Cakile edentula Range Dispersal

Kwiz Research Services

2022-05-10

library(RangeShiftR)  
require(sp)  
require(rgdal)  
require(sfheaders)  
require(raster)  
require(SDMSelect)  
require(tidyverse)

# Introduction

The goal of the analysis is to simulate dispersal of cakile edentula following introductions at the ports of “Sydney”, “Melbourne”, “Brisbane” and “Perth”.

Steps taken:

1. Using the BCCVL platform, and “occurance data” for cakile edentula in its native and observed habitats (globally) and the climate data “CRUclim (global), current climate (1976-2005), 30 arcmin (~50 km)” as input, predicted a species distribution map for the suitability of cakile edentula. The output is a CSV file with the suitability probabilities (0 - 1) for different points.
2. Transformed the data on suitability from BCCVL into a landscape map file *climate\_suitabilitya*. Extract the probabilities with Australia only i.e

* extent(xmin = 110, xmax = 155, ymin = -45, ymax = -9)

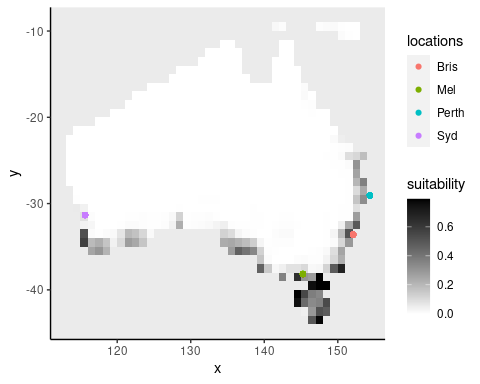
#### suitability map from csv  
AllData <- read.csv("Inputs/Unknown.species\_AllData\_Full.csv")  
colnames(AllData)[1:2] = c("lon", "lat")  
coordinates(AllData) = ~ lon + lat  
proj4string(AllData)=CRS("+init=EPSG:4326") # set it to lat-long  
AllData = spTransform(AllData,CRS("+init=EPSG:4326"))  
e <- extent(110, 155, -45, -9)  
r <- raster(e, ncols = 1000, nrows = 1000)  
# you need to provide a function 'fun' for when there are multiple points per cell  
AllData\_a <- rasterize(AllData, r, fun = mean)  
AllData2a = raster::resample(AllData\_a[[2]], raster(ext=extent(extent(AllData\_a)), resolution= 1,crs=projection(AllData)))  
raster::writeRaster(AllData2a, format="ascii", filename = "Inputs/climate\_suitabilitya", NAflag = -9, overwrite = T, bylayer = T, datatype = "FLT4S")

1. We simulate a species distribution map such that we have 100 individuals each in 4 locations i.e the ports of “Sydney”, “Melbourne”, “Brisbane” and “Perth”. Save this data into a species distribution map file *news\_locations.asc*

introduction = read.csv("Inputs/aus\_ports3.tsv", sep = "\t")[, 3:5] |> rename(lon = X, lat = Y, Individuals = NInds)  
introduction$Individuals = as.numeric(introduction$Individuals)  
a3 = introduction %>% filter(!is.na(Individuals)) %>% mutate("locations" = rep(c("Syd", "Mel", "Bris", "Perth"), 100)) %>% na.omit()  
coordinates(introduction) = ~ lon + lat  
proj4string(introduction)=proj4string(AllData) # set it to lat-long  
introduction = spTransform(introduction,proj4string(AllData))  
e <- extent(110, 155, -45, -9)  
r <- raster(e, ncols = 1000, nrows = 1000)  
# you need to provide a function 'fun' for when there are multiple points per cell  
introduction2 <- rasterize(introduction, r, fun = mean)  
introduction3 = raster::resample(introduction2[[2]], raster(ext=extent(extent(introduction2)), resolution= 1,crs=projection(introduction)))  
raster::writeRaster(introduction3, format="ascii", filename = "Inputs/news\_locations.asc", NAflag = -9, overwrite = T, datatype = "INT2U")

Visualization of the survival probabilities and the introduction points for cakile edentula.

lp = raster("Inputs/climate\_suitabilitya.asc") |> rasterToPoints() |> as.data.frame() |> rename(value = climate\_suitabilitya) |> filter(!is.na(value))  
ggplot( ) +   
 geom\_tile(data = lp , aes(x = x, y = y, fill = value)) +   
 scale\_fill\_gradient("suitability", low = "white", high = "black") +  
 theme\_grey() + theme(panel.border = element\_blank(), panel.grid.major = element\_blank(),  
panel.grid.minor = element\_blank(), axis.line = element\_line(colour = "black")) +   
 geom\_point(data = a3, aes(x = lon, y = lat, color = locations))



