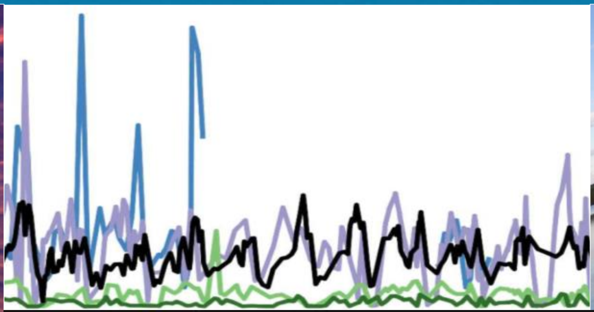
TETRA TECH

Open-Source Scripting
Day 1

September 20, 2021



complex world | CLEAR SOLUTIONS™

TETRA TECH

Tetra Tech Team

Brian Pickard, PhD

Landscape Ecology, Geospatial Science

brian.pickard@tetratech.com

Kateri Salk, PhD

Aquatic Ecology, Data Science

kateri.salkgundersen@tetratech.com

Kellie Dubay

Environmental Science,
Public Outreach

Workshop Goals

1. Review the current landscape of data tools & resources for water professionals
2. Increase familiarity with using R and Python across the data pipeline
3. Create and modify code for future use

Workshop Outline

1. Climate Data
2. Land Use/Land Cover Data
3. Hydrologic and Water Quality Data

Each Day:

- Intro presentation and discussion
- Hands-on demonstration
- Example presentations of state applications
- Hands-on demonstration

Ground Rules

- **Mute your mic unless asked to speak**
- **Questions during sessions**
  - During session: add your question/comment to the chat
  - After session: may solicit people to raise hand and turn on mic
- **Tech issues?**
  - Add your issue to the chat, but may not have time to troubleshoot on the fly
  - Follow along on the screen, code is available for use later
  - Can attempt to fix issues following session or after hours

Day 1 Agenda

- **Intro: workshop goals, outline, introduce people, logistics (30 min)**
  - Suitability of R vs. Python for various data tasks – Kateri Salk and Brian Pickard, Tetra Tech
  - Benefits of reproducible analysis & best practices– Kateri Salk and Brian Pickard, Tetra Tech
  - State perspectives – Nicholas von Stackelberg, Utah
- **Climate data acquisition and harmonization (90 min)**
- **Break (10 min)**
- **Examples of using climate data in modeling (45 min)**
  - General Web Scraping – Eric Hettler, Wisconsin
  - Scraping Climate Data from DAYMET – Eric Hettler, Wisconsin
  - Developing R packages to automate data analysis – Ansel Bubel, Florida
- **Break (5 min)**
- **Data management, documentation, and export (45 min)**
- **Wrap-up and next day preview (15 min)**

TETRA TECH

R vs. Python

Benefits of Reproducible Analysis

R vs. Python

- R is mainly used for statistical analysis while Python provides a more generalized approach to data science
  - R: data analysis and stats
  - Python: deployment and production

- R provides flexibility through available packages, Python enables construction of new models from scratch

- Both can handle big data, machine learning and make replicability and accessibility easier

