Lab 6: Landscape Resistance

Norah Saarman

2024-11-18

Section 13 Exercise

a. Load libraries

```
library(LandGenCourse)
#library(EcoGenetics)
library(GeNetIt)
```

```
## Loading required package: nlme
```

```
library(hierfstat)
library(adegenet)
```

```
## Loading required package: ade4
```

```
##
##    /// adegenet 2.1.10 is loaded ////////////
##
##    > overview: '?adegenet'
##    > tutorials/doc/questions: 'adegenetWeb()'
##    > bug reports/feature requests: adegenetIssues()
```

```
##
## Attaching package: 'adegenet'
```

```
## The following objects are masked from 'package:hierfstat':
##
##     Hs, read.fstat
```

```
require(gstudio)
```

```
## Loading required package: gstudio
```

```
## Warning: replacing previous import 'dplyr::union' by 'raster::union' when
## loading 'gstudio'
```

```
## Warning: replacing previous import 'dplyr::intersect' by 'raster::intersect'
## when loading 'gstudio'
```

```
## Warning: replacing previous import 'dplyr::select' by 'raster::select' when
## loading 'gstudio'
```

```
##
## Attaching package: 'gstudio'
```

```
## The following objects are masked from 'package:adegenet':
##
##     alleles, ploidy
```

```
## The following object is masked from 'package:hierfstat':
##
##     Ho
```

```r
require(dplyr)
```

```
## Loading required package: dplyr

##
## Attaching package: 'dplyr'

## The following object is masked from 'package:nlme':
##
##     collapse

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
require(tibble)
```

```
## Loading required package: tibble
```

```r
require(sf)
```

```
## Loading required package: sf

## Linking to GEOS 3.10.2, GDAL 3.4.1, PROJ 8.2.1; sf_use_s2() is TRUE
```

```r
require(popgraph)
```

```
## Loading required package: popgraph
```

```r
require(RgoogleMaps)
```

```
## Loading required package: RgoogleMaps

##
## Thank you for using RgoogleMaps!

##
## To acknowledge our work, please cite the package:

##  Markus Loecher and Karl Ropkins (2015). RgoogleMaps and loa: Unleashing R
##   Graphics Power on Map Tiles. Journal of Statistical Software 63(4), 1-18.
```

```r
require(geosphere)
```

```
## Loading required package: geosphere
```

```r
require(proto)
```

```
## Loading required package: proto
```

```r
require(sampling)
```

```
## Loading required package: sampling

##
## Attaching package: 'sampling'
```

```
## The following object is masked from 'package:adegenet':
##
##     strata
```

```r
require(seqinr)
```

```
## Loading required package: seqinr
##
## Attaching package: 'seqinr'
## The following object is masked from 'package:dplyr':
##
##     count
## The following object is masked from 'package:nlme':
##
##     gls
```

```r
require(spacetime)
```

```
## Loading required package: spacetime
```

```r
require(spdep)
```

```
## Loading required package: spdep
## Loading required package: spData
## To access larger datasets in this package, install the spDataLarge
## package with: `install.packages('spDataLarge',
## repos='https://nowosad.github.io/drat/', type='source')`
## Registered S3 method overwritten by 'spdep':
##   method   from
##   plot.mst ape
##
## Attaching package: 'spdep'
## The following object is masked from 'package:ade4':
##
##     mstree
```

```r
require(here)
```

