Week 2: Bonus Material

1. Overview

Author: Helene Wagner

a) Goals

This bonus material expands the Worked Exakmple to show:

- How to use a new standard for spatial data ('sf' package).
- How to plot site attributes in space ('sf' package)
- How to export the site data to a shapefile for import into a GIS ('sf' package).
- How to plot a categorical raster map with a predefined color scheme.

Try modifying the code to import your own data!

b) Required R libraries

```
require(sp)
require(sf)
require(raster)
require(GeNetIt)
require(rasterVis)
require(tmaptools)
root <- rprojroot::find_root("DGS_Week1_Lab.Rproj")
```

2. Convert 'SpatialPointsDataFrame' to 'sf' object

The following code checks whether the object 'RALU.site.sp' exists, and if not, creates it again (without the additional site variables derived in the Worked Example).

Note: the exclamation mark '!' means 'is not', hence 'if(!exists("RALU.site.sp"))' translates to 'if RALU.site.sp does not exist'. If the condition is met, then the code between the curly brackets will be executed.

```
if(!exists("RALU.site.sp"))
{
  RALU.site <- read.csv(file.path(root, "Data", "RALU_site_all.csv"), header=TRUE)
  RALU.site.sp <- RALU.site
  coordinates(RALU.site.sp) <- ~coords.x1+coords.x2
  proj4string(RALU.site.sp) <- get_proj4("utm11")
}
```

While 'sp' was a major achievement, it is being replaced now by 'sf', which is short for "simple feature". The conversion is easy. The resulting S3 object is a data frame and also an 'sf' object.

```
RALU.site.sf <- st_as_sf(RALU.site.sp)
class(RALU.site.sf)
```

```
## [1] "sf"         "data.frame"
```

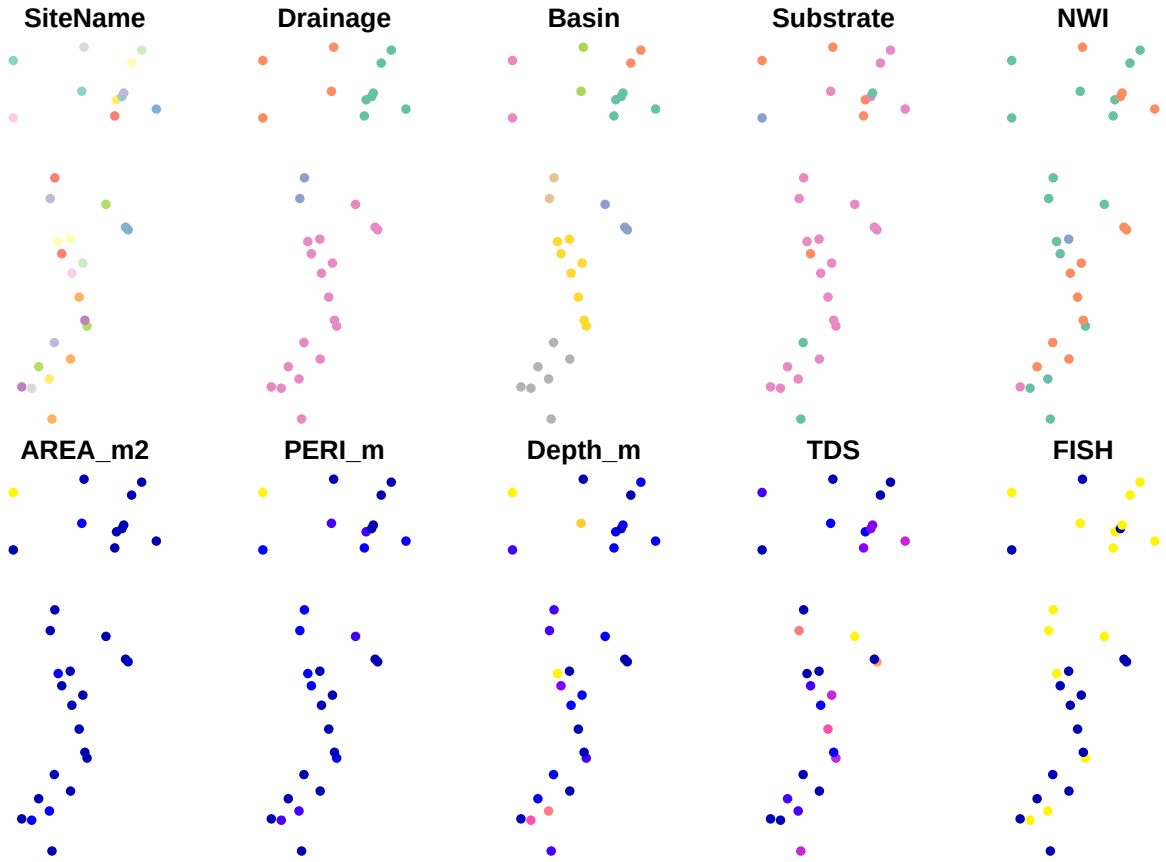
3. Plot variables (first 10 in data set)