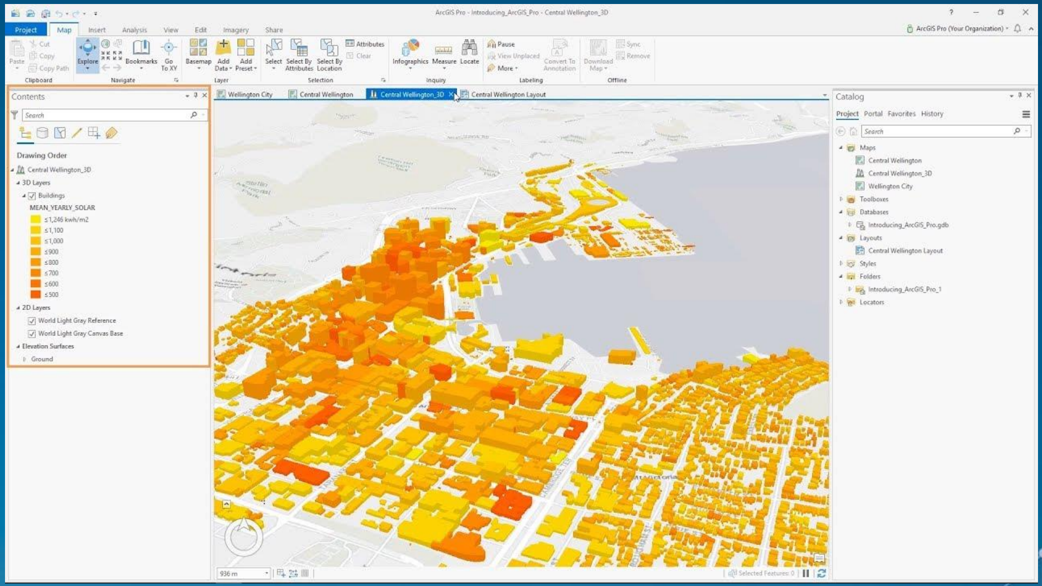
TETRA TECH

R vs. Python

Parameter	R	Python
Objective	Data analysis and statistics	Deployment and production
Primary Users	Scholar and R&D	Programmers and developers
Flexibility	Easy to use available library	Easy to construct new models from scratch. I.e., matrix computation and optimization
Learning curve	Difficult at the beginning	Linear and smooth
Popularity of Programming Language. Percentage change	4.23% in 2018	21.69% in 2018
Integration	Run locally	Well-integrated with app
Task	Easy to get primary results	Good to deploy algorithm
Database size	Handle huge size	Handle huge size
IDE	Rstudio	Spyder, Ipython Notebook
Important Packages and library	tidyverse, ggplot2, caret, zoo	pandas, scipy, scikit-learn, TensorFlow, caret
Disadvantages	Slow High Learning curve Dependencies between packages	Not as many packages as R
Advantages	•Graphs are made to talk. R makes it beautiful •Large catalog for data analysis •GitHub interface •RMarkdown •Shiny	•Jupyter notebook: Notebooks help to share data with colleagues •Mathematical computation •Deployment •Code Readability •Speed •Function in Python