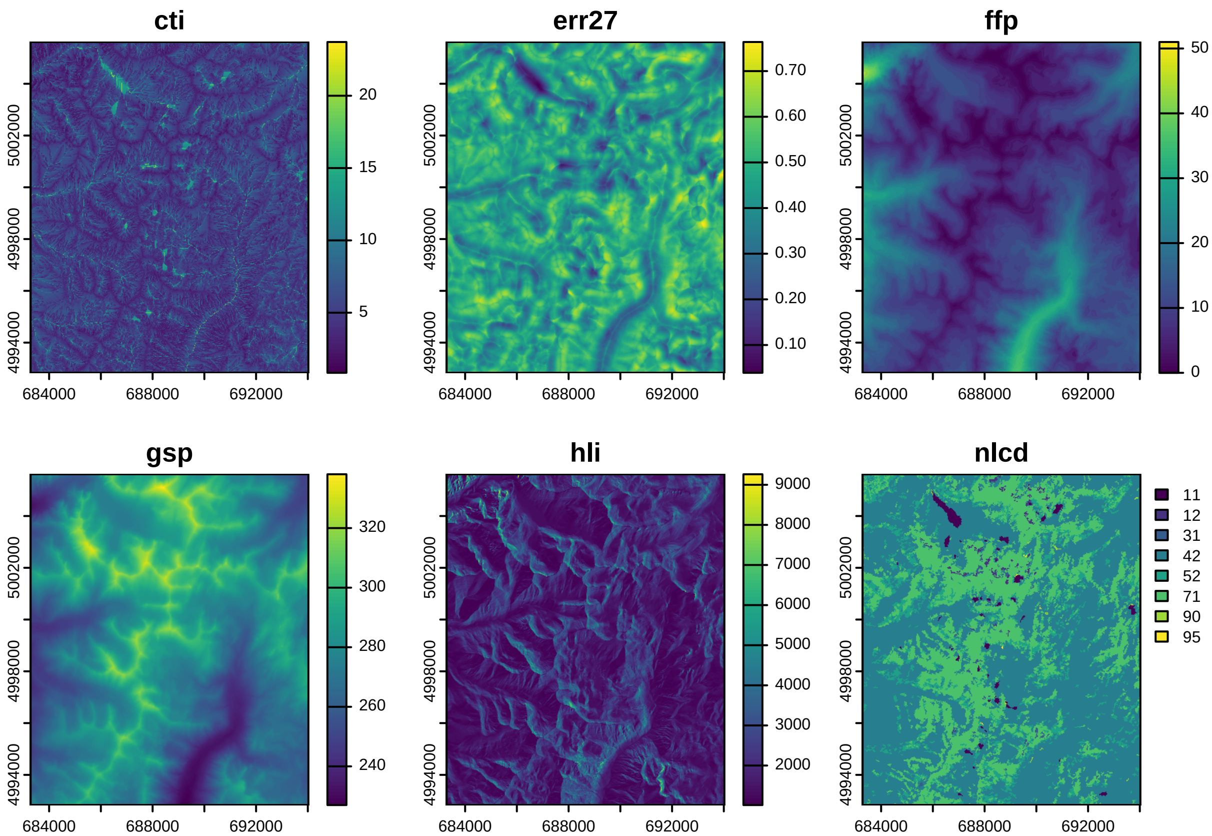
```
## Loading required package: here
## here() starts at /uufs/chpc.utah.edu/common/home/u6036559/git/usu-biol4750
```

```r
require(terra)
```

```
## Loading required package: terra
## terra 1.7.83
##
## Attaching package: 'terra'
## The following object is masked from 'package:seqinr':
##
##     query
```

d. Import rasters

```r
library(LandGenCourse)
library(GeNetIt)

RasterMaps <- terra::rast(system.file("extdata/covariates.tif", package="GeNetIt"))

plot(RasterMaps)
```



e. sites info (GPS coordinates for the study)

```r
data(ralu.site, package="GeNetIt")
sites <- ralu.site
str(sites)
```

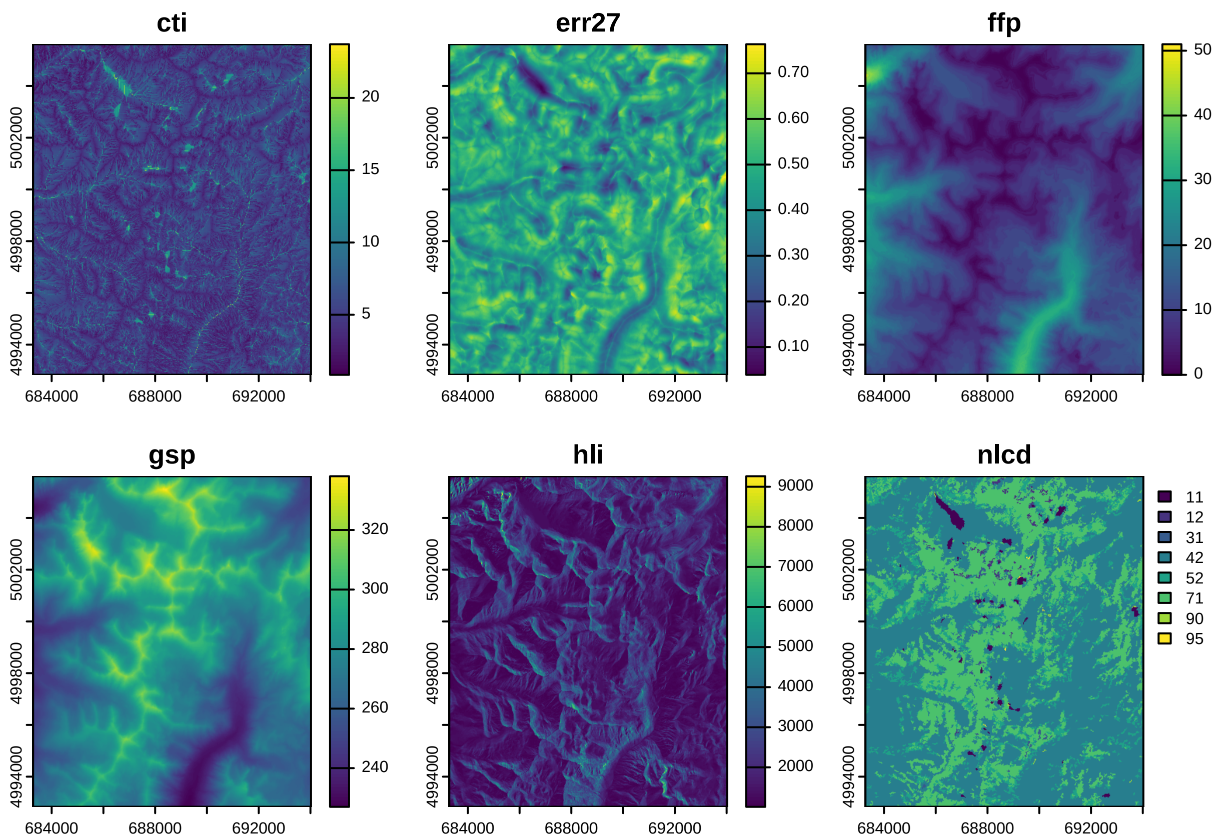
```
## Classes 'sf' and 'data.frame':   31 obs. of  18 variables:
##  $ SiteName : chr  "AirplaneLake" "BachelorMeadow" "BarkingFoxLake" "BirdbillLake" ...
##  $ Drainage : chr  "ShipIslandCreek" "WilsonCreek" "WaterfallCreek" "ClearCreek" ...
##  $ Basin    : chr  "Sheepeater" "Skyhigh" "Terrace" "Birdbill" ...
##  $ Substrate: chr  "Silt" "Silt" "Silt" "Sand" ...
##  $ NWI      : chr  "Lacustrine" "Riverine_Intermittent_Streambed" "Lacustrine" "Lacustrine" ...
##  $ AREA_m2  : num  62582 225 12000 12359 4600 ...
##  $ PERI_m   : num  1143 60 435 572 321 ...
##  $ Depth_m  : num  21.64 0.4 5 3.93 2 ...
##  $ TDS      : num  2.5 0 13.8 6.4 14.3 10.9 10 2.4 0 3.6 ...
##  $ FISH     : int  1 0 1 1 0 0 1 0 0 0 ...
##  $ ACB      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ AUC      : num  0.411 0 0.3 0.283 0 0 0.415 0 0 0 ...
##  $ AUCV     : num  0 0 0 0 0 0 0.171 0.047 0 0 ...
```

```
##  $ AUCC     : num  0.411 0 0.3 0.283 0 0 0.585 0.047 0 0 ...
##  $ AUF      : num  0.063 1 0.7 0.717 0.5 0.556 0.341 0.686 0 1 ...
##  $ AWOOD    : num  0.063 0 0 0 0 0.093 0 0.209 0 0 ...
##  $ AUFV     : num  0.464 0 0 0 0.5 0.352 0.073 0.058 0 0 ...
##  $ geometry :sfc_POINT of length 31; first list element:  'XY' num  688817 5003207
##  - attr(*, "sf_column")= chr "geometry"
##  - attr(*, "agr")= Factor w/ 3 levels "constant","aggregate",..: NA NA NA NA NA NA NA NA NA NA ...
##   ..- attr(*, "names")= chr [1:17] "SiteName" "Drainage" "Basin" "Substrate" ...
```