If we use 'plot' on this 'sf' object, it will plot site attribute data in space! Here we set the point character 'pch' to a filled circle, which is symbol #16.

For an overview of 'pch' symbol numbers, and colors, check: http://vis.supstat.com/2013/04/plotting-symbols-and-color-palettes/

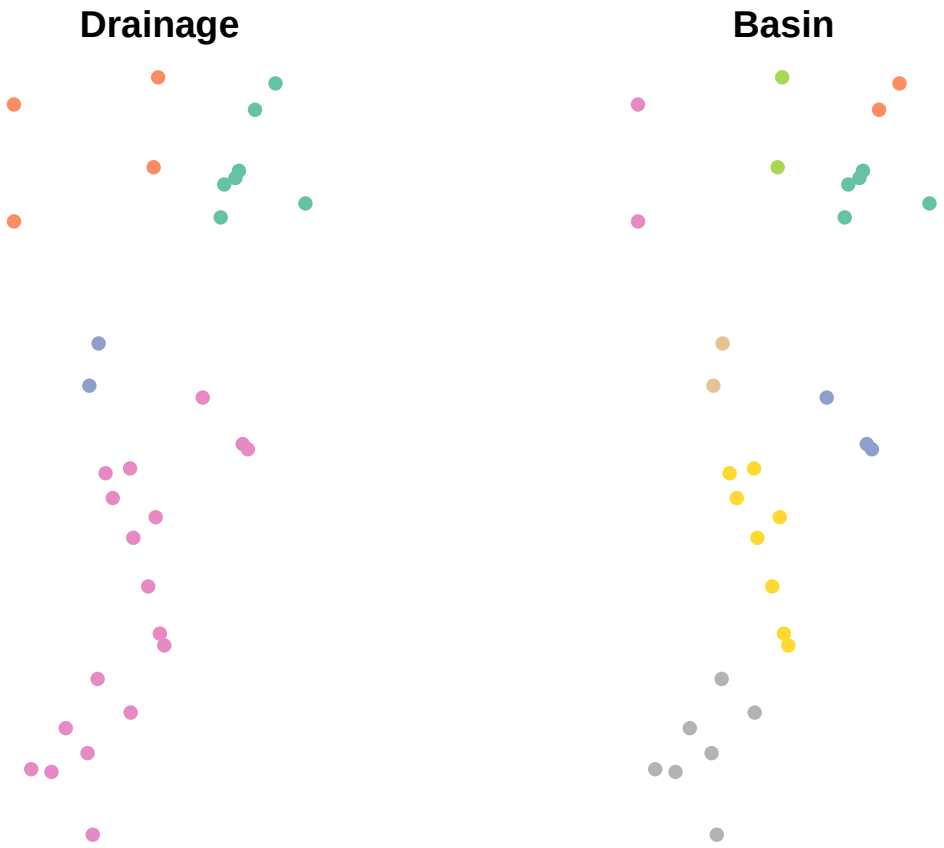
```
plot(RALU.site.sf, pch=16)
```

```
## Warning: plotting the first 10 out of 17 attributes; use max.plot = 17 to
## plot all
```



This is pretty cool! Let's have a closer look at the variables 'Drainage' and 'Basin'. These are factors, and each factor level is assigned a different color:

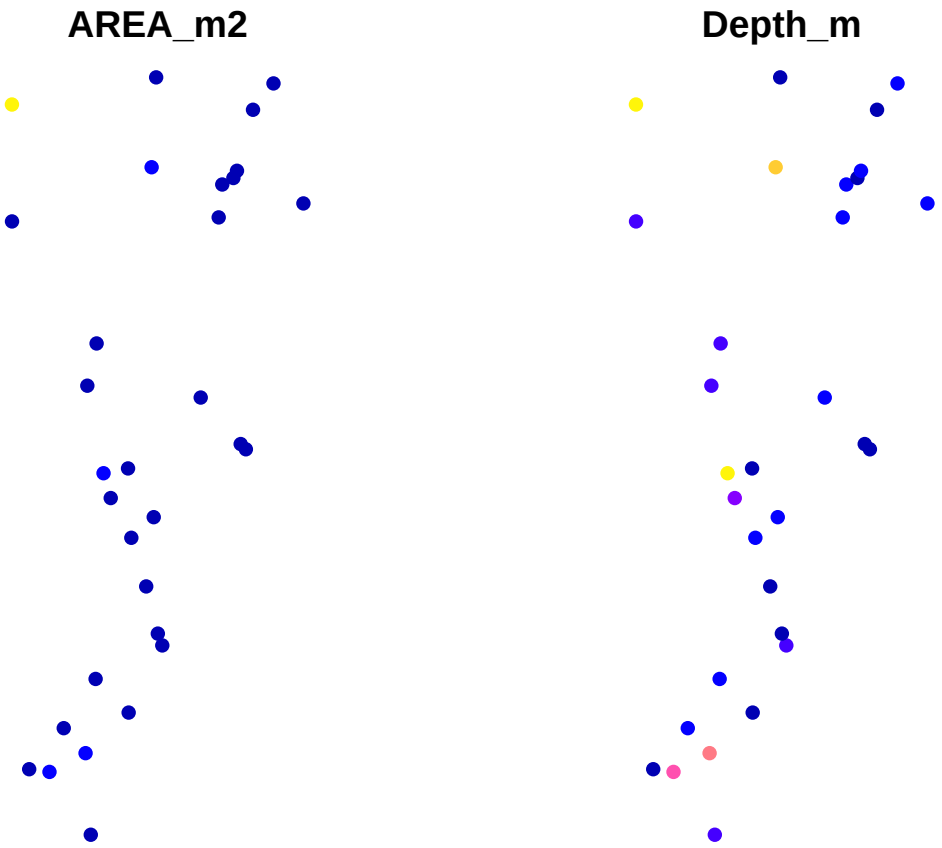
```
plot(RALU.site.sf[,c("Drainage", "Basin")], pch=16)
```



Now let's look at two quantitative variables, 'AREA_m2' and 'Depth_m'. R automatically uses a color ramp (from blue to pink to orange) to indicate variation in the values.

Note: To learn about options for the 'plot' function for 'sf' objects, access the help file by typing '?plot' and select 'Plot sf object'.

```
plot(RALU.site.sf[,c("AREA_m2", "Depth_m")], pch=16)
```



4. Export site data as ESRI shapefile

Just in case you want to know how you could get your results into a GIS. This is really easy with the 'sf' package. The following code may produce a warning that column names were abbreviated, and writes the component files for the ESRI shapefile into the pre-existing folder 'Output' (it won't create a new folder, you need to create it first).

The argument 'delete_dsn' specifies whether any existing file with the same name should be deleted first (i.e., overwritten).

```
st_write(RALU.site.sf, file.path(root, "Output", "RALU.site.shp"),
                                 delete_dsn = TRUE)
```

```
## Deleting source `/Users/hhwagner1/Desktop/DGS_Week1_Lab/Output/RALU.site.shp' using driver `ESRI Shap
## Writing layer `RALU.site' to data source `/Users/hhwagner1/Desktop/DGS_Week1_Lab/Output/RALU.site.shp
## features:       31
## fields:         17
## geometry type:  Point
```

5. Display categorical map with a predefined color table

Now to a more tricky topica. Recall that the last raster layer in the Worked Example, 'nlcd', contains categorial land cover data that are coded numerically. The 'raster' package actually misinterpreted them as numeric data.

