Analysis of Environmental Data

Deck 1: Introductions

Eco 602 – University of Massachusetts, Amherst – Fall 2021
Michael France Nelson

Welcome to Analysis of Environmental Data!

Course Meetings

- Tuesday and Thursday:
  - 8:30 – 9:45 AM
  - Holdsworth Hall 308
- Course meetings consist of:
  - [Brief] Lecture segments
  - Small group activities
  - Q+A and collaborative work time
  - Many more details in the syllabus!

Office Hours

- Mike's office hours:
  - Tuesday 1:00 PM – 2:00 PM
  - By appointment
  - Holdsworth Hall 311
  - Remotely via Zoom (see link in Moodle)
- Anastasia's office hours:
  - TBA

About me

My Interests and Roles at UMass

- Plant biology, invasion ecology and climate change
  - RISCC
- Spatial analysis
- Modeling Data: statistical and simulation models
  - QSG: Quantitative Sciences Group
- [Less un-] sustainable agriculture: Umass Cranberry Bog

Courses I Teach

- Analysis of Environmental Data + Lab
- Intro to Quantitative Ecology
- Spatial Analysis using R
- Intro to Geographic Information Systems (GIS)

About Anastasia

My Interests	Courses I have taken

• The intersections between climate change, the environment and health
  • Green prescriptions
  • Policy recommendations for health professionals, city planners, and environmental agencies
  • Education
  • Sustainability
  • Modeling current and future ecosystem services
  • Data analysis
	• Courses I took when completing my Masters at UMASS:
	  • ECO 602, 675, 699, 691A, 601
	  • GEO-SCI 591CM, 701
	  • NRC 528, 585
	  • EPI 630
	• Courses I'm taking this fall as a PhD student:
	  • ECO 899 (PhD dissertation)
	  • EPI 690 (Environmental Epidemiology)
	  • ECO 622 (Conservation Biology)
	  • NRC 597 (Analytical Methods, Energy & Climate Policy)

Mask Policy

I'd like to keep it on, please.

https://www.umass.edu/coronavirus/

Are face coverings required on campus?

- The university has adopted a face-covering requirement for indoor public spaces designed to deter spread of the COVID-19 Delta variant.
- The mask requirement starts Aug. 11 and will be reviewed in mid-September.
- All members of the UMass community — students, faculty and staff —as well as contractors and visitors are required to wear face coverings in public indoor spaces on the UMass Amherst campus.

What is this course about?

- This course is really about Modeling and Model Thinking.
- It’s easy to forget the big-picture as we dig deep into technicalia, but Model Thinking is the overarching theme.

Data	**Statistics**

• How to record data?
• Data types/scales.
• Messy, noisy, or incomplete data.
	• Dual Model Paradigm
	    • Deterministic functions
	    • Probability distributions
	• Quantifying uncertainty
	• The constellation of statistical models

How can I succeed in this course?

- Collaborate with your peers
- Try not to get behind.
  - Life happens. If you do get behind, it's ok. Please reach out to me ASAP if you are having trouble.
- Know that you will be frustrated with R, this is ok. Remember that R is a tool we use to turn data into information.
  - A complicated tool, but a tool that you can learn (I promise) with practice.
- Work with your peers on the 'individual' assignments.
  - Just make sure to list who you worked with!
- Ask lots of questions of me, your peers, and Dr Google.
- Be active in the in-class activities.
- Have fun and be creative. Laugh at my corny puns, jokes, and gaffes!
- Attend office hours
- Work with your classmates as much as possible.

n

- There is a flurry of set-up and foundational assignments at the beginning.
  - You should already be working on the Software Setup assignment already.
    - If you haven't done so, request Azure Virtual Desktop ASAP. It takes a few days; you won't be able to submit the assignment until you gain access.
  - You should begin the DataCamp and RNotebook assignments too.

- We're not going to cover the whole syllabus in class, you should check it out on your own. An important point, however:
  - Academic Honesty:
    - You'll be collaborating a lot. When you submit individual assignments you should work with your peers, but your answers must be in your own words. You'll have a chance to list the students you worked with on the individual assignments.
    - Do not cut and paste from online sources, R help entries, or Wikipedia articles without crediting the source. I can tell when you do.

- Assignments will generally be due Sunday night by 11:59PM.
  - I'll try to point out exceptions, but it is up to you to make sure you are aware of due dates.
- Speaking of due dates, the first few assignments are due relatively soon:
  - Software setup: Sep 10 (this is a Friday)
  - Using R Notebooks: Sep 12
  - Data camp – intro to R: Sep 12
- I've designed the in-class assignments to be completable within the class period.
  - In general, you should aim to submit your group's work by the end of class, or at least by the end of the day.
  - Your feedback regarding timing will be helpful.
- Slide decks: I'm working to get all of the 2020 slides updated and posted ASAP.
  - I frequently tinker with slides up until the day of lecture. You should plan to download a fresh copy after the corresponding lecture.

Outline for Thursday Sep 2

- Brief lecture on computing for scientists
- In-Class Model Thinking assignment
- Time for questions at the end

Model Thinking

What is a model?

Objectives

Understand what **model** means in the context of this course.

Learn some of the important classes of models.

Understand the basics of model thinking and the model building process.

What is a model?

Here's a (very) simple, but effective definition:

- A model is a simplified version of reality.

