Predator, prey and strandings: An IBM approach to modelling the 'Blue Fleet'

J.M. Sassen

2023-04-25

1. Introduction

2. Model Elaboration

2.1 Conceptual Overview

The model is designed to investigate the effects of physical factors such as wind and ocean currents on the movement and behavior of the two species. We created a 50km x 10km grid, with a spatial resolution of 1 meter (5*10^9 total cells). Cells with x-values > 1km were assigned a 'land' status and all other cells where assigned an 'ocean' status. This resulted in a straight coastline stretching the left side of the grid, emulating the Eastern Australian Coastline. Each individual cell was then assigned a value for the following physical parameters: Wind Speed, Wind Direction, Current Speed, Current Direction.

The environment was then populated with individual Glaucus and Physalia, each with their own unique attributes. For glaucus, we assigned the following attributes: initial X coordinate, initial Y coordinate, chemodetection range, and movement speed. As both chemodetection and speed are traits that are expected to vary between Glaucus, these were assigned by drawing one value from a normal distribution with mean = LITERATURE and sd = LITERATURE. For Physalia, the following attributes were assigned: initial X coordinate, initial Y coordinate and orientation. Orientation refers to the position of the sail relative to the tentacles: this comes in a 'right-handed' or 'left-handed' variation. This variable was assigned randomly for each individual.

Whilst grid resolution was 1m, our model allows individuals to move through continuous space. The movement of an individual depends on the winds and current within the current cell, however the effect of these drivers might only amount to a movement of 20 cm within a timestep. This value is used to update the position of the individual. On the next iteration, the individual will simply 'feel' the same physical drivers, as it still finds itself in the same cell. This enables us to include the biological interactions that occur on the scale of centimeters.

2.2 Movement Equations

The main purpose of this model is to simulate the drifting patterns of two members of the Blue Fleet. The distribution of these animals is mainly driven by winds and currents. However, our model also accounts for the capability of ***Glaucus*** to make small directed movements towards detected prey, as well as the right-and-left-handedness of ***Physalia***. Moreover, once a Glaucus has succesfully latched on to a prey item, it is likely that it will follow the drift path of they prey, at least until it stops feeding. Although minor compared to the main physical drivers of movement, including these biological nuances in our movement equations allows us to build a realistic model.

The movement of an individual e

2.2.1 Glaucus atlanticus

$X_{ij+1} = X_{ij} + WindSpeed_{xy,i} * sin(WindDirection_{xy,i}) + CurrentSpeed_{xy,i} * sin(CurrentDirection_{xy,i}) + C * (GlaucusSpeed_{ij}$

$Y_{ij+1} = Y_{ij} + WindSpeed_{xy,i} * cos(WindDirection_{xy,i}) + CurrentSpeed_{xy,i} * cos(CurrentDirection_{xy,i}) + C * (GlaucusSpeed_{ij} *$

Where:

X_{ij+1} = Position along the East-West Axis for $Individual_j$ at Time = $i+1 X_{ij}$ = Position along the East-West Axis for $Individi$

The p

```
# Movement of Glaucus
glaucusMovement <- function(glaucus, physalia){

  # Check the glaucus is not beached
  if(glaucus$x > 1){
    glaucus$col <- 'steelblue'


    ### Predator module

    # Glaucus are predators. They can detect prey from distance using chemical cues. We want to simulat
    # by allowing glaucus limited movement to a target, if the target is within reasonable range.

    # scan area for each glaucus
```

```r
    # Convert to spatial geometries to allow geometric operations.
    spat.point <- st_point(c(glaucus$x, glaucus$y))

    # We can decide what a reasonable buffer is. This might be a parameter tied to the
    # individual Glaucus!
    detection.zone <- st_buffer(spat.point, glaucus$chemodetection)

    # Find the Physalia that are within the detection zone.
    # needs to be in df format!!!1 vectorise all at once.

    physalia.df <- rbindlist(physalia, fill = T)

    physalia.df.spat <- st_as_sf(physalia.df , coords = c('x', 'y'))

    # Any prey in the detection zone?
    if(any(st_intersects(physalia.df.spat, detection.zone, sparse = F))){
      # Find the nearest Physalia
      print('Attack!')
      glaucus.target <- physalia.df.spat[[st_nearest_feature(spat.point, physalia.df.spat),'geometry']]
      # Now make it move towards the target. We want to use a random walk-esque
      # movement with bias towards the physalia.

      target_angle <- atan2(glaucus.target[2] - glaucus$y, glaucus.target[1] - glaucus$x) # atan2 calcu
      glaucus$x <- glaucus$x + glaucus$speed * sin(target_angle)
      glaucus$y <- glaucus$y +  glaucus$speed * cos(target_angle)

    }


    # Now account for effect of current and wind on glaucus.

    # Positional update rules: this IS the movement of a Glaucus
    glaucus$x <- max(glaucus$x + 0.005*wind_strength[round(glaucus$x,digits = 2),
                                            round(glaucus$y, digits = 2)] * sin(wind_direction[round
                                                                                                 round
                       current_strength[round(glaucus$x,digits = 2),
                                     round(glaucus$y, digits = 2)] * sin(current_direction[round(glau
                                                                                             round(glau

    # Y movement (north - south) uses cosine function
    glaucus$y <- max(glaucus$y + 0.005*wind_strength[round(glaucus$x,digits = 2),
                                            round(glaucus$y, digits = 2)] * cos(wind_direction[round
                                                                                                 round
                       current_strength[round(glaucus$x,digits = 2),
                                     round(glaucus$y, digits = 2)] * cos(current_direction[round(glau
                                                                                             round(glau

  } else{
    glaucus$col <- 'red'
    glaucus$status <- 'BEACHED' # status 1 is beached
  }
  return(glaucus)
}
```