1. Define the *landscape parameter* in RangeshiftR for simulation with the climate map and the species distribution and the parameters:
   * resolution = .
   * HabPercent = TRUE: since the data in our suitability map is a probability value between 0 and 1.
   * K\_or\_DensDep = 10000: to infer that a location with a probability of 1 would have a carrying capacity of 10000. The carrying capacities for all locations are calculated based on the probability value and the 100 threshold.

landscapes = ImportedLandscape(LandscapeFile = "climate\_suitabilitya.asc",   
 Resolution = 1,   
 HabPercent = TRUE,   
 K\_or\_DensDep = 10000,   
 SpDistFile = "news\_locations.asc",   
 SpDistResolution = 1)

1. We define the Species parameters which include:

* *demography*: Assumptions include:
  + the population dynamics are described by a female-only model
  + generations are non-overlapping
  + the maximum growth rate = 1000 : mean number of offspring per female and reproductive event at very low density. (AM Payne, 1984: “Reproduction ans survivorship of cakile edentula var.lacutris along the Lake Huron shoreline” suggests that “The largest number of of upper fruit segments on a mature plant was 1723 or a potential 3446 seeds”)
  + bc = 0.000001 : competition coefficient (under-compensatory)

demos <- Demography(Rmax = 1000, bc = 0.000001, ReproductionType = 0)  
demos

## Demography:  
## Unstructured population:  
## Rmax : 1000   
## bc : 1e-06   
## Reproduction Type : 0 (female only)

* *dispersal*: This contains information on *Emigration* probability, *Transport* modeled with a dispersal kernel and *Settlement* whose default options assume that an individual will die if it arrives in an unsuitable cell and settle if it’s suitable. The assumption:
  + Emigration probability = 0.1
  + *short range* and *long range transfer* at with and with respectively (You provided this values). This suggests that we have a 3rd possible dispersal distance equal to which has a probability of . However, rangeshiftR provides for either a single or double kernel to define distance. To fit this, I considered the odds between the short dispersal distance (20km) and the long distance (100km) and used this to recalculate their probabilities assuming the 3rd option of is not present.

*Settlement* is defined with the parameters:

- MaxSteps = 10: Maximum number of steps before mortality  
- Settle = 2: if cell is not suitable, individual settles in one of the suitable cells around the cell it landed (relaxed settlement condition)

dist = matrix(c(20000, 100000, 0.83 ), ncol = 3, byrow = T)  
disps <- Dispersal(Emigration = Emigration(EmigProb = 0.1), Transfer = DispersalKernel(DoubleKernel = T, Distances = dist)) + Settlement(MaxSteps = 0, Settle = 2)  
disps

## Dispersal:   
## Emigration:  
## Emigration probabilities:  
## [,1]  
## [1,] 0.1  
##   
## Transfer:  
## Dispersal Kernel  
## DoubleKernel = TRUE   
## Dispersal kernel traits:  
## [,1] [,2] [,3]  
## [1,] 20000 1e+05 0.83  
## Constant mortality prob = 0   
##   
## Settlement:  
## Settlement conditions:  
## [,1]  
## [1,] 2  
## FindMate = FALSE

1. Initialize the parameters for simulation

inits <- Initialise(InitType = 1, # initialisation from a loaded species distribution map  
 SpType = 0,# all suitable cells within all distribution presence cells  
 InitDens = 2,  
 IndsHaCell = 100) # = at carrying capacity  
inits

## Initialisation:   
## InitType = 1 : Initialisation from loaded species distribution map  
## all presence cells/patches.  
## InitDens = 2 : 100 individuals per cell/hectare

1. Simulation parameters

We will run with simulating the spread of cakile edentula over and record the *population*, *occurance* and *range* every

sims <- Simulation(Simulation = 1,  
 Years = 1000,  
 Replicates = 10,  
 OutIntRange = 1,  
 OutIntPop = 1,  
 OutIntOcc = 1)  
sims

## Simulation # 1   
## -----------------  
## Replicates = 10   
## Years = 1000   
## Absorbing = FALSE   
## File Outputs:  
## Range, every 1 years  
## Occupancy, every 1 years  
## Populations, every 1 years, starting year 0

1. Combine all the predefined parameters into a master object and run the simulation. We will run two separate simulations, one for the short range dispersal and the other for the long range dispersal.

simulate <- RSsim(seed = 13976) + landscapes + demos + disps + sims + inits  
validateRSparams(simulate) ## check parameter validity

