Overlays

HES 505 Fall 2022: Session 18

Matt Williamson

Objectives

By the end of today you should be able to:

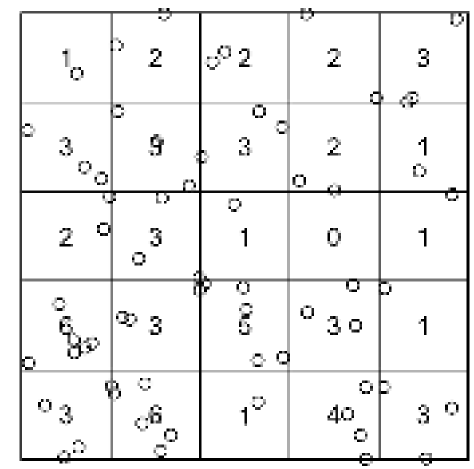
- integrate a covariate into KDE's
- Describe the utility and shortcomings of overlay analysis
- Describe and implement different overlay approaches

Re-visiting Density Methods

- The overall *intensity* of a point pattern is a crude density estimate

$$\hat{\lambda} = \frac{\#(S \in A)}{a}$$

- Local density = quadrat counts



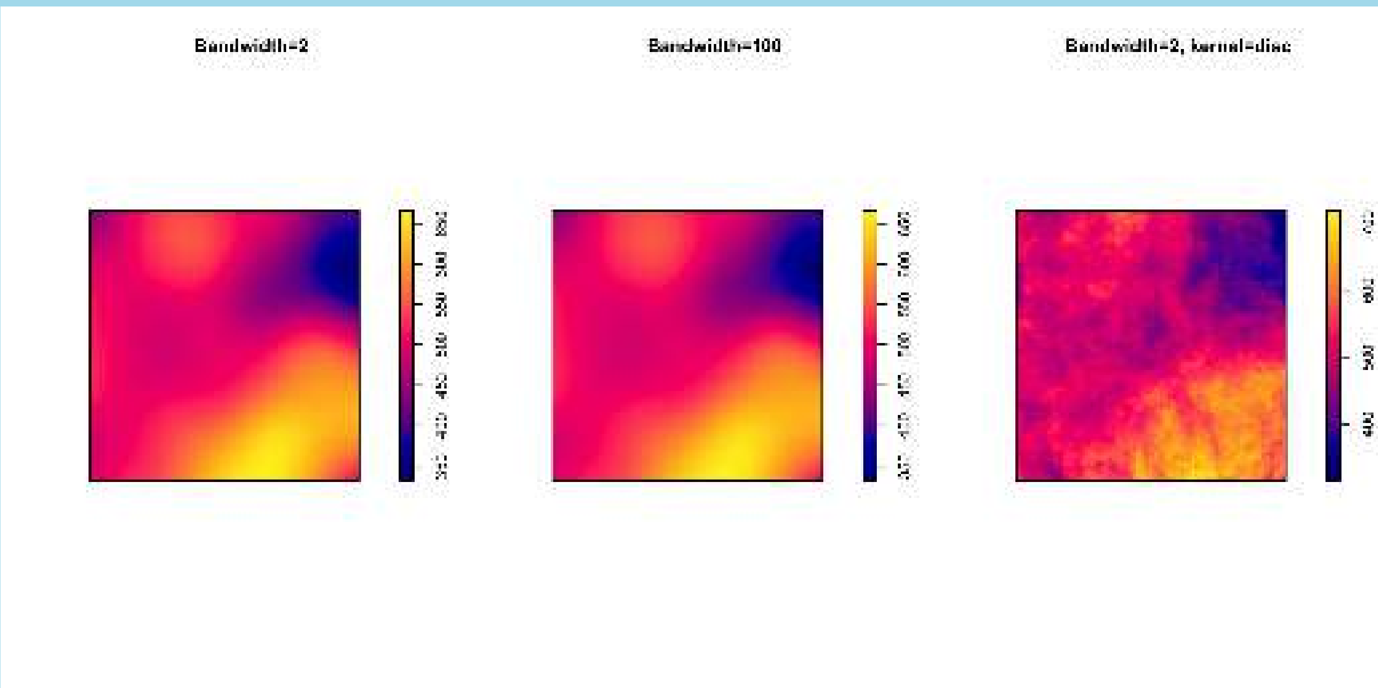
Kernel Density Estimates (KDE)

$$\hat{f}(x) = \frac{1}{nh_x h_y} \sum_{i=1}^{n} k\left(\frac{x - x_i}{h_x}, \frac{y - y_i}{h_y} \right)$$

