Lab 1

Environmental Analysis in R

Srishti Roy Chowdhury

Task- 1.1

```
> #Task-1.1
> d1.counties <- d.counties %>% group_by(STATEFP10) %>% mutate(stateLand = sum(ALAND10,AWATER10))
> #view(d1.counties)
> d11.counties <- d1.counties %>% group_by(COUNTYFP10) %>% mutate(countyLandpercent = ((ALAND10/stateLand)*100))
> d11.counties %>% group_by(COUNTYFP10) %>% dplyr::select(STATEFP10,COUNTYFP10,countyLandpercent) %>% head()
Simple feature collection with 6 features and 3 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -79.38264 ymin: 37.69574 xmax: -76.95493 ymax: 40.72605
Geodetic CRS:  WGS 84
# A tibble: 6 × 4
# Groups:   COUNTYFP10 [6]
  STATEFP10 COUNTYFP10 countyLandpercent                                                                           geometry
  <chr>     <chr>                  <dbl>                                                                 <MULTIPOLYGON [°]>
1 51        540                   0.0342 (((-78.47082 38.04893, -78.47086 38.04893, -78.47096 38.0489, -78.47125 38.0…
2 51        510                   0.0502 (((-77.06129 38.79457, -77.0612 38.79454, -77.06092 38.79444, -77.06083 38.7…
3 51        530                   0.0224 (((-79.36668 37.7267, -79.36655 37.72627, -79.36653 37.72617, -79.3665 37.72…
4 51        600                   0.0208 (((-77.31476 38.86701, -77.31534 38.86702, -77.31537 38.86702, -77.31538 38.…
5 42        021                   2.26   (((-79.03546 40.31539, -79.03611 40.31477, -79.0363 40.31459, -79.03725 40.3…
6 42        001                   1.70   (((-77.46594 39.85958, -77.46589 39.85924, -77.46586 39.85902, -77.46584 39.…
```

Task- 1.2

```
> #Task-1.2
> d1.counties <- d1.counties %>% group_by(COUNTYFP10) %>% mutate(countyWLand = (AWATER10/stateLand))
> #view(d1.counties)
> d1.counties %>% group_by(STATEFP10) %>% dplyr::select(STATEFP10,COUNTYFP10,countyWLand) %>% dplyr::filter(countyWLand == max (countyWLand))
Simple feature collection with 7 features and 3 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -80.89419 ymin: 37.44857 xmax: -74.98417 ymax: 43.70813
Geodetic CRS:  WGS 84
# A tibble: 7 × 4
# Groups:   STATEFP10 [7]
  STATEFP10 COUNTYFP10 countyWLand                                                                           geometry
* <chr>     <chr>            <dbl>                                                                 <MULTIPOLYGON [°]>
1 42        071           0.00131  (((-76.42964 40.23898, -76.43194 40.23868, -76.43203 40.23867, -76.43415 40.23839,…
2 51        001           0.0287   (((-75.46228 38.0082, -75.46734 38.00775, -75.46812 38.0077, -75.46982 38.00759, -…
3 24        019           0.0356   (((-76.14771 38.63003, -76.14871 38.63028, -76.1491 38.63037, -76.15131 38.63116, …
4 36        011           0.0107   (((-76.73804 42.96123, -76.73814 42.96113, -76.73823 42.96104, -76.73826 42.96102,…
5 54        025           0.000614 (((-80.73152 37.81926, -80.73147 37.81918, -80.72914 37.81585, -80.72651 37.81209,…
6 10        005           0.105    (((-75.70147 38.55944, -75.701 38.55352, -75.7005 38.54712, -75.7005 38.54708, -75…
7 11        001           0.107    (((-76.95602 38.92942, -76.95627 38.92961, -76.95759 38.93063, -76.95767 38.93069,…
```

