Spatiotemporal models

"Space is the place" - Sun Ra

Samuel Robinson, Ph.D.

Oct 27, 2023

Outline

- Spatial and temporal data
  - Some basic GIS (`sf`)
- How to think about space and time
  - Plotting
  - Variograms
  - "Continuous" random effects
  - Kernels and
- Some common modeling approaches
  - GLS (covariance)
  - Basis functions (GAMs)

Spaaaaace

Some common problems

- My data were sampled over time or space. I'm not really interested in time or space *per se*, so can I just ignore them and run my models?
- I am actually interested in how something changes over time or space. Can I just use day or location (lat/lon) as another term in my model?
- My supervisor told me to look for something called autocorrelation, and it sounds scary

A common approach: random effects

"Can I just use day or site as a random effect?"

- Short answer: "Yes"
- Long answer: You might be able to do better, because of the **1st Law of Geography**:

"...everything is related to everything else, but near things are more related than distant things." Waldo Tobler

- If you have spatial or temporal information, this can help R to estimate random effects more accurately
  - Can improve prediction accuracy (smaller p-values)
  - Can give you hints about the underlying causal mechanisms

Part 1: Time and Space in R

How R deals with time

- Dealing with time in R is somewhat annoying, but not complicated
- Common methods: `as.Date` (days), `as.POSIXlt` (date + time)
- Both require a date/time format: see `?strptime` for examples
- You can transform to specific formats (e.g. day of year) using `format`
- `difftime` is useful for getting differences in time points

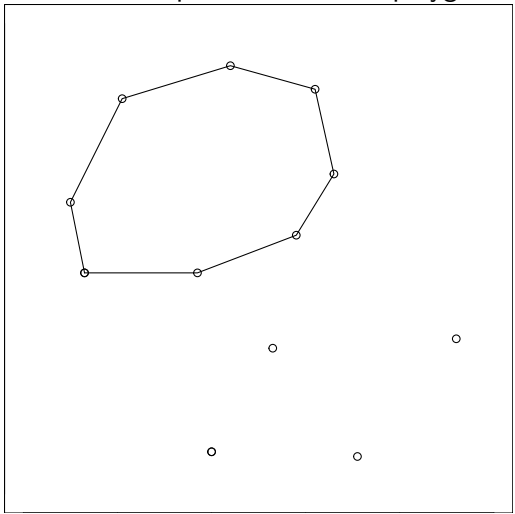
```
##    x         d1         d2
## 1  5 2010-05-06 2010-06-13
## 2 10 2021-11-14 2022-10-14
```

```
#Convert data to Date format
dateForm <- '%Y-%m-%d'
dExamp %>%
  mutate(across(c(d1,d2),
                ~as.Date(.x,format=dateForm))) %>%
  #Get day of year
  mutate(doy=format(d1,format='%j')) %>%
  #Get difference in time between d2 and d1
  mutate(dChange=difftime(d2,d1,units='days'))
```

```
##    x         d1         d2 doy  dChange
## 1  5 2010-05-06 2010-06-13 126  38 days
## 2 10 2021-11-14 2022-10-14 318 334 days
```