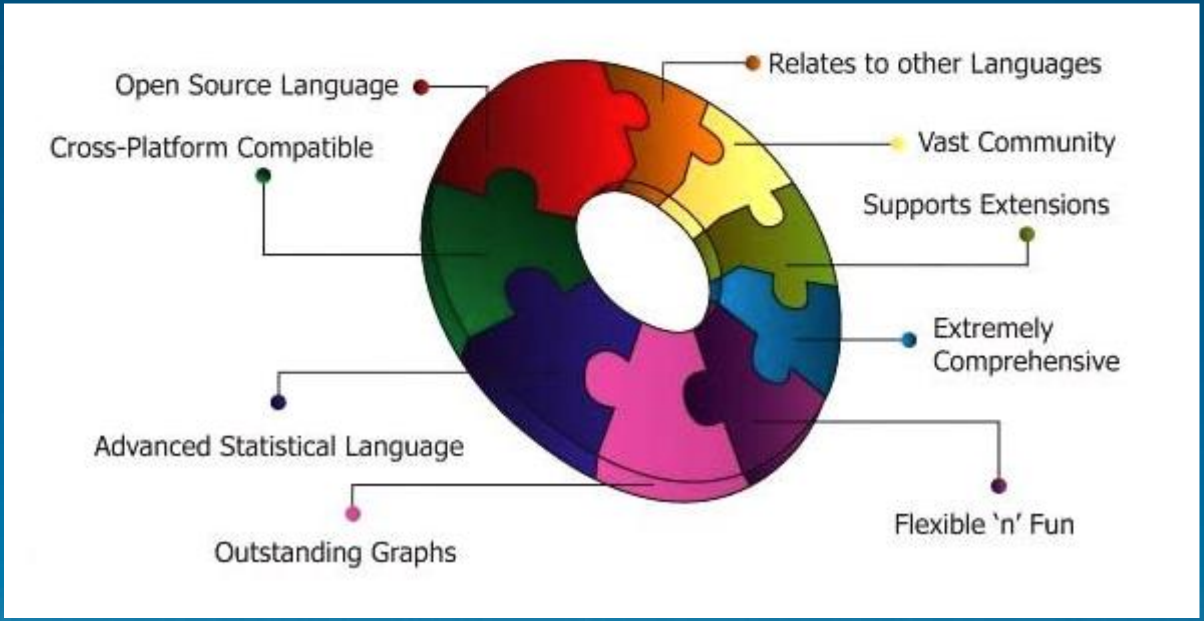
R vs Python

- Python is the programming language used by all ESRI products (e.g. ArcGIS, ArcPro)



- Therefore, Python is far superior for handling spatially explicit data and all related tasks.

R vs Python



- R has user-driven package development and online community
- Integration with interactive web apps through R Shiny

TETRA TECH

Benefits of Reproducible Analysis

Spreadsheet-based workflow

to

Scripting-based workflow

	A	B	C	D	E	F	G	H	I	J	K
1	lakeid	lakename	year4	daynum	sampledate	depth	temperatu	dissolved	irradiance	irradiance	comments
2	L	Paul Lake	1984	148	5/27/1984	0	14.5	9.5	1750	1620	NA
3	L	Paul Lake	1984	148	5/27/1984	0.25	NA	NA	1550	1620	NA
4	L	Paul Lake	1984	148	5/27/1984	0.5	NA	NA	1150	1620	NA
5	L	Paul Lake	1984	148	5/27/1984	0.75	NA	NA	975	1620	NA
6	L	Paul Lake	1984	148	5/27/1984	1	14.5	8.8	870	1620	NA
7	L	Paul Lake	1984	148	5/27/1984	1.5	NA	NA	610	1620	NA
8	L	Paul Lake	1984	148	5/27/1984	2	14.2	8.6	420	1620	NA
9	L	Paul Lake	1984	148	5/27/1984	3	11	11.5	220	1620	NA
10	L	Paul Lake	1984	148	5/27/1984	4	7	11.9	100	1620	NA
11	L	Paul Lake	1984	148	5/27/1984	5	6.1	2.5	34	1620	NA
12	L	Paul Lake	1984	148	5/27/1984	6	5.5	1.6	7.6	1620	NA
13	L	Paul Lake	1984	148	5/27/1984	7	5	0.4	1.3	1610	NA
14	L	Paul Lake	1984	148	5/27/1984	8	4.5	0.3	NA	NA	NA
15	L	Paul Lake	1984	148	5/27/1984	9	4.5	0.3	NA	NA	NA
16	L	Paul Lake	1984	148	5/27/1984	10	4.5	0.3	NA	NA	NA
17	L	Paul Lake	1984	148	5/27/1984	11	4.5	0.3	NA	NA	NA
18	L	Paul Lake	1984	148	5/27/1984	12	4.5	0.3	NA	NA	NA
19	R	Peter Lake	1984	149	5/28/1984	0	14.8	9.2	1630	1540	NA

```
## Objectives
1. Describe the usefulness of data wrangling and its place in the data pipeline
2. Wrangle datasets with dplyr functions
3. Apply data wrangling skills to a real-world example dataset

## Set up your session