Task- 1.3

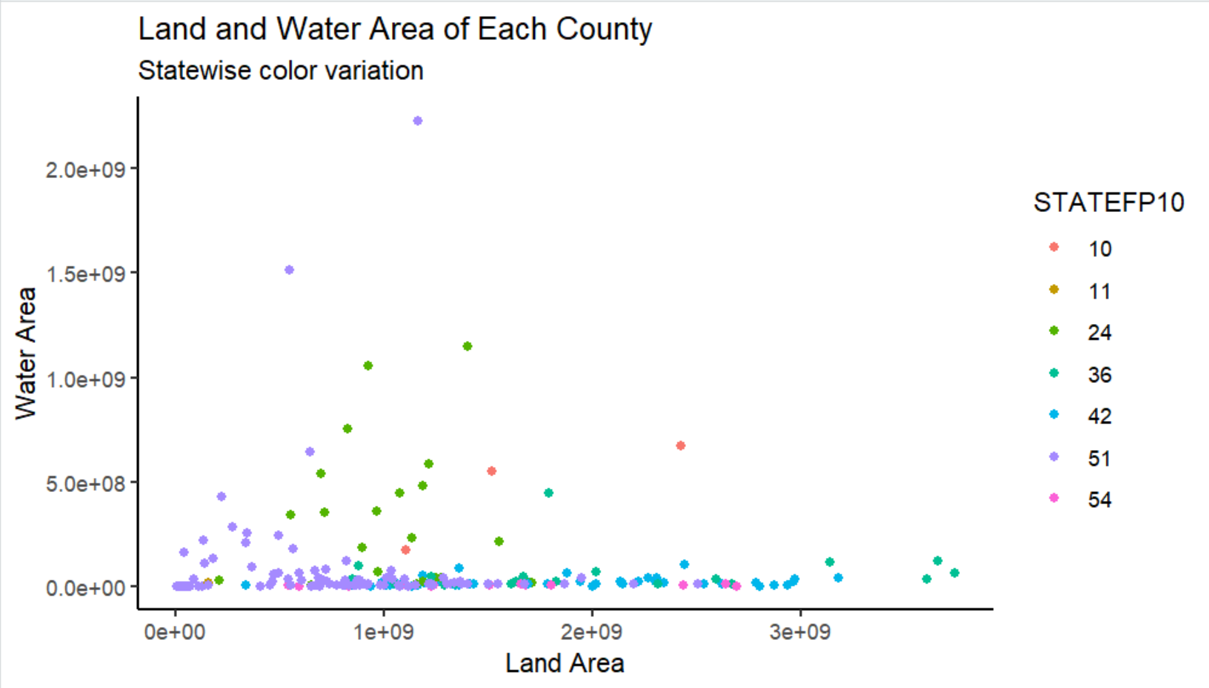
```
> #Task-1.3
> d.counties %>% as_tibble() %>% group_by(STATEFP10) %>% summarise(statecounty = n())
# A tibble: 7 × 2
  STATEFP10 statecounty
  <chr>           <int>
1 10                  3
2 11                  1
3 24                 24
4 36                 20
5 42                 43
6 51                102
7 54                 14
```

Task- 1.4

```
> #Task-1.4
> d.stations %>% dplyr::slice_min(nchar(STATION_NA))
Simple feature collection with 2 features and 11 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: -79.17917 ymin: 37.63376 xmax: -76.69635 ymax: 39.36667
Geodetic CRS:  WGS 84
# A tibble: 2 × 12
  OBJECTID MAP_ID USGS_STATI STATION_NA                 MAJOR_WATE Drainage_A START_DATE END_DATE   Lat  Long STAID              geometry
     <int>  <int>      <int> <chr>                      <chr>           <dbl>      <int>    <int> <dbl> <dbl> <chr>            <POINT [°]>
1      105    106    1595300 ABRAM CREEK AT OAKMONT,… Potomac          42.6       2013     2018  39.4 -79.2 0159… (-79.17917 39.36667)
2      114    116    1669520 DRAGON SWAMP AT MASCOT,… Virginia        109.        2011     2018  37.6 -76.7 0166… (-76.69635 37.63376)
```

