Ag Carpentry - Trial Data

Brittani

10/19/2019

Motivating Questions:

- What are the common file types in agricultural data?

Objectives with Agricultural Data Types

- Import agricultural datasets into R
- Describe the contents of files generated during planting, fertilization, and harvest
- Describe the contents of files used to control seeding and fertilization rate
- Assess the accuracy of machinery in applying the target rates

Keypoints:

- sf is prefereable for data analysis; it is easier to access the dataframe
- Projecting your data in utm is necessary for many of the geometric operations you perform (e.g. making trial grids and splitting plots into subplot data)
- Compare different data formats, such as gpkg, shp(cpg,dbf,prj,sbn,sbx),geojson,tif

Below are the packages that we will use in this episode.

```
library(sf)
library(httr)
library(rgdal)
```

```
## Loading required package: sp
```

```
## rgdal: version: 1.4-7, (SVN revision 845)
##  Geospatial Data Abstraction Library extensions to R successfully loaded
##  Loaded GDAL runtime: GDAL 2.4.2, released 2019/06/28
##  Path to GDAL shared files: /Library/Frameworks/R.framework/Versions/3.6/Resources/library/rgdal/gda
##  GDAL binary built with GEOS: FALSE
##  Loaded PROJ.4 runtime: Rel. 5.2.0, September 15th, 2018, [PJ_VERSION: 520]
##  Path to PROJ.4 shared files: /Library/Frameworks/R.framework/Versions/3.6/Resources/library/rgdal/p
##  Linking to sp version: 1.3-1
```

```
library(rgeos)
```

```
## rgeos version: 0.5-2, (SVN revision 621)
##  GEOS runtime version: 3.7.2-CAPI-1.11.2
##  Linking to sp version: 1.3-1
##  Polygon checking: TRUE
```

```
library(maptools)
```

```
## Checking rgeos availability: TRUE
```

```
require(tmap)
```

```
## Loading required package: tmap
```

```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```
require(gridExtra)
```

```
## Loading required package: gridExtra
```

```
library(readr)
library(measurements)
```

In this lesson we will explore the files that are generated during a trial season. These data include yield, as-applied, as-planted, and sometimes electricalconductivity. While you are likely using your yield maps every year to asses productivity, you might not be looking at your application maps if you normally use uniform rates. But if you use variable rate applications or have completed an agricutlural trial, your application map contains information about how well the machine applied the target rates.

Reading the Files

In the next exercise we will bring these data into the R environment. We've already looked at the asplanted data in the geospatial lesson. Now let's see what variables are contained in the other files.

Exercise Read the yield, as-planted, and as-applied files and explore the variables. For each file, identify what variables might we be interested in and why?

```
planting <- read_sf("asplanted_clean.shp")
nitrogen <- read_sf("asapplied_clean.shp")
yield <- read_sf("yield_clean.gpkg")
```

Solutions and Discussion

```
names(nitrogen)
```

```
##  [1] "Field"    "Dataset"  "Product"  "Obj__Id"  "Trck_d_"  "Swth_W_"
##  [7] "Dstnc_f"  "Durtn_s"  "Elevtn_"  "Are_Cnt"  "Dff_Stt"  "Time"
## [13] "Y_Offs_"  "X_Offs_"  "Rt_A_M_"  "Pass_Nm"  "Spd_mp_"  "Prd_c_h"
## [19] "Date"     "Rt_Appl"  "Rt_Ap_1"  "geometry"
```

As-Applied File

The nitrogen file contains 22 variables. The nitrogen type of is reported in `Product`, and this field used NH3. We have columns with the applied rate in both gallons (`Rt_Ap_1`) and pounds (`Rt_Appl`). We typically use pounds in order to compare across different nitrogen types.

There is not a clear column with the target rate. This is not a problem as we will later merge the trial map and the application map, so we can compare the rate applied to the intended rate.

We can also see the automatic-section control reacting when the applicator is off of the guidance line, with the swatch width changing in `Swth_W_`.