We use models every day

- Can you think of any that you have already used today?

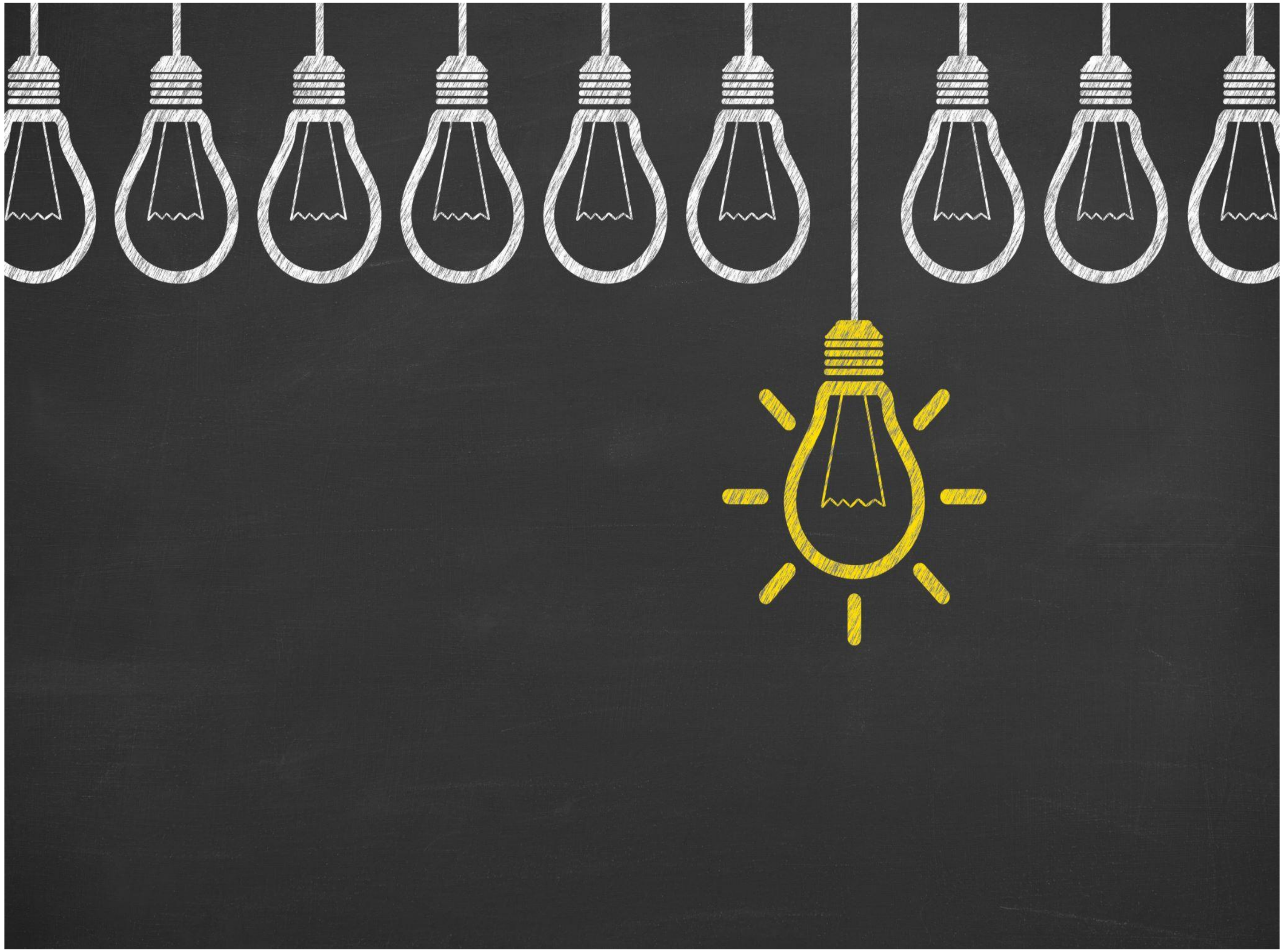
We might take these for granted, but they are all models:

- Sunrise/sunset times
- Seasons: expected phenology
- Tides and moon phase predictions
- Weather forecasts
- Mental geospatial models:
  - The layout of your house/apartment
  - The streets of your hometown
- Social cues
- Language?

What is model thinking?

Model thinking means:

- Learning to see potential models everywhere.
- Being aware of models we already use.
- Recognizing benefits, pitfalls, and limitations of a model.
- Considering multiple models.
- Thinking **iteratively.** Updating your model



A very cute example

Clarification: What is a system?

