Wildlife Ecology Modeling

Class Stuff

- **Instructor:** Brian Gerber
- **Classroom**: NR 243
  - Lecture/Lab/Discussion
- **When**: Tu 11am - 12:15pm and Fr 10am - 1pm

. . .

- **My Office**: 202A Wagar, Colorado Cooperative Research Unit
  - Office hours: TU 1:30pm - 2:30pm and by appointment
  - brian.gerber@colostate.edu

Class Stuff

- **Computers**: Bring laptop to class (especially Friday)
  - Software/R Packages should be installed prior to lab (posted on website)
- **Registration**:
  - 670-001 (CRN 77992) and FW 670-L01 (CRN 77993)

What is this course?

A mix of...

- statistics
- modeling
- math / notation
- coding
- science philosophy
- wildlife ecology and conservation

Assessment

Assessment Components	Percentage of Grade
Course Engagement	10%
Lab Assignments	40%
Discussions	10%
Quizzes	10%
Project	30%

Project

1) independent research project - highlighting a modeling application, data/code transparency, and communication of results
2) group development of a lecture and lab case-study that showcases a statistical application relevant to animal ecology and conservation

* more on the main site (5th tab)

Course Learning Objectives

Upon successful completion of this course students will be able to:

- think 'statistically'
- read quantitative ecology literature
- write code to fit and interpret complex statistical models relevant to wildlife ecology and conservation
- communicate statistical approaches and results

Why is this class useful?

- Able to read modern ecological literature
- Understand what you are doing when using data and models; **coding/statistics/modeling are related but not the same**
- Statistical modeling and coding skills are highly marketable
- Taking control of your analyses
- Collaborate with colleagues/statisticians

Software

Why learn to code?

- efficiency

Figure 1: R and RStudio logos

- transparency
- flexibility in application
- shareable
- marketable skill
- needed for publications

Why use R?

- open-source and free
- small total user base / large in ecology and statistics
- lots of resources online
- data management (meh)
- statistics
- plotting / graphics

Why use RStudio?

- Makes using R easier
- Projects (file mgmt)
- R Shiny: Interactive online apps
- R Markdown: Interactive documents
- Quarto: interactive articles, websites, blog, …
- Posit - Certified B corp

Type of Modeling

- parametric
- probabilistic
- generative
- inferential and/or predictive

Statistics in the Modern Age

. . .

Coding in the Modern Age

- Software changes all the time
- Code will become obsolete
- Base R functions change slower than packages
- Document/Annotate code and publish it online
- Organize your files! Use sub-folders

How do you use generative AI?

A Graduate Student's Dilemma

You need to know....

- field techniques
- logistics / planning
- people/advisor management
- institutional bureaucracy
- ecological theory
- wildlife mgmt principles
- conservation biology principles
- study design
- data management
- public speaking
- independent and team work
- graphic/visual arts
- 'the literature'
- the job market
- how to write a manuscript/thesis
- writing/sharing coding
- statistical modeling
- ...

Learning

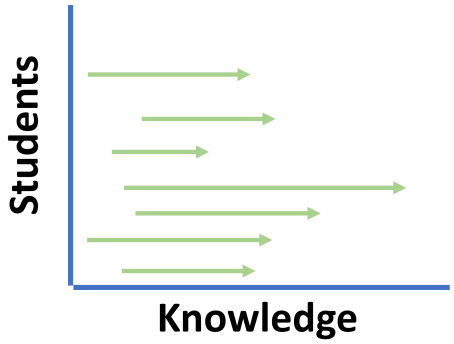

Figure 2: XY visualization of students and knowledge

I do not know where you are starting

Our Aim

What model did you fit? Why? How?