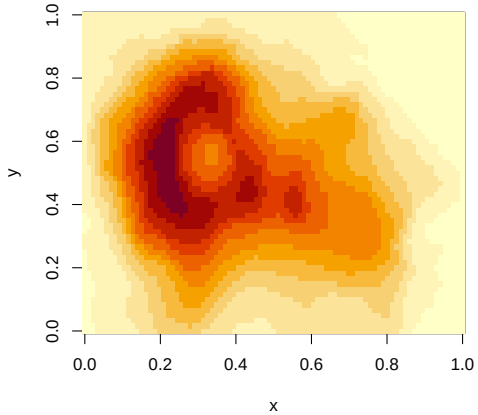
Two main types of spatial data

Vector data: points, lines, and polygons



Common R packages: `sf`, `sp`, `gstat`, `spdep`

Raster data: cells



Common R packages: `stars`, `terra`

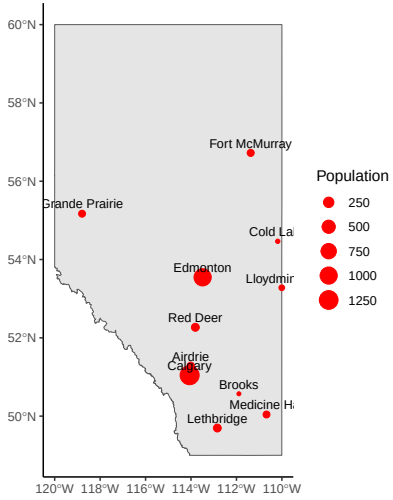
R as a GIS

- A **Geographic Information System** (GIS) is a system for organizing, analyzing, and displaying spatial information
- Common platforms and tools: ArcGIS, QGIS, PostGIS, Python
- A number of R packages are specifically written for dealing with GIS data, usually specific to raster or vector formats
- Ecologists mostly deal with vector data (site locations, boundary polygons) but raster data is sometimes used (NDVI, land cover classes)
- I'll show you a couple practical tips for using the `sf` package (see here also), but there are many other packages out there If you're dealing with large amounts of

  spatial data *I would encourage you to take a formal GIS course*, as there is a LOT to learn!

Common tasks: making maps

- Vector data are often encoded as *shapefiles* (set of several files)
- Point data can also be read in as *csv* files, which need to be turned into an `sf` object
- `sf` objects can be displayed in `ggplot` using `geom_sf`. Common aesthetics (colour, size) can be mapped onto the plot
  - Objects are layered on the map in order of coding
- Be careful: shapefiles can be very large, which can easily crash R!

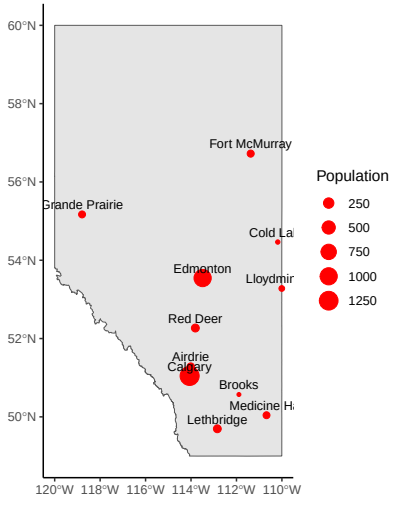


Common tasks: making maps (cont.)

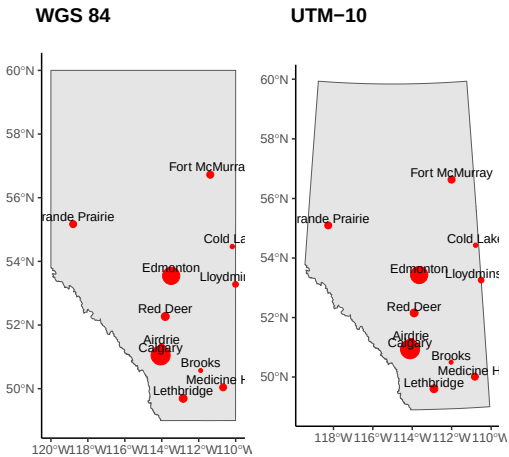
```
#Reads AB boundary shapefile
abBound <- read_sf('./shapefiles/AB_only.shp') %>%
  st_transform(4326)

#Reads city csv
csvPath <- './shapefiles/abCities.csv'
abCities <- read.csv(csvPath) %>%
  #Converts to sf
  st_as_sf(coords = c('lon','lat'),crs=4326)
#NOTE: crs 4326 is common lat/lon format

#Make map
(p1 <- ggplot()+
  #Add boundary
  geom_sf(data=abBound)+
  #Add cities
  geom_sf(data=abCities,aes(size=pop),col='red')+
  #Add labels
  geom_sf_text(data=abCities,aes(label=name),
  size=3,nudge_y=0.25)+
  labs(x=NULL,y=NULL,size="Population"))
```



