y, yhat = y_hat)
  return(df)
}
```

```
pred_df <- knn_est(x, y, 5)
g <- ggplot(pred_df) + geom_point(aes(x=x, y=y)) + geom_line(aes(x=x,y=yhat),colour="red")
plot(g)
```

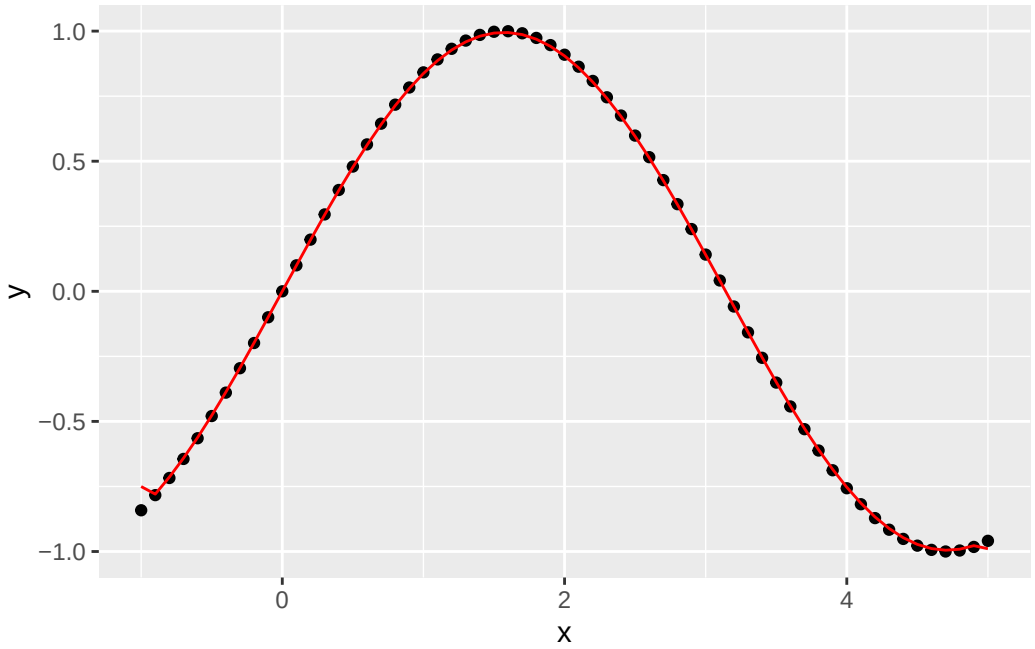


This gets rid of a lot of the weird effects at the edges but introduces some noise into the function. What if we make the window bigger, say 20, to get rid of some of the noise?



This seems to make the edge effects worse, as well as the estimates of the function overall worse.

What happens if we go in the opposite direction and shrink the window down to 2?



9.1.2 Discussion Questions

- Why does this appear to be more accurate for these data than $k = 10$ and $k = 5$?
- What would happen if we added observation noise to the values of y_i? Which one of the smoothers do you think would work better then?
- Is the one *best* value of k for all datasets? How might you go about picking the best one?
- How does our uniform weight function express Tobler's first law? What kind of weight function $w(x)$ might do a better job of capturing the notion of distance decay?

9.2 References

10 Spatial Density

When working with spatial data, our analytic task often falls into either of two rough categories.

1. *Estimating local averages of a continuous variable.* This could be something like neighborhood-by-neighborhood variation in average blood pressure or the intensity of some kind of environmental risk factor such as the intensity of fine (e.g. $PM_{2.5}$) dust which can cause respiratory illness.
2. *Estimating the local density of or incidence of a particular outcome.* For example, if we are trying to understand spatial variation in the incidence of a particular disease, we are interested in knowing how many cases of that disease are present in a given location or are likely to be present in some nearby unobserved location.

In this tutorial, we'll dig into the problem of spatial density estimation in one dimension along a spatial *transect*. The techniques discussed here form the basis of a number of approaches for cluster or hotspot analysis. For examples of smoothing local values of a continuous variable, see Chapter 9.

10.1 A motivating example

A spatial transect is an area of space along a line crossing a landscape. These are often used in ecology and forestry to assess the health of an environment, species diversity and other factors. Using a transect can help simplify the problem of spatial analysis down to one dimension rather than the usual two, while still providing a tremendous amount of useful information.

For example, (10) were interested in characterizing the intensity of exposure to triatomine bugs and other insect vectors of the pathogen *T. cruzi*, which causes Chagas disease.

Imagine we are estimating the density of some unknown insect vector along a 1 kilometer transect with the goal of characterizing the risk of infection with a vector-borne illness.

Figure 10.1: Example of an ecological transect from the US National Park Service (source)

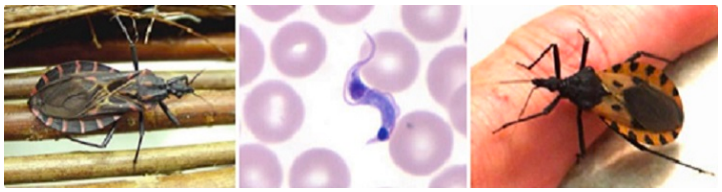


Figure 10.2: *Triatoma* (left- and right-hand panels) and *T. cruzi* (center) (source)

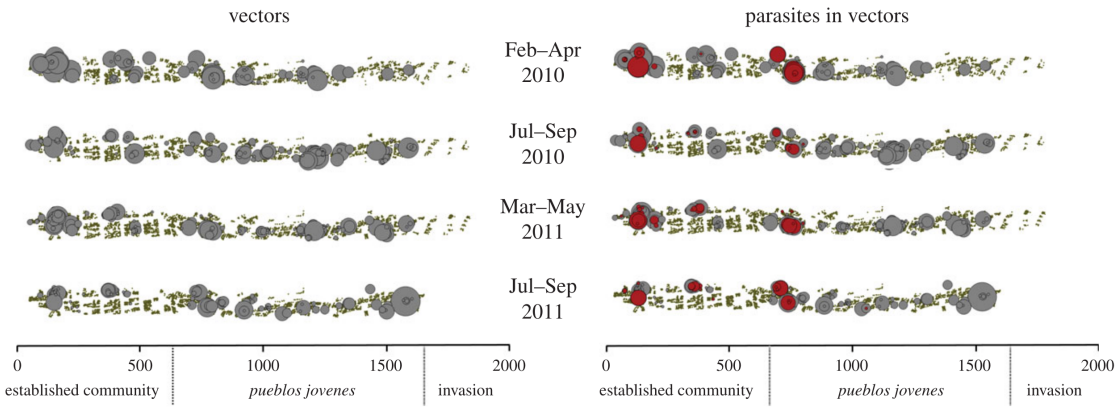


Figure 10.3: Intensity of *Triatomine* infestation along a 2km transect in Arequipa, Peru (Figure from (10))

10.2 Kernel density estimation in one dimension

Much like in our discussion of kernel smoothing of continuous outcomes, kernel functions play a key role in this setting as well. In this case, imagine that the locations of vectors along our transect have been sampled at random from some unknown function $f(x)$ which takes values from 0 (the beginning of the transect) to 1000m (the end).

We can use the Kernel function $K(d)$ to approximate the intensity of the outcome of interest at each observed case location x_i. Imagine that our observed data have locations $x_1, x_2, \ldots, x_n$ and that the distance between our point of interest, x_j and each observed point is $d_{ij} = |x_j - x_i|$.

Finally, lets include a bandwidth parameter, h, which controls the width of the window we will use for smoothing. When we put this all together, we can get an estimate of the density of our outcome of interest at location x_j as follows:

$$\hat{f}_h(x_j) = \frac{1}{n}\sum_{i=1}^{n} K(\frac{x_j - x_i}{h})$$

As you can see below, we can pick a range of kernel functions, but for the sake of simplicity, in this example, we will focus in on a Gaussian, or normal, kernel, which uses the probability density function of a normal distribution to weight points.

Lets start by sampling locations of observed points along a one dimensional line. To keep things interesting, we'll use a Gaussian mixture distribution with two components:

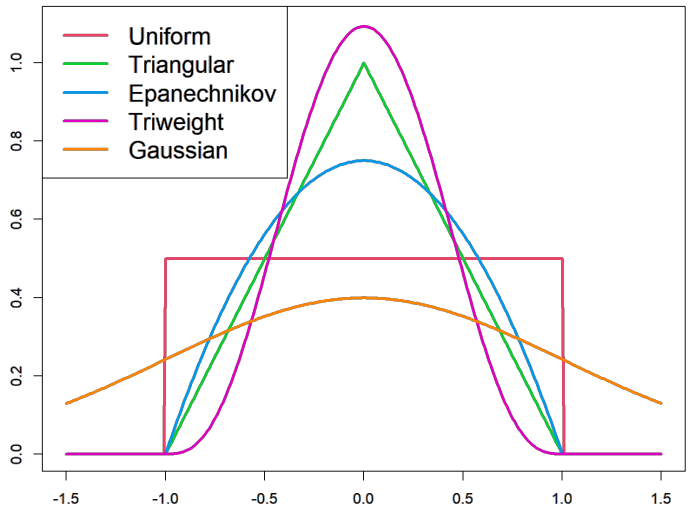


Figure 10.4: Comparison of different kernel functions (source)

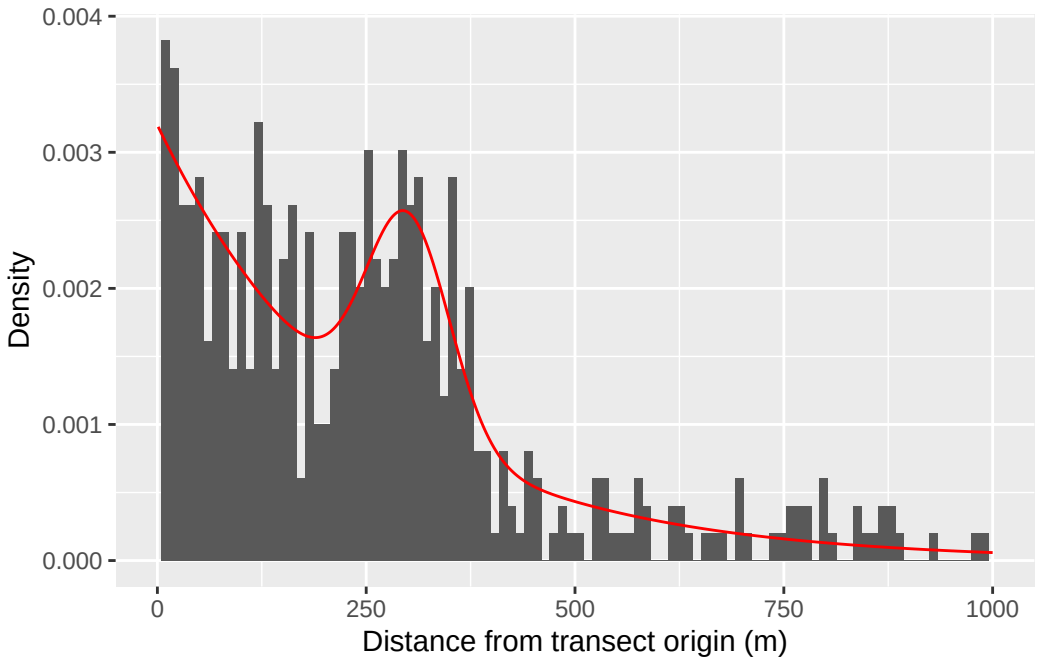
10.3 Worked example

First, lets imagine a scenario in which the risk of observing an insect vector steadily decreases as we walk along our transect. However, along the way there is a *hotspot* of increased risk beyond what we would expect from the smooth decline before and after that spot. For the purpose of this example, we'll assume that risk decays *exponentially* with distance from the origin, but that our hotspot is centered at a point 300 meters into the transect. The code below lets us sample the *locations* of the points along the transect where are observed from two distributions:

1. An exponential distribution representing smooth decay from the beginning to the end of the transect, and
2. A normal distribution representing a hotspot about 150m in width beginning 300m in

The figure below shows a histogram of locations sampled from $f(x)$ (vertical bars) overlaid with the true value of $f(x)$ in red:

```
library(ggplot2)
d_a <- dexp(1:1000, rate = 1/250)
d_b <- dnorm(1:1000, mean = 300, sd = 50)
y <- ((1-p_hot))*d_a + (p_hot*d_b)

dens_df <- data.frame(x = 1:1000, y = y)
xdf <- data.frame(x=x)


g <- ggplot(xdf) + geom_histogram(aes(x=x, y=..density..), bins=100) +
geom_line(data=dens_df, aes(x=x,y=y), colour="red") +
xlim(0, 1000) + ylab("Density") + xlab("Distance from transect origin (m)")
plot(g)
```



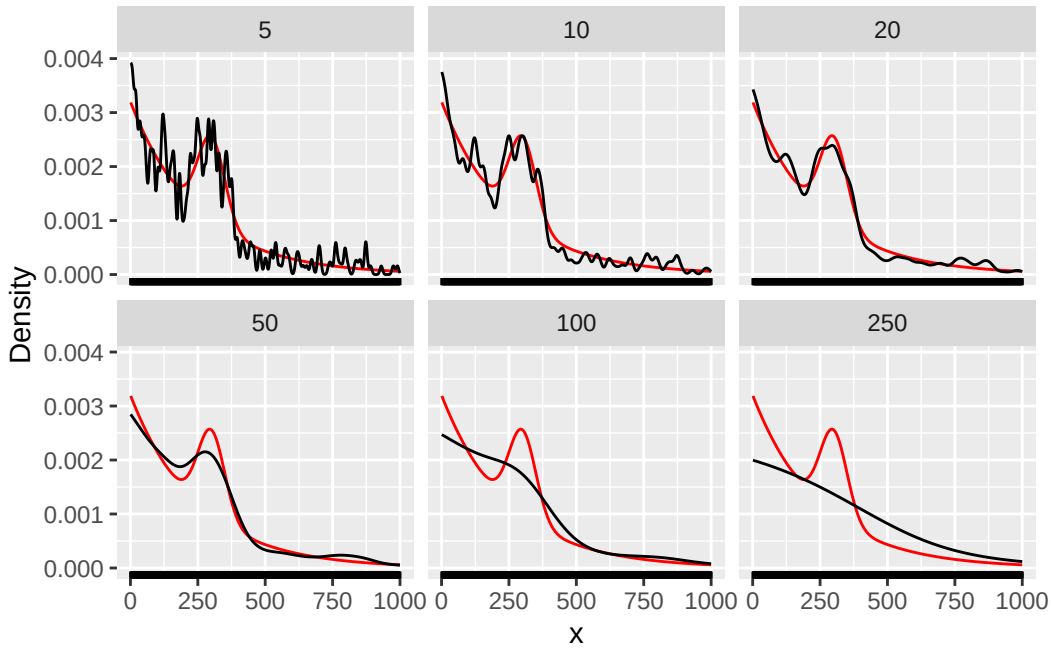
Now, imagine we have another set of finely spaced points along the line, and for each, we want to calculate the weight for each. The function below lets us do that:

The figure below shows the true value of our density function $f(x)$ in red, the density of points in the simulated data along the x-axis of the 'rug plot', and our smoothed density in black, for a bandwidth of $h = 10$:

```
library(ggplot2)
pred_df <- normal_smoother(x, h = 10)

g <- ggplot() + geom_rug(aes(x=x)) +
geom_line(data = pred_df, aes(x=x, y=y)) +
ylab("Density") + geom_line(data = dens_df, aes(x=x,y=y), colour="red") +
xlim(0, 1000)
dens_ojs <- dens_df
dens_ojs$y <- dens_ojs$y*cc
plot(g)
```

Now, lets see what happens if we try this for different values of h:

```
all_df <- data.frame()
for (hv in c(5, 10, 20, 50 ,100, 250)) {
  pred_df <- normal_smoother(x, h = hv)
  pred_df$h <- hv
  all_df <- rbind(all_df, pred_df)
}

  all_df$h <- as.factor(all_df$h)

g <- ggplot(all_df) + geom_rug(aes(x=x)) +
geom_line(data = dens_df, aes(x=x,y=y), colour="red") +
geom_line(aes(x=x, y=y)) +
ylab("Density") +
facet_wrap(~ h) +
xlim(0, 1000)

plot(g)
```




```
all_df <- data.frame()
hvals <- seq(1, 100, by = 2)
distvals <- seq(-100, 100, by = 1)
all_kernvals <- data.frame()
for (hv in hvals) {
  pred_df <- normal_smoother(x, h = hv)
  pred_df$h <- hv
  pred_df$smoother <- "gaussian"
  all_df <- rbind(all_df, pred_df)
  all_kernvals <- rbind(all_kernvals,data.frame(x=distvals, y=kdgaussian(distvals, bw = hv

  pred_df <- normal_smoother(x, h = hv, kern = kduniform, kernp=kpuniform)
  pred_df$h <- hv
  pred_df$smoother <- "uniform"
  all_df <- rbind(all_df, pred_df)
  all_kernvals <- rbind(all_kernvals,data.frame(x=distvals, y=kduniform(distvals, bw = hv)


  pred_df <- normal_smoother(x, h = hv, kern = kdtricube, kernp=kptricube)
  pred_df$h <- hv
  pred_df$smoother <- "tricube"
  all_df <- rbind(all_df, pred_df)
```

```
    all_kernvals <- rbind(all_kernvals, data.frame(x=distvals, y=kdtricube(distvals, bw =

  pred_df <- normal_smoother(x, h = hv, kern = kdtriangular, kernp=kptriangular)
  pred_df$h <- hv
  pred_df$smoother <- "triangular"
  all_df <- rbind(all_df, pred_df)
    all_kernvals <- rbind(all_kernvals, data.frame(x=distvals, y=kdtriangular(distvals, bw

}
all_df$y <- all_df$y * cc
```

10.4 Trying different bandwidths and kernels

You can adjust the range of the bandwidth here to get a better sense of the relationship between the smoothed curve (black) and true density (red). Adjust the bin width for the histogram of the underlying data to get a sense of the fit of the model to the underlying data.

```
// |echo: false
viewof h = Inputs.range([1, 100], {value: 10, step: 2, label: "Bandwidth (m)"})
viewof bw = Inputs.range([5, 100], {value: 10, step: 5, label: "Bin width (m)"})
viewof kern = Inputs.select(["gaussian", "uniform", "tricube", "triangular"], {value: "gau

numbins = Math.floor(1000/bw)

dtrans = transpose(hvals)
Plot.plot({
y: {grid: true,
label: "Density"},
x: {
label: "Distance from transect start (m) →"
},
    marks: [
    Plot.rectY(transpose(sample),Plot.binX({y: "count"}, {x: "loc", fill: "steelblue", thr
    Plot.lineY(dtrans, {filter: d => (d.h == h) && (d.smoother == kern), curve: "linear",
    Plot.lineY(transpose(dens), {x:"x", y: d => d.y * bw, curve:"linear", stroke: "red"})
    ]
})
```

The figure below shows the relative amount of weight placed on different points as a function of their distance from the point of interest (0, marked by the vertical red line):

```
// |echo: false
kv = transpose(kernvals).filter(d => d.h == h && d.smoother == kern)
Plot.plot({
y: {grid: true, label: "Relative weight of point as compared to origin"},
x: {
label: "Distance from point of interest (m)   "
},
marks: [
//Plot.lineY(kv, {filter: d => (d.smoother == kern), x:"x", y: d => d.y*1000}),
Plot.lineY(kv, Plot.normalizeY({x:"x", y: "y", basis: "extent"})),
Plot.ruleX([0], {stroke: "red"})
]})
```

10.4.1 Questions

- Which of the bandwidth options seems to do the best job in capturing the value of $f(x)$? Why?
- How does the choice of kernel impact the smoothing?
- How do the different kernel functions encode different assumptions about *distance decay*?
- What is the relationship between the histogram of the data and the smoother? What do you see as you change the histogram bin width relative to the smoothing bandwidth?

10.5 Additional Resources

Please see Matthew Conlen's excellent interactive KDE tutorial

10.6 References

11 Household Radon

In this tutorial, we are going to replicate the analysis of household-level variation in radon exposure originally presented in (11) (which is actually a tutorial version of (12)). Our goal is to run the models described in the paper using regression models from base `R` as well as a Bayesian hierarchical model from the `rstanarm` package. Finally, we will reproduce Figures 1 & 2 from the original paper using `ggplot2`:

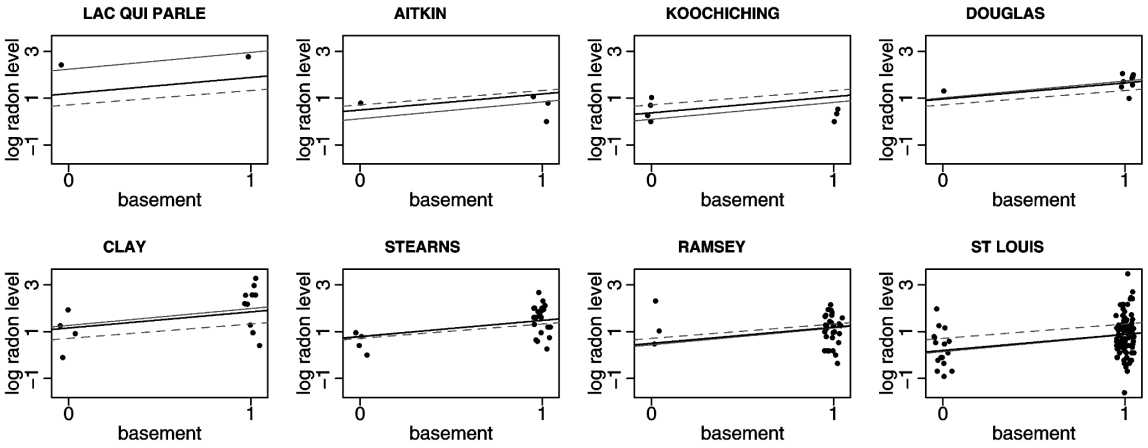


Figure 1. Multilevel (partial pooling) Regression Lines $y = a_j + \beta x$ Fit to Radon Data From Minnesota, Displayed for Eight Counties j With a Range of Sample Sizes. Light-colored dotted and solid lines show the complete-pooling and no-pooling estimates. The x-positions of the points are jittered slightly to improve visibility.

Figure 11.1: Original Fig 1 from (11)

11.1 Fitting the models

11.1.1 Setting up the workspace

First, we will load the relevant packages:

```
library(ggplot2)
library(tidyr)
library(dplyr)
```

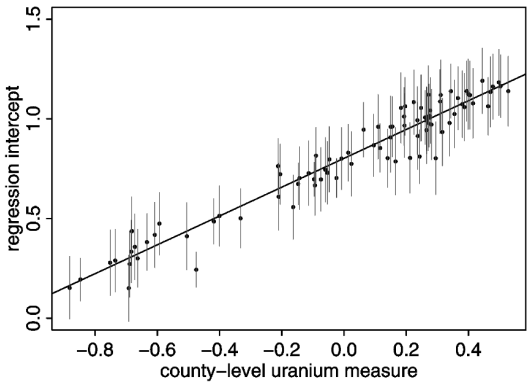


Figure 11.2: Original Fig 2 from (11)

```
library(bayesplot)
library(rstanarm)
library(purrr)
library(tidybayes)
```

11.1.2 Data Preparation

First, lets take the raw `radon` dataset from the `rstanarm` package and recode the `floor` variable to be interpretable as the `basement` one from the original paper: some minor modifications and additonal datasets that we'll use for the purposes of modeling and visualizing these data.