2. Explore the data set… plot with terra::plot() function

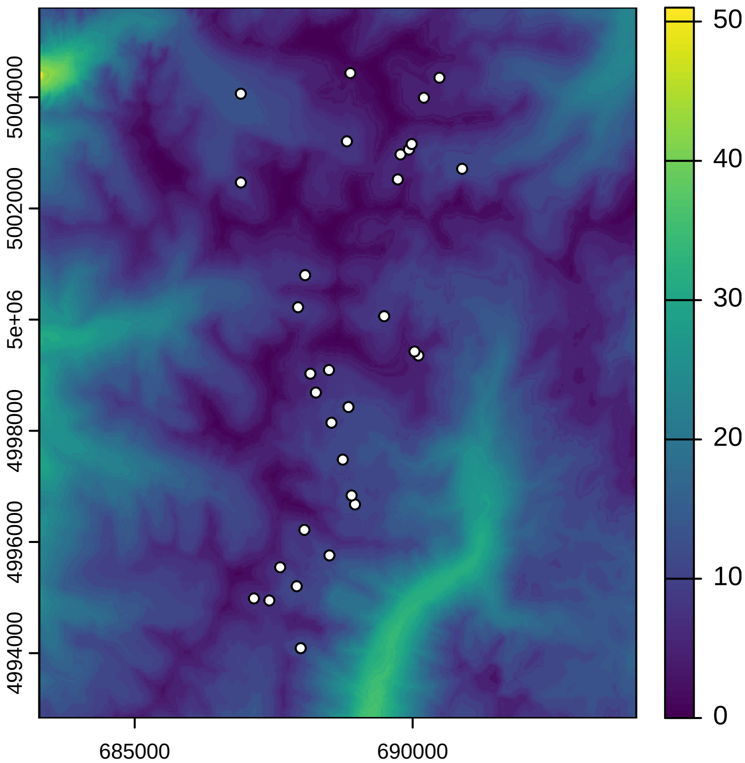
a. Plot all rasters

```
terra::plot(RasterMaps)
```



b. Plot spatial points over 'ffp' raster

```
par(mar=c(2,2,1,1))
terra::plot(RasterMaps[["ffp"]])
terra::points(sites, pch=21, col="black", bg="white")
```