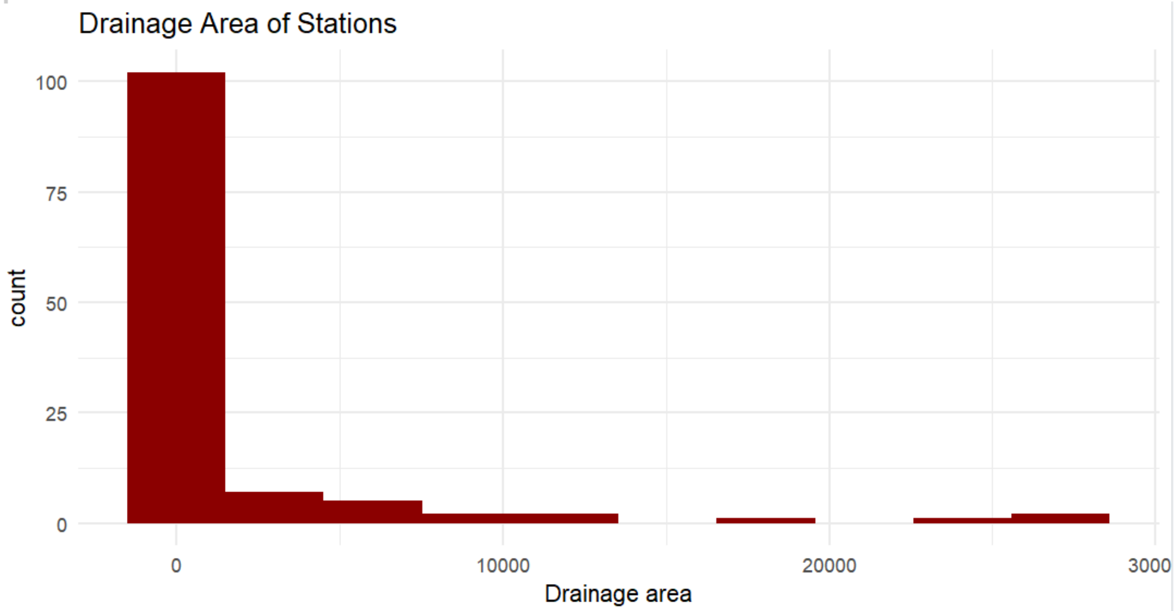
Task- 2.1

```
> #Task-2.1
> d.counties %>% ggplot(., aes(x = ALAND10,y=AWATER10,color = STATEFP10)) +
+   geom_point() +
+   theme_classic() +
+   labs(title = "Land and Water Area of Each County",
+        subtitle = "Statewise color variation",
+        x = "Land Area",
+        y = "Water Area")
```



