Working Draft

Patrick McKenzie & Rachel Swenie

April 24, 2017

Biogeography Model

Using data matrices, movement function, and speciation function to reconstruct ancestral ranges.

Sample tree

Five tips, (1(1000))((2(400)3(400))(4(200)5(200)))

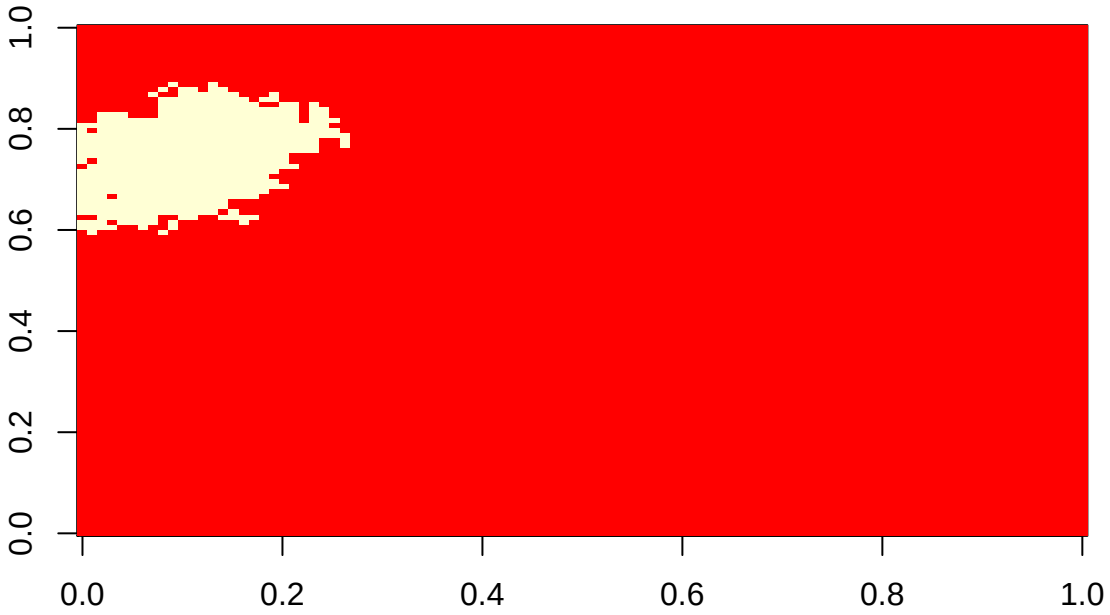
Generate a species range in a matrix with this cool function

Input the number of matrix rows and columns as the first two arguments, and the number of cells you want to be occupied by the species as the second argument. If you want to species' range to be centered around a particular point, designate that using a two-element vector as the argument for "startingcell" – otherwise, the range will be centered around a random point.

```
get_random_species_range <- function(nrow, ncol, numbercells, startingcell = NULL) {
  rows <- nrow
  cols <- ncol
  randommatrix <- matrix(rep(0,rows*cols), nrow = rows, ncol = cols)
  if (is.null(startingcell)) {
    index <- sample(rows*cols,1)
    randommatrix[index] <- 1
    pointindices <- arrayInd(index,c(rows,cols))
  } else {
    pointindices <- startingcell
    dim(pointindices) <- c(1,2)
  }

  while(sum(randommatrix) < numbercells) {
  workingcell <- pointindices[sample(nrow(pointindices),1),]
  random_adjacent <- sample(c(-1:1),2,replace = TRUE) #new cell row and col
  check_occupancy <- workingcell + random_adjacent
  if (sum(check_occupancy > 0) == 2 && sum(check_occupancy <= c(rows,cols)) == 2) {
    if (randommatrix[check_occupancy[1],check_occupancy[2]] == 0) {
      randommatrix[check_occupancy[1],check_occupancy[2]] <- 1
      pointindices <- rbind(pointindices,check_occupancy)
    }
  }
  }
  randommatrix
}
```

```r
example_range <- get_random_species_range(100,100,500)
image(example_range)
```



Movement and speciation

For each step in time, the following eqation is applied to each cell to calculate its new "probability" value. * These are normalized at each node and no longer represent true probabilities when moving up the tree.

$$P_{s,x,y}(t-1) = E_{x,y}(t-1)(((1-P_{s,x,y}(t)) * \alpha * \frac{\bar{N}}{8}) + (P_{s,x,y}(t) * \beta * \frac{\bar{N}}{8}))$$

The same equation is written out in code below. Note that `get_Nbar()` is a separate function written to calculate $\bar{N}$.

```r
get_prob_matrix <- function(Ps,Es, alpha, beta) {
  probs_older <- Ps
  for (i in 1:nrow(Ps)) { # rows
    for (q in 1:ncol(Ps)) { # columns
      Pcell <- Ps[i,q]
      Ecell <- Es[i,q]
      Nbar <- get_Nbar(Ps,i,q)
      Polder <- Ecell * ( ( (1-Pcell) * (Nbar/8) * alpha) + (Pcell * (Nbar/8) * beta) )
      probs_older[i,q] <- Polder
    }
  }
  probs_older
}
```

Scenario 1: Homogeneous environment