3. Setting cost values and calculating conductance

a. Resistance vs conductance values

b. scale rasters

```
#cti <- terra::resample(cti, gsp, method= "bilinear")
```

c. Calculate conductance values

```
RasterMaps[["err27"]]
```

```
## class       : SpatRaster
## dimensions  : 426, 358, 1  (nrow, ncol, nlyr)
## resolution  : 30, 30  (x, y)
## extent      : 683282.5, 694022.5, 4992833, 5005613  (xmin, xmax, ymin, ymax)
## coord. ref. : NAD83 / UTM zone 11N (EPSG:26911)
## source      : covariates.tif
## name        :      err27
## min value   : 0.03906551
## max value   : 0.76376426
```

```
err.cost <- (1/RasterMaps[["err27"]])
err.cost
```

```
## class       : SpatRaster
## dimensions  : 426, 358, 1  (nrow, ncol, nlyr)
## resolution  : 30, 30  (x, y)
## extent      : 683282.5, 694022.5, 4992833, 5005613  (xmin, xmax, ymin, ymax)
## coord. ref. : NAD83 / UTM zone 11N (EPSG:26911)
## source(s)   : memory
## varname     : covariates
## name        :     err27
```

```
## min value   :  1.309305
## max value   : 25.598027
```

```r
RasterMaps[["ffp"]]
```

```
## class       : SpatRaster
## dimensions  : 426, 358, 1  (nrow, ncol, nlyr)
## resolution  : 30, 30  (x, y)
## extent      : 683282.5, 694022.5, 4992833, 5005613  (xmin, xmax, ymin, ymax)
## coord. ref. : NAD83 / UTM zone 11N (EPSG:26911)
## source      : covariates.tif
## name        : ffp
## min value   :   0
## max value   :  51
```

```r
ffp.cost <- (RasterMaps[["ffp"]]/5)
ffp.cost
```

```
## class       : SpatRaster
## dimensions  : 426, 358, 1  (nrow, ncol, nlyr)
## resolution  : 30, 30  (x, y)
## extent      : 683282.5, 694022.5, 4992833, 5005613  (xmin, xmax, ymin, ymax)
## coord. ref. : NAD83 / UTM zone 11N (EPSG:26911)
## source(s)   : memory
## varname     : covariates
## name        :  ffp
## min value   :  0.0
## max value   : 10.2
```

```r
RasterMaps[["gsp"]]
```

```
## class       : SpatRaster
## dimensions  : 426, 358, 1  (nrow, ncol, nlyr)
## resolution  : 30, 30  (x, y)
## extent      : 683282.5, 694022.5, 4992833, 5005613  (xmin, xmax, ymin, ymax)
## coord. ref. : NAD83 / UTM zone 11N (EPSG:26911)
## source      : covariates.tif
## name        :      gsp
## min value   : 227.0000
## max value   : 338.0697
```

```r
gsp.cost <- (RasterMaps[["gsp"]]-196)/15
gsp.cost
```

```
## class       : SpatRaster
## dimensions  : 426, 358, 1  (nrow, ncol, nlyr)
## resolution  : 30, 30  (x, y)
## extent      : 683282.5, 694022.5, 4992833, 5005613  (xmin, xmax, ymin, ymax)
## coord. ref. : NAD83 / UTM zone 11N (EPSG:26911)
## source(s)   : memory
## varname     : covariates
## name        :      gsp
## min value   : 2.066667
## max value   : 9.471311
```

```r
RasterMaps[["cti"]]
```

```
## class       : SpatRaster
```

```
## dimensions  : 426, 358, 1  (nrow, ncol, nlyr)
## resolution  : 30, 30  (x, y)
## extent      : 683282.5, 694022.5, 4992833, 5005613  (xmin, xmax, ymin, ymax)
## coord. ref. : NAD83 / UTM zone 11N (EPSG:26911)
## source      : covariates.tif
## name        :        cti
## min value   :  0.8429851
## max value   : 23.7147598
```

```r
cti.cost <- RasterMaps[["cti"]]/5
cti.cost
```

```
## class       : SpatRaster
## dimensions  : 426, 358, 1  (nrow, ncol, nlyr)
## resolution  : 30, 30  (x, y)
## extent      : 683282.5, 694022.5, 4992833, 5005613  (xmin, xmax, ymin, ymax)
## coord. ref. : NAD83 / UTM zone 11N (EPSG:26911)
## source(s)   : memory
## varname     : covariates
## name        :      cti
## min value   : 0.168597
## max value   : 4.742952
```

d. Create a single landscape conductance raster

```r
cost1 <- (gsp.cost + cti.cost + err.cost + ffp.cost)
cost1
```

```
## class       : SpatRaster
## dimensions  : 426, 358, 1  (nrow, ncol, nlyr)
## resolution  : 30, 30  (x, y)
## extent      : 683282.5, 694022.5, 4992833, 5005613  (xmin, xmax, ymin, ymax)
## coord. ref. : NAD83 / UTM zone 11N (EPSG:26911)
## source(s)   : memory
## varname     : covariates
## name        :       gsp
## min value   :  9.012874
## max value   : 35.900946
```

4. Convert conductance into effective distance The higher the conductance, the lower the cost or resistance of a cell, and vice versa. We want to integrate conductance across cells to derive some measure of effective (or ecological) distance.