A definition of system from Merriam-Webster:

1. a group of interacting bodies under the influence of related forces
2. a group of body organs that together perform one or more vital functions
3. a group of related natural objects or forces

Model Thinking: How to think like a modeler

A model thinker would first consider:	Then they might:
• What do I already know about this system? • What are some of the important parts of a system? • What are some of the important interactions? • What questions are important?	• ponder • draw some sketches • write down their ideas • discuss with friends or colleagues… • Give up and try again tomorrow

Model Thinking: How to think like a modeler

Modeling is Iterative

In the second round the model thinker might consider questions like:

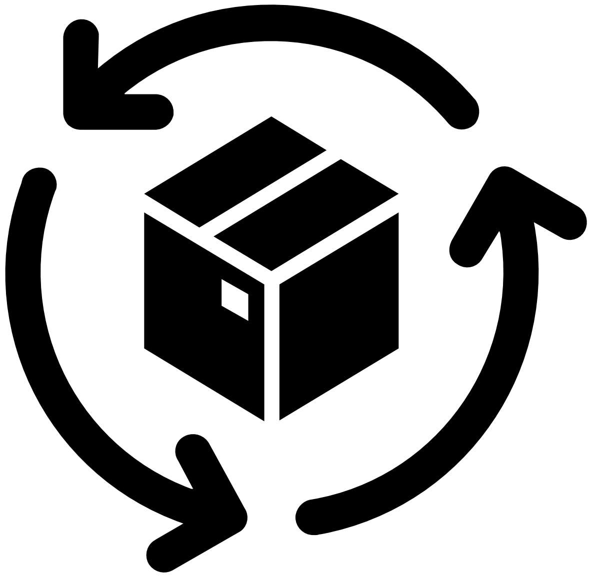
- What components are vital, and which could I ignore?
- Can I identify knowledge gaps and known unknowns?
- How could I create a computer simulation?
- What if my model is wrong?
  - Can I hypothesize a competing model?
- Can I identify biases in my model?
  - Cultural
  - Species
  - My personal background and training

Model Thinking: How to think like a modeler

As the model thinker refines their model, further questions they might consider are:

- Why do I want to build a model?
- What will I use the model for?
  - Prediction? Understanding? Experimenting?
- What data could I use to calibrate or validate my model?
- What are the sources of uncertainty and randomness?
- What if I discover my model is wrong?

A model thinker knows that modeling is iterative…



What can we learn from the dog's model?

I want to hear your ideas!

Classes of Models

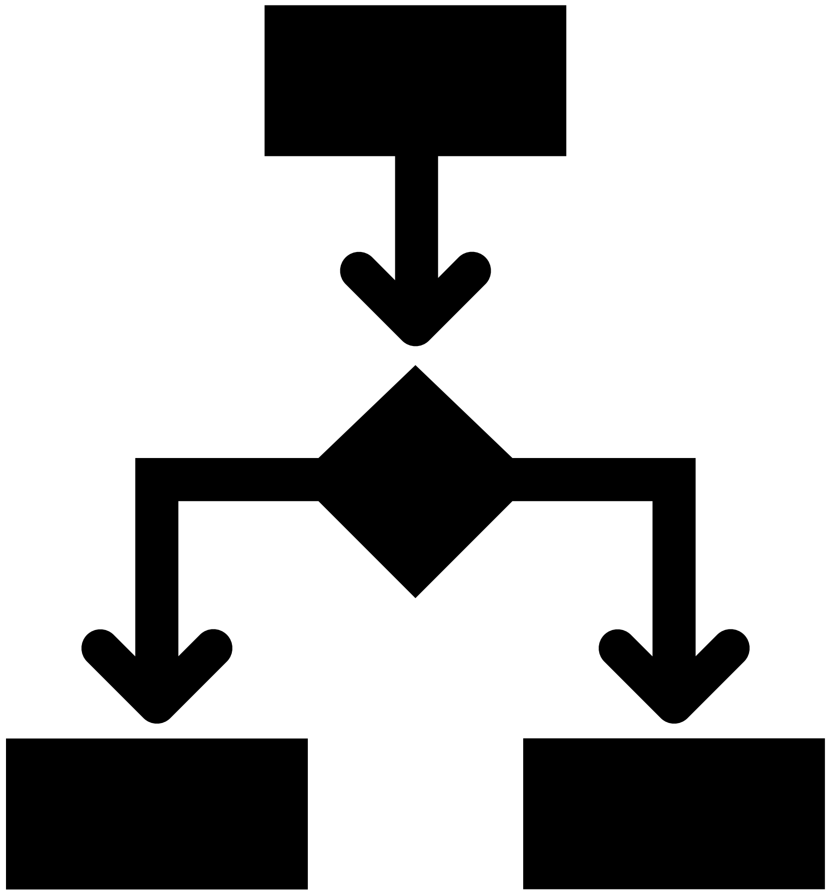
Modeling is a very broad concept.

The model we build depends on our purpose, what we already know, and the data we have!

Some categories of models include:	These categories are not mutually exclusive.
Conceptual Phenomenological Mechanistic	• Since modeling is iterative, you may use (for example) a conceptual model to inform an experiment. • You could then use a phenomenological model to explain patterns you observe in the results. • This may support or refute your conceptual model, or help you develop a mechanistic model.

