Solutions:

Spatial model dynamics

1. Non-Trivial Equilibrium Population Size

The population dynamics for n1 are:

$$\frac{dn_1}{dt} = (r_1 - kn_1^2)n_1$$

At equilibrium the above equation = 0

The solutions are:

1. n1=0 (trivial equilibrium),
2. $n_1 = \sqrt{\frac{r_1}{k}}$ (non-trivial equilibrium).

2. Stability of the Non-Trivial Equilibrium

The stability is determined by the derivative of the function with respect to n1:

$$\frac{d}{dn_1}\left[\frac{dn_1}{dt}\right] = r_1 - 3kn_1^2$$

At the equilibrium (plug in equilibrium from above) :

$$\frac{d}{dn_1}\left[\frac{dn_1}{dt}\right] = r_1 - 3k\left(\sqrt{\frac{r_1}{k}}\right)^2 = r_1 - 3r_1 = -2r_1$$

Since r1>0, the derivative is negative, making the equilibrium **stable**.

---

3. R Script and Plot for dndt\frac{dn}{dt}dtdn as a Function of nnn

R Script

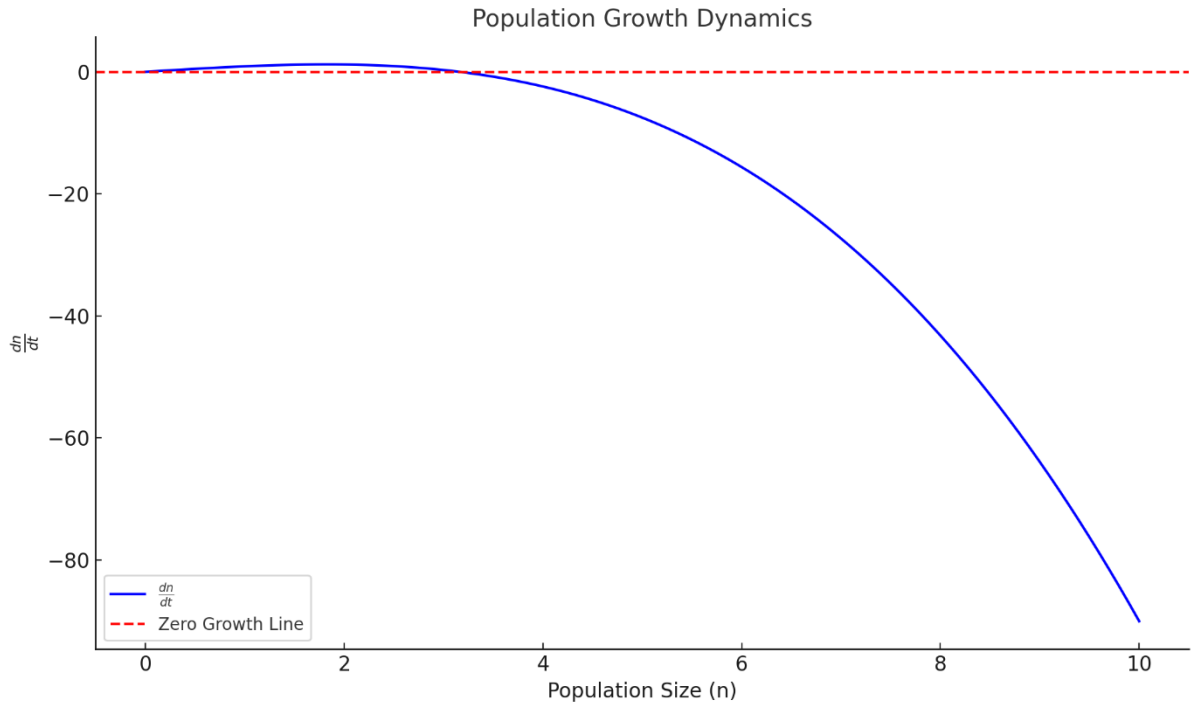
```
r
Kopiera kod
# Parameters
```

```
r1 <- 1        # Intrinsic growth rate
k <- 0.1       # Strength of density dependence

# Function to calculate dn/dt
dn_dt <- function(n) {
  (r1 - k * n^2) * n
}

# Population sizes
n_values <- seq(0, 10, length.out = 100)

# Plot dn/dt as a function of n
plot(n_values, dn_dt(n_values), type = "l", col = "blue",
     xlab = "Population Size (n)", ylab = "dn/dt",
     main = "Population Growth Dynamics")
abline(h = 0, col = "red", lty = 2)
```

Plot

Below is the plot of dn/dt as a function of n:

Population Growth Dynamics



4. Dynamics with One-Directional Dispersal

The new dynamics, with dispersal from n1n_1n1 to n2n_2n2 at a rate mn1m n_1mn1, are:

$$\frac{dn_1}{dt} = (r_1 - kn_1^2)n_1 - mn_1$$

$$\frac{dn_2}{dt} = (r_2 - kn_2^2)n_2 + mn_1$$

5. New Equilibrium of Population 1

At equilibrium dn1/dt =0:

$$(r_1 - kn_1^2)n_1 - mn_1 = 0$$

Factoring out n1:

$$r_1 - kn_1^2 - m = 0$$

Rearranging :

$$n_1^2 = \frac{r_1 - m}{k}$$

The new equilibrium for population 1 is:

$$\boldsymbol{n_1 = \sqrt{\frac{r_1 - m}{k}}}$$

6. Threshold for Population 1 Extinction

Population 1 goes extinct when n1≤0, which occurs when:

$$r_1 - m \leq 0 \quad \backslash implies \quad m \geq r_1$$

Thus, **population 1 goes extinct if m>r1** .