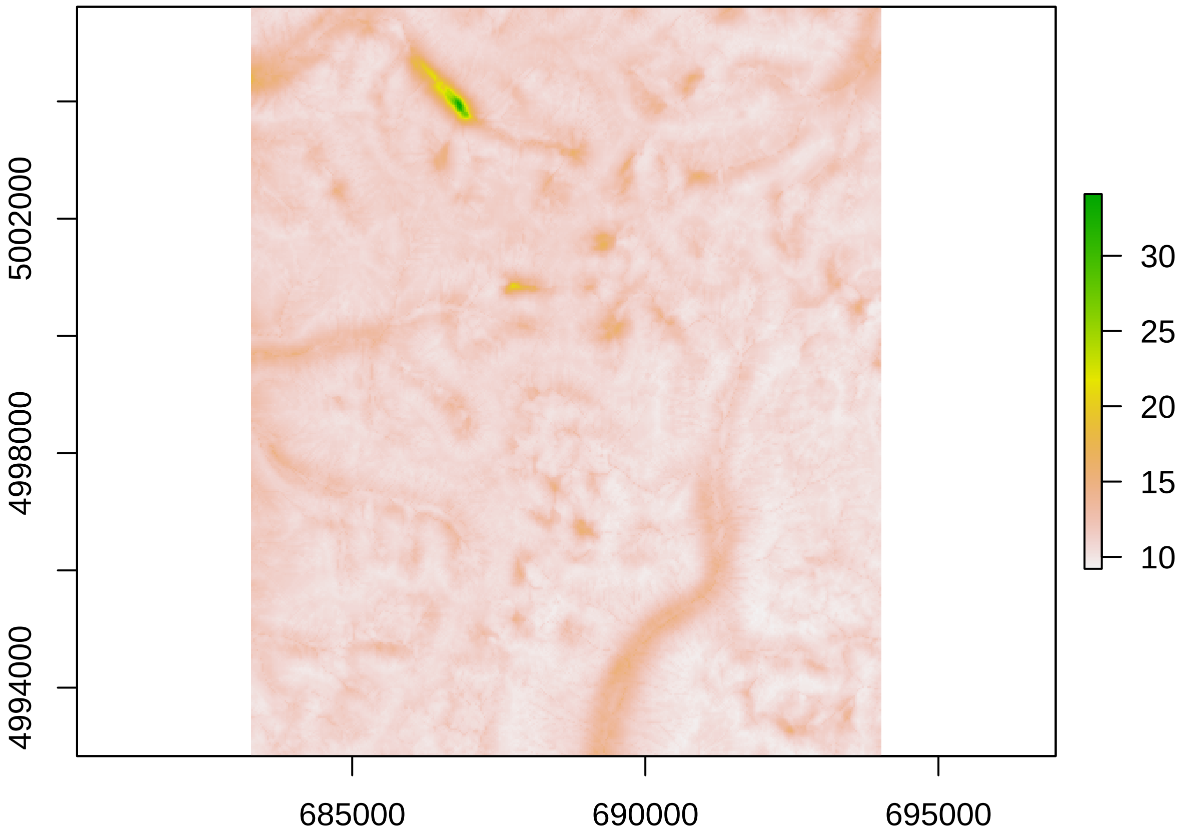
a. transition layer

```r
tr.cost1 <- gdistance::transition(raster::raster(cost1), transitionFunction=mean, directions=8)
tr.cost1
```

```
## class      : TransitionLayer
## dimensions : 426, 358, 152508  (nrow, ncol, ncell)
## resolution : 30, 30  (x, y)
## extent     : 683282.5, 694022.5, 4992833, 5005613  (xmin, xmax, ymin, ymax)
## crs        : +proj=utm +zone=11 +datum=NAD83 +units=m +no_defs
## values      : conductance
## matrix class: dsCMatrix
```

b. visually inspect the raster

```r
par(mar=c(2,2,1,1))
raster::plot(raster::raster(tr.cost1))
```

