SpatialOncoSimul con OncoSimulR. Simulación espacial del crecimiento tumoral.

Antonio Giráldez, María González, Mercedes Núñez

Universidad Autónoma de Madrid
MSc in Bioinformatics and Computational Biology

11/01/2023

Índice

¿Qué es un modelo espacial?

Cada individuo/célula además de caracterizarse por su genotipo tiene asociada una posición única y tiene la capacidad de moverse a otros espacios (en 1, 2 o 3D).

Modelo espacial de Waclaw

- Cada célula → 1 posición espacial
- Reproducción ∝ nº posiciones vacías + fitness
- Probabilidad M → migración tras replicación
- Pushing → si no hay posiciones vacías
- Replicación y muerte en superficie
- Modelo híbrido (factores microambientales + genotipo)

Modelo espacial de Sottoriva

- Concepto de deme → 10000 células en una posición espacial concreta
- Células en un deme → mismo fitness
- Reproducción y muerte a nivel de deme ∝ fitness
- Demes con mayor fitness → expansión
- Modelo híbrido

Modelo espacial de Sun

- Concepto de deme → >= 1000 células en una posición espacial concreta
- Primera fase: reproducción intrademe (evolución de células adimensional)
- Segunda fase: componente espacial
- Modelo híbrido (posiciones vacías)

Características de cada modelo adoptadas

Deme (Sottoriva y Sun) ⇨ Zona espacial definida por unas coordenadas (x,y,z en caso de un espacio tridimensional) que puede albergar una cantidad limitada de células. Para reducir los costes computacionales derivados de guardar la ubicación y simular el movimiento de cada célula por separado.

Migración cercana (en los tres modelos) ⇨ Fenómeno por el cuál las células (o demes) ocupan posiciones adyacentes en el espacio, normalmente al reproducirse.

Migración lejana (Waclaw) ⇨ Fenómeno por el cual las células (o demes) migran hasta posiciones no adyacentes en el espacio.

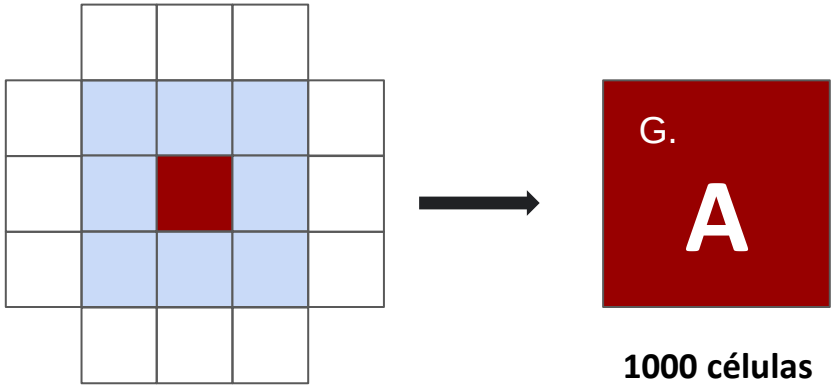
Algoritmo

1. Inicio
2. Fase intrademe
3. Fase interdeme
4. Condiciones de parada

Algoritmo

Inicio

- 1 deme de un tamaño concreto.
- 0 o más mutaciones iniciales.



Algoritmo

- **InitSize:** Población inicial de células WT en cada deme.
- **InitMutant:** Mutaciones iniciales con las que comienza el primer deme.

```
if (length(finalgrid_list) == 0) {
        simul_grid <- oncoSimulIndiv(fp, model, InitSize, InitMutant, …)
# Si en la lista de grids, o demes, aún no hay ninguno, se llama a la función base oncoSimulIndiv para
generar la primera simulación de la fase intrademe con un solo deme.

simul_grid <- data.frame(Genotype = simul_grid$GenotypesLabels, # Lista con todos los genotipos del deme
                         N = simul_grid$pops.by.time[nrow(simul_grid$pops.by.time), -1]) # Número de células por
                                                                                          genotipo
```

Algoritmo

- **InitSize:** Población inicial de células WT en cada deme.
- **InitMutant:** Mutaciones iniciales con las que comienza el primer deme.

```
if (SpatialModel == "1D"){
      simul_grid$Coordinates <- rep(list(0), length(
        simul_grid$Genotype))

} else if (SpatialModel == "2D"){
      simul_grid$Coordinates <- rep(list(c(0,0)), length(
        simul_grid$Genotype))
} else if (SpatialModel == "3D"){
      simul_grid$Coordinates <- rep(list(c(0,0,0)), length(
        simul_grid$Genotype))
}
```

Algoritmo

Fase intrademe

- Se modela el crecimiento dentro de cada deme utilizando la función oncoSimulIndiv.
- El output de oncoSimulIndiv se convierte en un data frame con el formato necesario para ser aceptado por las siguientes funciones.
- En los demes en los que el genotipo WT no está incluído en el resultado, se añade con una población = 0.

```
Final population composition:
  Genotype    N
1              0
2        i     0
3        u 1033
```

	Genotype	N	Coordinates.0	Coordinates.0.1	Coordinates.0.2
1		0	0	0	0
2	i	0	0	0	0
3	u	1033	0	0	0

Algoritmo

Fase intrademe

- **mu:** Probabilidad de mutación de cada gen del genotipo.
- **sampleEvery:** frecuencia con la que se muestrea la población.
- **finalTime:** tiempo durante el que corre el modelo en cada iteración y cada deme.

