Welcome to The Carpentries Etherpad!

This pad is synchronized as you type, so that everyone viewing this page sees the same text. This allows you to collaborate seamlessly on documents.

Use of this service is restricted to members of The Carpentries community; this is not for general purpose use (for that, try etherpad.wikimedia.org).

Users are expected to follow our code of conduct: https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

EVENT SITE: https://gwu-libraries.github.io/2019-08-19-gwu/
PRE-WORKSHOP SURVEY: https://www.surveymonkey.com/r/dcpreworkshopassessment?workshop_id=2019-08-19-GWU
SETUP: https://datacarpentry.org/geospatial-workshop/setup.html
DATA: https://ndownloader.figshare.com/articles/2009586/versions/10
(then unzip)

Introductions: Free space for anyone to introduce yourself and about your area of research/study
hi. Etherpad sounds cool.

Kean McDermott is the geographical information systems (GIS) specialist here at GW Libraries - he's your expert for GIS questions and consultations!
https://library.gwu.edu/scholarly-technology-group/geographic-information-systems-gis-support
Dan Kerchner is the Senior Software Developer and Librarian, he helps create applications for the library and consults with patrons on software development projects.
Megan Potterbusch is the Data Services Librarian. Available to help with data discovery and data management questions - including where to publish/preserve/share your data.
Jennifer-Major is Data Science Masters with Geospatial Certificate
Eve Bohnett is a Landscape Architecture PhD Candidate at University of Florida
Andrew Gepty is a fourth year PhD Student in Clinical Psychology here at GWU.
Jessa Henderson is a 2nd year PhD student in GSEHD's Human-Technology Collaboration Cross Disciplinary program
Fouzia Farooq - Public health/Epidemiology, 3rd year PhD student at University of Pittsburgh
This space for rent
Jaime Ashander is a postdoc fellow at Resources for the Future in DC
Kevin Heslin. Adjunct faculty in GWU dept of epidemiology. Associate director for science, NCHS.

INTRODUCTION TO GEOSPATIAL CONCEPTS
> Slides: https://gwu-libraries.github.io/2019-08-19-gwu/#syllabus

GLOSSARY OF TERMS:

- **Spatial extent** - the corners or bounds of the image (points, ines, polygons) applies for both raster and vector data
- **Resolution** (pixelation) - in raster data, not vector data - each pixel represents a latitude and

longitude location
- **Multi-band** - for a time series, colors (RGB) of rasters stacked in a single, composite dataset

Pros/Cons of Raster vs. Vector data:

- **Raster**
  - > Easier to "stack" and do calculations
  - > Continuous, able to display change over time
- **Vector**
  - > May be smaller files
  - > Vector is able to display multiple attributes (points, lines or polygons)
  - > When you're working with data involving boundaries (e.g. census tracts, cities)

- **Coordinate Reference System** - the earth, translated into data
- **Datum** - file formats for data (WGS, NAD etc.) to determine latitude/longitude - surveyors level their measurements down into a surface called a geod, every country has their own one
- **WGS84** - has the satelite data that we use today
- **Map Projections** - flatten maps into a flat, 2-D surface, which distorts countries' size (e.g. world mercator projection - the continent of Africa can fit many countries, but the area appears smaller - you can transform the lat/longs from degrees into meters, feet etc.
- **PROJ system** - open source way to transform datasets - uses the string of the **CRS system** (e.g. zones of the USA)
  - e.g. +proj=utm +zone=18 +datum=WGS84 +units=m +no_defs +ellps=WGS84

RASTER DATA SOURCES:

- **Google Earth Engine** - repository about climate, surface, satelite imagery (updated every 8 days) with different bands in a photo package to get different color spectrums - Powerful and free-ish (you have to register here: https://signup.earthengine.google.com/#!/)
- > **Google Earth Code Editor** - snippets here: https://developers.google.com/earth-engine/playground
- > **PanGEO, Sepal.io, Kepler.gl and Boundless Geospatial** - other repositories for raster data

R FOR GEOSPATIAL DATA

SETUP:

- **Download**: https://ndownloader.figshare.com/articles/2009586/versions/10
- **Packages:** Type into R Studio: install.packages(c("dplyr", "ggplot2", "raster", "rgdal", "rasterVis", "sf"))