::: {style="font-size: 0.7em"} * Assume each location in $\mathbf{s_i}$ drawn from unknown distribution

Kernel Density Estimates (KDE)

- h is the bandwidth and k is the kernel
- We can use `stats::density` to explore
- **kernel**: defines the shape, size, and weight assigned to observations in the window
- **bandwidth** often assigned based on distance from the window center



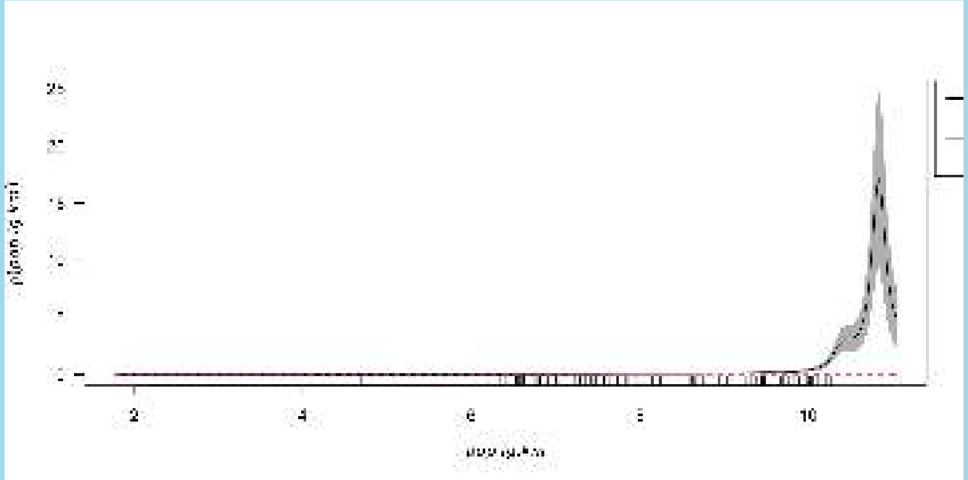
Kernel Density Estimates (KDE)

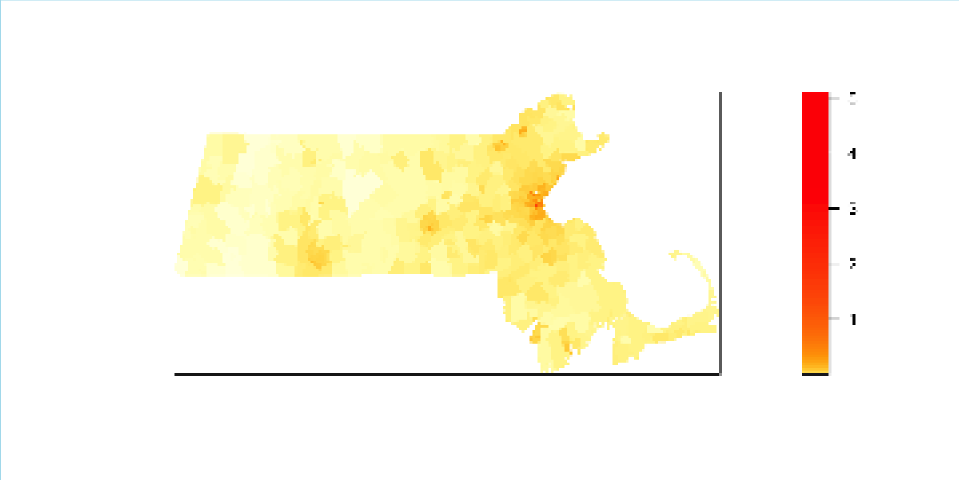
Adding a Covariate to KDE

- `rhohat` computes nonparametric intensity ρ as a function of a covariate

$$\lambda(u) = \rho(Z(u))$$

Adding a Covariate to KDE





Adding a Covariate to KDE

- We can also think more generatively
- Model explicitly as a Poisson Point Process using `ppm`