Yield File

```
names(yield)
```

```
##  [1] "Field"      "Dataset"    "Product"    "Obj__Id"    "Distance_f"
##  [6] "Track_deg_" "Duration_s" "Elevation_" "Time"       "Area_Count"
## [11] "Swth_Wdth_" "Y_Offset_f" "Crop_Flw_M" "Moisture__" "Yld_Mass_W"
## [16] "Yld_Vol_We" "Yld_Mass_D" "Yld_Vol_Dr" "Humidity__" "Air_Temp__"
## [21] "Wind_Speed" "Soil_Temp_" "Wind_Dir"   "Sky_Cond"   "Pass_Num"
## [26] "Speed_mph_" "Prod_ac_h_" "Crop_Flw_V" "Date"       "Yield__Dry"
## [31] "out"        "geom"
```

As-Planted File

```
names(planting)
```

```
##  [1] "Field"    "Dataset"  "Product"  "Obj__Id"  "Dstnc_f"  "Trck_d_"
##  [7] "Durtn_s"  "Elevtn_"  "Time"     "Are_Cnt"  "Swth_W_"  "Sd_Cn__"
## [13] "Plnt_P_"  "Rt_A_C_"  "SdFlw_k"  "Tgt_Rt_"  "Y_Offs_"  "DF_Mrg_"
## [19] "Hmdty__"  "Ar_Tm__"  "Wnd_Spd"  "Sl_Tmp_"  "Pass_Nm"  "Spd_mp_"
## [25] "Prd_c__"  "Prdt_Am"  "Date"     "Popultn"  "Rat__Cn"  "Trgt_Rt"
## [31] "Ppltn_1"  "Dt___Tm"  "geometry"
```

We see that the planting file has 33 variables, several of which appear to be identical. The main variables of interest are the planting rate (`Rt_A_C_`) and the target rate (`Tgt_Rt_`). These columns do appear under different names. We will discuss how to handle this below.

There are several other variables that could be useful. First, the hybrid is located in the `Product` column; we want to keep this in our records if we want to compare across years. We may also want to know things like the swath width (`Swth_W_`) or elevation (`Elevtn`).

Exercise Make a map of the seed application rate from the `planting` file. Find the column name with the variable seeding rate in the dataset. There are many variables in this dataset, and some of the names may look unclear. Looking at the summary of the variables will help determine which variable is the applied rate and which is the target rate.

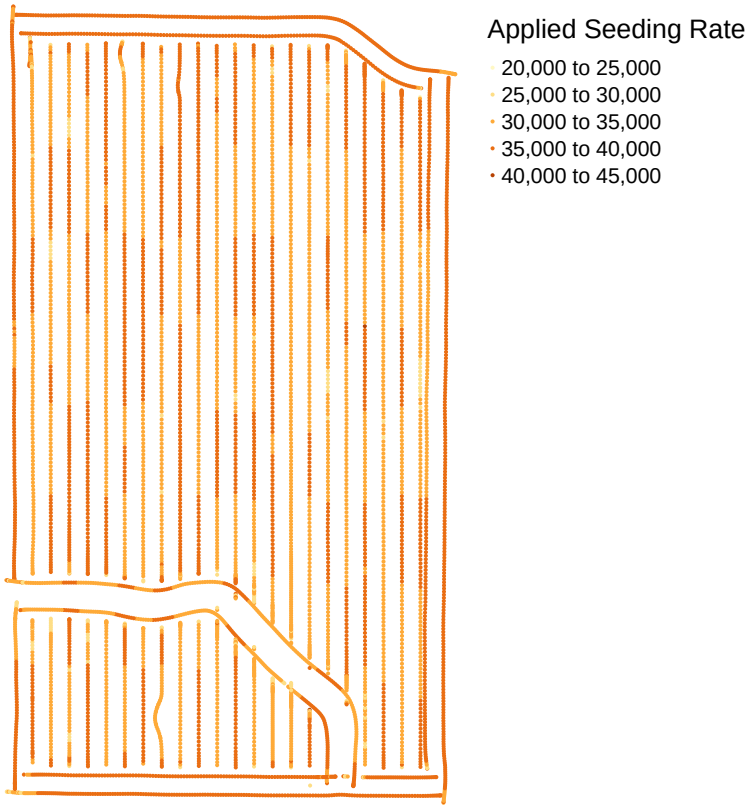
Solution

```
names(planting)
```

```
##  [1] "Field"    "Dataset"  "Product"  "Obj__Id"  "Dstnc_f"  "Trck_d_"
##  [7] "Durtn_s"  "Elevtn_"  "Time"     "Are_Cnt"  "Swth_W_"  "Sd_Cn__"
## [13] "Plnt_P_"  "Rt_A_C_"  "SdFlw_k"  "Tgt_Rt_"  "Y_Offs_"  "DF_Mrg_"
## [19] "Hmdty__"  "Ar_Tm__"  "Wnd_Spd"  "Sl_Tmp_"  "Pass_Nm"  "Spd_mp_"
## [25] "Prd_c__"  "Prdt_Am"  "Date"     "Popultn"  "Rat__Cn"  "Trgt_Rt"
## [31] "Ppltn_1"  "Dt___Tm"  "geometry"
```

We see many names in the planting file. It appears that `Rt_A_C_` is the applied rate and `Tgt_Rt_` is the target rate. We also know from when we loaded this file into the environment that it contains SpatialPoints not polygons. Thus, we cannot use `tm_polygons()`; instead we use `tm_dots()` to map the points with a colored dot for the level of applied seed.