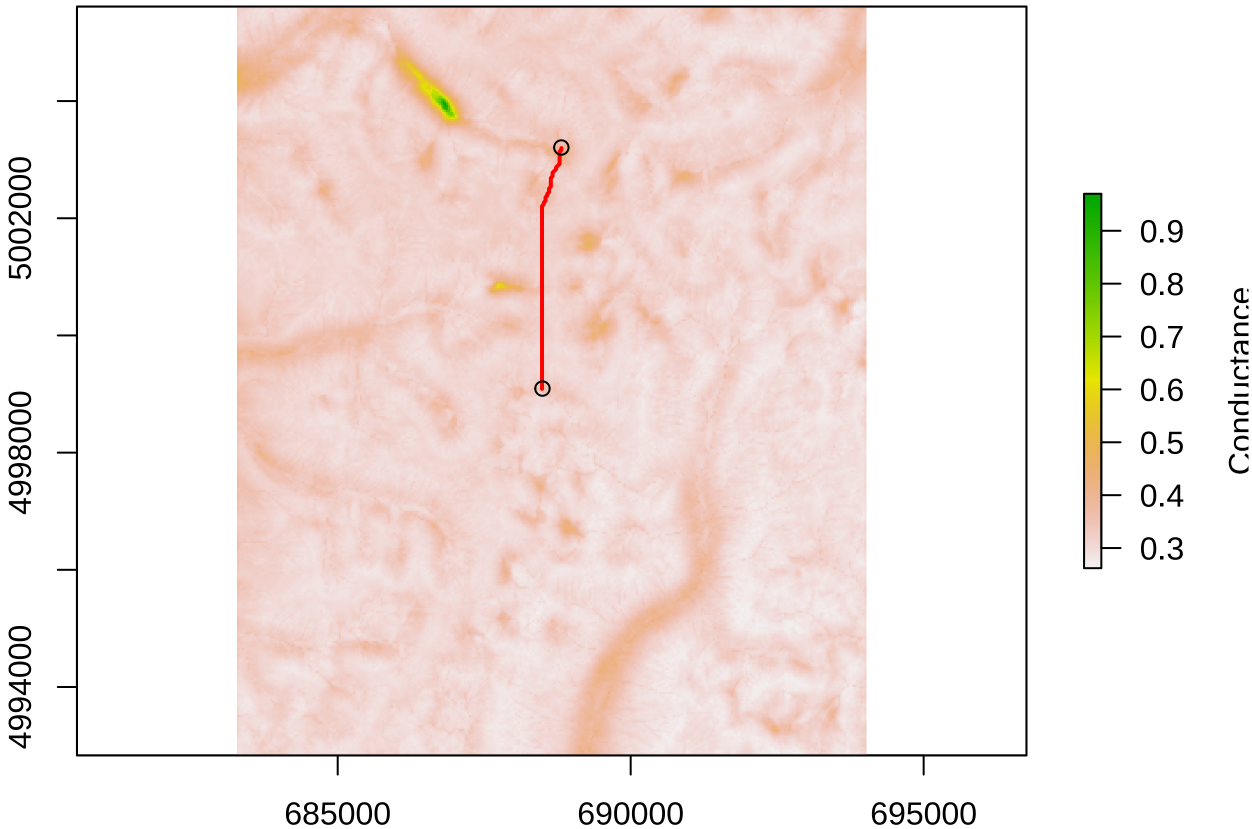


c. Correct for geometric distortion

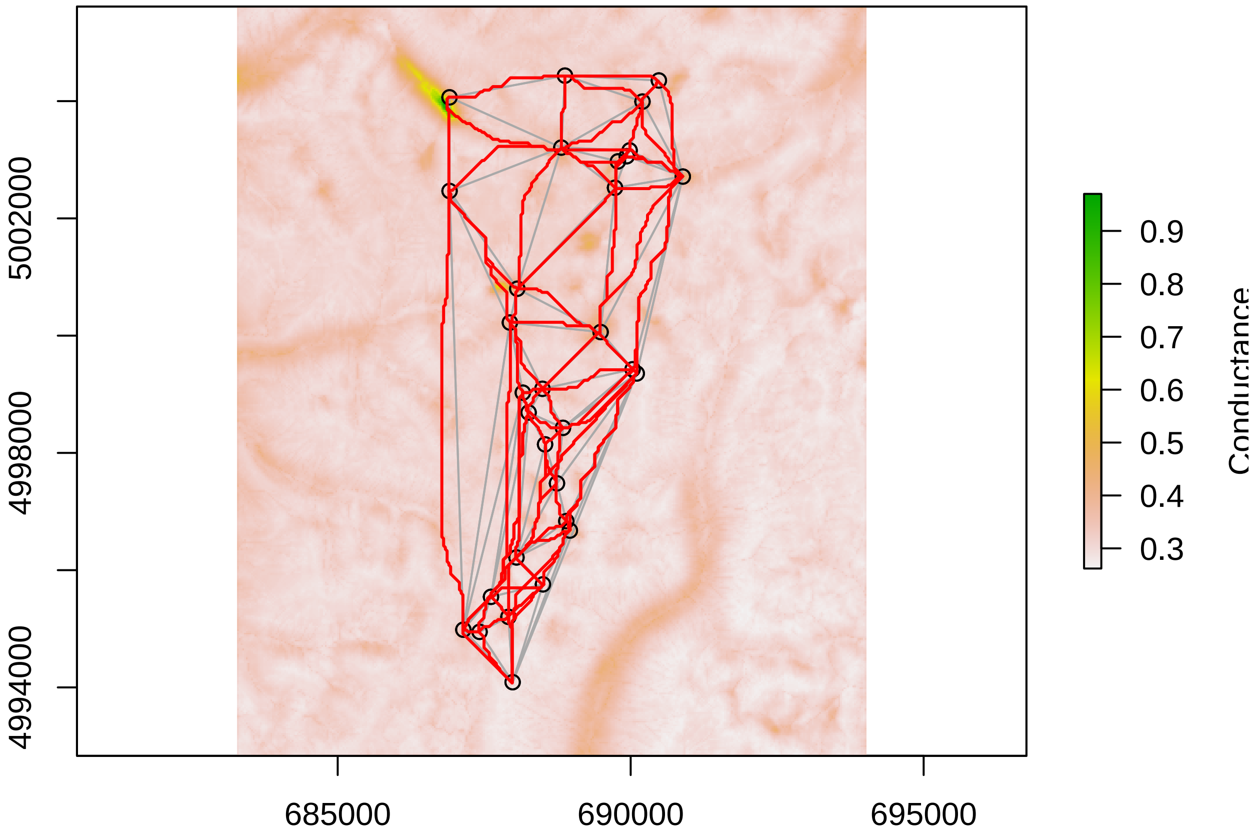
```r
tr.cost1 <- gdistance::geoCorrection(tr.cost1,type = "c",multpl=FALSE)
```