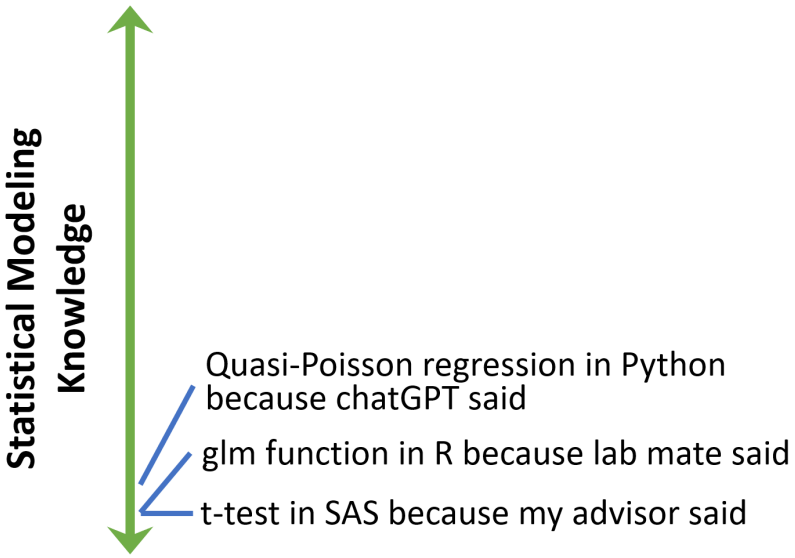

Figure 3: Extremes of how and what models get fit

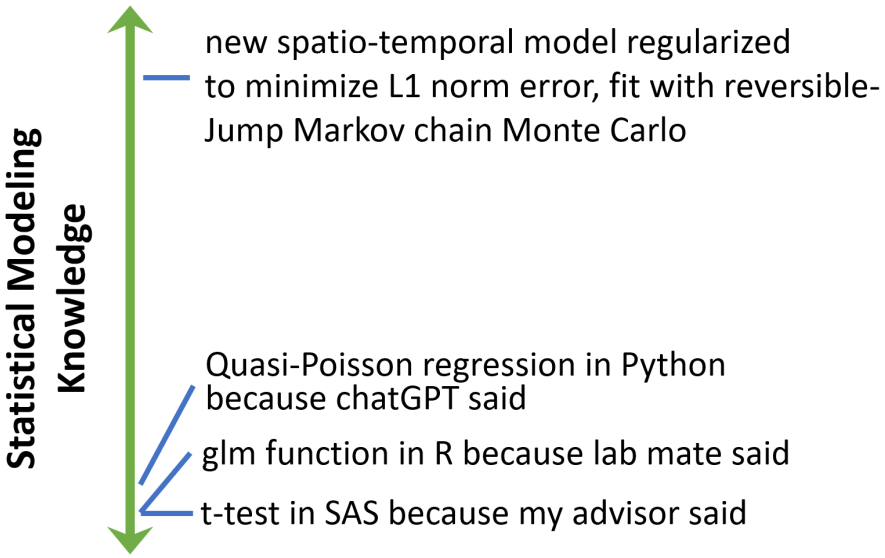

Figure 4: Extremes of how and what models get fit full

Think Statistically

Know...

- your objective in fitting a model
- the model and its properties (not just the name)
- how to interpret ALL the parameters
- how the parameters are being optimized
- the consequences of your modeling decisions
  - requires reading statistical literature
  - requires evaluating assumptions yourself

Why is statistics and ecological modeling so difficult?

My Background

Figure 5: Brian doing fieldwork

My Background

Figure 6: Contributing to workshops

Science Practice

I am a pragmatist

. . .

There are many ways to do ***great*** science

. . .

Disciplines have conventions

. . .

There are foundations of scientific and statistical learning

. . .

Know the *why* of your decisions

. . .

Ask lots of questions to everybody all the time

Teaching Philosophy

- Learning is a choice (in each moment)
- An inclusive environment is paramount for learning
- Communication is key
- Everyone has something to teach and something to learn
- Struggle is good. Solving problems leads to learning
- BUT….

Course Website

- **Lectures, code, readings, schedule, etc.**
  - https://bgerber123.github.io/FW670/
- **Homework, Discussion Posts, Quizzes, and Grading:**
  - https://colostate.instructure.com/courses/228738

Figure 7: Question Mark

RStudio

What does each panel do?