Simulación del crecimiento intrademe a partir de la nueva composición de genotipos

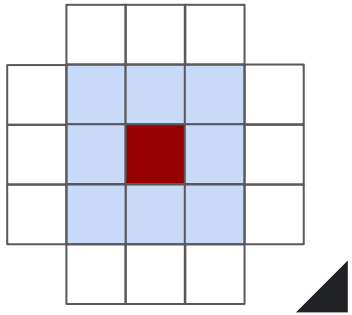
```
# 1. Function for tumor porgression simulation in a grid.
oncoSimulIndiv_grid <- function(grid){
    # If all the cells from a grid disappear N will be 0, so this grid
    # will be deleted from the list of current grids. The growth simulation
    # inside a grid will only be performed in grids with N > (exp(1) - 1) to
    # calculate K initial population equilibrium size.

    if (sum(grid$N) > (exp(1) - 1)){
        grid <- grid[which(grid$N > 0),]
        simul_grid <- invisible(oncoSimulIndiv(fp = fp, model = model,
                                numPassengers = numPassengers, mu = mu,
                                muEF = muEF,
                                detectionSize = detectionSize,
                                detectionDrivers = detectionDrivers,
                                detectionProb = detectionProb,
                                sampleEvery = sampleEvery,
                                initSize = grid$N,
                                initMutant = grid$Genotype,
                                s = s, sh = sh,
                                K = K, keepEvery = keepEvery,
                                minDetectDrvCloneSz = minDetectDrvCloneSz,
                                extraTime = extraTime,
                                finalTime = finalTime,
                                onlyCancer = onlyCancer
                                                        opGrowth,
```

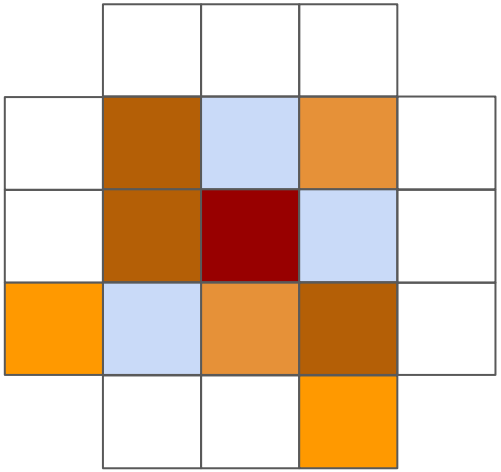
```
intragrid_list <- mclapply(finalgrid_list, function(x)
                           oncoSimulIndiv_grid(grid = x),
                           mc.cores = mc.cores)
```

Algoritmo

Fase Interdeme

- De cada deme se elige un número de células.
  - Se produce la migración a **demes adyacentes.**
  - Se produce la migración a **demes lejanos**





Algoritmo

- **migrationProb:** Probabilidad de que en un deme haya migración de células tumorales a demes adyacentes.
- **largeDistMigrationProb:** Probabilidad de que en un deme haya migración de células tumorales a demes lejanos.
- **maxMigrationPercentage:** Máximo porcentaje de células de cada deme que migrarían cuando se ha satisfecho la probabilidad de migración a regiones adyacentes o lejanas (o ambas).

```r
SimulMigration <- function(grid, migrationProb = 0.5,
                           largeDistMigrationProb = 1e-6,
                           maxMigrationPercentage = 0.2){
  # Create a function to define random coordinates for near and remote
  # migrating cells.
  randomcoordinates <- function(grid, MigrationType = "Near"){
    init_coordinates <- grid$Coordinates[[1]]
    coord_nearmigration <- init_coordinates
    coord_remotemigration <- init_coordinates

    # While loop to guarantee that coordinates for migrating cells are
    # different from the ones of the grid they belong to.
    while ((identical(coord_nearmigration, init_coordinates))
           & MigrationType == "Near"){
      coord_nearmigration <- c()
      for (coord in init_coordinates){
        new_coord <- as.numeric(sample(seq(coord - 1, coord + 1), 1))
        coord_nearmigration <- c(coord_nearmigration, new_coord)
      }
    }
    while ((identical(coord_remotemigration, init_coordinates))
           & MigrationType == "Remote"){
      coord_remotemigration <- c()
      for (coord in init_coordinates){
        new_coord <- as.numeric(sample(c(coord - sample(seq(10, 35), 1),
                                         coord +
                                           sample(seq(10, 35), 1)), 1))
        coord_remotemigration <- c(coord_remotemigration, new_coord)
      }
    }
    if (MigrationType == "Near"){
      return (list(coord_nearmigration))
    } else if (MigrationType == "Remote"){
      return (list(coord_remotemigration))
    }
  }
```