Task- 2.2

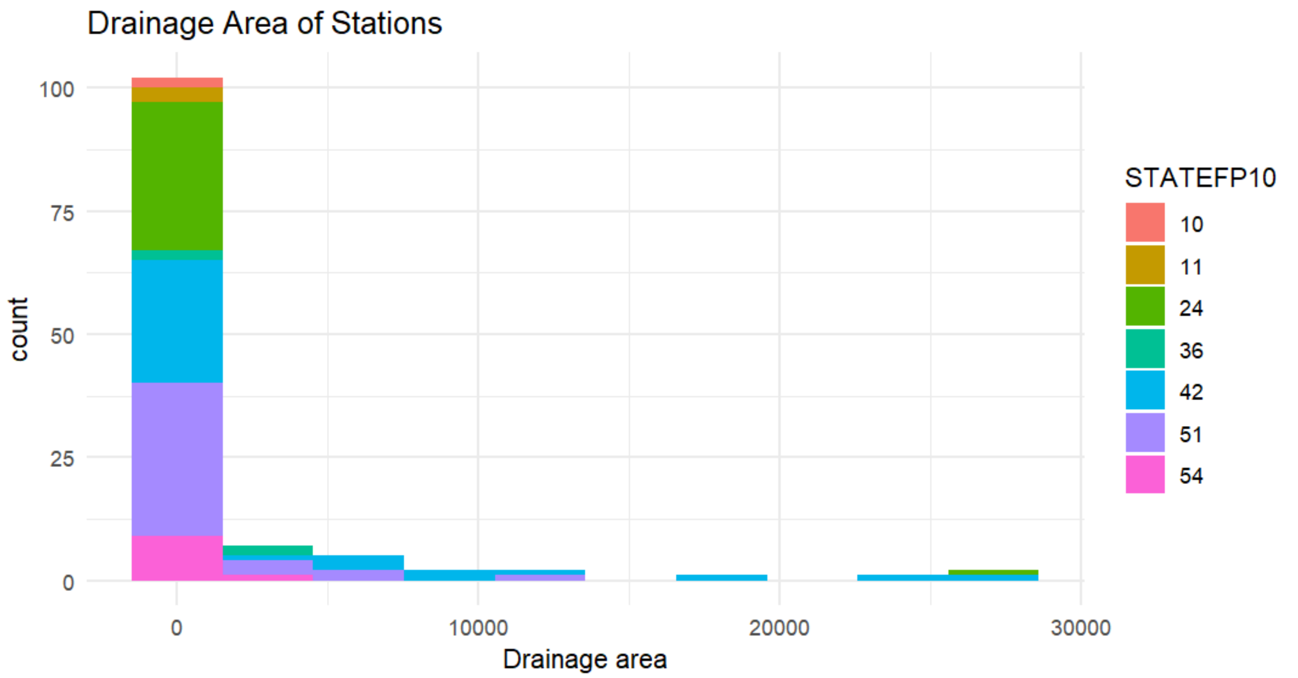
```
> #Task-2.2
> d.stations %>% ggplot(., aes(x = Drainage_A)) +
+   geom_histogram(fill = "dark red",bins=10) +
+   theme_minimal() +
+   labs(title = "Drainage Area of Stations",
+        x = "Drainage area")
```



```
#geometry files ready
d.counties %>% sf::st_crs() == d.stations %>% sf::st_crs()
d.stations %>% sf::st_is_valid()
d.counties %>% sf::st_is_valid()
d.counties <- d.counties %>% sf::st_make_valid()
dx.counties <- d.counties %>% dplyr::select(STATEFP10,NAMELSAD10,COUNTYFP10,GEOID10,ALAND10,Shape_Area,geometry)
county.station <- sf::st_intersection(dx.counties,d.stations)
#view(county.station)
```

Task- 2.3

```
> #Task-2.3
> ggplot(county.station, aes(x = Drainage_A,fill=STATEFP10)) +
+   geom_histogram(bins=10) +
+   theme_minimal() +
+   labs(title = "Drainage Area of Stations",
+        x = "Drainage area")
```



Task- 3

```
> #Task-3
> my_function <- function(numbers){
+   mean(numbers)
+   median(numbers)
+   min(numbers)
+   max(numbers)
+   sort(numbers, decreasing = FALSE)
+
+   if(is.numeric(numbers)==FALSE){
+     return("Error")
+   }
+
+   else{
+     return(
+        list(  mean(numbers),
+               median(numbers),
+               min(numbers),
+               max(numbers),
+               sort(numbers, decreasing = FALSE)
+           )
+         )
+     }
+ }
```

```
> my_function(c(1,0,-1))
[[1]]
[1] 0

[[2]]
[1] 0

[[3]]
[1] -1

[[4]]
[1] 1

[[5]]
[1] -1  0  1
```

```
> my_function(c(10,100,1000))
[[1]]
[1] 370

[[2]]
[1] 100

[[3]]
[1] 10

[[4]]
[1] 1000

[[5]]
[1]   10  100 1000
```

```
> my_function(c(.1, .001, 1e8))
[[1]]
[1] 33333333

[[2]]
[1] 0.1

[[3]]
[1] 0.001

[[4]]
[1] 1e+08

[[5]]
[1] 1e-03 1e-01 1e+08
```

```
> my_function(c("a", "b", "c"))
[1] "Error"
Warning message:
In mean.default(numbers) : argument is not
numeric or logical: returning NA
```

Task- 4.1

```
> #Task-4.1
> county.station %>% as_tibble() %>% group_by(STATEFP10) %>% summarise(statestations = n())
# A tibble: 7 × 2
  STATEFP10 statestations
  <chr>             <int>
1 10                    2
2 11                    3
3 24                   31
4 36                    4
5 42                   35
6 51                   37
7 54                   10
```