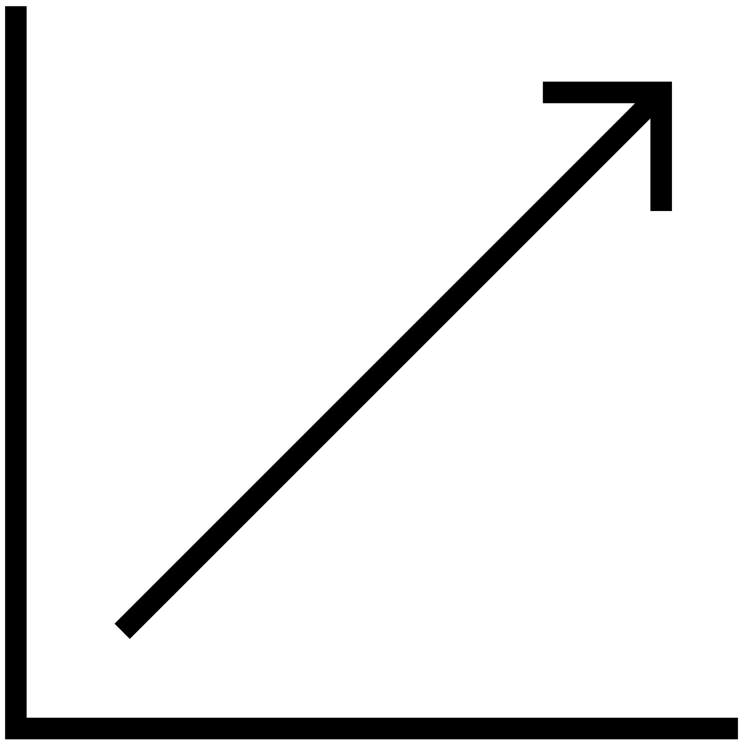
Conceptual Models

- Conceptual models are useful to identify important components and hypothesize interactions.
- Conceptual models don't have to be **quantitative**.
- A flow chart, or concept diagram can be a conceptual model.



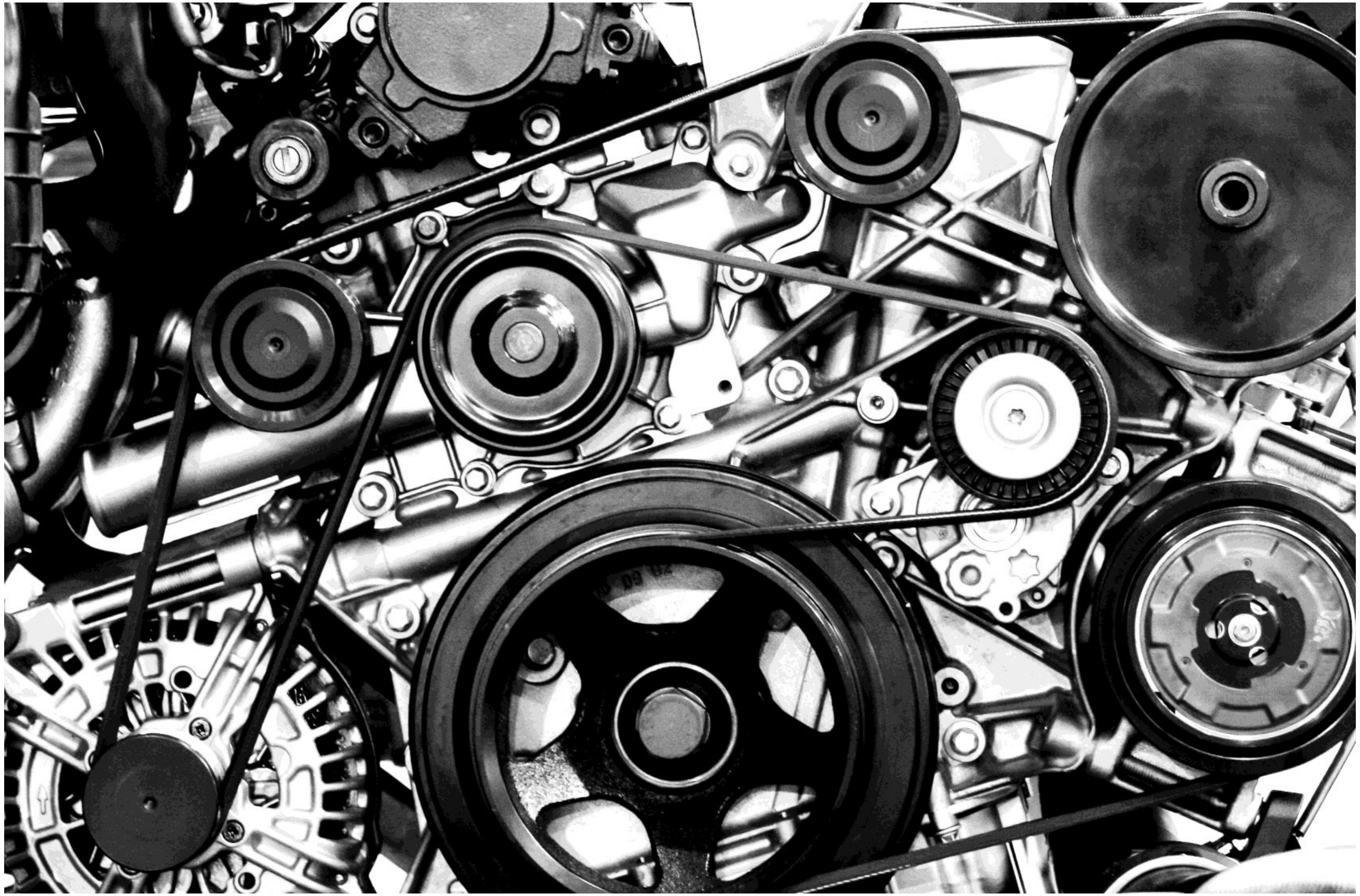
Phenomenological Models

- Phenomenological models use **quantitative tools** to *describe* observed patterns.
- The form of a phenomenological model doesn't have to reflect the structure of the underlying system.
- Note: *phenomenological* is incredibly hard to spell, and to pronounce so please be patient with me and each other!