 # Generación de coordenadas aleatorias para la migración a demes adyacentes y lejanos.

Algoritmo

```
near_probtest <- sample(c("Migration", "NoMigration"), 1,
                        prob = c(migrationProb, 1 - migrationProb))
remote_probtest <- sample(c("Migration", "NoMigration"), 1,
                          prob = c(largeDistMigrationProb,
                                   1 - largeDistMigrationProb))
```

 # Selección de migración adyacente o lejana

Algoritmo

```
MigPopulation <- sample(seq(from = 1, to = TotalPopSize *
                                    maxMigrationPercentage/100), 1)
```

```
# Población del deme que migra a zonas cercanas o alejadas.
```

```
MigPopulationcomp <- as.data.frame(table(
                                    sample(finalpopcomp_mut$Genotype,
                                           MigPopulation,
                                           replace = TRUE,
                                           prob = finalpopcomp_mut$N)))
```

```
# Composición de la población de células que migran.
```

Algoritmo

 # Determinación del número de células que migrarán a qué regiones del espacio cuando se satisface la probabilidad de migración a regiones alejadas y cercanas

 # Coordenadas del espacio a las que migran las células tumorales

```
  NearMigPop_number <- sample (from = 1, to = MigPopulation - 1,
                               1)
  RemoteMigPop_number <- MigPopulation - NearMigPop_number
  NearMigPopulationcomp <- as.data.frame(table(
                           sample(finalpopcomp_mut$Genotype,
                                  NearMigPop_number,
                                  replace = TRUE,
                                  prob = finalpopcomp_mut$N)))
  RemoteMigPopulationcomp <- as.data.frame(table(
                                sample(finalpopcomp_mut$Genotype,
                                       RemoteMigPop_number,
                                       replace = TRUE,
                                       prob = finalpopcomp_mut$N)))
  colnames(NearMigPopulationcomp) <-c("Genotype", "N")
  NearMigPopulationcomp$Coordinates <- randomcoordinates(
                                       grid = grid, "Near")
  colnames(RemoteMigPopulationcomp) <-c("Genotype", "N")
  RemoteMigPopulationcomp$Coordinates <- randomcoordinates(
                                       grid = grid, "Remote")
  MigPopulationcomp <- rbind(NearMigPopulationcomp,
                             RemoteMigPopulationcomp)
}
```

Algoritmo

```
  for (gen in unique(MigPopulationcomp$Genotype)) {
    Nmigration_per_genotype <- sum(MigPopulationcomp$N[which(
      MigPopulationcomp$Genotype == gen)])
    grid$N[which(grid$Genotype == gen)] <- (
      grid$N[which(grid$Genotype == gen)] - Nmigration_per_genotype)
  }
  return (list(grid, MigPopulationcomp))
```

 # Intercambio final de las células entre los demes al final de la fase interdeme

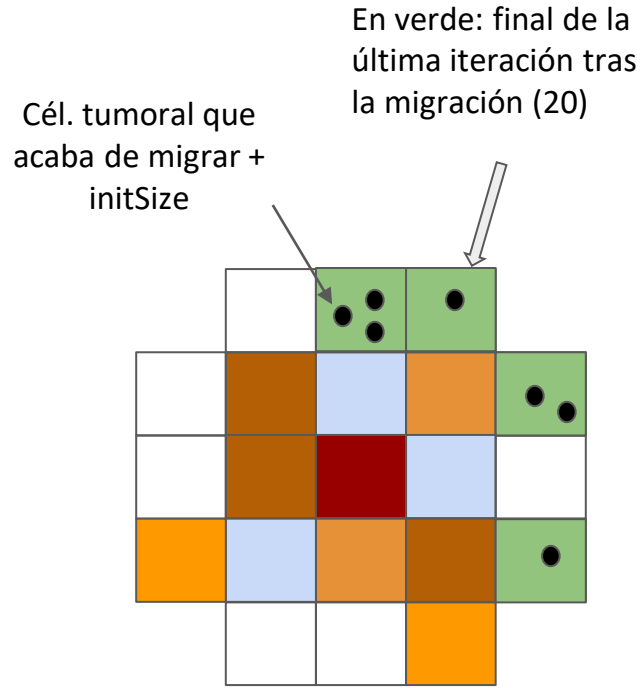
Algoritmo

Condiciones de parada

- Una iteración del algoritmo finaliza tras la fase intrademe de la iteración siguiente.
- Justificación: no parar la simulación en la última iteración tras la fase de migración → quedarían demes sin desarrollar