```
radon$basement <- 1 - radon$floor
```

Now we can see that that the dataset has all of the variables we need:

```
  floor county  log_radon log_uranium basement
1     1 AITKIN 0.83290912  -0.6890476        0
2     0 AITKIN 0.83290912  -0.6890476        1
3     0 AITKIN 1.09861229  -0.6890476        1
4     0 AITKIN 0.09531018  -0.6890476        1
5     0  ANOKA 1.16315081  -0.8473129        1
6     0  ANOKA 0.95551145  -0.8473129        1
```

11.1.3 Door 1: Full pooling!

This corresponds to a model in which we are assuming exactly no variation across locations in terms of the baseline level of radon. So, we can run a simple regression model where we assume that:

$$y_{ij} = \alpha + \beta x_{ij} + \epsilon_i$$

Where $x_{ij} = 1$ if a house has a basement and 0 otherwise.

In R, we can fit this model via least squares using a single line of code:

```
m1 <-lm(log_radon ~ basement, data = radon)
```

We can call the `summary` function to get a description of the key coefficients and the goodness-of-fit:

```
lm(formula = log_radon ~ basement, data = radon)
            coef.est coef.se
(Intercept) 0.78     0.06
basement    0.59     0.07
---
n = 919, k = 2
residual sd = 0.79, R-Squared = 0.07
```

11.1.4 Door 2: No pooling

The second approach is the "No Pooling" one in which we allow the baseline intensity of radon in each county (represented by the intercept term α_j) to vary, but we don't do anything to constrain that variation. In other words, we treat each county as though it was independent.

However, to estimate a consistent effect of having a basement across all counties, we estimate a single β term. This leads to a model that looks like this:

$$y_{ij} = \alpha_j + \beta x_{ij} + \epsilon_i$$

In `R` this is easy to implement, because we are implicitly asking the regression model to treat county as a categorical variable if we pass it to it as a `factor` datatype:

```
no_pool_m <- lm(log_radon ~ basement + log_uranium + county, data = radon)
```

```
lm(formula = log_radon ~ basement + log_uranium + county, data = radon)
                     coef.est coef.se
(Intercept)           0.42     0.37
basement              0.69     0.07
log_uranium           0.32     0.60
countyANOKA           0.09     0.44
countyBECKER          0.48     0.53
countyBELTRAMI        0.67     0.43
countyBENTON          0.39     0.48
countyBIGSTONE        0.31     0.68
countyBLUEEARTH       0.83     0.51
countyBROWN           0.80     0.60
countyCARLTON         0.05     0.38
countyCARVER          0.43     0.50
countyCASS            0.52     0.47
countyCHIPPEWA        0.56     0.60
countyCHISAGO         0.21     0.48
countyCLAY            0.79     0.54
countyCLEARWATER      0.28     0.50
countyCOOK           -0.23     0.60
countyCOTTONWOOD      0.06     0.63
countyCROWWING        0.26     0.40
countyDAKOTA          0.27     0.36
countyDODGE           0.63     0.63
countyDOUGLAS         0.60     0.49
countyFARIBAULT      -0.42     0.57
countyFILLMORE        0.18     0.75
countyFREEBORN        0.94     0.51
countyGOODHUE         0.80     0.48
countyHENNEPIN        0.32     0.34
countyHOUSTON         0.52     0.66
countyHUBBARD         0.30     0.44
countyISANTI          0.22     0.57
countyITASCA          0.07     0.42
countyJACKSON         0.83     0.59
countyKANABEC         0.18     0.50
countyKANDIYOHI       0.94     0.54
countyKITTSON         0.51     0.55
countyKOOCHICHING     0.04     0.52
countyLACQUIPARLE     1.75     0.71
countyLAKE           -0.40     0.44
countyLAKEOFTHEWOODS  0.99     0.51
countyLESUEUR         0.60     0.55
```

countyLINCOLN	1.08	0.67
countyLYON	0.75	0.59
countyMAHNOMEN	0.23	0.84
countyMARSHALL	0.53	0.44
countyMARTIN	-0.05	0.51
countyMCLEOD	0.17	0.46
countyMEEKER	0.13	0.49
countyMILLELACS	-0.07	0.60
countyMORRISON	0.11	0.41
countyMOWER	0.54	0.51
countyMURRAY	1.27	0.90
countyNICOLLET	0.99	0.59
countyNOBLES	0.71	0.68
countyNORMAN	0.10	0.63
countyOLMSTED	0.16	0.48
countyOTTERTAIL	0.60	0.40
countyPENNINGTON	0.11	0.54
countyPINE	-0.24	0.43
countyPIPESTONE	0.62	0.68
countyPOLK	0.55	0.60
countyPOPE	0.11	0.70
countyRAMSEY	0.22	0.33
countyREDWOOD	0.78	0.61
countyRENVILLE	0.46	0.67
countyRICE	0.70	0.49
countyROCK	0.06	0.79
countyROSEAU	0.64	0.36
countySCOTT	0.70	0.43
countySHERBURNE	0.24	0.44
countySIBLEY	0.10	0.58
countySTLOUIS	-0.03	0.31
countySTEARNS	0.38	0.43
countySTEELE	0.41	0.53
countySTEVENS	0.56	0.77
countySWIFT	-0.18	0.61
countyTODD	0.65	0.54
countyTRAVERSE	0.76	0.69
countyWABASHA	0.69	0.50
countyWADENA	0.43	0.48
countyWASECA	-0.47	0.58
countyWASHINGTON	0.31	0.34
countyWATONWAN	1.54	0.60
countyWILKIN	1.06	0.86

```
countyWINONA          0.41     0.60
countyWRIGHT          0.59     0.39
---
n = 919, k = 86
residual sd = 0.73, R-Squared = 0.29
```

11.1.5 Door 3: Partial Pooling

Finally, we get to the partial pooling, hierarchical model in which we introduce a *hierarchical prior* to the model to allow our model to shrink observations from places with few observations towards the population mean. This allows us to avoid the pitfalls of overfitting associated with the no-pooling approach while not making the homogeneity assumptions associated with the full-pooling approach.

This works out to a *multi-level* model that allows random variation in household-level radon measurements as well as variation at the county level in radon levels above or below the amount predicted by the county-level soil uranium measure. Much like the no-pooling model, we can write outcomes for *individuals* as:

$$y_{ij} = \alpha_j + \beta x_{ij} + \epsilon_i$$

However, rather than stopping there, we introduce a second level of random variation to the county-level *intercepts*, α_j.

$$\alpha_j = \gamma_0 + \gamma \zeta_j + \epsilon_j$$

Where $\epsilon_i \sim N(0, \sigma_i)$ and $\epsilon_j \sim N(0, \sigma_j)$.

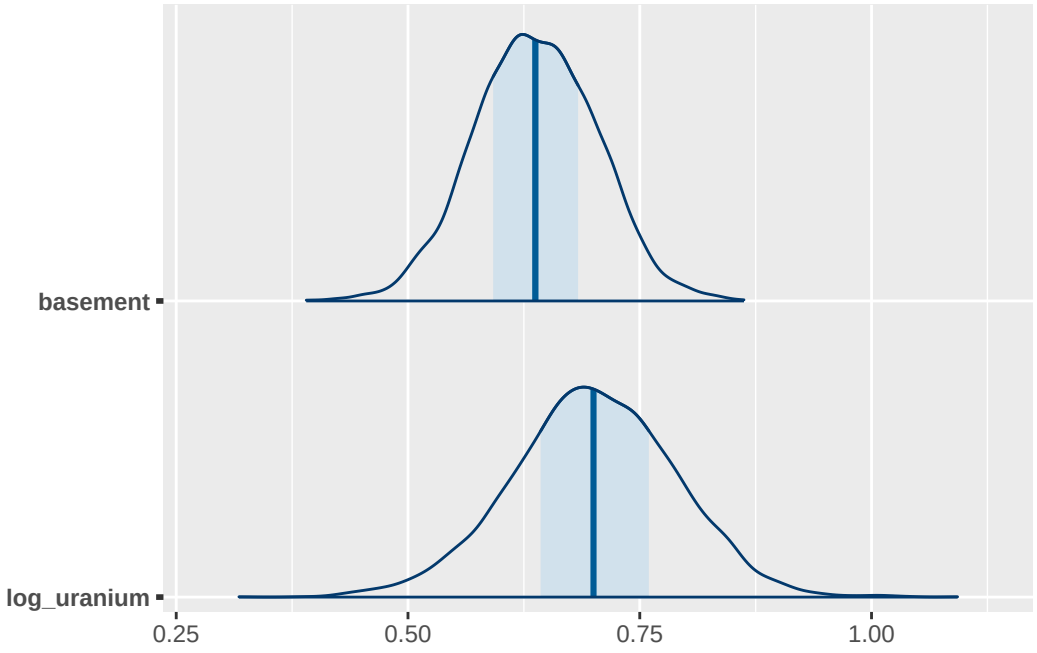
To fit this model, we'll use the `rstanarm` package, which uses the Stan Bayesian modeling language under the hook to fit the model. This model introduces another piece of syntax to our equation, which now reads `log_radon ~ basement + log_uranium + (1 | county)`. The interesting part of this is the `(1 | county)` which is a syntax used by `rstanarm` and other hierarchical modeling packages (such as `lme4`) to specify random intercepts (typically represented by a 1 in the matrix of regressors) for each of a set of clusters, in this case counties. In this model, the county-level intercept terms are implicitly assumed to be normally distributed with unknown variance σ_j which will be estimated when the model is fit.

We use the `stan_lmer` function to fit a hierarchical linear model with a normally-distributed response variable, as follows:

```
m2 <- stan_lmer(log_radon ~ basement + log_uranium + (1 | county), data = radon)
```

Because this model is fit by MCMC, we can use draws from the posterior distribution to understand uncertainty in the model. For example, this visualization of the median prediction and credible intervals for the basement and uranium effects can be visualized using the `mcmc_areas` function from the `bayesplot` package:

```
posterior <- as.matrix(m2)
g2 <- mcmc_areas(posterior, pars = c("basement", "log_uranium"))
plot(g2)
```



11.2 Making the Figures

11.2.1 Figure 1

Data Preparation

Since each row of `radon` dataset includes an observation of a single house, we need to work backwards to obtain the county-level soil uranium measure for each individual county. This is pretty straightforward to do using the `dplyr` package:

```
county_uranium <- radon %>%
  group_by(county) %>%
```

```
    summarize(log_uranium = first(log_uranium))
```

We will also make a second dataset that we will use for storing the predicted radon levels for households with and without basements each for county. This contains 2 entries for each county, representing observations taken in the basement or on the first floor.

```
county_uranium_tmp_1 <- county_uranium
county_uranium_tmp_1$basement <- 1
county_uranium_tmp_2 <- county_uranium
county_uranium_tmp_2$basement <- 0

county_dummy_df <- rbind(county_uranium_tmp_1, county_uranium_tmp_2)
```

Now, we will take each of our fitted models (fully pooled, unpooled and partially pooled) and put their predicted values into our plotting dataset

```
county_dummy_df$pooled_pred <- predict(m1, county_dummy_df)
county_dummy_df$no_pool_pred <- predict(no_pool_m, county_dummy_df)
```

```
Warning in predict.lm(no_pool_m, county_dummy_df): prediction from a rank-
deficient fit may be misleading
```

Because the partial pooling model was fit using MCMC, we will take a slightly different approach and use the median of the posterior predictive distribution for each observation, which is analogous to (but not exactly the same as) the OLS predictions from the other models:

```
## Gives posterior median for each prediction.
county_dummy_df$partial_pred <- posterior_predict(m2, county_dummy_df) %>%
  apply(2,median)
```

11.2.2 Plotting

To re-create Figure 1, we will subset out the observed data and predictions for the 8 counties included in the original figure:

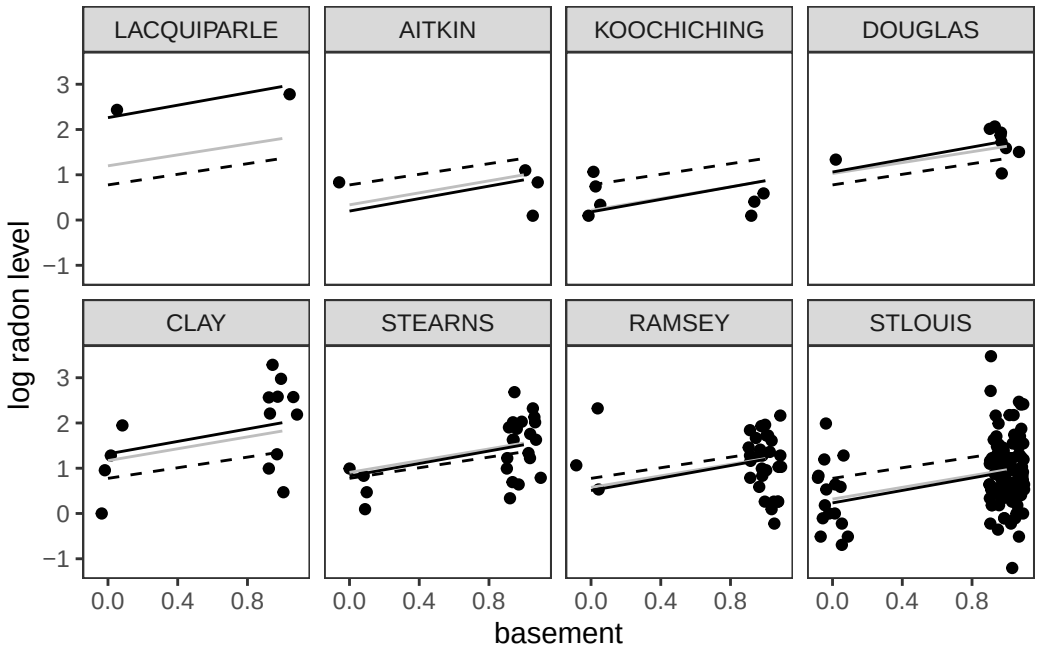
```
## Place the county names in a vector we will use to keep track of them
fig_1_counties <-
  c(
    "LACQUIPARLE",
    "AITKIN",
```