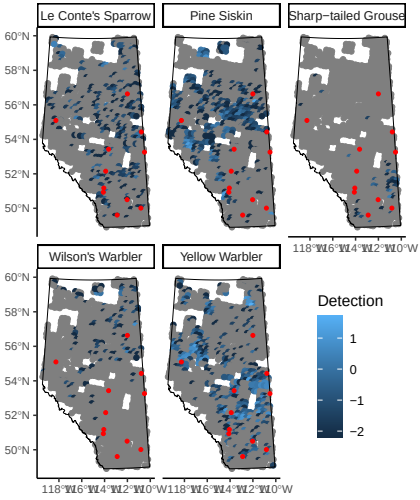
Common tasks: reprojection



- The world is not flat: all maps have to "bend" the data somehow. This is called the map **projection**
- Some map projections preserve *area*, others preserve *distance*. Degrees are not all the same distance apart!
- Usually we're interested in absolute distance between locations, so *Mercator* (UTM) is a good choice, but be careful which UTM zone you choose!
- `sf` uses `crs` codes: **4326** is for lat/lon (WGS 84), 3401 is an Alberta-specific UTM projection
- Many others are available

First challenge

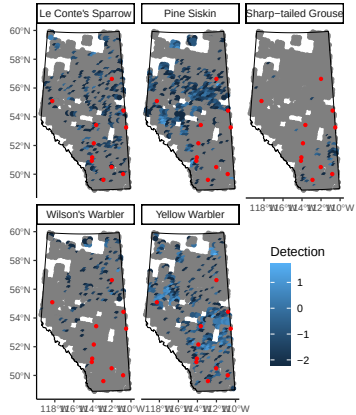
- Make this map of bird counts from the ABMI dataset
- `medDetects` is the median detection rate at each site over several years



First challenge results

```
abBound <- read_sf('./shapefiles/AB_only.shp') %>%
  st_transform(3401)
birdDat <- read.csv('./shapefiles/birdDat.csv') %>%
  st_as_sf(coords = c('lon','lat'),crs=4326) %>%
  st_transform(st_crs(abBound))
abCities <- read.csv('./shapefiles/abCities.csv') %>%
  st_as_sf(coords = c('lon','lat'),crs=4326) %>%
  st_transform(st_crs(abBound))

ggplot()+
  geom_sf(data=birdDat,aes(col=log(medDetects)))+
  geom_sf(data=abBound,fill=NA,col='black')+
  geom_sf(data=abCities,col='red',size=1)+
  facet_wrap(~Common.Name)+
  labs(col='Detection')+
  theme(axis.text = element_text(size=8),
        legend.position=c(0.85,0.25))
```



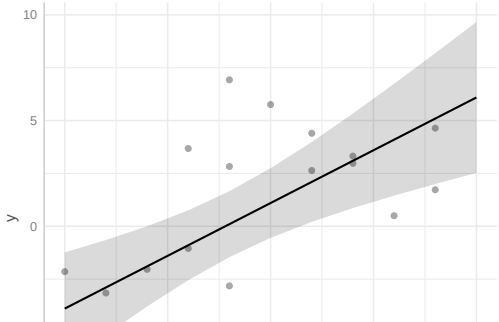
Part 2: Spatiotemporal modeling

Let's start with an example

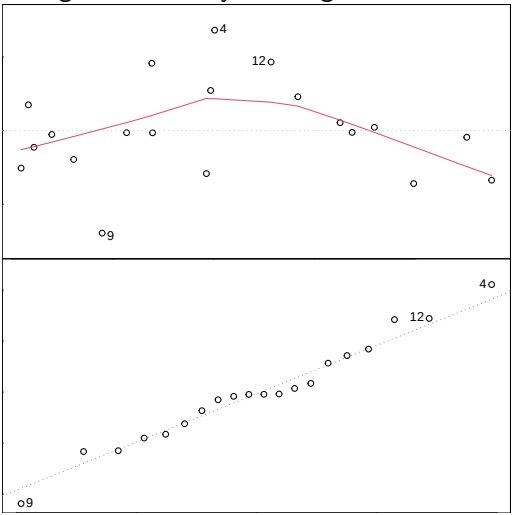
- Say we're fitting a simple linear regression on a dataset collected across space

```
## # A tibble: 6 x 5
##        y     x site    lat    lon
##    <dbl> <dbl> <chr> <dbl>  <dbl>
## 1 -3.46  -8.61 a     -1.79  5.79
## 2  0.378  5.76 b     -2.40 -3.67
## 3 -1.35  -4.61 c      3.62 -0.294
## 4  6.85  -2.15 d     -7.02  6.40
## 5  3.98   1.16 e     -7.33 -4.63
## 6 -1.86  -5.64 f     -1.20 -6.49
```