⇩

spatialIterMax: número máximo de iteraciones alcanzadas en una simulación



Algoritmo

Condiciones de parada

- Una iteración del algoritmo finaliza tras la fase intrademe de la iteración siguiente.
- Justificación: no parar la simulación en la última iteración tras la fase de migración → quedarían demes sin desarrollar

⇩

spatialIterMax: número máximo de iteraciones alcanzadas en una simulación

```
Iteration 1 (2 demes)
Iteration 2 (3 demes)
Iteration 3 (4 demes)
Iteration 4 (6 demes)
Iteration 5 (9 demes)
Iteration 6 (12 demes)
Iteration 7 (16 demes)
Iteration 8 (25 demes)
Iteration 9 (34 demes)
Iteration 10 (47 demes)
Iteration 11 (66 demes)
Iteration 12 (89 demes)
Iteration 13 (120 demes)
Iteration 14 (160 demes)
Iteration 15 (204 demes)
[1] "Final Population: 230717 (204 demes 15 iterations)"
[1] "Genotype: i, v --- Population: 2731 --- Demes: 4"
[1] "Genotype: u --- Population: 182975 --- Demes: 201"
```

Ejemplos:

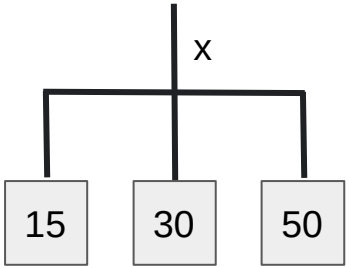
Ejemplo 1.

```
s1 <- allFitnessEffects(
  data.frame(parent = c("Root", "Root", "i"),
             child = c("u" , "i" , "v"),
             s = c(0.1 , -0.05 , 0.25),
             sh = -1,
             typeDep = "MN"),
             epistasis = c("u:i" = -1,"u:v" = -1))

evalAllGenotypes (s1 , order = FALSE, addwt = TRUE)

Spatial_3D <- SpatialOncoSimul(fp = s1,
                      model = "McFL",
                      onlyCancer = FALSE,
                      finalTime = 500,
                      mu = 1e-4,
                      initSize = 1000,
                      keepPhylog = FALSE,
                      seed = NULL,
                      errorHitMaxTries = FALSE,
                      errorHitWallTime = FALSE,
                      initMutant = c("i"),
                      spatialIterMax = 30,
                      SpatialModel = "3D")
```

Efectos de fitness

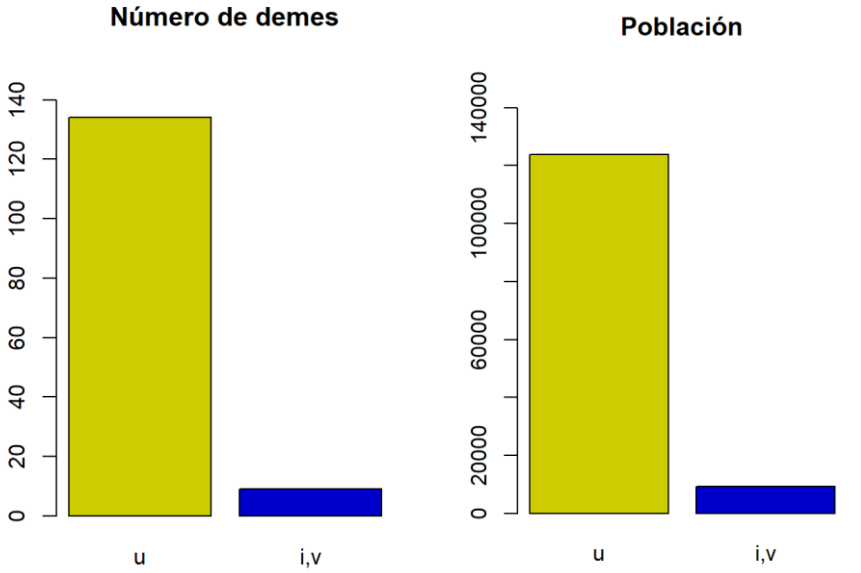
Genotype	Birth
WT	1.0000
i	0.9500
u	1.1000
v	0.0000
i, u	0.0000
i, v	1.1875
u, v	0.0000
i, u, v	0.0000

