8.2 Preparing Linear Measures

Manual of Applied Spatial Ecology

3/11/2022

First we will begin with determining the distance between several features. In our first example, we want to measure distance from each mule deer location to the nearest stream if it is determined a priori that water or riparian habitats influence mule deer distribution in our study area. While this may not seem like a very complicated process, there are numerous steps needed to achieve this feat. We will need to use the package spatstat that will help us in creating individual segments with nodes for linear features such as roads and streams/rivers.

1. Exercise 8.2 - Download and extract zip folder into your preferred location

2. Set working directory to the extracted folder in R under Session - Set Working Directory...

3. Now open the script LinearDistscript.Rmd" and run code directly from the script

4. First we need to load the packages needed for the exercise

```
library(spatstat)
library(maptools)#Needed for spatstat class "owin"
library(polyCub)#replaces gpclib function
library(rgdal)
library(raster)
library(rgeos)
library(OneR)#bin function
```

5. Now we will have a separate section of code to include projection information we will use throughout the exercise. In previous versions, these lines of code were within each block of code

```
utm12.crs<-"+proj=utm +zone=12 +datum=NAD83 +units=m +no_defs +ellps=GRS80 +towgs84=0,0,0"
Albers.crs <-CRS("+proj=aea +lat_1=29.5 +lat_2=45.5 +lat_0=23 +lon_0=-96 +x_0=0 +y_0=0
  +datum=NAD83 +units=m +no_defs +ellps=GRS80 +towgs84=0,0,0")
```

6. We will use the mule deer dataset we used in previous exercises

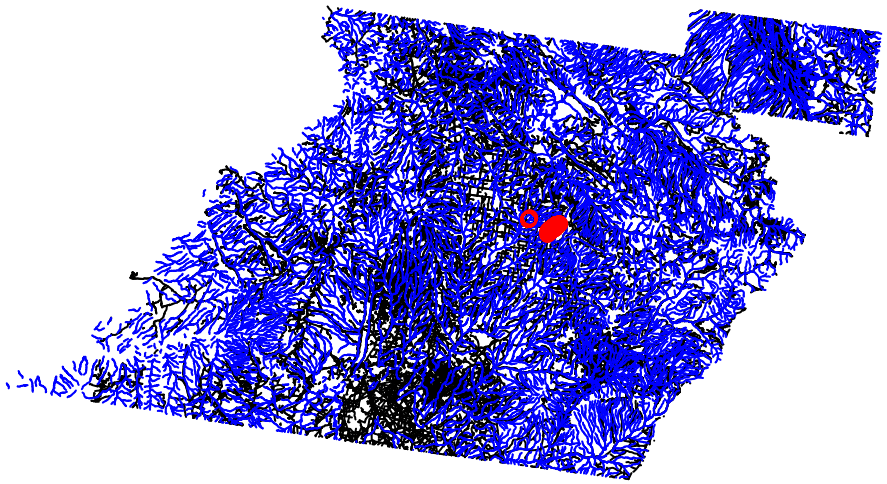
```
muleys <-read.csv("muleysexample.csv", header=T)

#Remove outlier locations
newmuleys <-subset(muleys, muleys$Long > -110.90 & muleys$Lat > 37.80)
muleys <- newmuleys
newmuleys <-subset(muleys, muleys$Long < -107)
muleys <- newmuleys

#Make a spatial data frame of locations after removing outliers
coords<-data.frame(x = muleys$X, y = muleys$Y)
deer.spdf <- SpatialPointsDataFrame(coords= coords, data = muleys, proj4string = CRS(utm12.crs))
```

7. Load the necessary road and rivers shapefiles already in Albers projection to match previous vegetation raster.

```
roads<-readOGR(dsn=".",layer="AlbersRoads",verbose=FALSE)
rivers<-readOGR(dsn=".",layer="AlbersRivers",verbose=FALSE)
```

8. We need to project the deer.spdf from utm to Albers to match other road and river shapefiles

```
deer.albers <-spTransform(deer.spdf, CRS=Albers.crs)
plot(roads)
plot(rivers,add=T, col="blue")
points(deer.albers, col="red")
```