$$\lambda(u) = \exp^{\mathrm{Int}+\beta X}$$

```
# Create the Poisson point process model
PPM1 <- ppm(starbucks.km ~ pop.lg.km)
# Plot the relationship
```

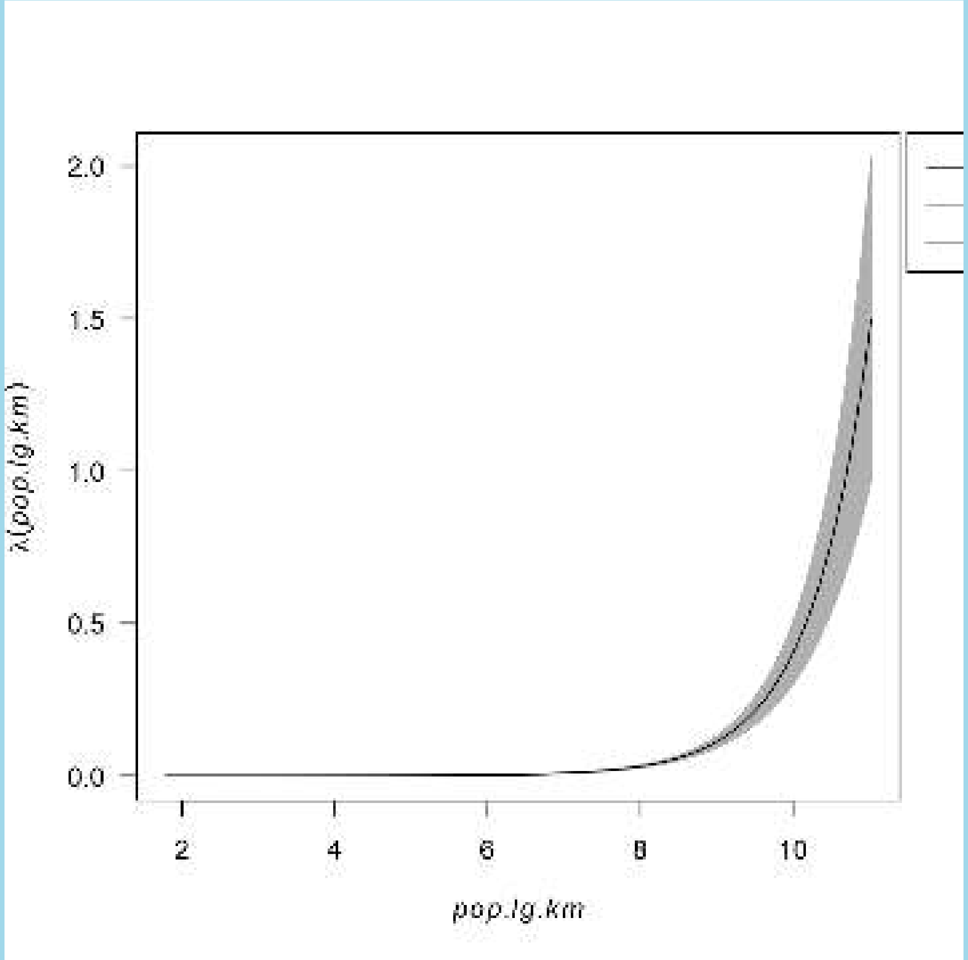
Adding a Covariate to KDE

```
Nonstationary Poisson process

Log intensity:  ~pop.lg.km

Fitted trend coefficients:
(Intercept)   pop.lg.km
 -13.710551    1.279928


              Estimate       S.E.
CI95.lo    CI95.hi Ztest      Zval
(Intercept) -13.710551 0.46745489
-14.626746 -12.794356   *** -29.33021
pop.lg.km     1.279928 0.05626785
1.169645   1.390211   ***  22.74705
Problem:
```