d. plot shortest paths

```r
sites.sp <- sf::as_Spatial(sites)

par(mar=c(2,2,1,2))
AtoB <- gdistance::shortestPath(tr.cost1, origin=sites.sp[1,],
                                goal=sites.sp[2,], output="SpatialLines")
raster::plot(raster::raster(tr.cost1), xlab="x coordinate (m)",
             ylab="y coordinate (m)",legend.lab="Conductance")
lines(AtoB, col="red", lwd=2)
points(sites.sp[1:2,])
```




```
par(mar=c(2,2,1,2))
raster::plot(raster::raster(tr.cost1), xlab="x coordinate (m)",
             ylab="y coordinate (m)", legend.lab="Conductance")
points(sites.sp)

Neighbours <- spdep::tri2nb(sites.sp@coords, row.names = sites.sp$SiteName)

plot(Neighbours, sites.sp@coords, col="darkgrey", add=TRUE)
for(i in 1:length(Neighbours))
{
  for(j in Neighbours[[i]][Neighbours[[i]] > i])
  {
    AtoB <- gdistance::shortestPath(tr.cost1, origin=sites.sp[i,],
                                    goal=sites.sp[j,], output="SpatialLines")
    lines(AtoB, col="red", lwd=1.5)
  }
}
```



5. Create cost-distance matrices a. Least cost distance

```
cost1.dist <- gdistance::costDistance(tr.cost1,sites.sp)
```

b. Cost-distance matrix based on random paths (similar to Circuitscape)

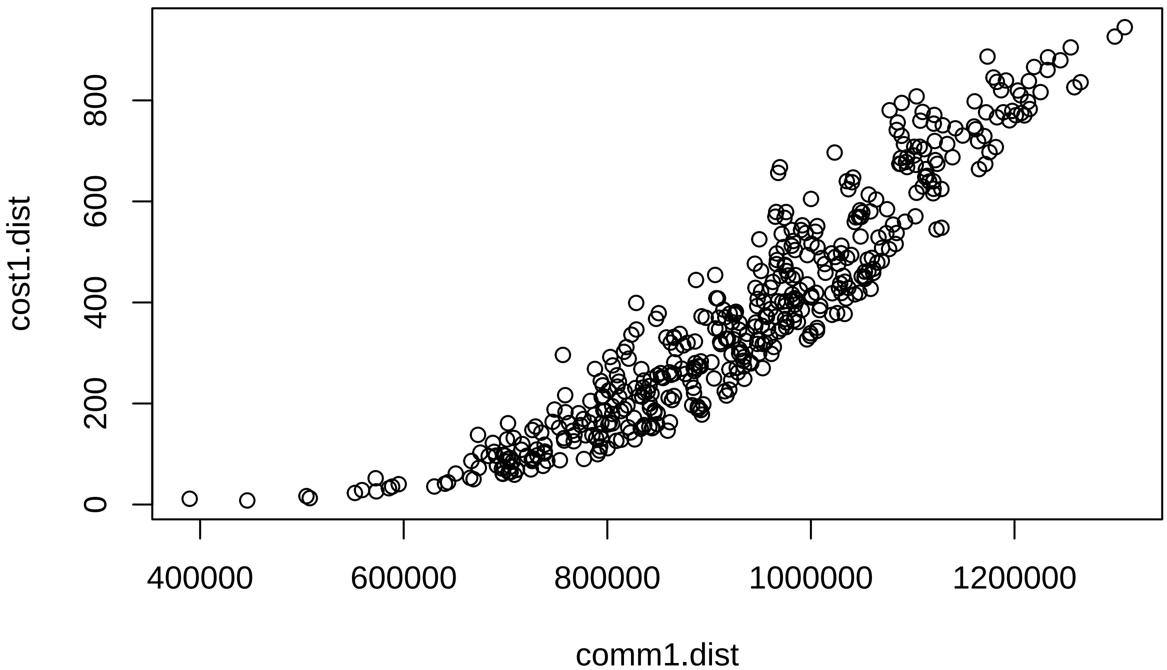
```
comm1.dist <- gdistance::commuteDistance(x = tr.cost1, coords = sites.sp)
```

c. Compare cost distances

```
dist_df <- data.frame("cost1.dist"=as.numeric(cost1.dist),
                      "comm1.dist"=as.numeric(comm1.dist))

corr.LCD.comm <- cor(dist_df$cost1.dist, dist_df$comm1.dist, method = "spearman")
corr.LCD.comm
```

```
## [1] 0.9519704
```

```
plot(cost1.dist~comm1.dist)
```