Mechanistic Models

- Mechanistic models use **quantitative tools** to *describe* observed patterns.
- Mechanistic models use a known or hypothesized structure of the underlying system to specify the model form.



Some Model Examples

- What are the important components in a mangrove estuary in southern Florida, and how do they interact?
    - Davis et al (2005)
- What are the advantages and differences among fragmentation, variegation, and contour landscape models?
    - Fischer, Lindenmeyer, and Fazey (2004)
- What predictor variables best explain plant presence/absence and cover in vernal pools?
    - Ray and Colligne (2014)
- How well can models of of connectivity describe plant presence/absence and cover in vernal pools?
    - Ray and Colligne (2014)
- Do black bear responses to resource availability conform the Ideal-Free Distribution Model?
    - Beckmann and Berger (2003)

What is a model?

In the words of some famous model thinkers:

"All models are wrong, but some are useful" - George Box

Companion In-Class Modeling Activity

Let's Model!

Now that you have an idea of what I mean by model thinking, let's put our skills to work using some real ecological scenarios.

Click here to link to the assignment description.

You can also find it on the course github site under 'In-Class Activities'

Please make sure you read through the assignment instructions and the scenarios before you come to class.

References for the example models

- Beckmann, J.P., and Berger, J. (2003). Using Black Bears to Test Ideal-Free Distribution Models Experimentally. Journal of Mammalogy 84, 594–606.
- Davis, S.M., Childers, D.L., Lorenz, J.J., Wanless, H.R., and Hopkins, T.E. (2005). A conceptual model of ecological interactions in the mangrove estuaries of the Florida Everglades. Wetlands 25, 832.
- Fischer, J., Lindenmayer, D.B., and Fazey, I. (2004). Appreciating Ecological Complexity: Habitat Contours as a Conceptual Landscape Model. Conservation Biology 18, 1245–1253.
- Ray, C., and Collinge, S.K. (2014). Quantifying the dominance of local control and the sources of regional control in the assembly of a metacommunity. Ecology 95, 2096–2108.

Computing for Scientists

Objectives

Understand	Understand	Understand
Understand filesystems and paths.	Understand differences among command-line, menu-driven, and script-based analyses.	Understand file concepts: binary and text encoding, extensions, and archive files.

Important Concepts and Details

Key Concepts

- File systems
- Path
- Graphical and text interfaces
- File formats

Key Terms

- Files and directories
- Absolute path, relative path
- Home directory, Working directory
- text files, binary files, archive files
- File extensions and application associations
- Operating System: Graphical user interface and command line interface.

Content Note

Some or all of the material in this lecture deck may be review for you.

However, in my teaching I've found that an incomplete mastery of these foundational concepts at the beginning of the course results in lots of headaches, frustration, and wasted time (yours and mine) down the road.

I expect that many of you will know some or all of these concepts, and that many of you won't be familiar with any of them!

If the material in this deck is unfamiliar, don't panic! There's an internal logic - we'll learn it in a systematic way.

If the material in this deck is easy for you, don't worry. Things will get more challenging soon!

Why do we have to know about filesystems?

If you can answer all of these questions, you may skip this lecture:

- How are files stored and organized on your computer?
- Where is my user directory?
- Where is that file that I just downloaded from my browser?
- What is wrong with **New Folder (3)**?
- Why shouldn't I submit an assignment with the filename **Untitled document (27).docx**?
- How do associate files with a '.Rdata' extension with Rstudio?
- What happens if I try to open a binary file using a text editor?
- Why is it better to use relative paths when you want to collaborate?
- My program just autosaved my file, hooray! Now where is it?

GUIs hide what goes on behind the scenes

You use a Graphical User Interface (GUI) to control the vast majority of modern computing devices.

GUIs have huge advantage because you can quickly and easily perform complex tasks via just a few clicks or swipes.

GUIs make using our computer more intuitive and save us from hours of repetitive tasks, that we might have to perform with a **command prompt**.

GUIs are very easy to use, but they are only an interface that hides all of the complex processes that happen behind the veneer.

To become proficient at working with data and computers as scientists, you have to learn some of the messy stuff that goes on inside.

What is a filesystem?

A file system is a kind of like **map** that your computer uses to organize **files** and **directories** on a data storage device.

What is a directory?

Directories, also known as folders, are like containers that can hold files or other directories.

What is a file?

A file is a collection of information that is stored as a single, discrete unit on your computer.

Both files and directories are identified by **names**.

Directory structures

Your computer's file system is organized similarly to a tree:

- There is a **root directory** on your storage medium that contains all of the other files and directories.
- To find a file, your computer traverses the branches of the tree following the directions contained in the *path*

An *absolute* path	**A *relative* path**
gives your computer the map to find a file starting from the *root directory* of your storage medium. • An *absolute path* is never ambiguous.	provides a map starting at the current *working directory*. • A working directory is just the directory that a program is currently pointing to.

Types of files

Text files

Text files contain characters (such as letters) that are human-readable.

- Text files encode the data in agreed-upon formats that many programs can read and interpret.

Text files are more universal in the sense that any program that can open a text file can read a text file created in any other program.