9. Determine boundary box around mule deer locations to create a layer to clip and zoom in.

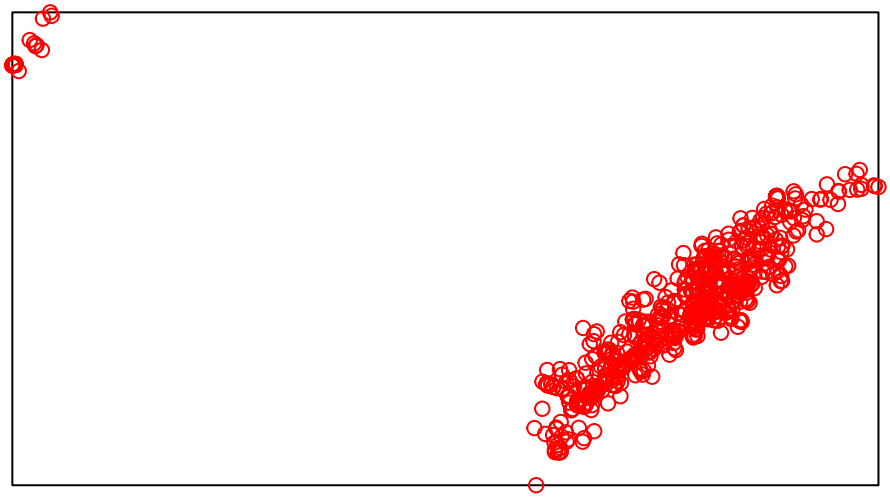
```
bbox(deer.albers)
```

```
##        min      max
## x -1127964 -1115562
## y  1718097  1724868
```

```
bb1 <- cbind(x=c(-1127964,-1127964,-1115562,-1115562,-1127964),
  y=c(1718097, 1724868, 1724868,1718097,1718097))
AlbersSP <- SpatialPolygons(list(Polygons(list(Polygon(bb1)),"1")),
            proj4string=CRS(proj4string(deer.albers)))
```

```
## Warning in proj4string(deer.albers): CRS object has comment, which is lost in output; in tests, see
## https://cran.r-project.org/web/packages/sp/vignettes/CRS_warnings.html
```

```
plot(AlbersSP)
points(deer.albers, col="red")
```



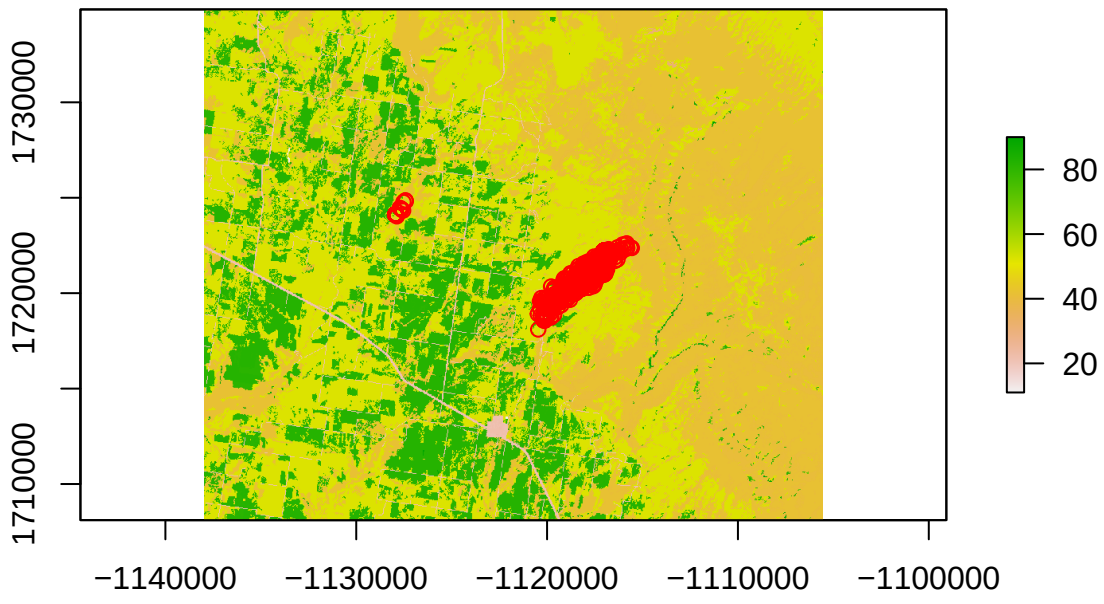
10. Load vegetation raster layer tif that came in the Albers projection from the online source.

```
veg <-raster("cropnlcd.tif")
```

```
## Warning in showSRID(SRS_string, format = "PROJ", multiline = "NO", prefer_proj =
## prefer_proj): Discarded datum unknown in Proj4 definition
```