Model: lm(y ~ x)

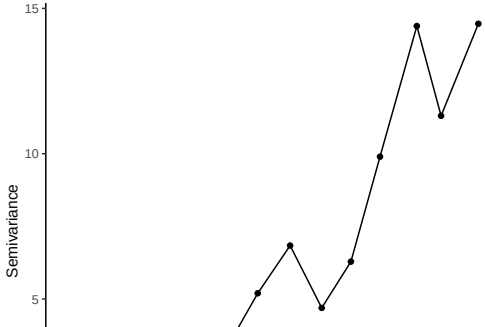


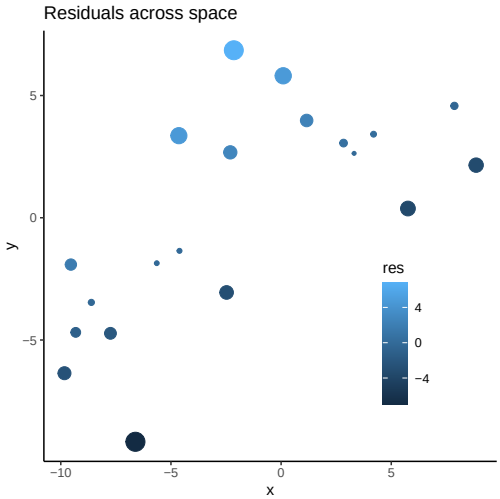
Things look mostly OK, right?



Spatial residual plot

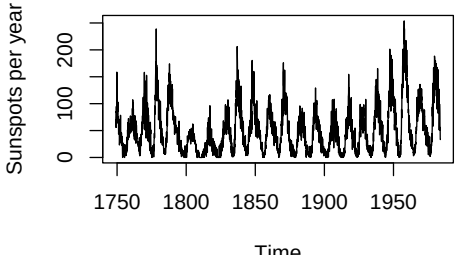
- Residuals are spatially *non-independent*!
- *Variograms* are a common tool to examine how variance changes with distance
- Uncorrelated spatial data will have a *flat* variogram (no change in semivariance with distance)

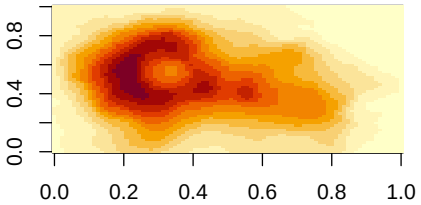




Temporal or Spatial Data

- Correlation is often present in temporal data or spatial data; causes may be unknown or "uninteresting"
- Usually we are interested in accounting for these patterns, in order to better estimate the "interesting" patterns on top of them
- Last week we talked about *cross*-correlation (i.e. correlation between columns of data); this week we're talking about *auto*-correlation (i.e. correlation between individual data points in a single column)





Covariance

- Normal distributions[1] don't just have a single σ, but a matrix of values
- If our data y are *independent*, then it looks like this:

$$y \sim Normal(M, \Sigma)$$

$$M = [\mu_1, \mu_2, \mu_3]$$

$$\Sigma = \begin{bmatrix} \sigma^2 & 0 & 0 \\ 0 & \sigma^2 & 0 \\ 0 & 0 & \sigma^2 \end{bmatrix}$$

- Zeros mean "μ_1, μ_2, & μ_3 aren't related to each other"
- Diagonal elements = *variance*, off-diagonal = *covariance*

[1] Multivariate Normal

Covariance and Correlation

In real life, things may not be independent from each other. For example:

- $\sigma = 2$ (variance $= \sigma^2 = 4$)
- μ_1 and μ_2 are strongly correlated (r=0.7), but μ_3 is not related to anything (r=0). Shown here as a *correlation matrix* (R):

$$R = \begin{bmatrix} 1 & 0.7 & 0 \\ 0.7 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- When multiplied by the variance, this becomes the *covariance matrix* (Σ)

$$\Sigma = \begin{bmatrix} \sigma^2 \times 1 & \sigma^2 \times 0.7 & \sigma^2 \times 0 \\ \sigma^2 \times 0.7 & \sigma^2 \times 1 & \sigma^2 \times 0 \\ \sigma^2 \times 0 & \sigma^2 \times 0 & \sigma^2 \times 1 \end{bmatrix} = \begin{bmatrix} 4 & 2.8 & 0 \\ 2.8 & 4 & 0 \\ 0 & 0 & 4 \end{bmatrix}$$