Task- 4.2

#Calculating the average size of NY counties considering intersected file

I tried to calculate the area of study area counties of New York using "*st_area*". However, the results returned zero (0), so I used the variable "*Shape_Area*" of the dataset to calculate the average size of the counties of New York.

```
> #Calculating area using "st_area" function
> ny.counties <- county.station %>% dplyr::filter(STATEFP10 == 36)
> ny.counties %>% st_area()
Units: [m^2]
[1] 0 0 0 0
> #avg. area calculation using Shape_area variable
> county.station %>% dplyr::filter(STATEFP10 == 36)%>% group_by(STATEFP10) %>% summarise(avgAcounty = mean(Shape_Area))
Simple feature collection with 1 feature and 2 fields
Geometry type: MULTIPOINT
Dimension:     XY
Bounding box:  xmin: -77.21667 ymin: 42.00223 xmax: -75.40611 ymax: 42.37779
Geodetic CRS:  WGS 84
# A tibble: 1 x 3
  STATEFP10  avgAcounty                                                                                  geometry
  <chr>           <dbl>                                                                          <MULTIPOINT [°]>
1 36        4059967165. ((-76.63472 42.00223), (-77.21667 42.25251), (-75.80306 42.03529), (-75.40611 42.37779))
```

#Calculating the average size of NY counties considering the county boundary dataset

```
> d.counties %>% dplyr::filter(STATEFP10 == 36)%>% group_by(STATEFP10) %>% summarise(avgAcounty = mean(Shape_Area))
Simple feature collection with 1 feature and 2 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -78.30932 ymin: 41.8483 xmax: -74.16468 ymax: 44.09697
Geodetic CRS:  WGS 84
# A tibble: 1 x 3
  STATEFP10  avgAcounty                                                                                       geometry
  <chr>           <dbl>                                                                                  <POLYGON [°]>
1 36        3870377716. ((-74.3497 42.78138, -74.35103 42.78128, -74.36261 42.78041, -74.36276 42.7804, -74.37331...
```

Task- 4.3

```
> #Task-4.3
> avgDrainage <- county.station %>% group_by(STATEFP10) %>% summarise(avgAdrainage = mean(Drainage_A))
> #view(avgDrainage)
> avgDrainage %>% dplyr::slice_max(avgAdrainage)
Simple feature collection with 1 feature and 2 fields
Geometry type: MULTIPOINT
Dimension:     XY
Bounding box:  xmin: -78.26529 ymin: 39.77251 xmax: -75.88075 ymax: 41.98473
Geodetic CRS:  WGS 84
# A tibble: 1 x 3
  STATEFP10 avgAdrainage                                                                                  geometry
  <chr>            <dbl>                                                                          <MULTIPOINT [°]>
1 42               3584. ((-75.88075 41.25092), (-75.89464 41.55842), (-76.36774 39.9465), (-76.3283 39.90594),...
```

Question-Answer

1. In my opinion, the following codes will work differently, because intersection functions behave differently with different types of spatial objects.

- sf::st_intersection(d.stations, del.counties)
- sf::st_intersection(del.counties, d.stations)

The intersection operation always returns features from that first argument that intersects with the second argument. The geometry that is used first in the code works as a base map. In ArcGIS, the geometry of the final map depends on the base map. Even in the new attribute table, the attribute table of the base map is placed first and then the table of second geometry is placed. Additionally, as a point feature does not take up too much space, it might not affect the final map because of the order. However, using two polygons or one polygon and one line feature might affect the final map based on the order used in the intersection function.

2. As this is my first experience using R, the whole experience is new to me. So, I found it interesting and challenging at the same time. During this lab, I experienced the fact that there are multiple ways to solve one problem which challenged me to do trial and error to come up with an efficient solution. Additionally, this compelled me to think critically.

3. Both individual and group activities would be interesting to perform. I would love to learn about areas or problems that can be solved easily with R language. Also, I want to learn more about utilizing R in ArcGIS.