## [1] TRUE

1. Run simulations

set.seed(13976)  
RunRS(RSparams = simulate, dirpath = paste0(getwd(), "/"))

## Checking Control parameters   
##   
## Control Parameters checked  
##   
## Run Simulation(s) with seed 13976 ...  
## LandscapeFile headers OK: /cloud/project/Range dispersal simulation/Inputs/climate\_suitabilitya.asc  
## Species Distribution map headers OK: /cloud/project/Range dispersal simulation/Inputs/news\_locations.asc  
##   
## Running simulation nr. 1  
##   
## starting replicate 0  
## RunModel(): completed initialisation   
## starting year 0...  
## starting year 1...  
## starting year 2...  
## starting year 3...  
## starting year 10...  
## starting year 20...  
## starting year 30...  
## starting year 100...  
## starting year 200...  
## starting year 300...  
##   
## starting replicate 1  
## RunModel(): completed initialisation   
## starting year 0...  
## starting year 1...  
## starting year 2...  
## starting year 3...  
## starting year 10...  
## starting year 20...  
## starting year 30...  
## starting year 100...  
## starting year 200...  
##   
## starting replicate 2  
## RunModel(): completed initialisation   
## starting year 0...  
## starting year 1...  
## starting year 2...  
## starting year 3...  
## starting year 10...  
## starting year 20...  
## starting year 30...  
## starting year 100...  
## starting year 200...  
## starting year 300...  
##   
## starting replicate 3  
## RunModel(): completed initialisation   
## starting year 0...  
## starting year 1...  
## starting year 2...  
## starting year 3...  
## starting year 10...  
## starting year 20...  
## starting year 30...  
## starting year 100...  
## starting year 200...  
## starting year 300...  
##   
## starting replicate 4  
## RunModel(): completed initialisation   
## starting year 0...  
## starting year 1...  
## starting year 2...  
## starting year 3...  
## starting year 10...  
## starting year 20...  
## starting year 30...  
## starting year 100...  
## starting year 200...  
## starting year 300...  
##   
## starting replicate 5  
## RunModel(): completed initialisation   
## starting year 0...  
## starting year 1...  
## starting year 2...  
## starting year 3...  
## starting year 10...  
## starting year 20...  
## starting year 30...  
## starting year 100...  
## starting year 200...  
##   
## starting replicate 6  
## RunModel(): completed initialisation   
## starting year 0...  
## starting year 1...  
## starting year 2...  
## starting year 3...  
## starting year 10...  
## starting year 20...  
## starting year 30...  
## starting year 100...  
## starting year 200...  
## starting year 300...  
##   
## starting replicate 7  
## RunModel(): completed initialisation   
## starting year 0...  
## starting year 1...  
## starting year 2...  
## starting year 3...  
## starting year 10...  
## starting year 20...  
## starting year 30...  
## starting year 100...  
## starting year 200...  
## starting year 300...  
##   
## starting replicate 8  
## RunModel(): completed initialisation   
## starting year 0...  
## starting year 1...  
## starting year 2...  
## starting year 3...  
## starting year 10...  
## starting year 20...  
## starting year 30...  
## starting year 100...  
## starting year 200...  
## starting year 300...  
##   
## starting replicate 9  
## RunModel(): completed initialisation   
## starting year 0...  
## starting year 1...  
## starting year 2...  
## starting year 3...  
## starting year 10...  
## starting year 20...  
## starting year 30...  
## starting year 100...  
## starting year 200...  
## starting year 300...  
##   
## \*\*\*\*\* Elapsed time: 27 seconds  
##   
## \*\*\*\*\*  
## \*\*\*\*\* Simulation completed   
## \*\*\*\*\* Outputs folder: /cloud/project/Range dispersal simulation/Outputs/  
## \*\*\*\*\*

## $Errors  
## [1] 0

# Summary of the results

## Range

range = read.csv("Outputs/Batch1\_Sim1\_Land1\_Range.txt", sep = "\t")  
range\_1000\_years = table(range$Rep) |> data.frame() |> rename(Replicate = Var1, Max\_Years = Freq) |> mutate(Max\_Years = Max\_Years -1)  
range\_1000\_years2 = range\_1000\_years[order(range\_1000\_years$Max\_Years, decreasing = T), ]  
range\_1000\_years2

## Replicate Max\_Years  
## 3 2 1000  
## 7 6 1000  
## 9 8 1000  
## 8 7 924  
## 1 0 588  
## 4 3 355  
## 10 9 333  
## 5 4 312  
## 2 1 254  
## 6 5 222

# Sample output: Range Output File

## Abundance