R STUDIO SCRIPTS:

- **Start a new script:** Ctrl + Shift + N (shows up in the console)
- **Run:** CTRL + ENTER - sends a new line down to the console to be evaluated (or click Run)
- Source - runs the entire console
- Comparisons:
  - 1 == 1
  - 1 > 2 (will get a True or False response in the console)

    - 2 < 2
    - 2 <= 2
- Programming:
    - **Assign a variable:** <- OR =

Challenge 1: http://datacarpentry.org/r-intro-geospatial/01-rstudio-intro/index.html

- **Variable Values**: Type into the R Script pane, hit run then the variables will change in the **Environment Tab (Values)**
- **Variable Names**: separate words with periods (.) or underscores (_)

PROJECT MANAGEMENT:
https://datacarpentry.org/r-intro-geospatial/02-project-intro/index.html

- **Data Folder**: File > New Project > New Directory
- **File location:** getwd()
- **Downloads**: Save files as .CSV instead of .TXT files and save into data folder
- **New Script**: Plus sign or Ctrl + Shift + N (then save)

EXPLORING DATA:

- variable$ - explores the categories of each column
    - gapminder[1:5, "pop"] If you wanted entries 1 through 5 of the pop column -
- gapminder <-read.csv("data/gapminder_data.csv")
- str(gapminder) - information on variable in the console
- class(1) - numeric class(1L) - integer
- dim - dimension (e.g. the number of columns and rows) of a matrix, array or data frame

Type coercion https://datacarpentry.org/r-intro-geospatial/03-data-structures-part1/index.html

- The data in nordic_2$lifeExp is stored as factors rather than numeric. This is because of the “or” character string in the third data point.
- **“Factor”** is R’s special term for categorical data.
- **c()** - combines data - https://www.rdocumentation.org/packages/base/versions/3.6.1/topics/c
- Adding rows:
    - gapminder$below_avg <- gapminder$lifeExp < 70.5
    - head(gapminder)
    - str(gapminder[1 ,])

PIPING
http://datacarpentry.org/r-intro-geospatial/06-dplyr/

- Install package - library(dplyr)
- Pipe: gapminder%>%
- select(year, country, gdpPercap)
- # Returning values for Africa not other continents:
    - year_country_gdp_euro <- gapminder %>%
    - filter(continent == "Europe") %>%
    - select(year, country, gdpPercap)
- # Counting things
- gapminder %>%

- filter(year == 2002) %>%
- count(continent)

PLOTS:

- #install.packages('ggplot2') - library(ggplot2)
- ggplot(data=gapminder, aes(x=lifeExp)) + geom_histogram()
- The figure shows the distribution of gdp per capita, rather than life expectancy - ggplot(data = gapminder, aes(x = gdpPercap)) + geom_histogram()
- install.packages("dplyr")
- library(dplyr)
- library(ggplot2)
- library(raster)
- library(rgdal)
- ggplot(data=gapminder,aes(x=lifeExp)) +
- geom_histogram(aes(fill=continent))+
- labs(title="a histogram")
-
- GDALinfo <- read.csv("Data/NEON-DS-Airborne-Remote-Sensing/NEON-DS-Airborne-Remote-Sensing/HARV/DSM/HARV_dsmCrop.tif")
- getwd()
- dsm_harv <-raster("Data/NEON-DS-Airborne-Remote-Sensing/NEON-DS-Airborne-Remote-Sensing/HARV/DSM/HARV_dsmCrop.tif")
- summary(dsm_harv)
- dsm_harv_df<-as.data.frame(dsm_harv,xy=TRUE)
- str(dsm_harv_df)
- ggplot()+
- geom_raster(data=dsm_harv_df,aes(x=x,y=y,fill=HARV_dsmCrop))+
- scale_fill_viridis_c() +
- coord_quickmap()

RASTERS:
https://datacarpentry.org/r-raster-vector-geospatial/01-raster-structure/index.html

COMMENTS
starts a line that is a comment. It won't be processed as code - ignored by R