Let's again extract the categorical raster layer into a new object 'NLCD'.

```
data(rasters)
NLCD <- raster(rasters[6])
NLCD
```

```
## class       : RasterLayer
## dimensions  : 426, 358, 152508  (nrow, ncol, ncell)
## resolution  : 30, 30  (x, y)
## extent      : 683282.5, 694022.5, 4992833, 5005613  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=utm +zone=11 +datum=NAD83 +units=m +no_defs +ellps=GRS80 +towgs84=0,0,0
## data source : in memory
## names       : nlcd
## values      : 11, 95  (min, max)
```

Then we use function 'ratify' to tell R that this is a categorical map (factor) that needs a 'raster attribute table' (rat). Nothing to do with rats...

Then create the raster attribute table from the factor levels. This is simply a vector with all factor levels present in 'NLCD'. The levels are codes as numbers between 11 and 95. Here's the legend information: https://www.mrlc.gov/nlcd06_leg.php

```
NLCD <- ratify(NLCD)
rat <- levels(NLCD)[[1]]
rat
```

```
##   ID
## 1 11
## 2 12
## 3 31
## 4 42
## 5 52
## 6 71
## 7 90
## 8 95
```

Let's add some columns with labels and predefined colors (using hex color code). A color table ("Colortable_LULC.csv") is already in the data folder.

It has more rows than we need, because not all US land cover classes occur in the study area. We need to make sure the colors and cover types are stored as 'character', because the missing factor levels would later create problems.

```
ColTab <- read.csv(file.path(root, "Data", "Colortable_LULC.csv"), header=TRUE)
ColTab$color <- as.character(ColTab$color)
ColTab$attribute <- as.character(ColTab$attribute)
ColTab
```

```
##    value   color                   attribute
## 1     11 #456DA8                  Open Water
## 2     12 #E6EEF9           Perennial Ice/Snow
## 3     21 #E1CBCD        Developed, Open Space
## 4     22 #DC9786     Developed, Low Intensity
## 5     23 #F40100  Developed, Medium Intensity
## 6     24 #B00206    Developed, High Intensity
## 7     31 #B2AEA3 Barren Land (Rock/Sand/Clay)
## 8     41 #6BA95C             Deciduous Forest
## 9     42 #16692E             Evergreen Forest
## 10    43 #B9CA8F                 Mixed Forest
## 11    51 #AD9439                  Dwarf Scrub
```

```
## 12    52 #D5B883                  Shrub/Scrub
## 13    71 #EDEFCA         Grassland/Herbaceous
## 14    72 #D3D27C             Sedge/Herbaceous
## 15    73 #A5CD53                     Lichens
## 16    74 #88B8A1                        Moss
## 17    81 #DED73E                  Pasture/Hay
## 18    82 #AD722C             Cultivated Crops
## 19    90 #BED8F6               Woody Wetlands
## 20    95 #6EA5C4 Emergent Herbaceous Wetlands
```

Now we need to match the codes in 'NLCD.rat' and 'ColTab' and extract the additional variables for the land use categories that occur in NLCD. We can use the function 'merge' to do this.

Then we change the order of columns so that 'attribute' is in the first column. This will be used to label the legend.

Finally we tell R that the factor levels of NLCD are defined in 'rat'.