Overlays

Overlays

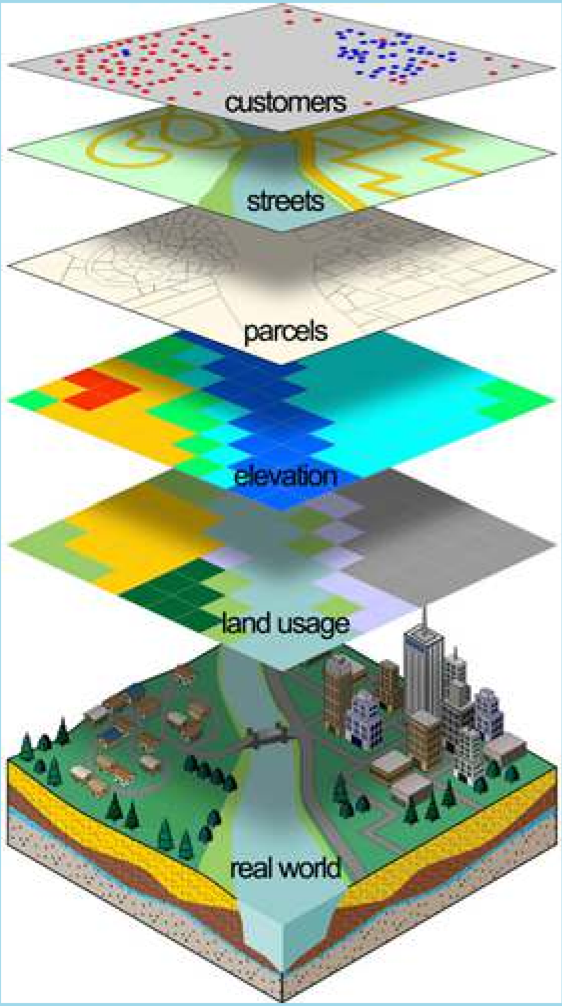
- Methods for identifying optimal site selection or suitability
- Apply a common scale to diverse or disimilar outputs

Getting Started

1. Define the problem.
2. Break the problem into submodels.
3. Determine significant layers.
4. Reclassify or transform the data within a layer.
5. Add or combine the layers.
6. Verify

Boolean Overlays

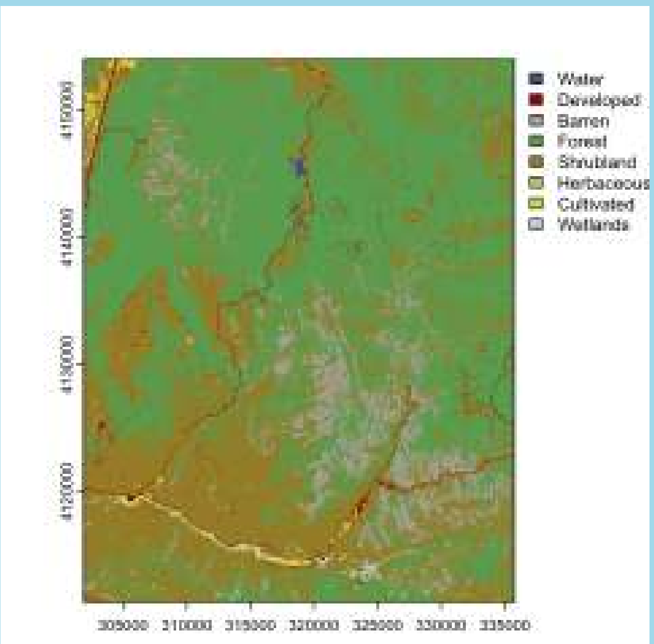
- Successive disqualification of areas
- Series of “yes/no” questions
- “Sieve” mapping