GEOSPATIAL RASTER AND VECTOR DATA WITH R

```
install.packages(c("dplyr", "ggplot2", "raster", "rgdal", "rasterVis", "sf"))
library(rgdal)
GDALinfo("data/NEON-DS-Airborne-Remote-Sensing/HARV/DSM/HARV_dsmCrop.tif")  # <-- gives the metadata info that's in the file


HARV_dsmCrop_info <- capture.output(
  GDALinfo("data/NEON-DS-Airborne-Remote-Sensing/HARV/DSM/HARV_dsmCrop.tif")
)
```

```
DSM_HARV <-
  raster("data/NEON-DS-Airborne-Remote-Sensing/HARV/DSM/HARV_dsmCrop.tif")

DSM_HARV_df <- as.data.frame(DSM_HARV, xy = TRUE)

ggplot() +
  geom_raster(data = DSM_HARV_df , aes(x = x, y = y, fill = HARV_dsmCrop)) +
  scale_fill_viridis_c() +
  coord_quickmap()

  Alpha - transparency level vs. fill
str() - Structure to inspect attributes

RGB_band1_HARV <-
raster("data/NEON-DS-Airborne-Remote-Sensing/HARV/RGB_Imagery/HARV_RGB_Ortho.tif")
RGB_band1_HARV_df <- as.data.frame(RGB_band1_HARV, xy=TRUE)

ggplot()+
  geom_raster(data=RGB_band1_HARV_df, aes(x=x, y=y, alpha=HARV_RGB_Ortho))+
  coord_quickmap()

str(RGB_band1_HARV)
```

MORE GGplot resources!!

the skills we used for rasters can be applied (with some tweaks) to other data as well. SEe this lesson from another data carpentry workshop: https://datacarpentry.org/R-ecology-lesson/04-visualization-ggplot2.html

VECTORS

```
- Libraries - add all the libraries you use at the top, so you can get a quick glance at the code
library(sf)
library(ggplot2)
- CRS = Coordinate Reference System - use datum=NULL to use the same CRS

aoi_boundary_HARV <- st_read("data/NEON-DS-Site-Layout-Files/NEON-DS-Site-Layout-Files/
HARV/HarClip_UTMZ18.shp")

# Specific functions to access metadata/attributes
st_geometry_type(aoi_boundary_HARV)
# This is a factor (looks like characters)

st_crs(aoi_boundary_HARV)
st_bbox(aoi_boundary_HARV)
aoi_boundary_HARV
class(lines_HARV)
nrow(lines_HARV)
```

```
#plot the data by attribute
ggplot() +
  geom_sf(data=aoi_boundary_HARV, size = 3, color="black", fill="cyan1") +
  ggtile("AOI boundary plot") +

# Use dataframe like functions to learn about attributes
ncol(lines_HARV)
nrow(lines_HARV)
dim(lines_HARV)
names(lines_HARV)
head(lines_HARV)
point_HARV$Ownership
names(point_HARV)

#plot and subset

stoneWall_HARV <- lines_HARV %>%
  filter(TYPE == "stone wall")nrow(stoneWall_HARV)
ggplot() +
  geom_sf(data = stoneWall_HARV, aes(color = factor(OBJECTID)), size = 1.5) +
  labs(color = 'Wall ID') +
  ggtitle("NEON Harvard Forest Field Site", subtitle = "Stonewalls") +
  coord_sf()

# Convert Raster to a Data Frame
chm_HARV <- raster("data/NEON-DS-Airborne-Remote-Sensing/HARV/CHM/HARV_chmCrop.tif")
chm_HARV_df <- as.data.frame(chm_HARV, xy = TRUE)
```

QUESTIONS (General):

How to convert .shp or .prj files into dataframes for future merge that allows color assignments, such as using a left_join() that carries a color variable?

In the lessons (likely tomorrow) on importing and plotting shapefiles we will see the sf library, which has a data.frame like structure that stores shapefile data. We will use ggplot2 to plot these features and see ways to change aesthetics (eg color, line width) of the images depending on attributes in the shape file. Something like adding a color value to the data itself is likely also possible (eg with dplyr's left_join) but we won't take that approach tomorrow.

Does R work with KML files as well?