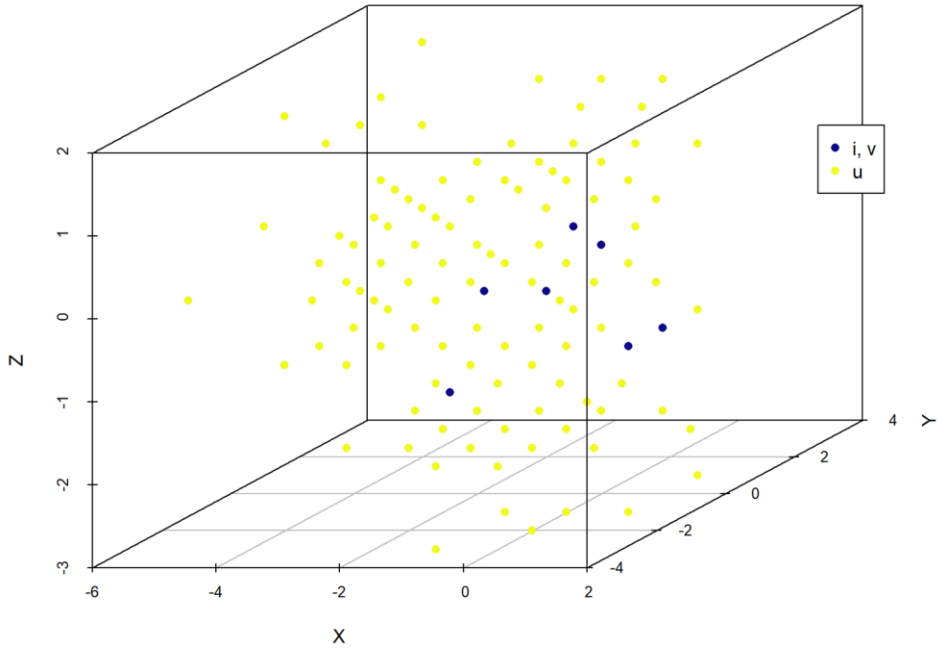
spatialIterMax



Ejemplos:

Ejemplo 1 (15 iteraciones)

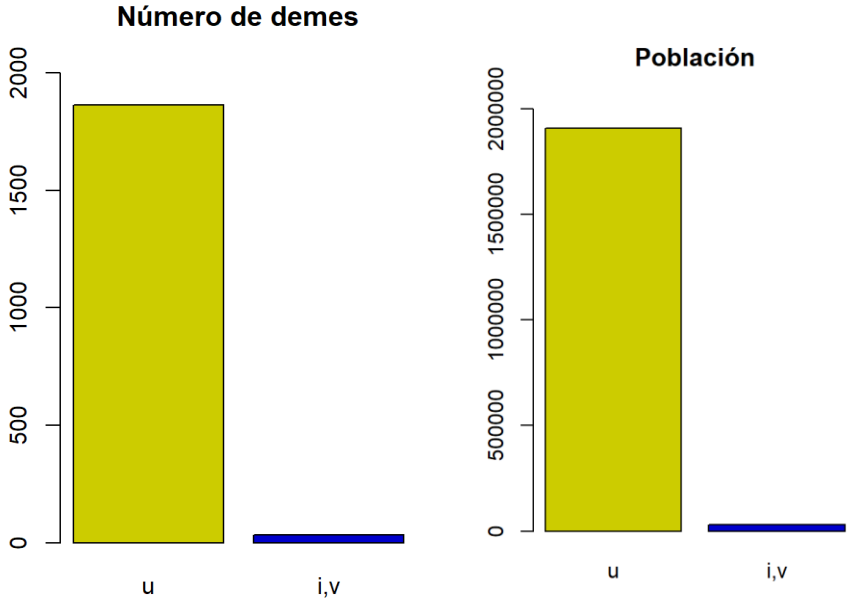


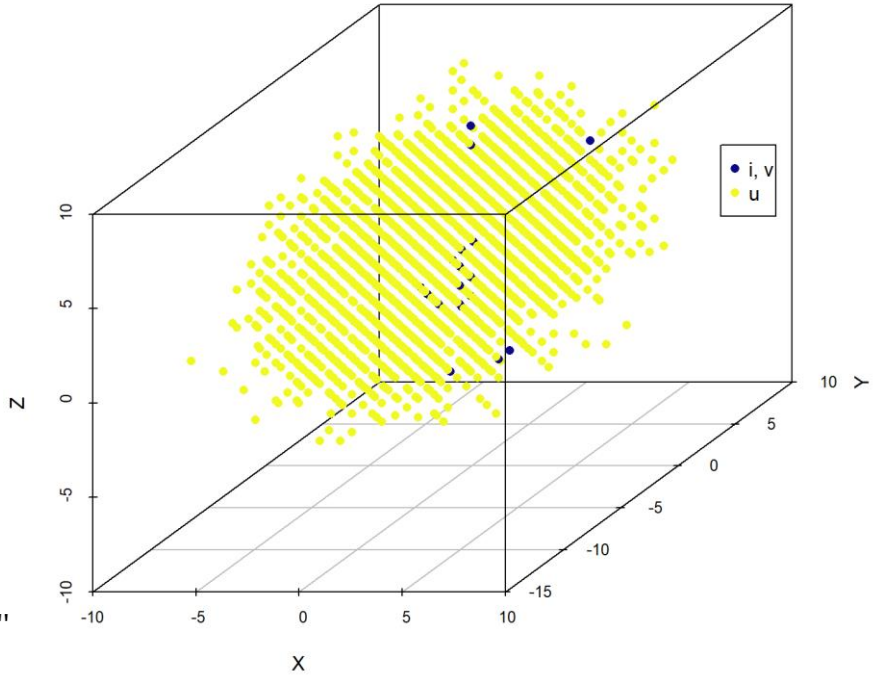



```
[1] "Final Population: 161978 (142 demes 15 iterations)
[1] "Genotype: i, v --- Population: 9251 --- Demes: 9"
[1] "Genotype: u --- Population: 123900 --- Demes: 134"
```