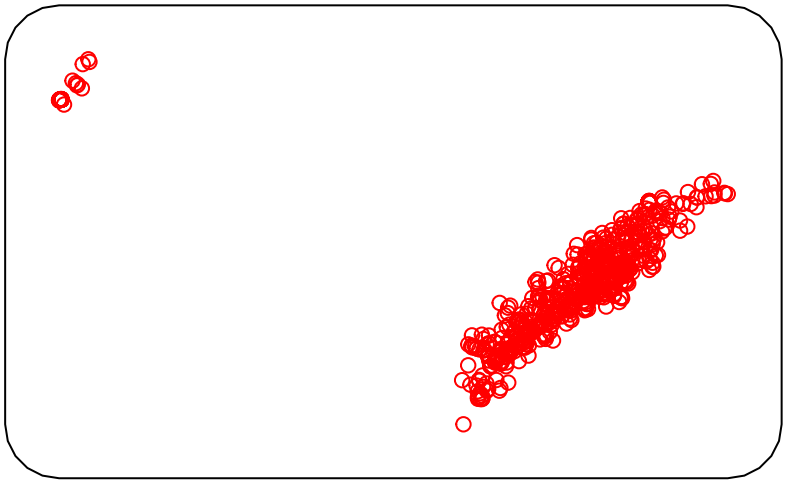
```
#Check to see all our layers are now in Albers projection
proj4string(veg)
proj4string(deer.albers)
proj4string(AlbersSP)
```

```
plot(veg)
points(deer.albers, col="red")
```



11. Then we need to expand the bounding polygon so all locations are included. We can then make the bounding polygon (AlbersSP) a class owin in order to proceed with functions in package spatstat.

```
buffSP <- gBuffer(AlbersSP,width=1000)
plot(buffSP)
points(deer.albers,col="red")
```



12. Code below will be for use with the spatstat package to convert segments of line layers (e.g., roads, rivers) to lines to enable distance to feature from deer locations. Most calculations with spatstat require 3 new classes so most code is created to achieve this goal:

"owin" Observation windows "ppp" Planar point patterns "psp" Planar segment patterns

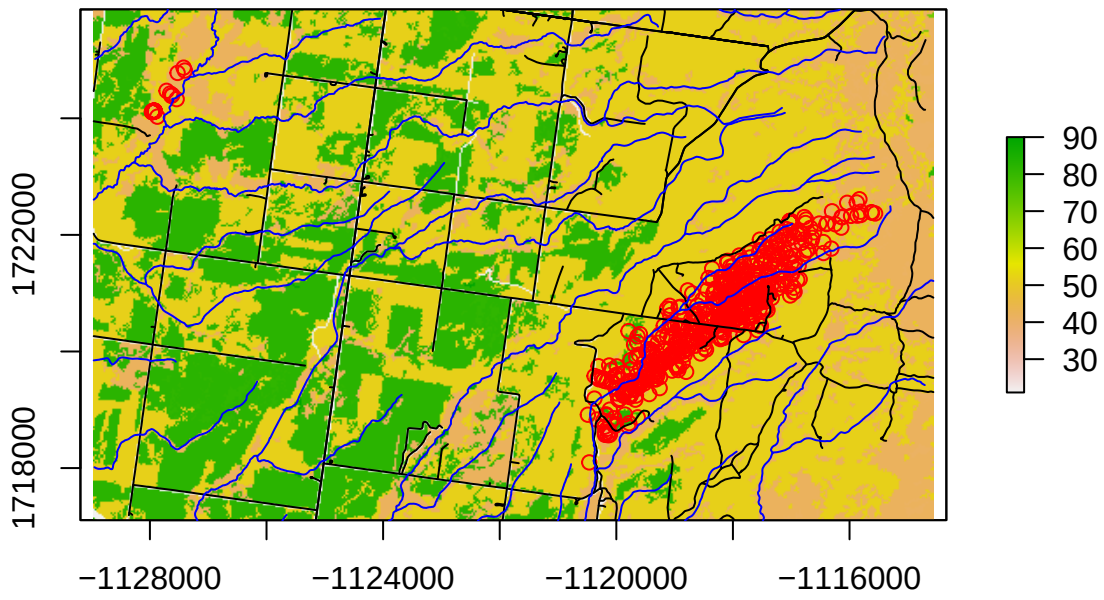
```
#Replace AlbersSP with buffSP
AlbersSPDF <- as(buffSP, "SpatialPolygonsDataFrame")
bdy.owin <- as.owin(AlbersSPDF)
is.owin(bdy.owin)
```

```
## [1] TRUE
```

```
#It is TRUE so now we can move forward with the analysis
```