Probably. Well definitely GDAL has tools to convert to/from KML. I haven't used these through R or otherwise.:

- Google suggests GDAL could help (https://gis.stackexchange.com/questions/58131/reading-kml-file-into-r) but

- the package 'maptools' may provide easier to use tool

Are there basemap packages in R?

- install.packages('rworldmap') # global maps and enviornmental data

- install.packages('maps') # countrya nd within-country boundaries
- see the CRAN task view on geospatial https://cran.r-project.org/web/views/Spatial.html in particular the section **Specific geospatial data sources of interest**
- ggmap to get online maps (but you need an API key from google maps first)
https://www.nceas.ucsb.edu/~frazier/RSpatialGuides/ggmap/ggmapCheatsheet.pdf

How to automatically color the states in the final challenge for shape data?
- try scale_fill_brewer() instead of the manual one!

How can we get ggplot to "truncate" the plot and "zoom in" to anareathat's smaller than the spatial extent of the data?

If anybody can figure this out (I can't): How can you get ggplot to give youa colored line segment in the legend, rather than a colored square?
- I think we showed this, eventually. Add this to geom_sf: show.legend="line" # yup it wasin the next question

*Can we keep the code up longer by extending the terminal and keeping the console smaller?

POLL:

1. How is the pace of the content so far?
a. way too slow

b. too slow
xx
c. just right
xxxxxxx
x (for the intro)
d. too fast
xxxxxxx
x (for rasters)
e. way too fast
xx
pace is good. I like typing!

SHAPEFILES

lesson 06: https://datacarpentry.org/r-raster-vector-geospatial/06-vector-open-shapefile-in-r/index.html

point_HARV <- st_read("data/NEON-DS-Site-Layout-Files/HARV/HARVtower_UTM18N.shp")

Note: difference to web lesson in getting plot to use UTM:

```
   # plot the data
ggplot() +
  geom_sf(data=aoi_boundary_HARV, size=3, color="black",
          fill="cyan1") +
  ggtitle("AOI boundary plot") +
  coord_sf(datum=NULL)

# raster file for CHM
chm_HARV <- raster("data/NEON-DS-Airborne-Remote-Sensing/HARV/CHM/HARV_chmCrop.tif")
```

Use legend tricks with multiple layers!!
https://datacarpentry.org/r-raster-vector-geospatial/08-vector-plot-shapefiles-custom-legend/index.html

Multiple fills require ggplot addon: https://eliocamp.github.io/codigo-r/2018/09/multiple-color-and-fill-scales-with-ggplot2/

When doing:
Challenge: Plot Raster & Vector Data Together, and the code in the solution, be careful because it uses CHM_HARV_df and that was used in the previous lesson, Raster Calculations.
It included "CHM_HARV <- DSM_HARV - DTM_HARV
CHM_HARV_df <- as.data.frame(CHM_HARV, xy = **TRUE**)"
...and that may still be in memory in your project. That CHM_HARV doesn't have a HARV_chmCrop.

```
# CRS and shpaes
state_boundary_US <- st_read("data/NEON-DS-Site-Layout-Files/US-Boundary-Layers/US-State-Boundaries-Census-2014.shp")
```

Writing data! and creating spatial data from CSVs

Quick version!

```
#####################################################
## read in a csv
plot_locations_HARV <-
  read.csv("data/NEON-DS-Site-Layout-Files/HARV/HARV_PlotLocations.csv")
str(plot_locations_HARV)
head(plot_locations_HARV, n=1)

## from gedeticCA and utmZone cols it looks like same CRS as others, eg point_HARV
utm18nCRS <- st_crs(point_HARV) ## so just use that one!

# st_as_sf function converts foreign objects to sf
plot_locations_sp_HARV <- st_as_sf(plot_locations_HARV,
                                   coords = c("easting", "northing"),
                                   crs = utm18nCRS)

st_write(plot_locations_sp_HARV, "results/PlotsProcessed_HARV.shp",
         driver="ESRI Shapefile")
#####################################
```

Full details ----> https://datacarpentry.org/r-raster-vector-geospatial/10-vector-csv-to-shapefile-in-r/index.html

POLL2 :