Ejemplos:

Ejemplo 1 (30 iteraciones)

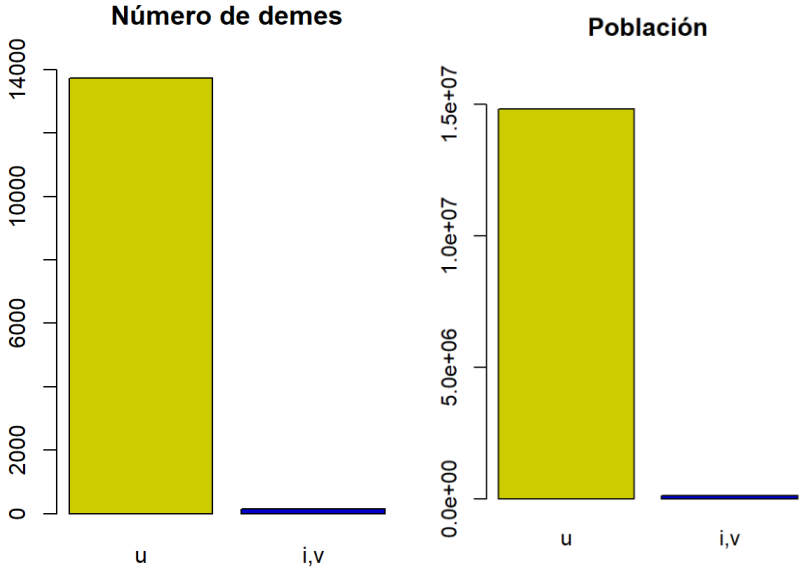


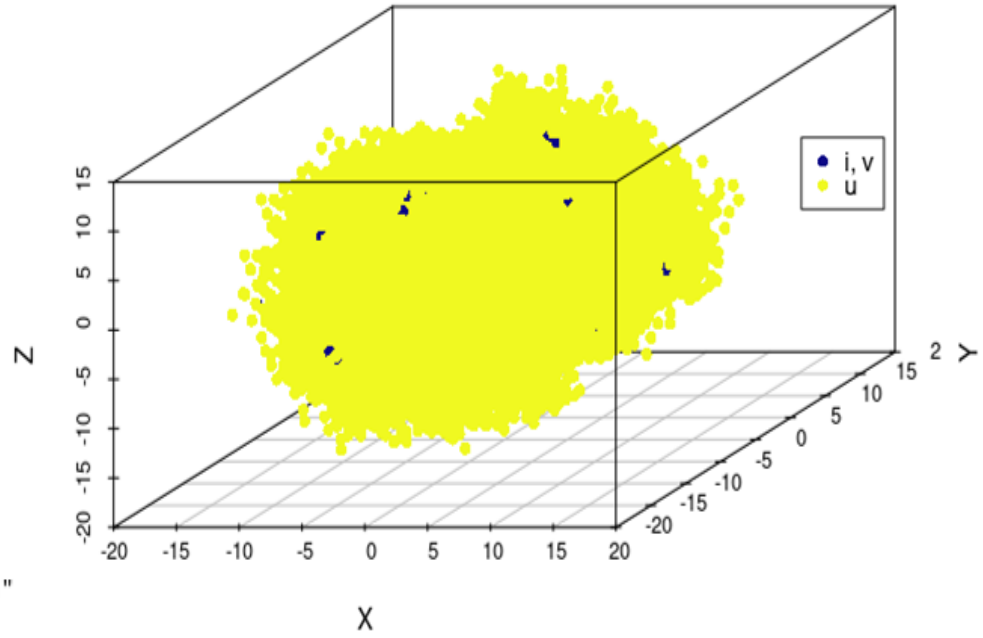



```
[1] "Final Population: 2158689 (1884 demes 30 iterations)"
[1] "Genotype: i, v --- Population: 29267 --- Demes: 34"
[1] "Genotype: u --- Population: 1909348 --- Demes: 1862"
```

Ejemplos:

Ejemplo 1 (50 iteraciones)






```
[1] "Final Population: 15931596 (13800 demes 50 iterations)"
[1] "Genotype: i --- Population: 1 --- Demes: 1"
[1] "Genotype: i, v --- Population: 130000 --- Demes: 140"
[1] "Genotype: u --- Population: 14828918 --- Demes: 13731"
```