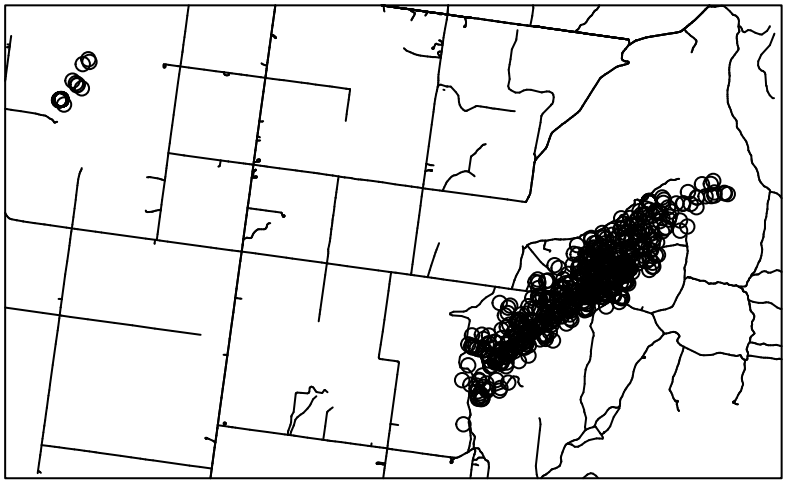
12. Now clip the raster using the buffered bounding box (buffSP) created in step 5.

```
bbclip <- crop(veg, buffSP)
cliproads <- gIntersection(roads, buffSP, byid=TRUE)
cliprivers <- gIntersection(rivers, buffSP, byid=TRUE)

plot(bbclip)
points(deer.albers, col="red")
plot(cliproads, add=T)
plot(cliprivers, col="blue",add=T)
```



13. We will start with the road layer by converting a single line to a set of segments packaged as a function.

```
foo <- function(cliproads){
x <- cliproads@Lines[[1]]@coords
cbind(
head(x,-1),
tail(x,-1))}
#The function can be applied successively to each line in the list we extracted from roads.
#Results are output as a list, then converted to a matrix.

segs.lst <- lapply(cliproads@lines,foo)
segs <- do.call(rbind,segs.lst)

segs.x <- c(segs[,c(1,3)])
segs.y <- c(segs[,c(2,4)])
segs.owin <- as.owin(c(range(segs.x),range(segs.y)))#create a new "owin" class because
#roads occur outside our bdy.owin created above

#The segments as a planar segment pattern:
segs.psp <- as.psp(segs, window=segs.owin)
plot(segs.psp)
points(deer.albers)
```



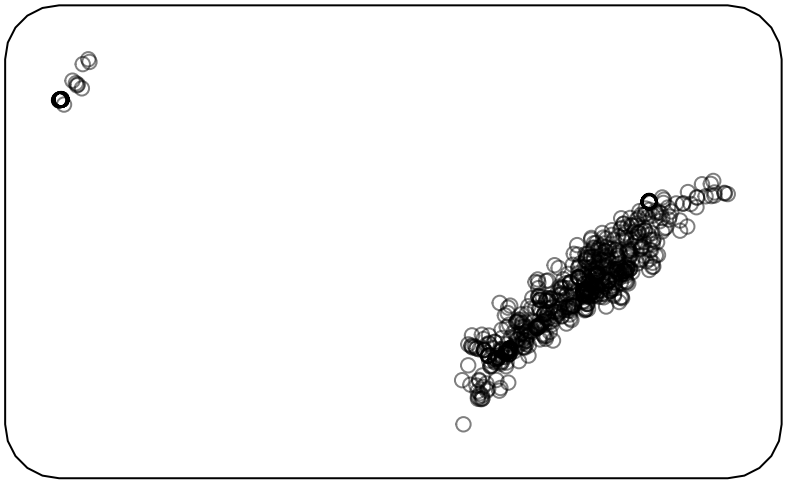

```
segs.psp[1:5]
```

```
## planar line segment pattern: 5 line segments
## window: rectangle = [-1128964, -1114562] x [1717097, 1725868] units
```

```
#lengths.psp(segs.psp[1:10])

#We can cut road segments into distances we control
dist <- pointsOnLines(segs.psp, eps=1000)
```