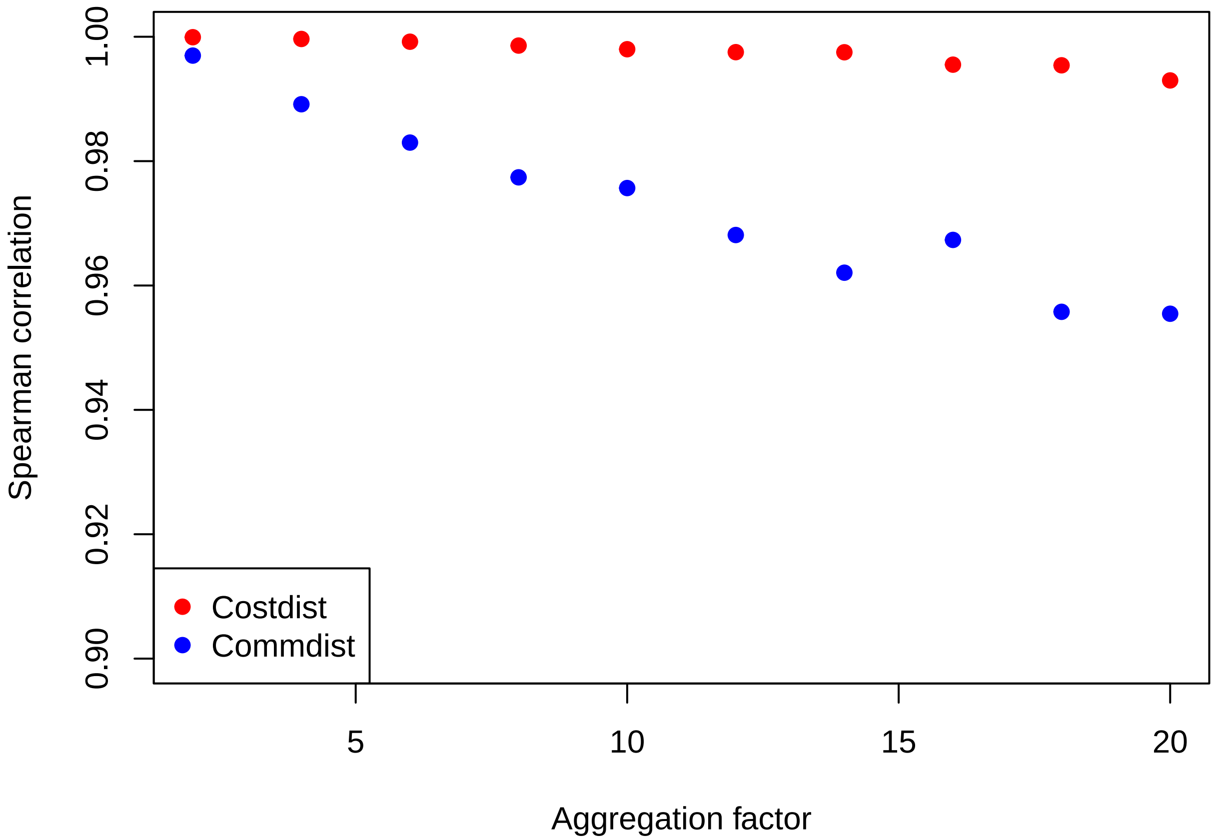
6. How does changing resolution affect these metrics?

a. Create a loop

```
cor_cost <- c()
cor_comm <- c()
res_fact <- seq(2,20,2)
for(fac in res_fact){
  cost1_agg <- raster::aggregate(raster::raster(cost1), fact = fac)
  tr.cost_agg <- gdistance::transition(cost1_agg,
                   transitionFunction=mean, directions=8)
  tr.cost_agg <- gdistance::geoCorrection(tr.cost_agg,type = "c",multpl=FALSE)
  cost.dist_agg <- gdistance::costDistance(tr.cost_agg, sites.sp)
  comm.dist_agg <- gdistance::commuteDistance(x = tr.cost_agg, coords = sites.sp)
  cost.dist_agg <- as.numeric(cost.dist_agg)
  comm.dist_agg <- as.numeric(comm.dist_agg)
  cor_cost <- c(cor_cost,cor(dist_df$cost1.dist, cost.dist_agg,
                            method = "spearman"))
  cor_comm <- c(cor_comm,cor(dist_df$comm1.dist, comm.dist_agg,
                            method = "spearman"))
}
```

b. Plot the results

```
par(mar=c(4,4,1,1))
plot(y = cor_cost, x = res_fact, col = "red", pch = 19,
     ylim = c(0.9,1), xlab = "Aggregation factor", ylab = "Spearman correlation")
points(y = cor_comm, x = res_fact, col = "blue", pch = 19)
legend("bottomleft", legend = c("Costdist","Commdist"),
       pch = 19, col = c("red", "blue"))
```



Spearman correlation
1.00
0.98
0.96
0.94
0.92
0.90
5
10
15
20
Aggregation factor
Costdist
Commdist