Binary files

- Binary files are not human-readable.
- Binary file formats can be platform-specific, such as `.exe` files which are runnable on Windows.
- May be proprietary, or standardized (like image file formats)

File extensions and associations

Filenames usually consist of two or more blocks of letters separated by periods:

- `file_01.txt`

The extension is meant to communicate the *file type* or *file format*.

- Think of image files that have extensions like `.png` or `.jpg`.
- A changing a file extension does not change the contents of the file, but it can change the *association*.

Your operating uses the *file extension* to guess the file type and which program it should open with!

Archive files

- Archive files can contain copies of a directory structure, which itself can contain files and other directories.
- The most familiar archive format is .zip.
- Archive files are typically (but not always) *compressed*
  - They are usually in a *binary* format.
- Archive files are a convenient way to share multiple files and directories as a single unit.

Course Software: R and RStudio

Key concepts

- What is R?
- Why should I learn it?
- What is RStudio?

What is R?

R is a statistical *programming language*. R has many desirable qualities, for example:

- It is open-source and extensible
- There are thousands of *packages* for just about any kind of analysis you might need to do.
- R is *somewhat easy* to learn compared to many other languages.
    - It is a *high level* language: many technicalities occur behind the scenes so code is compact.
    - It is [mostly] *vectorized*
    - It is an *interpreted* language, so more forgiving of errors than a *compiled* language.

Do I really need to learn R?

Why should I learn R

- Reproducibility, Collaboration, and Communication
- I'm enrolled in this course
- Graphics capabilities
- Tons of learning materials
- Increasingly a standard tool for research, government, and industry
- Dominant platform in many fields
- Huge user knowledge base
- Did I mention that it's free and open source?

Why shouldn't I learn R?

- Point-and-click programs can be easier to learn:
  - Excel and other spreadsheets can handle some calculations and analyses
- It's too hard to learn
  - It's *hard* to learn, but *too hard*…. I'm not convinced.
  - Any programming language has a steep learning curve
- It's not established in my field
- [Somewhat] idiomatic syntax

What is RStudio

RStudio is an Integrated Development Environment (IDE)

- Provides a user-friendlier interface to R
- Provides lots of tools to organize projects
- RSTudio and kinttr + RMarkdownMarkdown/markup language for creating webpages, pdfs, presentations and other documents
- Embed R code and graphics

Where to get R and RSTudio

R is maintained by the Comprehensive R Archive Network:

- https://cran.r-project.org/

RSTudio is at https://rstudio.com/

- The basic RStudio desktop application is free and open-source!

Some R Examples

What can I do with R?

- Create graphics
- Create *models*!
- Use it as a calculator.

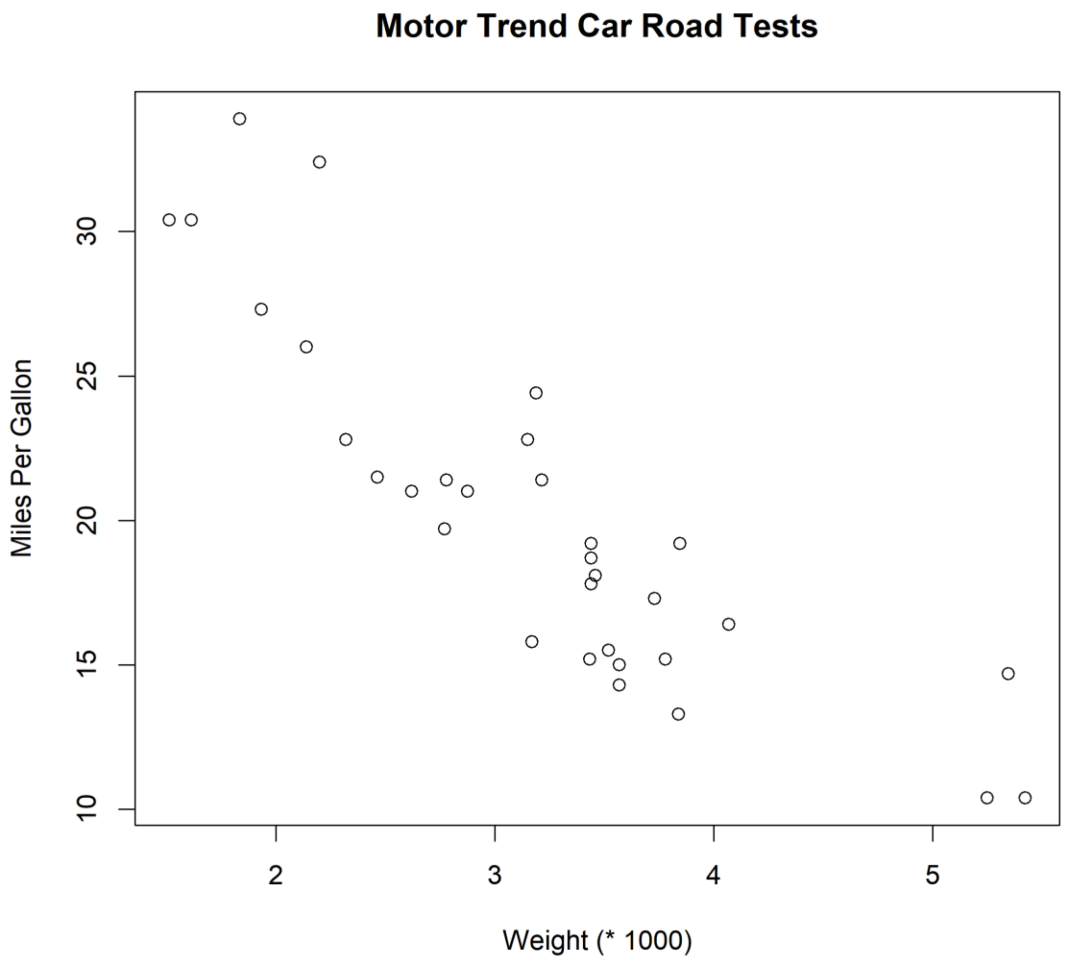
I'll show some examples using the built-in dataset 'mtcars'.