Boolean Overlays

- Reclassifying
- Which types of land are appropriate

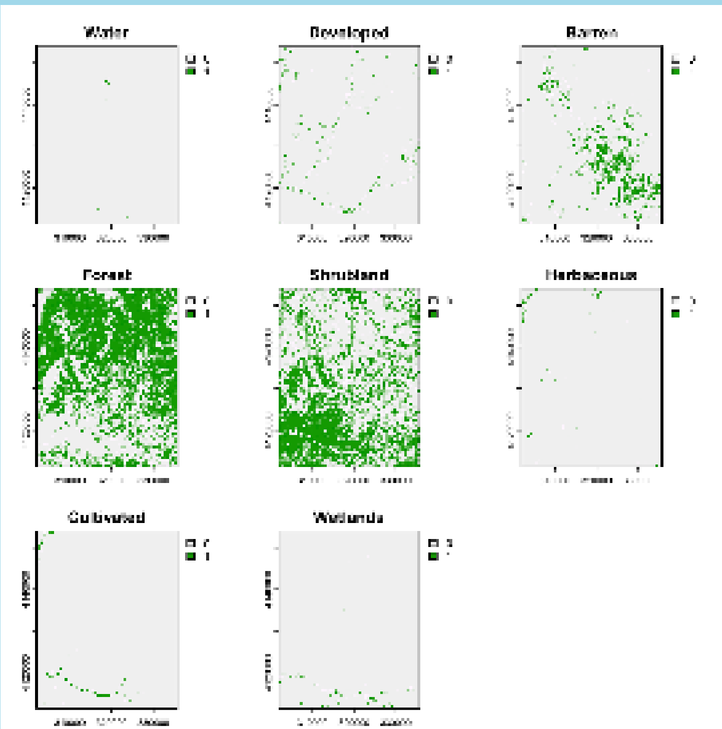
```r
nlcd <-  rast(system.file("raster/nlcd.tif", package = "spDataLarge"))
plot(nlcd)
```



Boolean Overlays

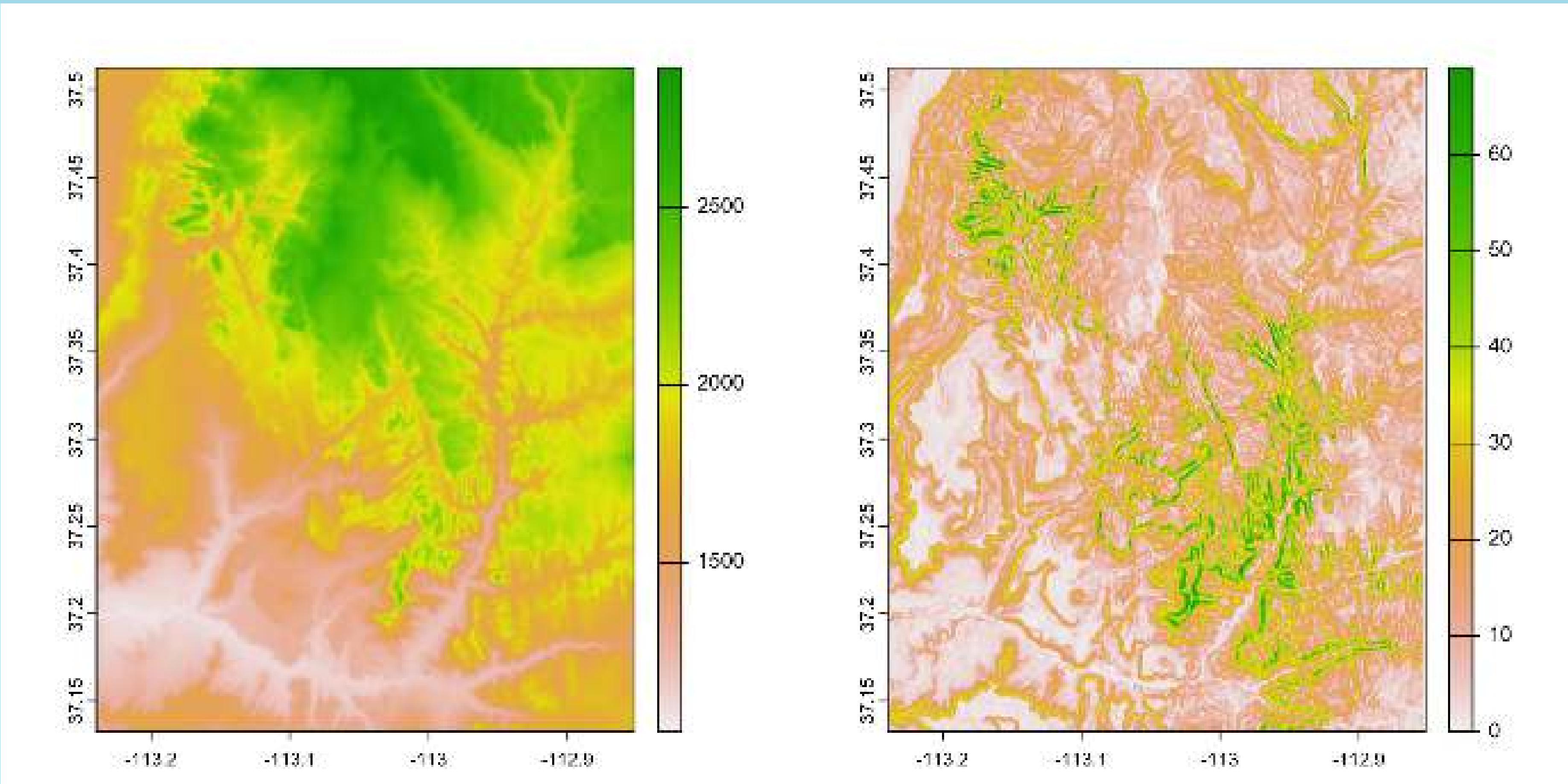
- Which types of land are appropriate?

```r
nlcd.segments <- segregate(nlcd)
names(nlcd.segments) <- levels(nlcd)[[1]][-1,2]
plot(nlcd.segments)
```