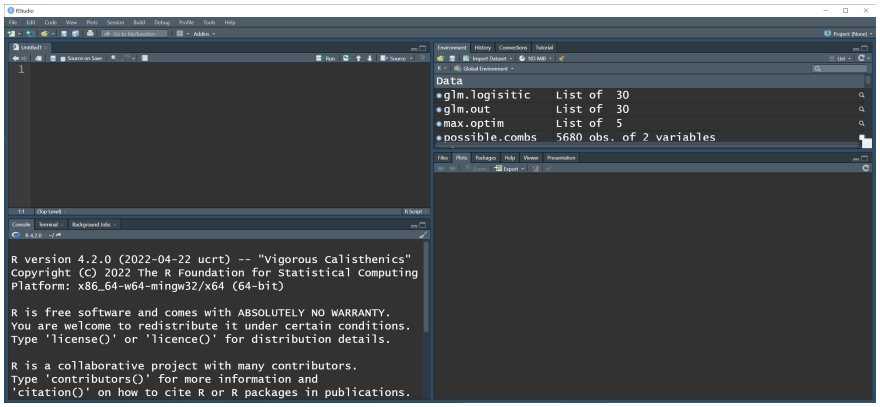

Figure 8: RStudio 4 panels

The language of R

. . .

Objects

A storage place for information; stored in the "Environment"

'Attributes' describes the structure or information of the object

The language of R

Objects

```
# y is an 'object' that is assigned the value 3
y = 3
y
```

```
[1] 3
```

The language of R

Values

- numeric
- integer
- character
- factor

Objects

- vector

- matrix
- array
- list
- dataframe
- S3, S4, S5, and beyond

The language of R

Functions

'does stuff'; creates or manipulates objects

'*Arguments'* are the types of things a function is asking for; the inputs

The language of R

object = function(argument1 = input1, argument1 = input2)

. . .

object = function(input1, input2)

. . .

this = sign(x = -5)

```
sign(-5)
```

```
[1] -1
```

```
sign(5)
```

```
[1] 1
```

Some useful functions

for loops

```
save.this = c()
for(i in 1:10){
  save.this[i] =  1-i
}
```

Some useful functions

Create your own function

```
my.mean.func = function(x){
                            sum(x)/length(x)
                           }
```

```
my.mean.func(
             c(5,4,7,2,7,1)
             )
```

```
[1] 4.333333
```

Some useful functions

apply/sapply/lapply/vapply

```
mat= matrix(rnorm(100),nrow = 10, ncol = 10)

apply(mat, 2, FUN = median)
```

```
 [1]  0.1701061  0.4506905  0.3500996  0.4001541  0.6768034 -
0.1834669
 [7] -0.0160180 -0.6371736 -0.3363563 -0.6696376
```

```
apply(mat, 2, FUN = function(x){

  length(which(x>1))/length(x)

})
```

```
 [1] 0.2 0.2 0.2 0.2 0.3 0.3 0.3 0.2 0.1 0.2
```

Code Organization

- Hierarchical code organization
  - code structure using indenting
  - top –> bottom execution

. . .

Help! My code doesn't work…

```
                cor.sp.route.cor=vector("list",n.species)
cor.sp=rep(NA,n.species)
            for(s in 1:n.species){
route=new.cov.species.long.scaled[[s]]$routeID
cor.sp[s]=cor(patch.size20.species.scaled.mat.center.route[s,],patch.count20.species.scaled.
    for(i in 1:nroutes){
temp1=patch.size20.species.scaled.mat.center.route[s,which(route==route.id[i])]
  temp2=patch.count20.species.scaled.mat.center.route[,][s,which(route==route.id[i])]
  if(length(temp1)>5){
  cor.sp.route.cor[[s]]=abs(c(cor.sp.route.cor[[s]],cor(temp1,temp2)))
  }}}
```

Better...

```
# Create Storage objects
  cor.sp.route.cor=vector("list",n.species)
  cor.sp=rep(NA,n.species)

#loop over species
  for(s in 1:n.species)
    {
      route = new.cov.species.long.scaled[[s]]$routeID

      cor.sp[s] = cor(patch.size20.species.scaled.mat.center.route[s,],
                      patch.count20.species.scaled.mat.center.route[s,]
                      )
        # loop over species and routes
        for(i in 1:nroutes)
          {
            temp1 = patch.size20.species.scaled.mat.center.route[s,which(route==route.id[i])
            temp2 = patch.count20.species.scaled.mat.center.route[,][s,which(route==route.id
            if(length(temp1)>5){
                                cor.sp.route.cor[[s]] =abs(c(cor.sp.route.cor[[s]],cor(temp1
                                } #End if statement
          } #End routes loop

    } #End species loop
```

Figure 9: Question Mark