2.2.2 Physalia physalis

$X_{ij+1} = X_{ij} + 0.0266 * (WindSpeed_{xy,i} * sin(WindDirection_{xy,i})) + CurrentSpeed_{xy,i} * sin(CurrentDirection_{xy,i} + D_{ij}))Y_{ij+1}$

Where:

X_{ij+1} = Position along the East-West Axis for $Individual_j$ at Time = $i+1 X_{ij}$ = Position along the East-West Axis for $Individi$

```
# Movement of Physalia
physaliaMovement <- function(physalia, glaucus){

  # Boundary conditions for movement
  if(physalia$x >= 100 | physalia$y >= 100 | physalia$y <= 0){
    physalia$col <- 'green'
    physalia$status <- 2
    return(physalia)}

  # Live condition
    if(physalia$x > 1 & physalia$status != 'EATEN'){
    physalia$col <- 'purple'

    # This is a very important part. We have right and left=handed bluebottles
    # They drift in opposite directions - presumably to sustain populations.
    # We need to account for this properly. In addition, we want to add some
    # stochasticity to the movement. This is due to inherent variability
    # in the shape and size of bluebottles, but also due to waves etc.

    if(physalia$orientation == 'right') {direction_offset <- rnorm(1,1,0.1)*pi/3} # right-handed drift
    if(physalia$orientation == 'left') {direction_offset <- rnorm(1,-1,0.1)*pi/3} # left handed drift a

    # Positional update rules: this IS the movement of physalia
    # Wind larger impact on physalia due to sail
    # physalia also have an offset - their sails change the way they interact
    # with wind - this is one of the cool parts in the model.
    # 0.0266 - see Lee, Schaeffer, Groeskamp (2021)
    physalia$x <- max(physalia$x + 0.0266*(wind_strength[round(physalia$x,digits = 2),
                                        round(physalia$y, digits = 2)]) * sin(wind_direction

                      current_strength[round(physalia$x,digits = 2),
                                  round(physalia$y, digits = 2)] * sin(current_direction[round(p
                                                                        round(p

    # Y movement (north - south) uses cosine function
    physalia$y <- max(physalia$y + 0.0266*(wind_strength[round(physalia$x,digits = 2),
                                        round(physalia$y, digits = 2)]) * cos(wind_direction

                       current_strength[round(physalia$x,digits = 2),
                                  round(physalia$y, digits = 2)] * cos(current_direction[round(p
                                                                        round(p

    ### Predator module

    # We want to simulate damage to the physalia, i.e. being killed by a predator.
    # We use a simple rule: if the glaucus is within feeding range of the physalai
```

```r
    # for 3 timesteps, the physalia has been eaten completely. The glaucus will then also
    # move on as we delete the physalia.

    # Convert to spatial geometries to allow geometric operations.
    spat.point <- st_point(c(physalia$x, physalia$y))

    # We can decide what a reasonable buffer is. This is the same for
    # all physalia. 50 cm seems reasonable
    under.attack.zone <- st_buffer(spat.point, 0.0005)

    # Check if there is a predator nearby
    glaucus.df <- rbindlist(glaucus, fill = T)
    glaucus.df.spat <- st_as_sf(glaucus.df , coords = c('x', 'y'))

    # Any predators in the detection zone?
    if(any(st_intersects(glaucus.df.spat, under.attack.zone, sparse = F))){
      # If yes, the physalia is 'under attack'
      physalia$underattack <- physalia$underattack + 1
    }

    if(physalia$underattack >= 3){
      physalia$status <- 'EATEN'
    }

  } else{
    physalia$col <- 'red'
    if(physalia$x < 1){
      physalia$status <- 'BEACHED' # status 1 is beached
    }
  }
  return(physalia)
}
```

2.3 Biological Interactions