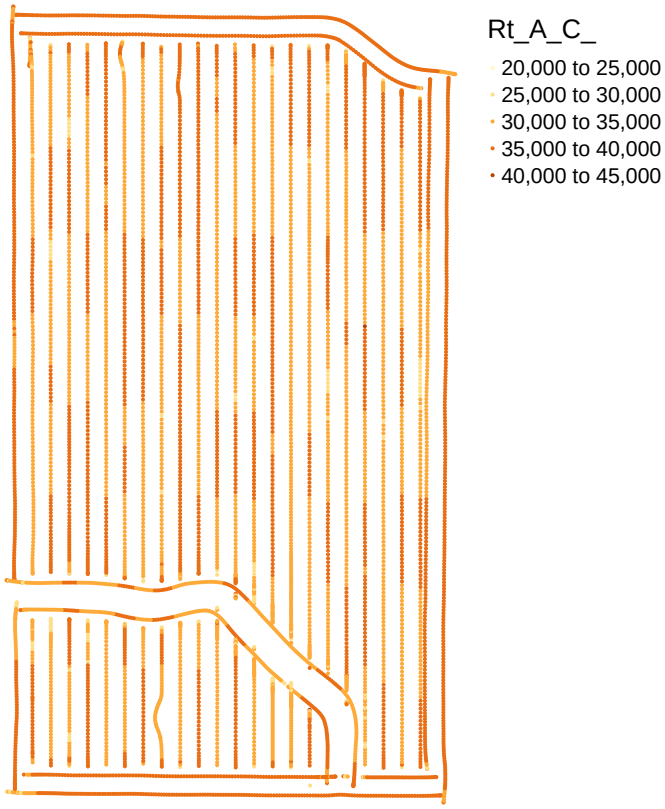
```
tm_shape(planting) + tm_dots('Rt_A_C_', title = "Applied Seeding Rate") +
  tm_layout(legend.outside = TRUE, frame = FALSE) +
  tm_legend(text.size = .7,
            title.size = 1,
            width = 100,
            bg.color = "white")
```



Challenge Try modifying code above to answer the following questions: Why do we supply a title in `tm_dots` and why is `legend.outside = TRUE`?

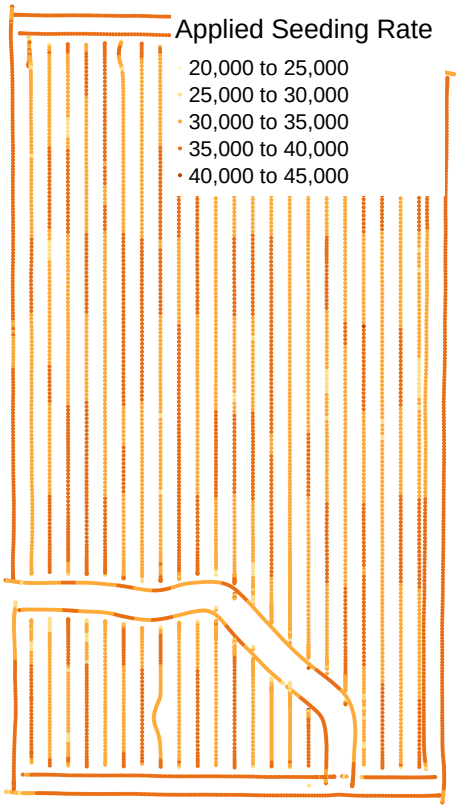
Solution If we remove the title from `tm_dots()`, we will have the original column name as the legend label. We might understand that this is the applied seed, but someone else looking at the graph likely would not. Also, if you create a map and want to look back it, you're unlikely to remember what this name meant.

```
tm_shape(planting) + tm_dots('Rt_A_C_') +
  tm_layout(legend.outside = TRUE, frame = FALSE) +
  tm_legend(text.size = .7,
            title.size = 1,
            width = 100,
            bg.color = "white")
```



If we add `legend.outside = FALSE`, the legend will block part of the map. In some cases, the map may have a vacant section where the legend can go, but here we need to move the legend outside to be able to see all of the points on the field.