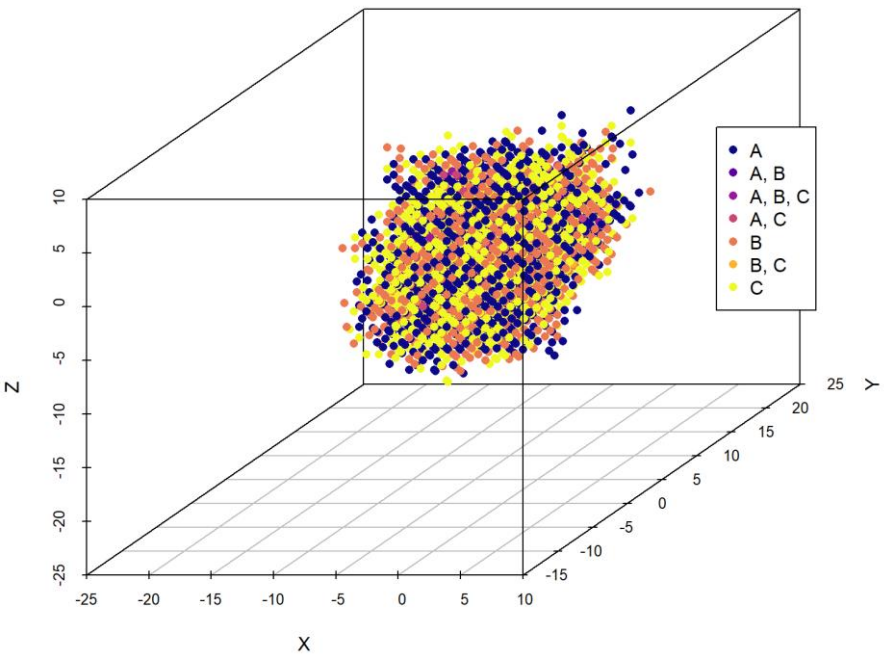
Ejemplos:

Ejemplo 2 (Análisis del parámetro de migración: MigrationProb y largeDistMigrationProb)

```
s2 <- allFitnessEffects(epistasis = c ("A:-B:-C"=0.05, "-A:B:-C"=0.05,
                                       "-A:-B:C"=0.05 ,"A:B:-C"=0.045,
                                       "A:-B:C"=0.045 , "-A:B:C"=0.045,
                                       "A:B:C"=0.055))
Spatial_3D_2 <- SpatialOncoSimul (fp = s2,
                                  onlyCancer = FALSE,
                                  finalTime = 200,
                                  mu = 1e-4,
                                  migrationProb = 1,
                                  largeDistMigrationProb = 1e-5,
                                  maxMigrationPercentage = 0.2,
                                  initSize = 1000,
                                  initMutant = c("A"),
                                  keepPhylog = FALSE,
                                  seed = NULL,
                                  errorHitMaxTries = FALSE,
                                  errorHitWallTime = FALSE,
                                  spatialIterMax = 20,
                                  SpatialModel = "3D")
```

```
[1] "Final Population: 4170428 (3925 demes 20 iterations)"
[1] "Genotype: A --- Population: 1070862 --- Demes: 1999"
[1] "Genotype: A, B --- Population: 46593 --- Demes: 892"
[1] "Genotype: A, B, C --- Population: 50020 --- Demes: 194"
[1] "Genotype: A, C --- Population: 55521 --- Demes: 976"
[1] "Genotype: B --- Population: 987181 --- Demes: 1865"
[1] "Genotype: B, C --- Population: 44679 --- Demes: 851"
[1] "Genotype: C --- Population: 1063634 --- Demes: 1930"
```