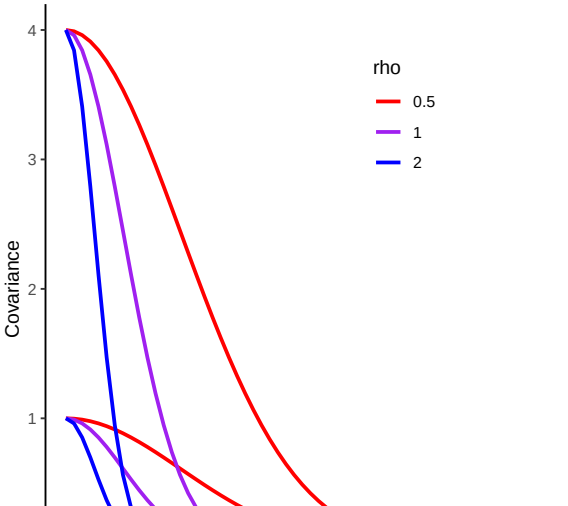
Gaussian Process Modelling

- We can model covariance between things as a function of *distance*, either in time or space
- Squared-exponential is fairly common[2]:

$$\Sigma = covariance$$
$$\Sigma = variance \times correlation$$
$$\Sigma = \sigma^2 \times e^{-\rho^2 Dist^2}$$

- Instead of finding a single σ value, R now looks for σ (maximum covariance) and ρ (decay with distance)

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplo
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this war
## generated.
```

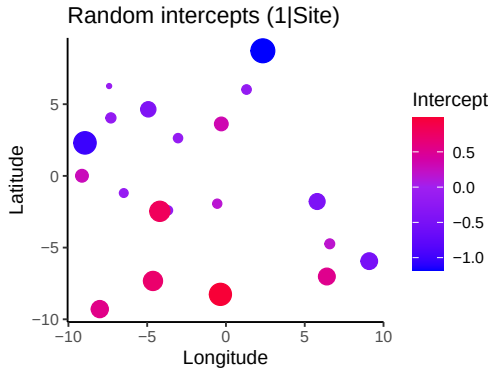


Spatial random effects

- Say that we collected data at 16 sites, and we're interested in the effect of y on x
- Let's first fit a model with a random intercept for site

```
#Same syntax as lmer models:
lmm2 <- glmmTMB(y~x+(1|site),data=dat2)
```

- If we plot the intercepts for each site, we see that they are clustered:



Spatial random effects (cont.)

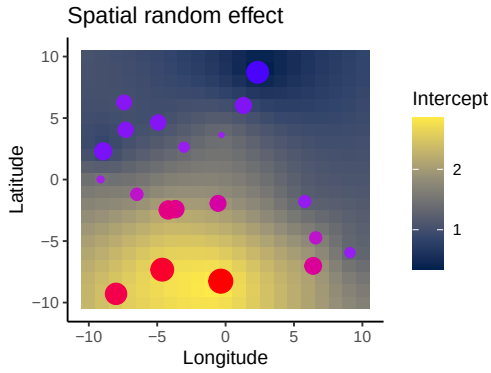
- Re-fit model with a spatial (exponential) random effect

```
#Coordinates
dat2$coords <- numFactor(dat2$lon,dat2$lat)

#Group factor (only 1 here)
dat2$group <- factor(rep(1,nrow(dat2)))

#Fit model with spatial random effect
lmm3 <- glmmTMB(y~x+exp(coords+0|group),data=dat2)
```

- Clustering effect modeled as a spatial random effect



Challenge

Problem: hard for large datasets

Solution: basis function

A challenger approaches

- Ho ho ho! Merry Christmas! In order to maximize the number of presents that you get from Santa Claus, you've decided to apply an analytic approach, and have collected data across Alberta on *number of Christmas presents received*
- You've also collected data on things that might influence Saint Nick's generosity (*naughtiness*, *presence of milk and cookies*, *chimney width*)
- Fit a GLMM to the present data, one using spatial random intercepts, and one using "regular" random intercepts
- Which type of snack should you leave out for Santa? Which area might you consider moving to??

Two-column slide