1. How is the pace of the content so far?
a. way too slow

b. too slow

c. just right
XXXXXXx
d. too fast

e. way too fast

2. What do you perfer?
a. seeing more ways to make complex maps, without understanding all the details

b.understanding all the details, even if we don't get as far
XXXXxX

BACK TO RASTERS

```
point_HARV <- st_read("data/NEON-DS-Site-Layout-Files/HARV/HARVtower_UTM18N.shp")

aoi_boundary_HARV <- st_read(
  "data/NEON-DS-Site-Layout-Files/HARV/HarClip_UTMZ18.shp"
)


## challenge 2
CHM_HARV <- raster("data/NEON-DS-Airborne-Remote-Sensing/HARV/CHM/HARV_chmCrop.tif")
CHM_HARV_df <- as.data.frame(chm_HARV, xy=TRUE)

CHM_HARV_Cropped <- crop(x = CHM_HARV, y = as(aoi_boundary_HARV, "Spatial"))
CHM_HARV_Cropped_df <- as.data.frame(CHM_HARV_Cropped, xy = TRUE)

# READING AND WRITING

## read in a csv

## from gedeticCA and utmZone cols it looks like same CRS as others, eg point_HARV
utm18nCRS <- st_crs(point_HARV) ## so just use that one!

# st_as_sf function converts foreign objects to sf
plot_locations_sp_HARV <- st_as_sf(plot_locations_HARV,
```

```
                    coords = c("easting", "northing"),
                    crs = utm18nCRS)


st_write(plot_locations_sp_HARV, "results/PlotsProcessed_HARV.shp",
        driver="ESRI Shapefile")
```

CHOROPLETH CRIME MAPPING and MAKING MAPS

Site: https://hautahi.com/rmaps

Go to the Github link in the text --> download zip here:
https://github.com/hautahi/Rmaps/archive/master.zip
Unzip, and save Rmaps-master to your workshop folder
If you save Rmaps-master to your project folder you've been using, then the first few lines of code would be:

```
library(rgdal)
port <- readOGR(dsn = "Rmaps-master/data", layer = "Police_Districts_Portland") #note the direction of the slashes
crime <- readOGR(dsn = "Rmaps-master/data", layer = "NIJ_Nov2016_Crime")

# you will need this prior to --> library(leaflet)
install.packages("leaflet")

#Instead of typing Proj (+UTM etc) -
crime_agg<-spTransform(crime_agg, CRS("+init=epsg:4326"))
proj4string(crime_agg)

# You will also need (for the later section)
install.packages('rgeos')
library(rgeos)
```

*****
EVERYBODY PLEASE FILL OUT THE POST-WORKSHOP SURVEY!
https://www.surveymonkey.com/r/dcpostworkshopassessment?workshop_id=2019-08-19-GWU
*****
FOLLOWING UP - GW LIBRARIES RESOURCES FOR AFTER THE WORKSHOP:
Upcoming workshops at GW related to Coding/Programming:
https://library.gwu.edu/events?series=coding
including Python and more R!

Coding, Programming, Data, Statistical, and GIS Consultations

Book a consultation with a Geographic Information Systems (GIS) specialist (https://calendly.com/gis-consultations)
Learn more about GIS assistance (https://library.gwu.edu/scholarly-technology-group/geographic-

information-systems-gis-support), including curricular support for instructors and workshops
Book a consultation for help with statistical analyses: https://calendly.com/statistical-consulting-gw in C++, Excel, Java, MatLab, mySQL, Python, R, SAS, SPSS, SQL, STATA, and Tableau.

- Statistical analysis using statistical computing packages
- Quantitative sampling
- Data preparation
- Hypotheses testing

Book a consultation with our data services librarian (https://calendly.com/data-consultation) for help organizing your data and preparing your data for long-term storage and sharing.
Book a consultation to research social media data (https://calendly.com/social-media-consulting-gw) or learn more about the Social Feed Manager (https://library.gwu.edu/scholarly-technology-group/social-feed-manager)
Book a programming or software development consultation (https://calendly.com/gwul-coding) with a member of the Scholarly Technology Group

https://www.usgs.gov/faqs/what-are-band-designations-landsat-satellites?qt-news_science_products=0#qt-news_science_products