- It contains data from an experiment about driving velocity and stopping distance.

R examples: Basic plotting 1

Creating basic graphics is super easy.
Here's a simple scatterplot:

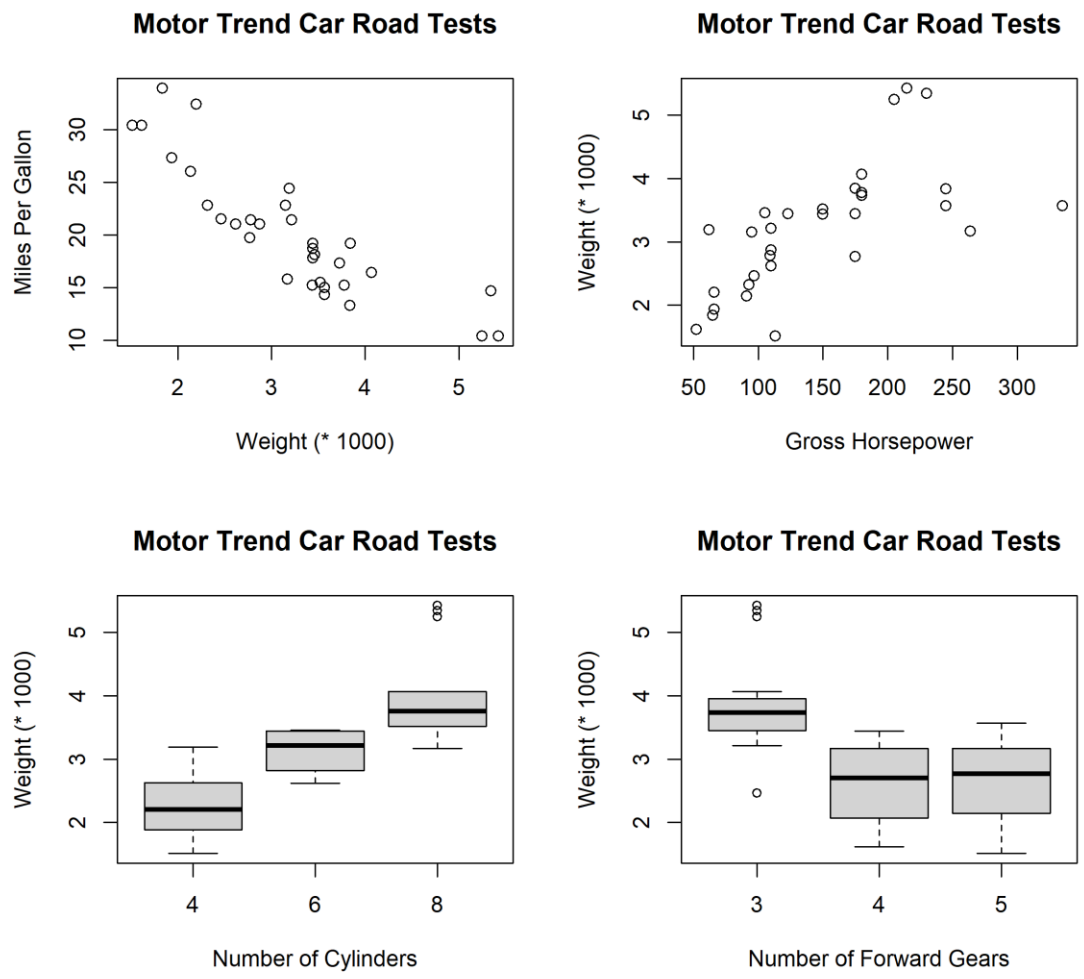
```
plot(
    mtcars$wt, mtcars$mpg,
    main = "Motor Trend Car Road Tests",
    xlab = "Weight (* 1000)",
    ylab = "Miles Per Gallon")
```



R examples: Basic plotting 2

You can also include multiple panels within the same figure using slightly more complicated code:

```
par(mfrow = c(2, 2))
plot(
  mtcars$wt, mtcars$mpg,
  main = "Motor Trend Car Road Tests",
  xlab = "Weight (* 1000)",
  ylab = "Miles Per Gallon")
plot(
  mtcars$hp, mtcars$wt,
  main = "Motor Trend Car Road Tests",
  ylab = "Weight (* 1000)",
  xlab = "Gross Horsepower")
boxplot(
  wt ~ cyl, data = mtcars,
  main = "Motor Trend Car Road Tests",
  ylab = "Weight (* 1000)",
  xlab = "Number of Cylinders")
boxplot(
  wt ~ gear, data = mtcars,
  main = "Motor Trend Car Road Tests",
  ylab = "Weight (* 1000)",
  xlab = "Number of Forward Gears")
```