largeDistMigrationProb → 1e-5

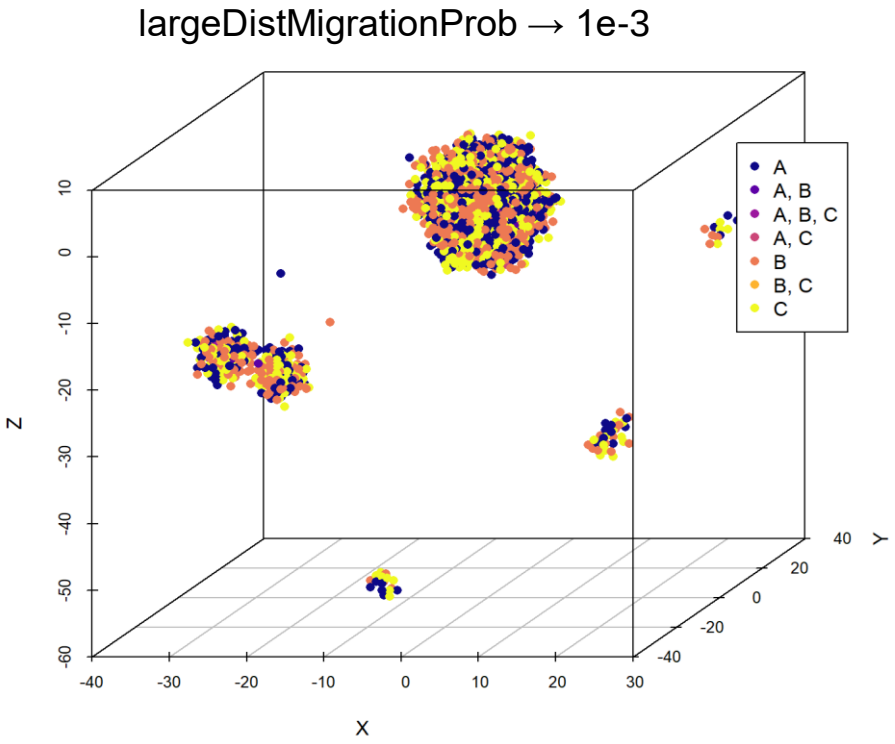


Ejemplos:

Ejemplo 2 (Análisis del parámetro de migración: MigrationProb y largeDistMigrationProb)

```
s2 <- allFitnessEffects(epistasis = c ("A:-B:-C"=0.05, "-A:B:-C"=0.05,
                                       "-A:-B:C"=0.05 ,"A:B:-C"=0.045,
                                       "A:-B:C"=0.045 , "-A:B:C"=0.045,
                                       "A:B:C"=0.055))
Spatial_3D_2 <- SpatialOncoSimul (fp = s2,
                                  onlyCancer = FALSE,
                                  finalTime = 200,
                                  mu = 1e-4,
                                  migrationProb = 1,
                                  largeDistMigrationProb = 1e-5,
                                  maxMigrationPercentage = 0.2,
                                  initSize = 1000,
                                  initMutant = c("A"),
                                  keepPhylog = FALSE,
                                  seed = NULL,
                                  errorHitMaxTries = FALSE,
                                  errorHitWallTime = FALSE,
                                  spatialIterMax = 20,
                                  SpatialModel = "3D")
```

```
[1] "Final Population: 3939096 (3717 demes 20 iterations)"
[1] "Genotype: A --- Population: 951000 --- Demes: 1820"
[1] "Genotype: A, B --- Population: 46657 --- Demes: 884"
[1] "Genotype: A, B, C --- Population: 45180 --- Demes: 179"
[1] "Genotype: A, C --- Population: 44262 --- Demes: 829"
[1] "Genotype: B --- Population: 953353 --- Demes: 1826"
[1] "Genotype: B, C --- Population: 42016 --- Demes: 806"
[1] "Genotype: C --- Population: 932886 --- Demes: 1780"
```



Discusión

- Si un deme está rodeado por otros demes, no estamos teniendo en cuenta que la probabilidad de migración podría ser menor que en un deme que tiene más espacios vacíos alrededor.
- El usuario pueda parar la simulación en el momento que desee sin tener que predefinir un número máximo de iteraciones.
- Añadir un nuevo parámetro que permita al usuario determinar el número máximo de demes de la simulación. Pudiendo así limitar el número de iteraciones y el número de demes final.
- Permitir al usuario acotar el espacio en el que el tumor puede crecer.
- Considerar la participación de factores ambientales externos presentes en diferentes zonas del espacio y que pueda afectar al crecimiento y a la probabilidad de migración en cada deme.

Bibliografía

Parramón Castillo, A. (2018). Simulación de un modelo espacial de evolución tumoral con R y C++. [TFM]. Universidad Complutense de Madrid.

Diaz-Uriarte, R. (2015). Identifying restrictions in the order of accumulation of mutations during tumor progression: effects of passengers, evolutionary models, and sampling.

Bozic, I., et al., (2010). Accumulation of driver and passenger mutations during tumor progression. Proceedings of the National Academy of Sciences of the United States of America, 107, 18545--18550.

R Diaz-Uriarte. (2017). OncoSimulR: genetic simulation with arbitrary epistasis and mutator genes in asexual populations. Bioinformatics, 33, 1898--1899.

R Diaz-Uriarte and C. Vasallo. (2019). Every which way? On predicting tumor evolution using cancer progression models 2019 PLoS Computational Biology.

R Diaz-Uriarte. (2017). Cancer progression models and fitness landscapes: a many-to-many relationship 2017 Bioinformatics.

Gerstung et al., 2011. The Temporal Order of Genetic and Pathway Alterations in Tumorigenesis. PLoS ONE, 6.

McFarland, C.~D. et al. (2013). Impact of deleterious passenger mutations on cancer progression. Proceedings of the National Academy of Sciences of the United States of America, 110(8), 2910--5.

Mather, W.~H., Hasty, J., and Tsimring, L.~S. (2012). Fast stochastic algorithm for simulating evolutionary population dynamics. Bioinformatics (Oxford, England), 28(9), 1230--1238.