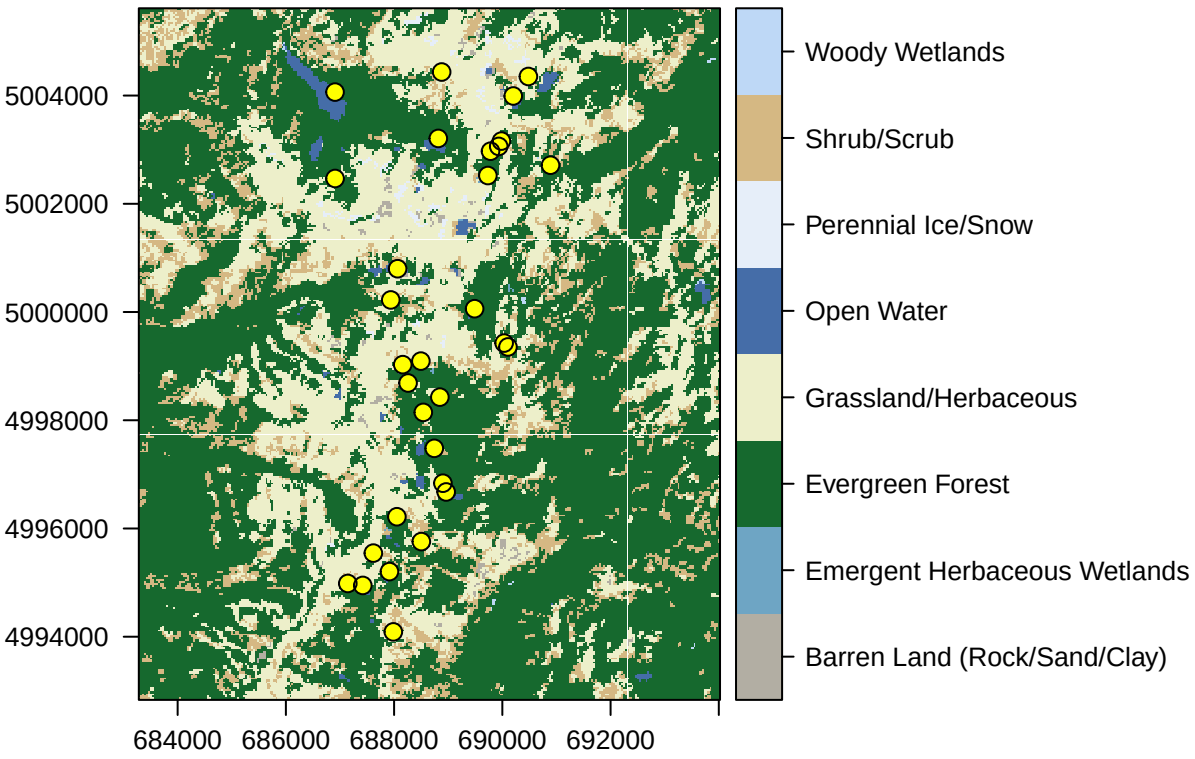
```
rat <- merge(rat, ColTab, by.x="ID", by.y="value", all=FALSE, sort=TRUE)
rat <- rat[,c(1,3,2)]
levels(NLCD) <- rat
rat
```

```
##   ID                    attribute   color
## 1 11                   Open Water #456DA8
## 2 12           Perennial Ice/Snow #E6EEF9
## 3 31 Barren Land (Rock/Sand/Clay) #B2AEA3
## 4 42              Evergreen Forest #16692E
## 5 52                   Shrub/Scrub #D5B883
## 6 71         Grassland/Herbaceous #EDEFCA
## 7 90                Woody Wetlands #BED8F6
## 8 95 Emergent Herbaceous Wetlands #6EA5C4
```

Now we can use the function 'levelplot' (package 'rasterVis') to plot the land cover map with the correct color scheme and legend. We will overlay the sampling locations, using yellow circles with black outlines.

Note: this code uses advanced plotting 'language', where we first create the levelplot of the categorical map (assigning a specific color to each factor level of NLCD), then add a layer with the yellow filled circles at the sampling locations, and another layer with the black symbol outlines. These layers are added to the plot with '+ layer()'

```
levelplot(NLCD, col.regions = rat$color[order(rat$attribute)], colorkey=list(height=1)) +
  layer(sp.points(RALU.site.sp, pch=16, col="yellow", cex=1.1)) +
  layer(sp.points(RALU.site.sp, pch=1, col="black", cex=1.1))
```


5004000
5002000
5000000
4998000
4996000
4994000
684000
686000
688000
690000
692000
Woody Wetlands
Shrub/Scrub
Perennial Ice/Snow
Open Water
Grassland/Herbaceous
Evergreen Forest
Emergent Herbaceous Wetlands
Barren Land (Rock/Sand/Clay)