```
    "KOOCHICHING",
    "DOUGLAS",
    "CLAY",
    "STEARNS",
    "RAMSEY",
    "STLOUIS"
  )


# First, using the `county_dummy_df` with the basement/non-basement predictions in it,
# subset out the relevant counties and make a new county factor variable which
# will be used to ensure that the counties in Fig. 1 plot in the right order

county_df_fig_1 <- county_dummy_df %>%
  filter(county %in% fig_1_counties) %>%
  mutate(county2 = factor(county, levels = fig_1_counties)) %>%
  arrange(county)

## Now select out the households in the original data that
## are in each county and create another county-level factor
## variable in the same order

pred_counties <- radon %>% filter(county %in% fig_1_counties) %>%
  mutate(county2 = factor(county, levels = fig_1_counties))
```

Once we have the datasets together for the figure, we can begin constructing it using `ggplot2`:

```
g <- ggplot() +
  ## The geom_jitter geom plots the log_radon values for each household and
  ## jitters the points slightly to avoid overplotting.
  geom_jitter(
    data = pred_counties,
    aes(x = basement, y = log_radon, group = county2),
    height = 0,
    width = 0.1
  ) +

  ## This superimposes the partial-pooling ( +  x_i + _i + _j) predictions
  ## over the raw data
  geom_line(
```

```
    data = county_df_fig_1,
    aes(x = basement, y = partial_pred, group = county2),
    linetype = "solid",
    colour = "gray"
  ) +

  ## No-pooling predictions ( _{ij} +  x_i + _i)
  geom_line(
    data = county_df_fig_1,
    aes(x = basement, y = no_pool_pred, group = county2)
  ) +

  ## Full pooling predicitons ( +  x_i + _i)
  geom_line(
    data = county_df_fig_1,
    aes(x = basement, y = pooled_pred, group = county2),
    linetype = "dashed"
  ) +

  ## Finally, use facet_wrap to arrange the panels in two
  ## rows of four
  facet_wrap(vars(county2), nrow = 2) +
  xlab("basement") +
  ylab("log radon level") +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())

plot(g)
```



11.2.3 Figure 2

Figure 2 reproduces the relationship between the county-level random intercepts, α_j and the expected level of radon at a county level as a function of county-level soil uranium.

Data Preparation

The following code allows us to extract predictions at the county level using our prediction dataset. To do this, we use the `predicted_draws` function from the `tidybayes` package, which lets us sample from the posterior distribution of the fitted model. The `median_qi` function, also from tidybayes, lets us calculate the width of a 1 standard error interval (equivalent to the range containing ~17% of the posterior probability mass around the posterior median) used in the original Figure 1 from (11):

```
dd <- predicted_draws(m2, county_dummy_df) %>%
  median_qi(.width = 0.17) %>%
  filter(basement == 0)
```

In order to calculate the predicted mean radon at a county level, we need to access the coefficients corresponding to the level two model, including the intercept γ_0 and the effect of a 1-log change in log-uranium on predicted log-radon, γ_1. In order to get these values out of

the model, we can use the `gather_draws` function from tidybayes, which allows us to access the posterior distributions for each of these parameters:

```
uranium_coefs <-
  gather_draws(m2, c(`(Intercept)`, log_uranium)) %>% median_qi()
```

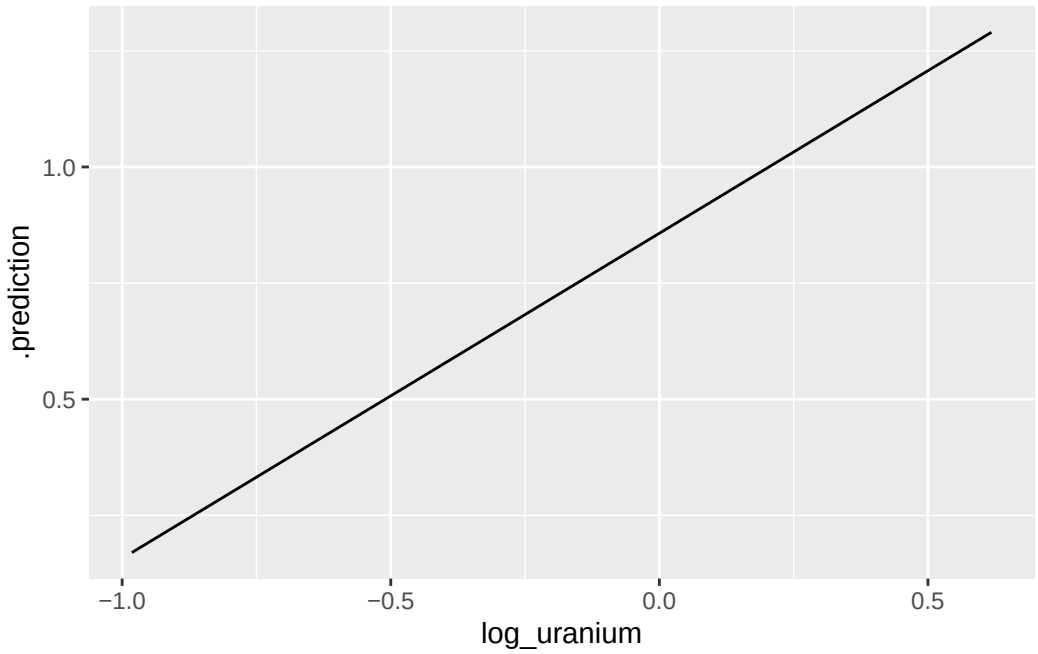
Now it is as simple as calculating the linear predictor $\gamma_0 + \gamma_1 z_j$, where z_j is the log-uranium measure for the j-th county, and storing this information in a data frame we will use for plotting:

```
log_uranium_range <-
  seq(min(county_uranium$log_uranium) - .1,
      max(county_uranium$log_uranium) + .1,
      by = 0.1)

pred_log_radon <-
  uranium_coefs$.value[1] + uranium_coefs$.value[2] * log_uranium_range

median_radon_pred <-
  data.frame(log_uranium = log_uranium_range, .prediction = pred_log_radon)
```

Plotting

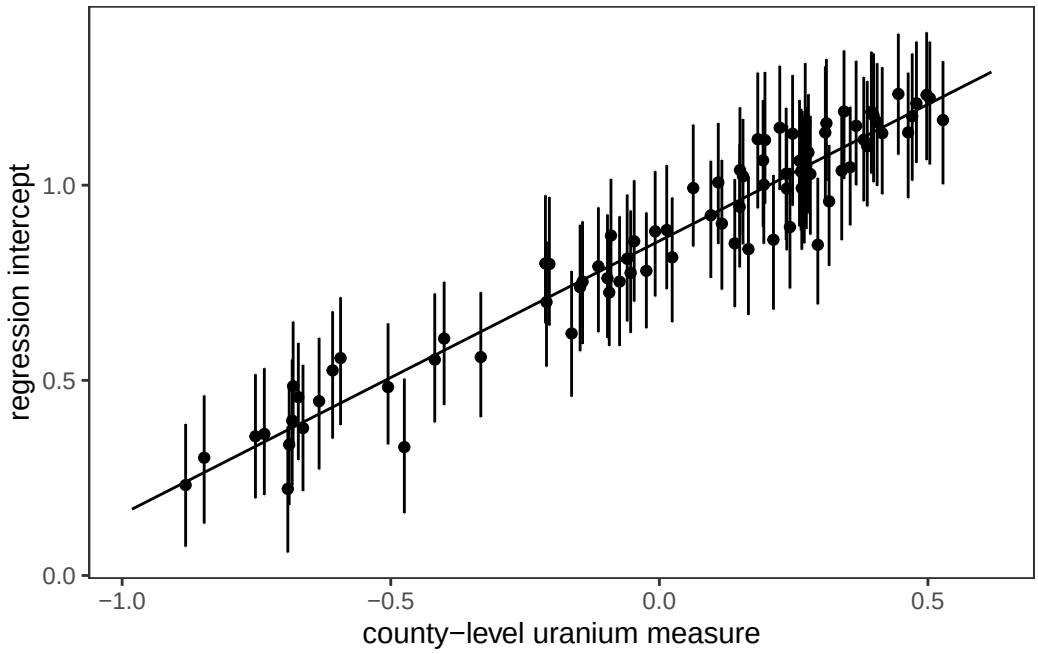
Now, we can build this figure up one step at a time, starting with our mean predictions:

```
g <- ggplot(dd) +
     geom_line(data = median_radon_pred, aes(x = log_uranium, y = .prediction))

plot(g)
```



The next step is to then add the median predictions (points) and 1 SE errorbars to the plot, and then fix the theme to match the original figure, et voilà!

```
g <- g +  geom_point(aes(x = log_uranium, y = .prediction, group = county)) +
  geom_errorbar(aes(
    x = log_uranium,
    y = .prediction,
    ymin = .lower,
    ymax = .upper
  )) +
  theme_bw() + theme(panel.grid.major = element_blank(),
                     panel.grid.minor = element_blank()) +
  xlab("county-level uranium measure") +
  ylab("regression intercept")

plot(g)
```


regression intercept
1.0
0.5
0.0
−1.0
−0.5
0.0
0.5
county−level uranium measure

References

12 Taking a spatial perspective on the `radon` data

This tutorial is a follow-up to a prior exercise using these data. So if you haven't already, please go back and take a look at the original multi-level modeling radon example in Chapter 11.

12.1 Learning Goals

The primary goals of this tutorial are to introduce you to:

1. Merging of non-spatial health exposure or outcome data with spatial metadata.
2. Calculation of important spatial summary statistics, e.g. Moran's I, from such data.
3. Spatial analysis of residuals from aspatial regression models of spatially-referenced data.

> **Look out for stretch exercises!**
>
> If you see a box with a like this, it's in an invitation to go a bit further. This could be a conceptual question or a chance to write a bit of code to explore the data or outputs of the analysis a bit more.

12.2 Setting up the environment

```
library(ggplot2)
library(arm)
library(tidycensus)
library(dplyr)
library(rstanarm)
library(stringr)
library(spdep)
knitr::opts_chunk$set(message = FALSE, warning=FALSE, tidy=TRUE)
```

12.3 Data Preparation

Before diving into the analysis steps, there are several key things we need to do to be able to easily work with these data.

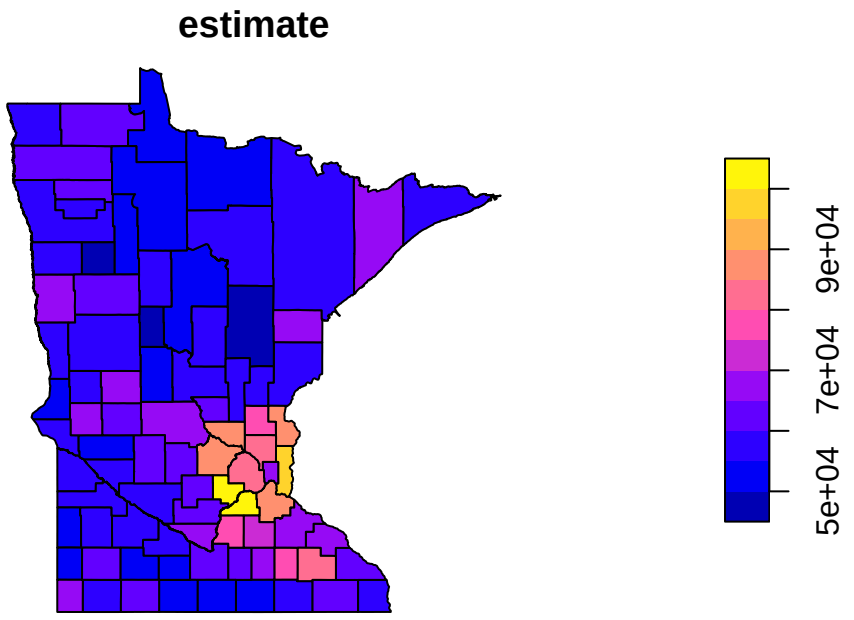
12.3.1 Download a shapefile for Minnesota

First, we need to download a shapefile for the state of Minnesota in which each polygon represents an individual county. Thankfully, in R, this is made easy using the excellent `tidycensus` package:

```
options(tigris_use_cache = TRUE)

minnesota <- get_acs(state = "MN", geography = "county", variables = "B19013_001",
    geometry = TRUE, year = 2020)
```

Tidycensus gives us the data as an `sf` dataframe containing a number of fields including population estimates, which we can plot straightforwardly using the `plot` function supplied by the `sf` package:

```
plot(minnesota["estimate"])
```



12.3.2 Merge the spatial data with the radon data

In its raw form, this spatial dataset isn't quite ready to merge with the radon data. If we take a peek at the county names in the shapefile, we can see that they don't quite match the formatting of the ones in the original data:

```
head(sort(minnesota$NAME))
```

```
[1] "Aitkin County, Minnesota"    "Anoka County, Minnesota"
[3] "Becker County, Minnesota"    "Beltrami County, Minnesota"
[5] "Benton County, Minnesota"    "Big Stone County, Minnesota"
```

Whereas in the radon data we see:

```
head(unique(as.character(radon$county)))
```

```
[1] "AITKIN"   "ANOKA"    "BECKER"   "BELTRAMI" "BENTON"   "BIGSTONE"
```

The big differences here are that the shapefile uses: 1) mixed-case county names and 2) includes the name of the state in each label. To make these match the `radon` dataset, we can use some tools from the `stringr` package as well as some base R functions:

```
minnesota <-
  minnesota %>% mutate(
    ## Since all of the original county names have the same substring " County, Minnesota"
    ## we can use the str_remove function to pull them out of all of them
    county = str_remove(NAME, " County, Minnesota") %>%
      ## Since some of the counties  officially have two-word names (e.g. Big Stone)
      ## which are collapsed in the radon dataset, we will use this function to remove all
      str_replace_all(" ", "") %>%
      ## A few county names include abbreviations indicated by the presence of a '.' (e.g.
      ## so we will get rid of that bit of punctuation since it is not in the original dat
      str_replace_all("\\.", "") %>%
      ## Finally, convert all the county names to uppercase
      toupper()
  )
```

Now, the county labels should match:

```
head(sort(minnesota$county))
```

```
[1] "AITKIN"   "ANOKA"    "BECKER"   "BELTRAMI" "BENTON"   "BIGSTONE"
```

12.3.3 Preparing the `radon` dataset

We will repeat the steps from the earlier tutorial in order to prepare our data for analysis:

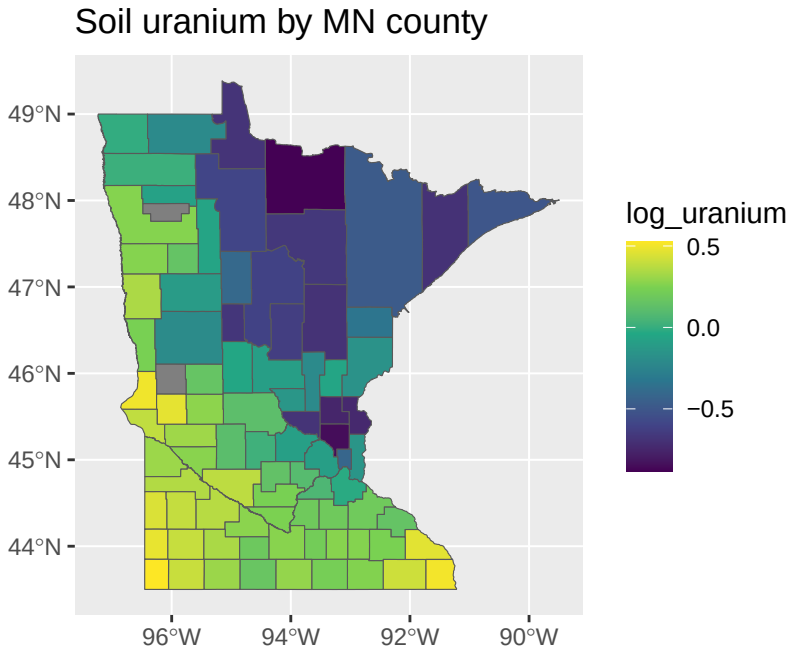
```
radon <- radon %>%
    mutate(basement = 1 - floor)

county_uranium <- radon %>%
    group_by(county) %>%
    summarize(log_uranium = first(log_uranium), mean_radon = mean(log_radon))
```

Because the sf dataset returned by tidycensus is a dataframe, we can then easily merge the county-level soil uranium concentrations we derived above into the shapefile. We use the `left_join` function from `dplyr` to ensure that all of the counties in the original shapefile are represented in the final dataset, even if a soil uranium measure is unavailable for them in the original data:

```
minnesota_radon <- left_join(minnesota, county_uranium)
```

We can then plot the log-uranium measures on the map and see that, in fact, they are quite spatially correlated. We can also see that there appear to be two counties which are missing soil uranium data in the `radon` dataset. To have a bit more control over our plots, we'll switch here to using the `geom_sf` function of `ggplot2`, which makes plotting geographies from sf objects easy:

```
g <- ggplot(minnesota_radon) + geom_sf(aes(fill = log_uranium)) + scale_fill_viridis_c() +
    ggtitle("Soil uranium by MN county")
plot(g)
```



12.4 Measuring Spatial Correlation

To validate our hunch that soil uranium is spatially concentrated in Minnesota, we can calculate the value of Moran's I for these data using some functions from the `spdep` package. First, we use the `poly2nb` function to obtain the neighbors for each polygon, which will be used to calculate Moran's I.

```
nb <- poly2nb(minnesota_radon)
```

This function yields an R list in which each entry is a vector with the indices for the neighbors of the i-th county. For example, this prints the neighbors of the first three counties in the dataset:

```
print(nb[1:3])
```

```
[[1]]
[1] 12 47

[[2]]
[1] 27

[[3]]
```

```
[1]  8 24 46 57 67 76 83 87
```

We then pass this function to the `nb2listw` function to obtain weights for the relationships between neighbors. Here, we use the simplest option available, "B", for binary weights equal to 1 if the areas are neighbors and 0 otherwise:

```
lw <- nb2listw(nb, style = "B", zero.policy = TRUE)
print(lw$weights[1:3])
```

```
[[1]]
[1] 1 1

[[2]]
[1] 1

[[3]]
[1] 1 1 1 1 1 1 1 1
```

Finally, we can pass these weights, along with some additional information including the outcome of interest at each location, the total number of locations, and the sum of all the weights to the `moran` function. The `NAOK=TRUE` option used here also allows the function to drop locations where data are missing:

```
radon_i <- moran(minnesota_radon$log_uranium, lw, length(nb), Szero(lw), NAOK = TRUE)$I
```

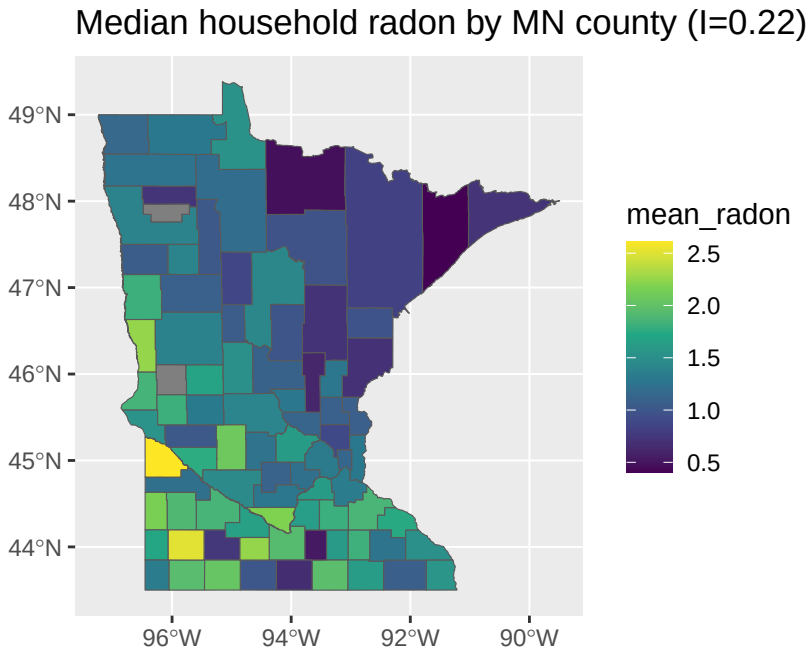
When we do this, we find that the value of Moran's I = 0.71, which is close to the maximum value of 1. Since we'll be returning to the calculation of Moran's I using our spatial data, lets pack it up into a function:

```
moranFromSF <- function(x, sfdf, style = "B") {
    nb <- poly2nb(sfdf)
    lw <- nb2listw(nb, style = style, zero.policy = TRUE)
    mi <- moran(x, lw, length(nb), Szero(lw), NAOK = TRUE)$I
    return(mi)
}

print(moranFromSF(minnesota_radon$log_uranium, minnesota_radon))
```

```
[1] 0.712615
```

Of course, our key quantity of interest isn't soil uranium but the concentration of radon at the household level. When we constructed the `county_uranium` dataset above, we also calculated the median radon concentration in the data for each county. When we plot it, we see something similar to the soil uranium, but perhaps a bit less clear:

```
g <- ggplot(minnesota_radon) + geom_sf(aes(fill = mean_radon)) + scale_fill_viridis_c() +
    ggtitle(paste0("Median household radon by MN county (I=", round(moranFromSF(minnesota_
        minnesota_radon), 2), ")"))
plot(g)
```



As you can see in the figure, the value of Moran's I is smaller than we got for log-uranium but still substantial.

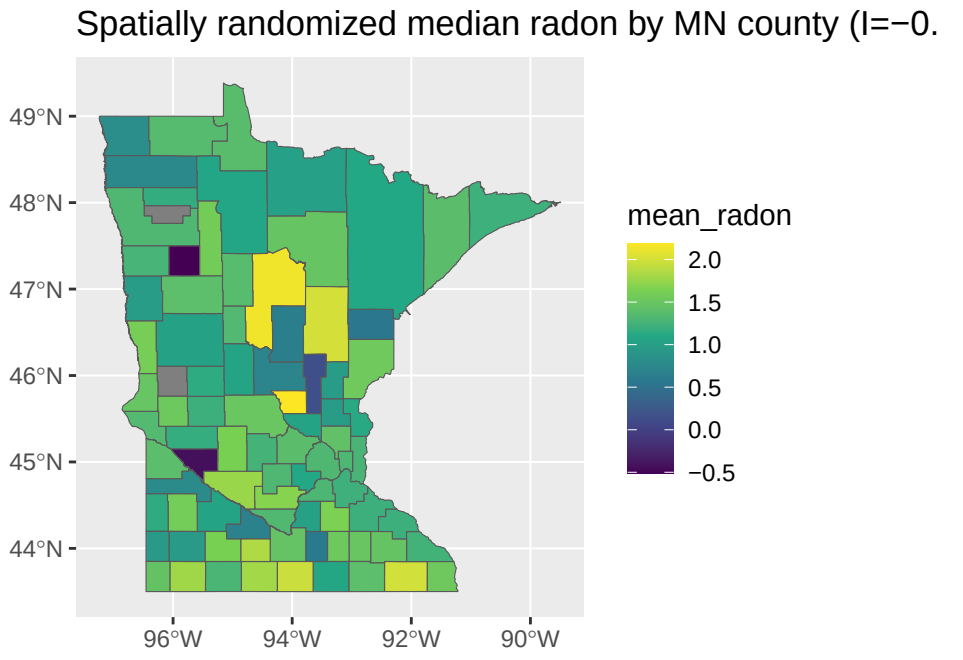
> What's going on?
>
> Pause here and take a moment to try to figure out what might account for the difference in this intensity of clustering in radon vs. soil uranium measurements.

12.4.1 Testing, testing

One way to determine whether the spatial aggregation of the radon measurements is meaningful is to compare it to a *counterfactual scenario* in which the distribution of radon concentrations

is uncorrelated with space. This assumption, known as complete spatial randomness (or CSR), allows us to provide a benchmark against which we determine whether the value of Moran's I we determined is highly likely to occur by chance alone. Thankfully, it is easy to generate a dataset in which the median radon values are distributed randomly across the map:

```
## Make a new dataset representing 'random minnesota': Use the sample function
## to resample household radon values without replacement, we then recalculate
## county values based on these suffled values
county_uranium_random <- radon %>%
    mutate(log_radon = sample(log_radon, nrow(.), replace = FALSE)) %>%
    group_by(county) %>%
    summarize(log_uranium = first(log_uranium), mean_radon = mean(log_radon))

random_minnesota <- left_join(minnesota, county_uranium_random)


## Plot the new randomized data
g <- ggplot(random_minnesota) + geom_sf(aes(fill = mean_radon)) + scale_fill_viridis_c() +
    ggtitle(paste0("Spatially randomized median radon by MN county (I=", round(moranFromSF
        random_minnesota), 2), ")"))
plot(g)
```



This yields something that looks pretty randomly distributed, which is reflected in a Moran's

I estimate closer to the null value of 0. This doesn't necessarily tell us whether this result is meaningful rather than an artifact of random chance.

> What is the same? What is different?
>
> Take a minute to explore the distribution of different quantities between some random minnesotas and the observed one. For example, look at distributions of the number of observations per county, the proportion of households in each county that have basements, etc. Which are similar and which are different?

12.4.2 Complete Spatial Randomness

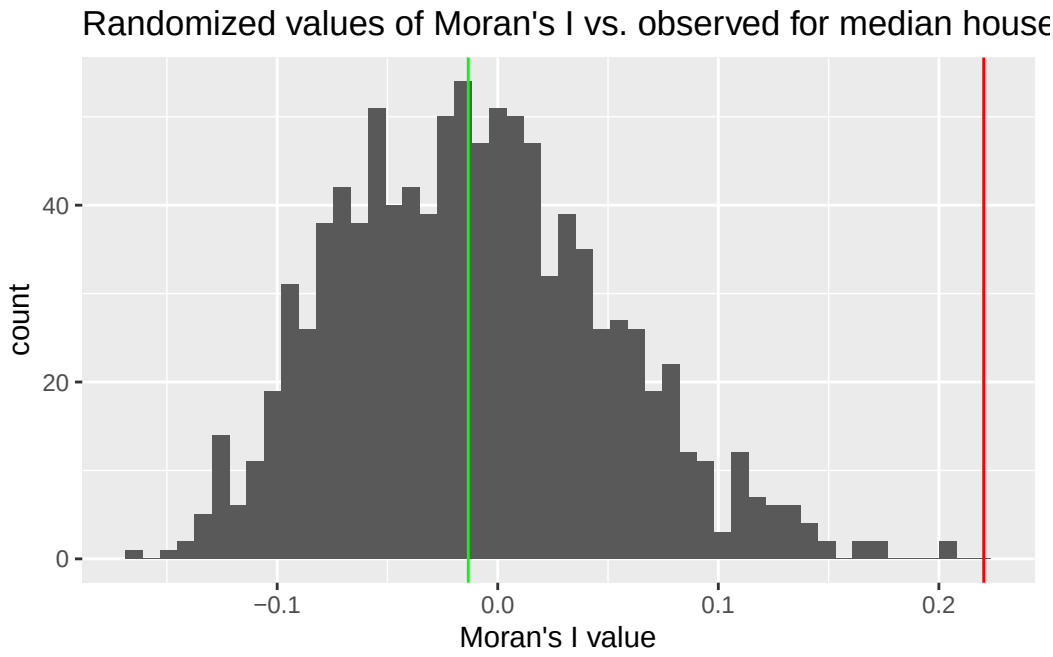
What we can do, though, is to generate a bunch of random Minnesotas in which there is no relationship between geographic location and median radon, calculate Moran's I for each of those, and see how our observed data stack up.

```
csrMorans <- function(radon, minnesota, trials = 1000, style = "B") {
    county_uranium <- radon %>%
        group_by(county) %>%
        summarize(log_uranium = first(log_uranium), mean_radon = mean(log_radon)) %>%
        left_join(minnesota, .)

    nb <- poly2nb(minnesota)
    lw <- nb2listw(nb, style = style, zero.policy = TRUE)
    mv <- moran(county_uranium$mean_radon, lw, length(nb), Szero(lw), NAOK = TRUE)$I

    moran_vals <- rep(0, trials)
    for (i in 1:trials) {

        county_uranium_random <- radon %>%
            mutate(log_radon = sample(log_radon, nrow(.), replace = FALSE)) %>%
            group_by(county) %>%
            summarize(log_uranium = first(log_uranium), mean_radon = mean(log_radon))

        random_minnesota <- left_join(minnesota, county_uranium_random)
        moran_vals[i] <- moran(random_minnesota$mean_radon, lw, length(nb), Szero(lw),
            NAOK = TRUE)$I
    }

    return(list(midist = moran_vals, mi = mv))
}
```

```
csr_dist <- csrMorans(radon, minnesota)
```

We can use the distribution of Moran's I values taken from the randomized datasets to benchmark how likely our observed value is to occur by purely random chance. The figure below shows that this is quite unlikely:

```
g <- ggplot() + geom_histogram(aes(x = csr_dist$midist), bins = 50) + xlab("Moran's I valu
    geom_vline(xintercept = csr_dist$mi, colour = "red") + geom_vline(xintercept = median(
    colour = "green") + ggtitle("Randomized values of Moran's I vs. observed for median ho

plot(g)
```



And we can directly estimate this probability as follows:

```
real_moran <- moranFromSF(minnesota_radon$mean_radon, minnesota_radon)
p_moran <- sum(csr_dist$midist >= csr_dist$mi)/length(csr_dist$midist)
print(p_moran)
```

```
[1] 0
```

From 1000 samples, it appears that none of our random datasets yielded a value of Moran's I

$\geq$ to the observed value, suggesting that it is unlikely that we would observe this value as a simple function of sampling variability.

> What could go wrong?
>
> Before you move on, take a minute to think about what some of the potential flaws in our CSR-based approach to assessing the meaningfulness or signficance of this result might be.

12.5 Models!

Up to this point, we have relied on county-level summaries of the household-level radon data. For the final section of this tutorial, we are going to go back to using the full dataset and implement regression models that are able to characterize variation at the household and community level. Specifcially, we are going to first fit the full-pooling, no-pooling and partial pooling models from the original (11) paper. We won't go into detail on these as they have been discussed in depth in the original paper and the previous post.

> Confused?
>
> For more detail on the implementation on interpretation of these models, please check out Chapter 11.

12.5.1 Full-pooling model

The full-pooling model has the following form, in which the variable x_{ij} indicates whether house i in county j has a basement (1) or not (0).

$$y_{ij} = \alpha + \beta x_{ij} + \epsilon_i$$

```
full_pooling_model <- lm(log_radon ~ basement, data = radon)
radon$full_pooling_resid <- resid(full_pooling_model)
radon$full_pooling_pred <- predict(full_pooling_model)
```

> Storing Model Predictions
>
> Note that we are storing the residuals and predictions for this model (and the ones below) as a column inside the `radon` dataframe.

12.5.2 No Pooling

The no-pooling model assumes essentially that each county is indepenedent, and includes a categorical variable for the county that the observed household is in:

$$y_{ij} = \alpha_j + \beta x_{ij} + \epsilon_i$$

```
no_pooling_model <- lm(log_radon ~ basement + county, data = radon)
radon$no_pooling_resid <- resid(no_pooling_model)
radon$no_pooling_pred <- predict(no_pooling_model)
```

12.5.3 Partial pooling model

The partial-pooling model is the multi-level analogue to the no-pooling model. For more detail, please see the partial pooling section of the original tutorial.

```
partial_pool_model <- stan_lmer(log_radon ~ basement + log_uranium + (1 | county),
    data = radon)
radon$partial_pooling_resid <- resid(partial_pool_model)
radon$partial_pooling_pred <- posterior_predict(partial_pool_model) %>%
    apply(2, mean)
```

12.6 Residual Analysis

One thing that is important to note is that none of the regression models we are looking at directly account for spatial clustering. In other words, the spatial arrangement of the counties is not an input to the model. This doesn't mean that they cannot adequately account for spatial correlation through the inclusion of key covariates, however.

One way to assess how well a model is accounting for observed and unobserved spatial hererogeneity is to examine the model *residuals* for evidence of spatial clustering, which is what we will do in this section.

Since the residuals for each model are generated at the level of individual households, we will go back to working with county-level summaries of both the prediction error (residuals) and predicted household radon values:

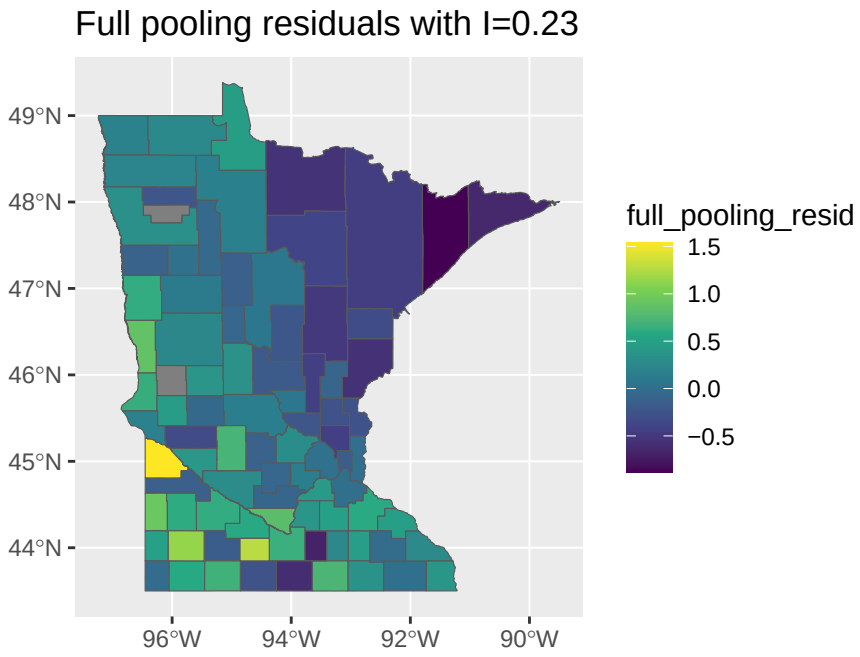
```
results_by_county <- radon %>%
    group_by(county) %>%
```

```
    summarize(p_basement = sum(basement)/n(), full_pooling_resid = mean(full_pooling_resid
        no_pooling_resid = mean(no_pooling_resid), full_pooling_pred = mean(full_pooling_p
        no_pooling_pred = mean(no_pooling_pred), partial_pooling_pred = mean(partial_pooli
        partial_pooling_resid = mean(partial_pooling_resid))
results_by_county <- left_join(minnesota, results_by_county)
```

12.6.1 Full Pooling Residuals

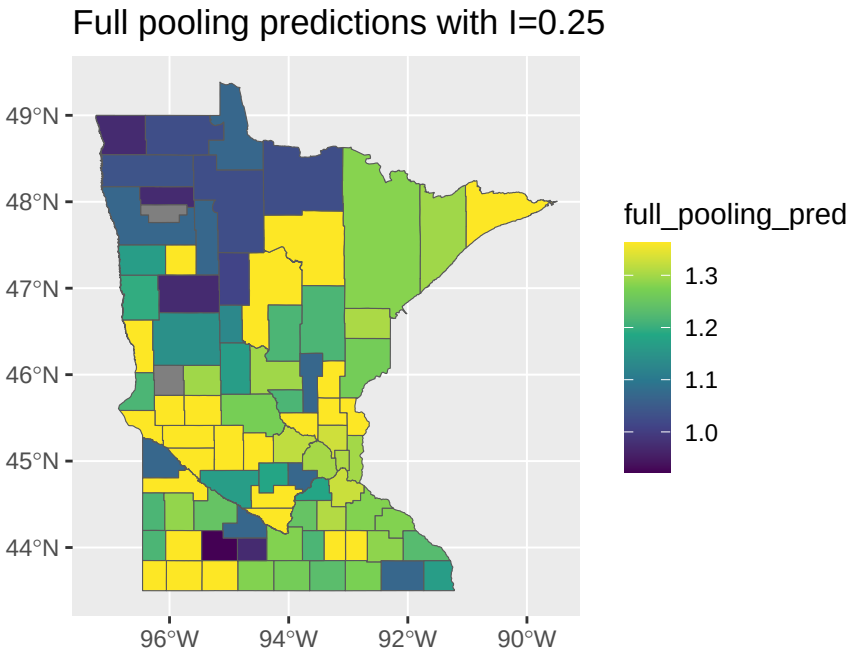
When we look at the results of the full-pooling model, the residuals that still look pretty spatially clustered, and this is reflected in the value of Moran's I > 0:

```
mi <- round(moranFromSF(results_by_county$full_pooling_resid, results_by_county),
    2)
g <- ggplot(results_by_county) + geom_sf(aes(fill = full_pooling_resid)) + scale_fill_viri
    ggtitle(paste0("Full pooling residuals with I=", mi))
plot(g)
```



This is probably intuitive: the full pooling model didn't include any county-level information, so it might not account for all of the sptial variation. On the flipside, if we look at the predictions of the model - reflecting the expected household levels of radon in each county - should we should expect to find that they are spatially un-clustered or also clustered?

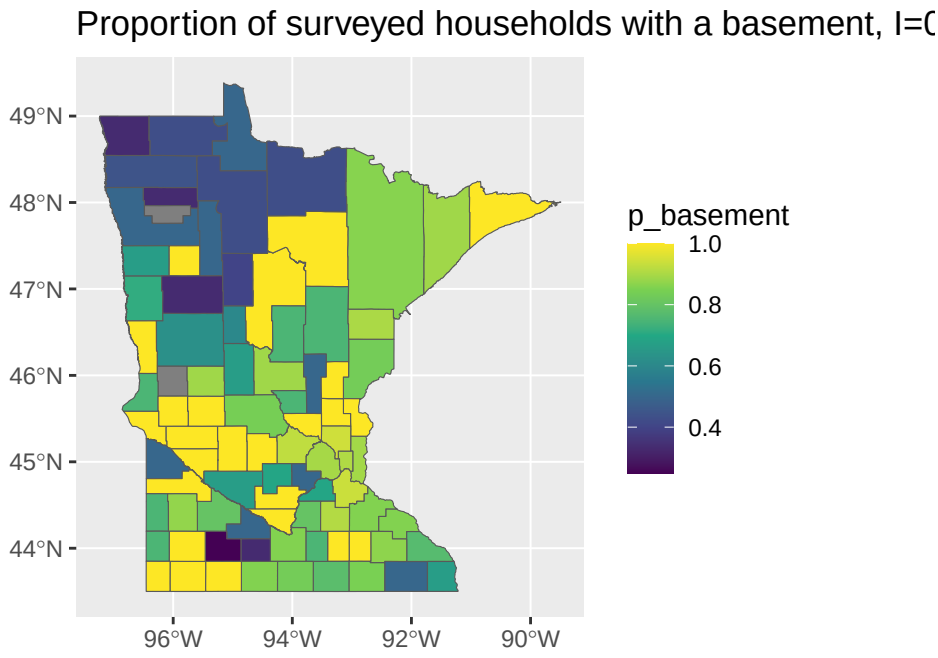
```
mi <- round(moranFromSF(results_by_county$full_pooling_pred, results_by_county),
    2)
g <- ggplot(results_by_county) + geom_sf(aes(fill = full_pooling_pred)) + scale_fill_virid
    ggtitle(paste0("Full pooling predictions with I=", mi))
plot(g)
```



Wait - what? The predictions are also quite clustered, although the pattern looks a bit like a photographic negative of the residual map. It looks like our model is predicting lower values in the northwest corner of the state relative to the rest of the state. How is this possible, if our model doesn't include contextual information?

This might be explained by differences in *composition* at the county level: maybe houses in some counties are more likely to have basements than in others? If this is the case, then those high-basement counties may have higher avg. levels of radon. So, lets just check and see if our one predictor - the presence or absence of a basement - exhibits any spatial variability?

```
mi <- round(moranFromSF(results_by_county$p_basement, results_by_county), 2)
g <- ggplot(results_by_county) + geom_sf(aes(fill = p_basement)) + scale_fill_viridis_c()
    ggtitle(paste0("Proportion of surveyed households with a basement, I=", mi))
plot(g)
```



Whoops...that looks familiar! It seems like the pattern of spatial variation in the presence/absence of basements may be driving the clustering in our predictions and - by consequence - our residuals!

> **Spatially correlated predictors → Spatially correlated predictions**
>
> Sometimes, it is easy to forget that the input data may be as or more correlated than the outcome data. In this example, the presence or absence of a basement in a house seems to have a spatial pattern and this impacts the spatial patterning of our predictions and model residuals!

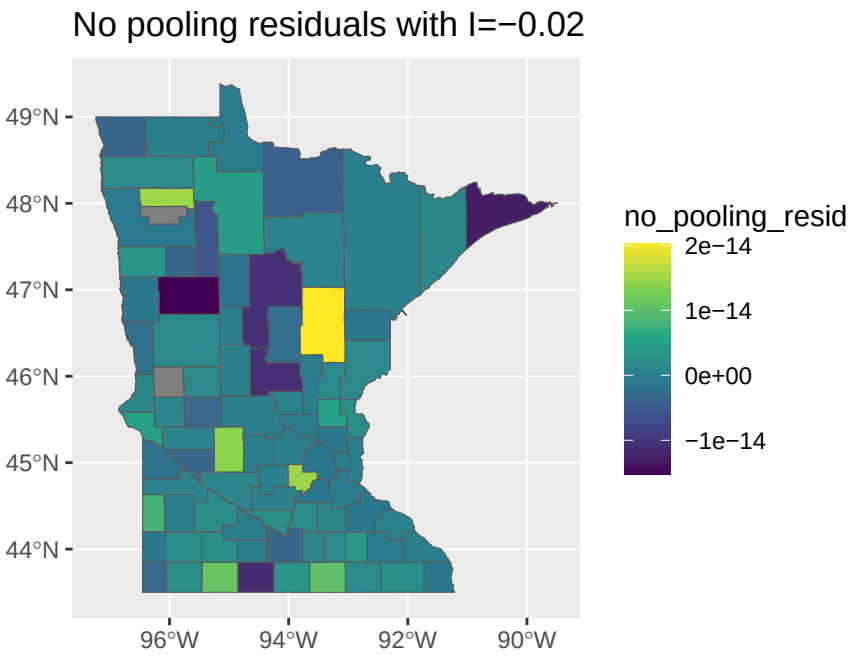
So it looks like we are over-predicting risk in some areas where more surveyed households have basements and under-predicting it in other places where fewer households have basements.

12.6.2 No Pooling

Ok, so lets try this again with our no-pooling model which at least includes the counties as categorical covariates. Unless something weird is going on, this model should do a good job of explaining spatial variation:

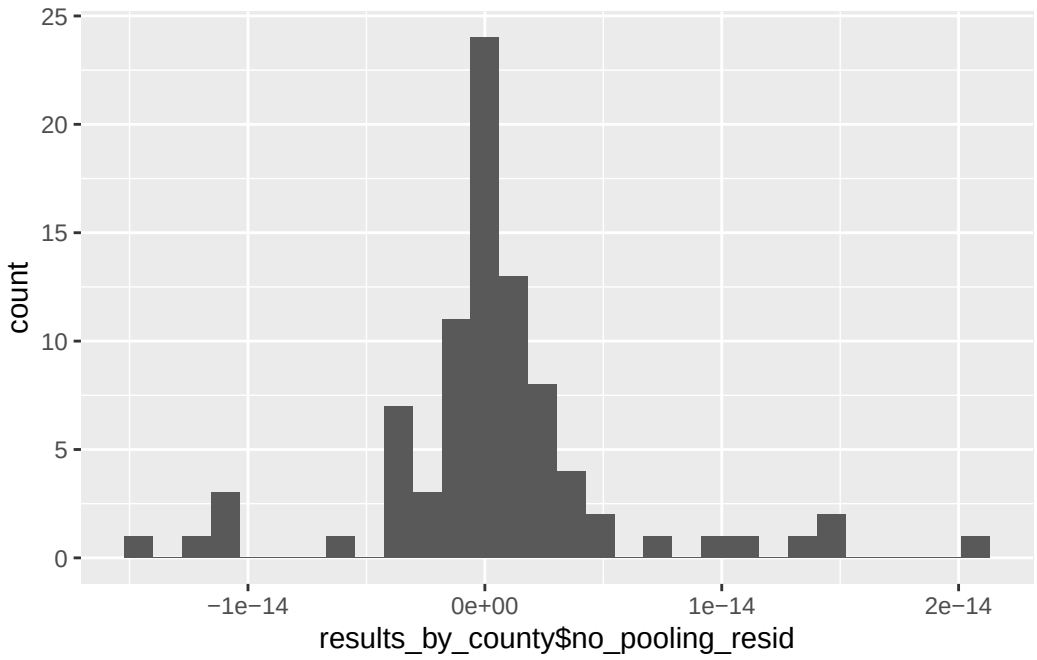
```
mi <- round(moranFromSF(results_by_county$no_pooling_resid, results_by_county), 2)
g <- ggplot(results_by_county) + geom_sf(aes(fill = no_pooling_resid)) + scale_fill_viridi
```

```
    ggtitle(paste0("No pooling residuals with I=", mi))
plot(g)
```



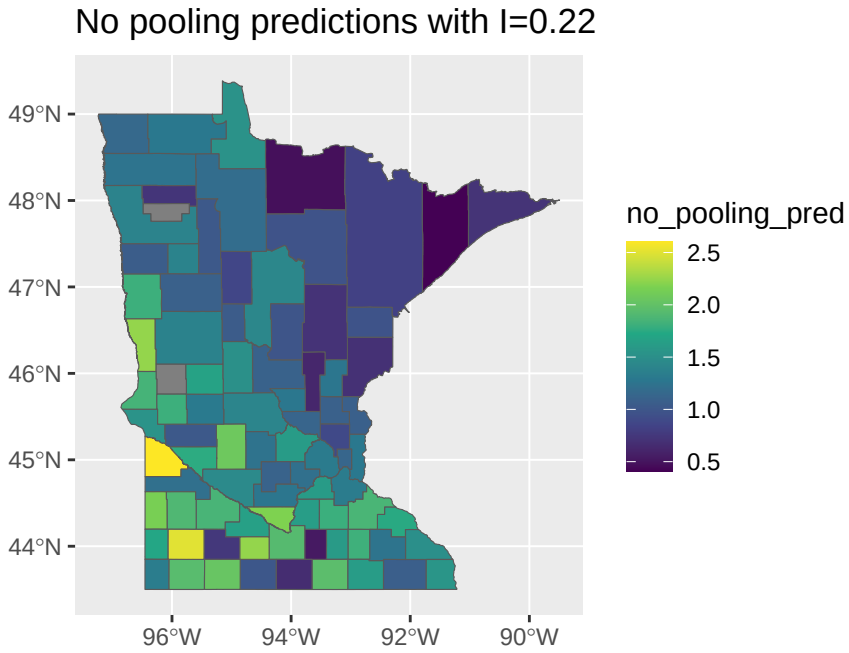
Well, that's a bit better, although it does such a good job at explaining away the overall variability in our measurments, we might be concerned that it is overfitting the model through the inclusion of the county level random effects. This is evidenced in the tiny size of the residuals and their minimal variation:

```
g <- ggplot() + geom_histogram(aes(x = results_by_county$no_pooling_resid))
plot(g)
```



Unsurprisingly, this model does an excellent job of predicting the spatial patterns in the original data:

```
mi <- round(moranFromSF(results_by_county$no_pooling_pred, results_by_county), 2)
g <- ggplot(results_by_county) + geom_sf(aes(fill = no_pooling_pred)) + scale_fill_viridis
    ggtitle(paste0("No pooling predictions with I=", mi))
plot(g)
```



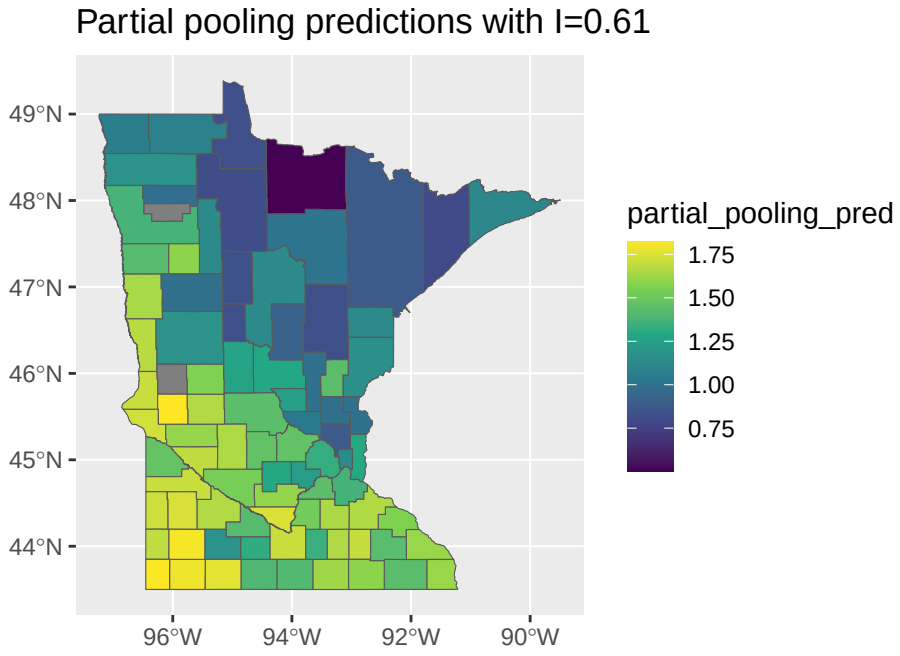
I love this model! It's perfect! It captures almost the exact same clustering and spatial patterning of risk as the original data.

> ⚠ Danger!
>
> What is problematic about this model? What limits its usefulness for both interpretation and prediction?

12.6.3 Partial Pooling

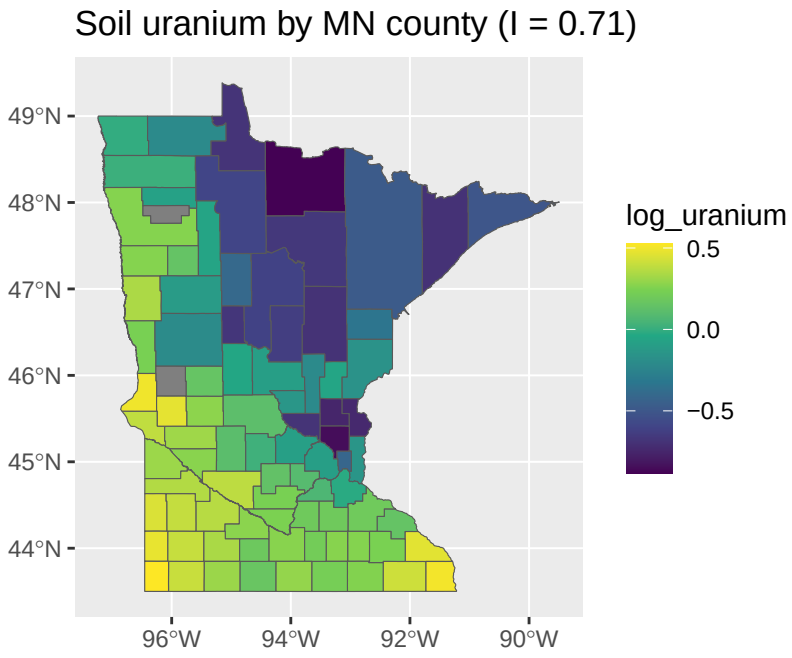
When we look at the predictions of the partial pooling model, they are notably smoother and more clustered than those of the full- and no-pooling models:

```
mi <- round(moranFromSF(results_by_county$partial_pooling_pred, results_by_county),
    2)
g <- ggplot(results_by_county) + geom_sf(aes(fill = partial_pooling_pred)) + scale_fill_vi
    ggtitle(paste0("Partial pooling predictions with I=", mi))
plot(g)
```



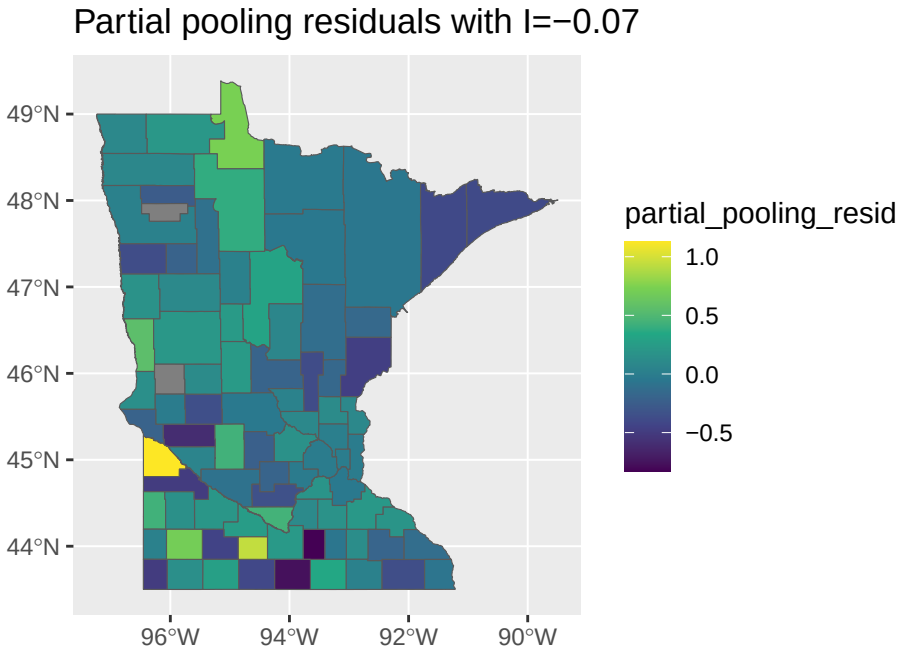
If we compare this pattern and intensity of clustering to the log-uranium data, it is clear that the smoothness in the model predictions reflects the relative smoothness and clustering of the soil uranium data:

```
mi <- round(moranFromSF(minnesota_radon$log_uranium, minnesota_radon), 2)
g <- ggplot(minnesota_radon) + geom_sf(aes(fill = log_uranium)) + scale_fill_viridis_c() +
    ggtitle(paste0("Soil uranium by MN county (I = ", mi, ")"))
plot(g)
```



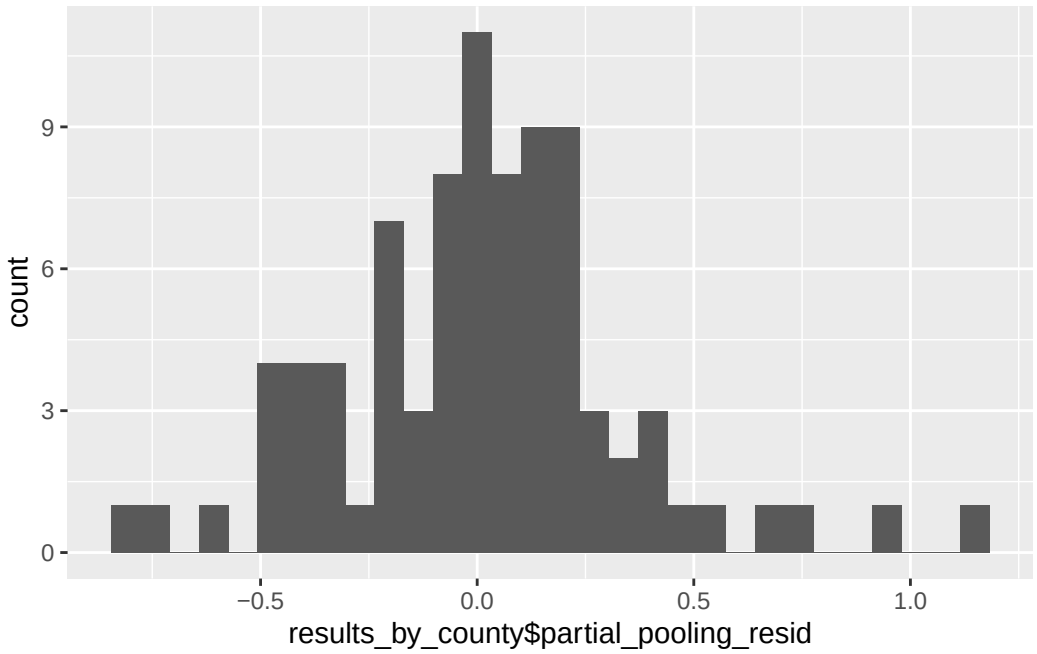
When we look at the residuals, they are still quite un-clustered - similar to the no pooling model, but their magnitude is larger, suggesting that the multi-level model is less suceptible to overfitting:

```
mi <- round(moranFromSF(results_by_county$partial_pooling_resid, results_by_county),
    2)
g <- ggplot(results_by_county) + geom_sf(aes(fill = partial_pooling_resid)) + scale_fill_v
    ggtitle(paste0("Partial pooling residuals with I=", mi))
plot(g)
```



By contrast, the aggregated residuals at the county level are less indicative of overfitting than the no-pooling model, but are still a bit fat-tailed, suggesting that some counties may still be over- or under-fit.

```
g <- ggplot() + geom_histogram(aes(x = results_by_county$partial_pooling_resid))
plot(g)
```



12.7 What's next?

In this tutorial, we have thoroughly reviewed the spatial implications of the three types of models reviewed in the (11) analysis of household-level radon in Minnesota. While our results suggest that the partial-pooling model provides the most compelling explanation of spatial variability in our data, we are not done yet! In the next tutorial, we will look specifically at the predictive capabilities of each of these models and use the ability to predict risk for counties in which household-level measures are unavaialble or missing as the final guide in our odyssey of model comparison.

References

1. McElreath R. Statistical Rethinking: A Bayesian Course with Examples in R and STAN. 2nd edition. Boca Raton: Chapman and Hall/CRC; 2020.
2. Hilborn R, Mangel M. The Ecological Detective: Confronting Models with Data. 1st edition. Princeton, NJ: Princeton University Press; 1997.
3. Gelman A. Regression and Other Stories. 1st edition. Cambridge: Cambridge University Press; 2020.

4. Zelner JL, Trostle J, Goldstick JE, et al. Social Connectedness and Disease Transmission: Social Organization, Cohesion, Village Context, and Infection Risk in Rural Ecuador. *American Journal of Public Health* [electronic article]. 2012;102(12):2233–2239. (http://ajph.aphapublications.org/doi/10.2105/AJPH.2012.300795). (Accessed December 15, 2019)

5. Zelner JL, Murray MB, Becerra MC, et al. Age-Specific Risks of Tuberculosis Infection From Household and Community Exposures and Opportunities for Interventions in a High-Burden Setting. *American Journal of Epidemiology* [electronic article]. 2014;180(8):853–861. (https://academic.oup.com/aje/article-lookup/doi/10.1093/aje/kwu192). (Accessed December 15, 2019)

6. Morris SE, Zelner JL, Fauquier DA, et al. Partially observed epidemics in wildlife hosts: Modelling an outbreak of dolphin morbillivirus in the northwestern Atlantic, June 2013–2014. *Journal of The Royal Society Interface* [electronic article]. 2015;12(112):20150676. (https://royalsocietypublishing.org/doi/10.1098/rsif.2015.0676). (Accessed December 15, 2019)

7. Thompson CN, Zelner JL, Nhu TDH, et al. The impact of environmental and climatic variation on the spatiotemporal trends of hospitalized pediatric diarrhea in Ho Chi Minh City, Vietnam. *Health & Place* [electronic article]. 2015;35:147–154. (https://linkinghub.elsevier.com/retrieve/pii/S1353829215001094). (Accessed December 15, 2019)

8. Zelner JL, Murray MB, Becerra MC, et al. Identifying Hotspots of Multidrug-Resistant Tuberculosis Transmission Using Spatial and Molecular Genetic Data. *Journal of Infectious Diseases* [electronic article]. 2016;213(2):287–294. (https://academic.oup.com/jid/article-lookup/doi/10.1093/infdis/jiv387). (Accessed December 15, 2019)

9. Tobler WR. A Computer Movie Simulating Urban Growth in the Detroit Region. *Economic Geography* [electronic article]. 1970;46:234–240. (http://www.jstor.org/stable/143141). (Accessed January 13, 2022)

10. Levy MZ, Barbu CM, Castillo-Neyra R, et al. Urbanization, land tenure security and vector-borne Chagas disease. *Proceedings of the Royal Society B: Biological Sciences* [electronic article]. 2014;281(1789):20141003. (https://royalsocietypublishing.org/doi/full/10.1098/rspb.2014.1003). (Accessed January 23, 2023)

11. Gelman A. Multilevel (Hierarchical) Modeling: What It Can and Cannot Do. *Technometrics* [electronic article]. 2006;48(3):432–435. (http://www.tandfonline.com/doi/abs/10.1198/004017005000000661). (Accessed December 15, 2019)

12. Price PN, Nero AV, Gelman A. Bayesian prediction of mean indoor radon concentrations for Minnesota counties. *Health Physics*. 1996;71(6):922–936.