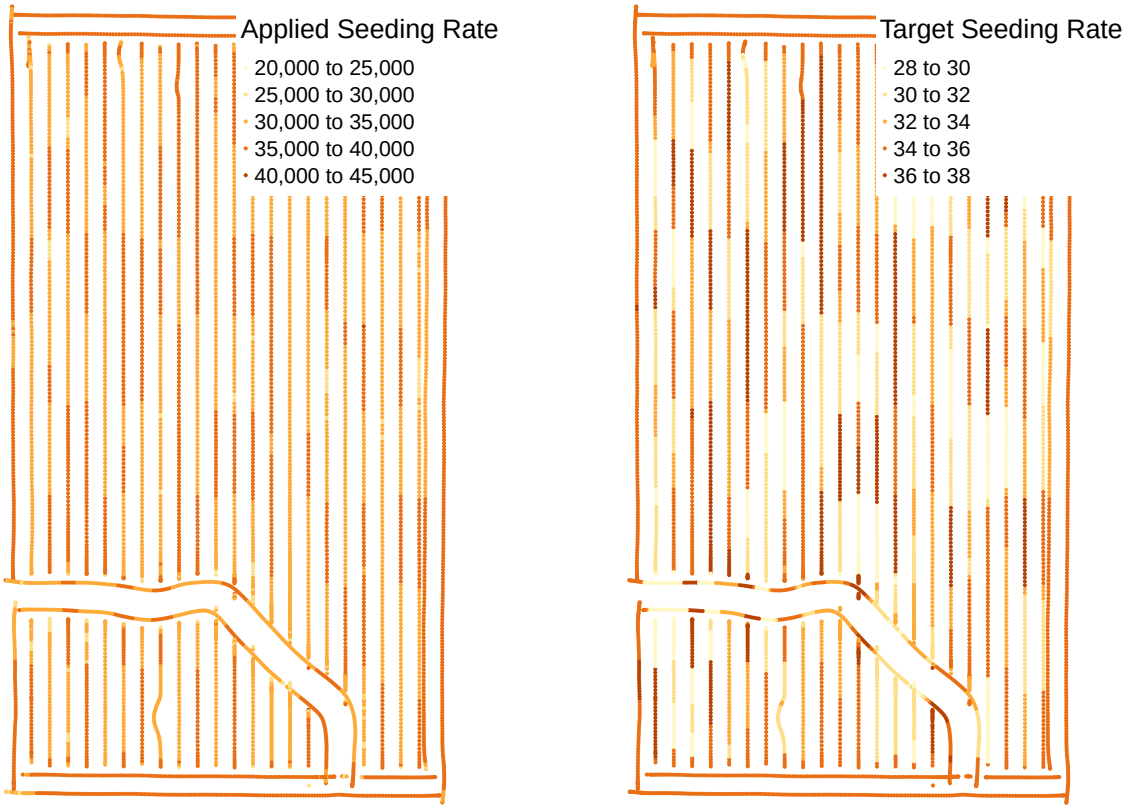
```
tm_shape(planting) + tm_dots('Rt_A_C_', title = "Applied Seeding Rate") +
  tm_layout(legend.outside = FALSE, frame = FALSE) +
  tm_legend(text.size = .7,
            title.size = 1,
            width = 100,
            bg.color = "white")
```



Side-by-Side Maps

Some kinds of maps you want to see close together. For example, perhaps we want to asses how well the as-applied rates lined up with the target rates for seed and nitrogen. We can use `tmap_arrange()` to make a grid of `tmap` objects. We define `ncol` and `nrow`, and the command will arrange the objects given into the grid. In this case, we have two objects we want to see next to each other, so we have two columns and one row.

```
map_applieds <- tm_shape(planting) + tm_dots('Rt_A_C_', title = "Applied Seeding Rate") +
  tm_layout(legend.outside = FALSE, frame = FALSE) +
  tm_legend(text.size = .7,
            title.size = 1,
            width = 100,
            bg.color = "white")
map_tgts <- tm_shape(planting) + tm_dots('Tgt_Rt_', title = "Target Seeding Rate") +
  tm_layout(legend.outside = FALSE, frame = FALSE) +
  tm_legend(text.size = .7,
            title.size = 1,
            width = 100,
            bg.color = "white")
tmap_arrange(map_applieds, map_tgts, ncol = 2, nrow = 1)
```



From the applied and target seeding rate maps, we can see that this trial had a very accurate application of the designed seeding rates. This is a common result for seed, which has more accurate application than nitrogen. However, we still have maximum and minimum applied rates that are much higher than the designed rates.