```{r, message = FALSE}
getwd()
library(plyr)
library(tidyverse)
library(lubridate)
NTL.phys.data.PeterPaul <- read.csv("./Data/Processed/NTL-LTER_Lake_ChemistryPhysics_PeterPaul_Processed.csv")
NTL.nutrient.data <- read.csv("./Data/Raw/NTL-LTER_Lake_Nutrients_Raw.csv")
```

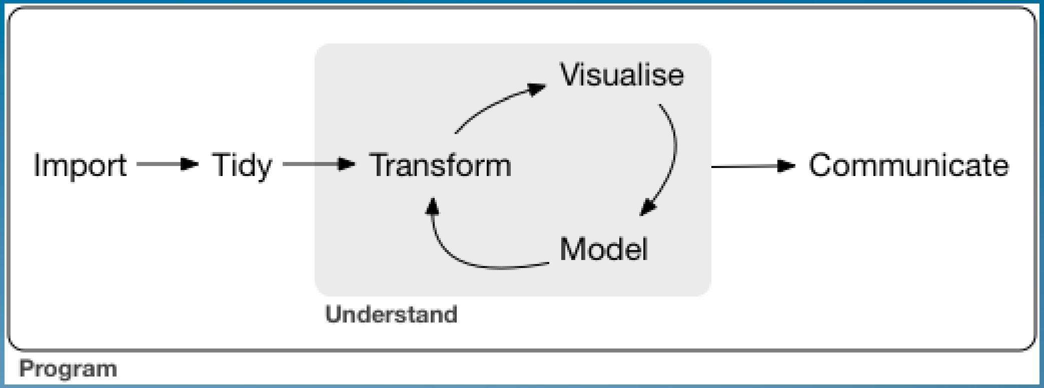
Review of basic exploration and wrangling
```{r}
# Data summaries for physical data
colnames(NTL.phys.data.PeterPaul)
dim(NTL.phys.data.PeterPaul)
str(NTL.phys.data.PeterPaul)
summary(NTL.phys.data.PeterPaul$comments)
class(NTL.phys.data.PeterPaul$sampledate)

# Format sampledate as date
NTL.phys.data.PeterPaul$sampledate <- as.Date(NTL.phys.data.PeterPaul$sampledate, format = "%Y-%m-%d")

# Select Peter and Paul Lakes from the nutrient dataset
NTL.nutrient.data.PeterPaul <- filter(NTL.nutrient.data, lakename == "Paul Lake" | lakename == "Peter Lake")
```
```

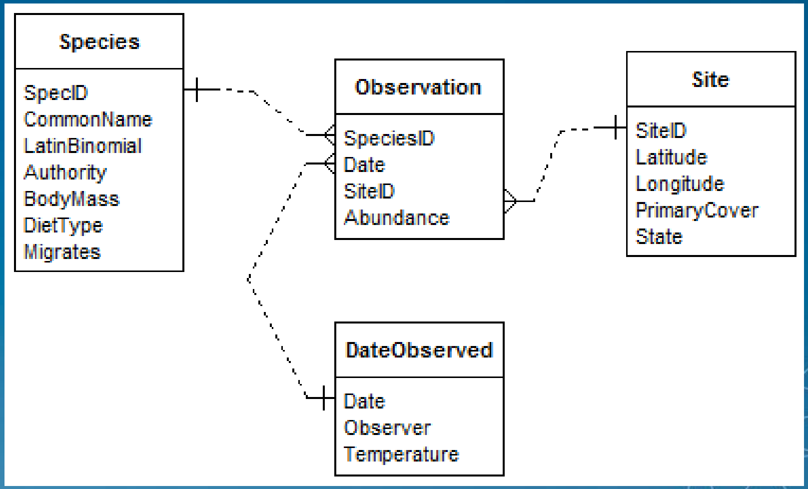
Benefits of Reproducible Analysis

- Automating manual tasks → faster, more efficient, more consistent
- Code can be reused → apply to future projects, share with others
- Steps are well-documented and transparent → increased capacity for QA/QC and project hand-offs
- Ability to fix errors and re-flow into analysis



Best Practices for Reproducible Analysis

- **Separate raw from processed data, link w/code**
  - Never edit raw data
  - Plan to spend ~75% of time cleaning data
- **Data tables**
  - Header: row 1 (and ONLY row 1)
  - Columns: Variables/attributes
  - Rows: Measurements
  - Cells: Observations
- **Star schema**



From: https://dynamicecology.wordpress.com/2016/08/22/ten-commandments-for-good-data-management/

Where do I start?

- **Learning curve from spreadsheet to coding can be steep!**
- **Progress can/should be made incrementally**
- **Tools should scale with:**
  - Data complexity
  - Analytical complexity
  - Capacity to re-use in future applications

TETRA TECH

Prep for data session: download materials now!

github.com/KateriSalk/ACWA_OpenSourceWorkshop_2021