Boolean Overlays

- Which types of land are appropriate?

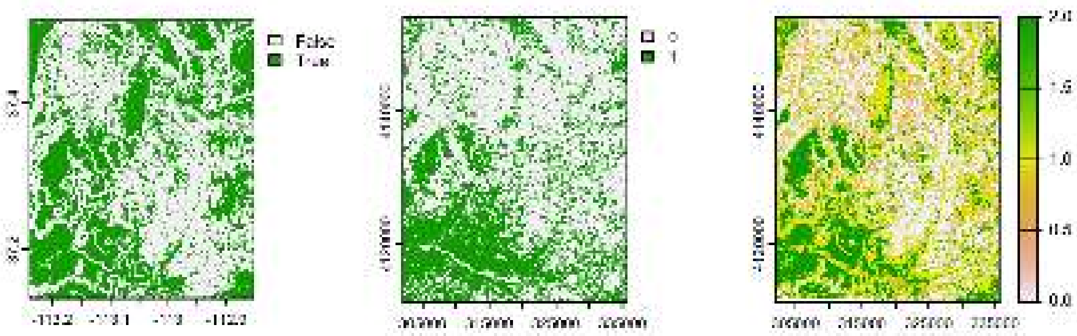


Boolean Overlays

- Make sure data is aligned!

```
suit.slope <- slope < 10
suit.landcov <- nlcd.segments["Shrubland"]
suit.slope.match <- project(suit.slope, suit.landcov)
suit <- suit.slope.match + suit.landcov
```

Boolean Overlays



Challenges with Boolean Overlays

1. Assume relationships are really Boolean
2. No measurement error
3. Categorical measurements are known exactly
4. Boundaries are well-represented

A more general approach

- Define a *favorability* metric

$$F(\mathbf{s}) = \prod_{M=1}^{m} X_m(\mathbf{s})$$

Weighted Linear Combinations

$$F(\mathbf{s}) = \frac{\sum_{M=1}^{m} w_m X_m}{\sum_{M=1}^{m} w_i}$$