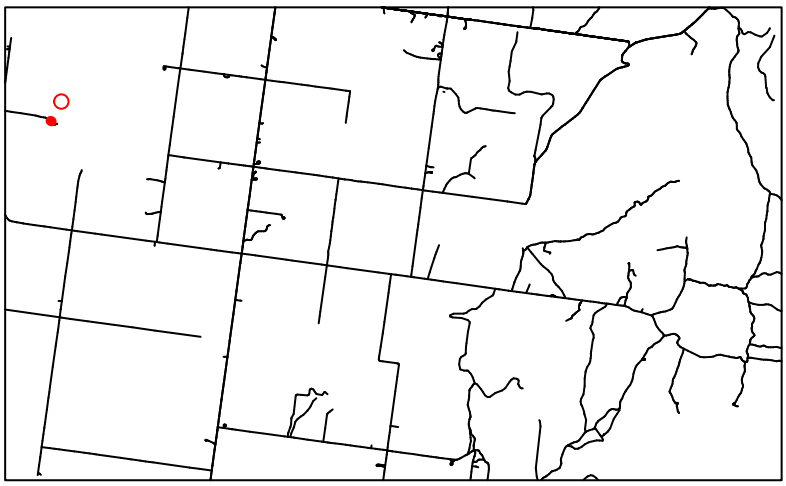
14. We first need to handle the mule deer locations. We need to make mule deer xy coordinates a planar point pattern (i.e., ppp) for use in package spatstat.

```
deer2 <-as.data.frame(deer.albers)
newdeer <-cbind(deer2$x,deer2$y)
xy.ppp <- as.ppp(newdeer,W=bdy.owin)
plot(xy.ppp)
```



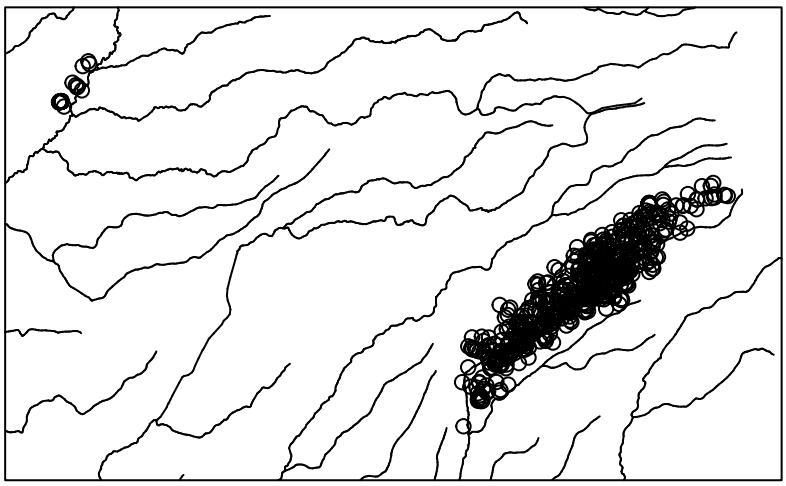
15. Now we can determine the distance from mule deer locations (xy.ppp) to the nearest road

```
roaddist <- nncross(xy.ppp, segs.psp)$dist
#Or identify segment number closest to each point
v <- nearestsegment(xy.ppp,segs.psp)#Identifies segment number not a distance
plot(segs.psp)
plot(xy.ppp[101], add=TRUE, col="red")
plot(segs.psp[v[101]], add=TRUE, lwd=5, col="red")
```



16. Now we do the same to a river layer by converting a single line to a set of segments packaged as a function.

```
foo <- function(cliprivers){
x <- cliprivers@Lines[[1]]@coords
cbind(
head(x,-1),
tail(x,-1))}
#The function can be applied successively to each line in the list we extracted from roads.
#Results are output as a list, then converted to a matrix.
rivs.lst <- lapply(cliprivers@lines,foo)
rivs <- do.call(rbind,rivs.lst)
rivs.x <- c(rivs[,c(1,3)])
rivs.y <- c(rivs[,c(2,4)])
rivs.owin <- as.owin(c(range(rivs.x),range(rivs.y)))

#The segments as a planar segment pattern:
rivs.psp <- as.psp(rivs, window=rivs.owin)
plot(rivs.psp)
points(deer.albers)
```