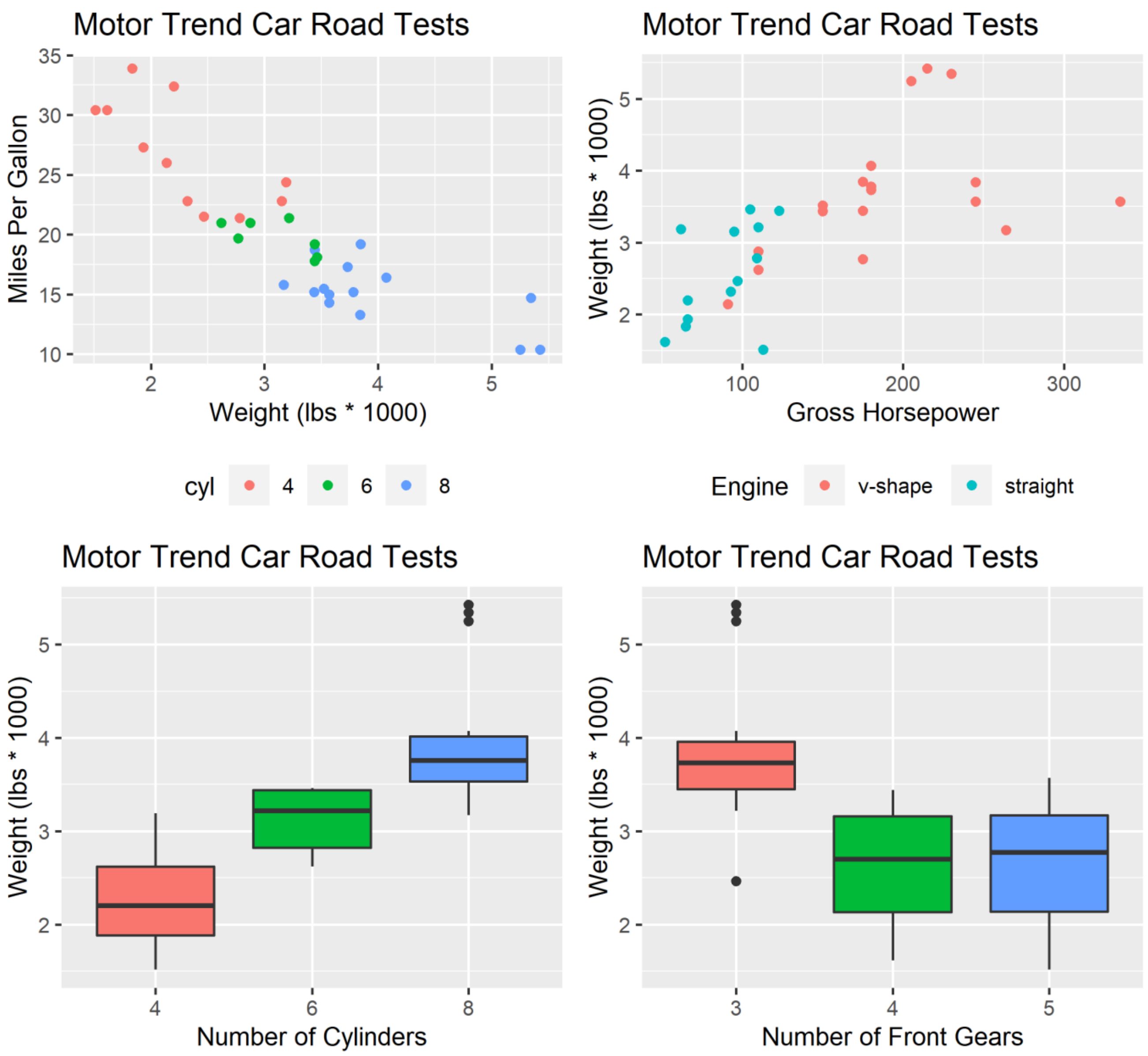


R examples: Fancier plotting

The previous figures used *base R* plotting.

There are many other *packages* for creating fancier plots or plots for specialized purposes.

This plot was created using the `ggplot2` package:



R examples: Linear Regression

R has a *formula notation* which makes specifying a model very intuitive. For example, we can create a simple linear regression of displacement and miles per gallon:

```
data(mtcars)
fit_1 =
  lm(
    disp ~ mpg,
    data = mtcars)
```

We can easily view model coefficients `summary()` function:

```
Residuals:
     Min       1Q   Median       3Q      Max 
-103.05   -45.74    -8.17    46.65   153.75 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  580.884     41.740  13.917 1.26e-14 ***
mpg          -17.429      1.993  -8.747 9.38e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 
1

Residual standard error: 66.86 on 30 degrees of freedom
Multiple R-squared:  0.7183,	Adjusted R-squared:  0.709 
F-statistic: 76.51 on 1 and 30 DF,  p-value: 9.38e-10
```

R examples: Linear Regression

- It's easy to plot the model with 95% confidence intervals using ggplot!
- We'll talk a lot more about regression and confidence intervals soon enough…