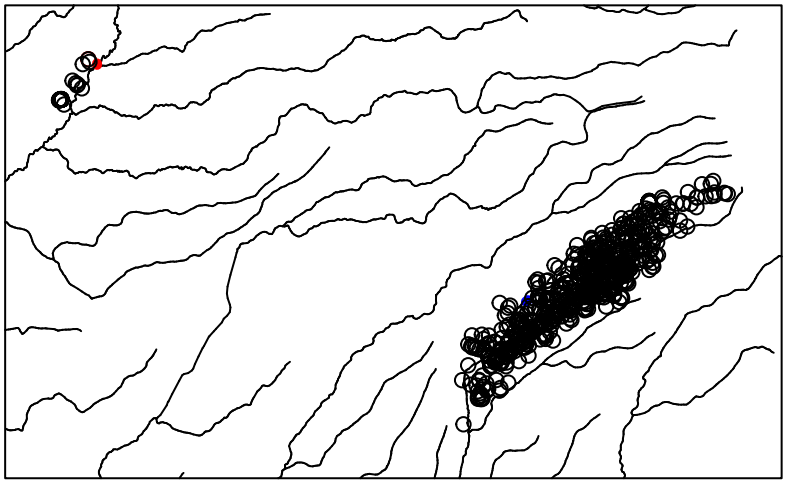

```
is.psp(rivs.psp)
```

```
## [1] TRUE
```

```
#All is TRUE so now we can move forward with the analysis
```

17. Now we can determine the distance from mule deer locations (xy.ppp) to the nearest river.

```
rivdist <- nncross(xy.ppp, rivs.psp)$dist

#Or identify segment number closest to each point
riv <- nearestsegment(xy.ppp,rivs.psp)
plot(rivs.psp, lwd=1)
plot(xy.ppp[1], add=TRUE, col="red")
plot(rivs.psp[riv[1]], add=TRUE, lwd=5, col="red")
plot(xy.ppp[290], add=TRUE, col="blue")
plot(rivs.psp[riv[290]], add=TRUE, lwd=5, col="blue")
points(deer.albers)
```



18. We can then summarize the distances in some meaningful way for analysis. Instead of representing distance to road as individual numerical values we can bin the distances in some categories we determine appropriate for our research objective.

```
br <- seq(0,1000,200)
lbl <- paste(head(br,-1),tail(br,-1),sep="-")
road.tbl <- table(cut(roaddist,breaks=br,labels=lbl))
Rdresults <- road.tbl/sum(road.tbl)
Rdresults
```

```
##
##        0-200      200-400      400-600      600-800     800-1000
## 0.5121951220 0.1866791745 0.2579737336 0.0422138837 0.0009380863
```

```
br1 <- seq(0,4000,500)
lbl1 <- paste(head(br1,-1),tail(br1,-1),sep="-")
river.tbl <- table(cut(rivdist,breaks=br1,labels=lbl1))
Rivresults <- river.tbl/sum(river.tbl)
Rivresults
```

```
##
##      0-500   500-1000  1000-1500  1500-2000  2000-2500  2500-3000  3000-3500
## 0.97942002 0.02057998 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
##  3500-4000
## 0.00000000
```

19. Or we can place each distance into a category or Bin for each deer

```
BinRoad <- bin(roaddist, nbins=5, method='content', labels=c('1','2','3','4','5'))
BinRoad2 <- cut(roaddist, 5, method='intervals', include.lowest=TRUE, labels=c('1','2','3'
  ,'4','5'))
table(BinRoad)
```

```
## BinRoad
##   1   2   3   4   5
## 216 212 214 215 212
```

```
BinRivers <- bin(rivdist, nbins=5, method='content', labels=c('1','2','3','4','5'))
BinRivers <- cut(rivdist, 5, method='intervals', include.lowest=TRUE, labels=c('1','2','3'
    ,'4','5'))
table(BinRivers)
```

```
## BinRivers
##   1   2   3   4   5
## 508 276 153  85  47
```

```
#Now use cbind function to add binned distances to muleys dataset.
Dist <- cbind(BinRoad,BinRivers)
muleys <- cbind(muleys, Dist)
```