---

7. R Script and Plot for Coupled Dynamics

R Script

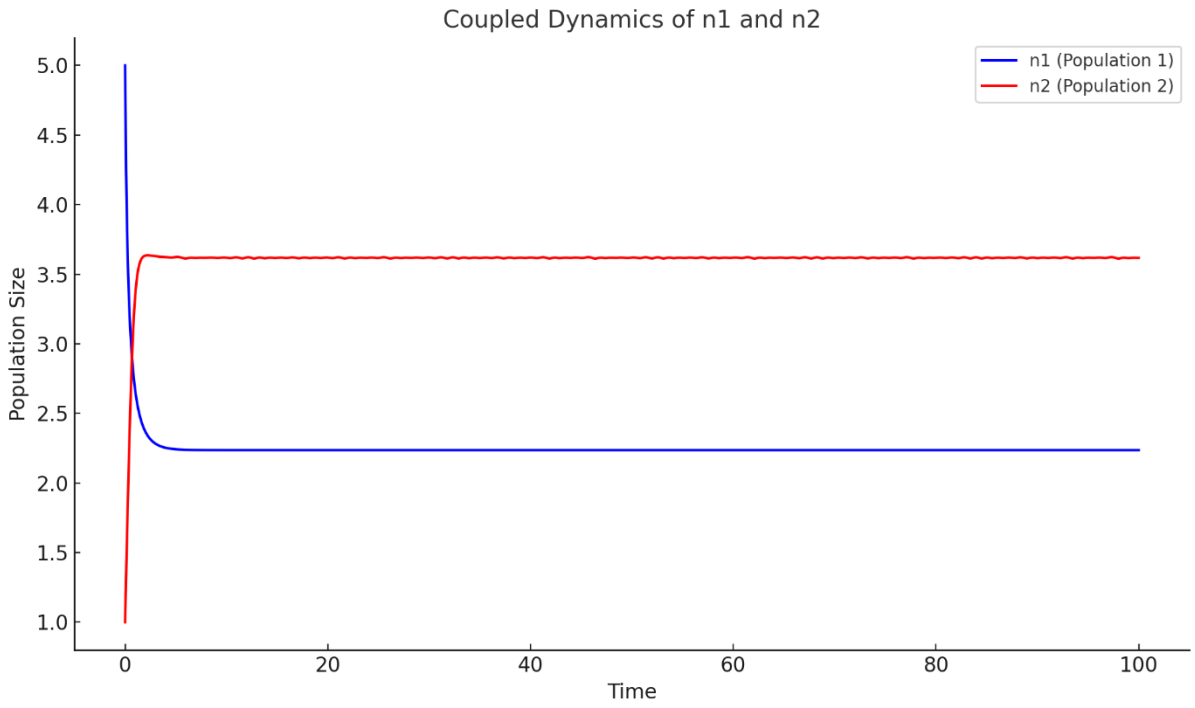
```
r
Kopiera kod
# Parameters
r1 <- 1       # Intrinsic growth rate for population 1
```

```
r2 <- 1       # Intrinsic growth rate for population 2
k <- 0.1      # Strength of density dependence
m <- 0.5      # Migration rate

# Time sequence
time <- seq(0, 100, by = 0.1)

# Differential equations
coupled_dynamics <- function(t, state, parameters) {
  n1 <- state[1]
  n2 <- state[2]

  dn1 <- (r1 - k * n1^2) * n1 - m * n1
  dn2 <- (r2 - k * n2^2) * n2 + m * n1

  list(c(dn1, dn2))
}

# Initial conditions
initial_state <- c(n1 = 5, n2 = 1)

# Solve the system
library(deSolve)
result <- ode(y = initial_state, times = time, func = coupled_dynamics,
parms = NULL)

# Plot the results
plot(result[, "time"], result[, "n1"], type = "l", col = "blue", lwd = 2,
     xlab = "Time", ylab = "Population Size",
     main = "Coupled Dynamics of n1 and n2")
lines(result[, "time"], result[, "n2"], col = "red", lwd = 2)
legend("topright", legend = c("n1 (Population 1)", "n2 (Population 2)"),
       col = c("blue", "red"), lty = 1, lwd = 2)
```

Plot

Below is the plot of the coupled dynamics of n1n_1n1 and n2n_2n2 over time:

Coupled Dynamics of n1 and n2



Here is the plot of the coupled dynamics of n1n_1n1 and n2n_2n2 over time. You can also download the plot as a PNG file.

The dynamics of a chemical reaction (4p)

A mixture has two chemical compounds, X and Y. Two units of X can combine and form a single unit of Y. The compound Y is, however, unstable and spontaneously disintegrates into two units of X. The chemical reactions can be written

$$2X \leftrightarrow Y$$

The dynamics of the corresponding concentrations, denoted *x* and *y*, respectively, follow

$$\begin{cases} \frac{dx}{dt} = -2kx^2 + 2\mu y \\ \frac{dy}{dt} = kx^2 - \mu y \end{cases},$$

where *k* and *μ* are positive constants.

a) Show that the system has a whole suite of equilibrium states (which depend on the initial conditions, i.e. x(0) and y(0)) (1p)

$$\frac{dx}{dt} = -2kx^2 + 2\mu y = 0$$

$$y = \frac{k}{\mu} x^2$$

Samma för båda ekvationerna

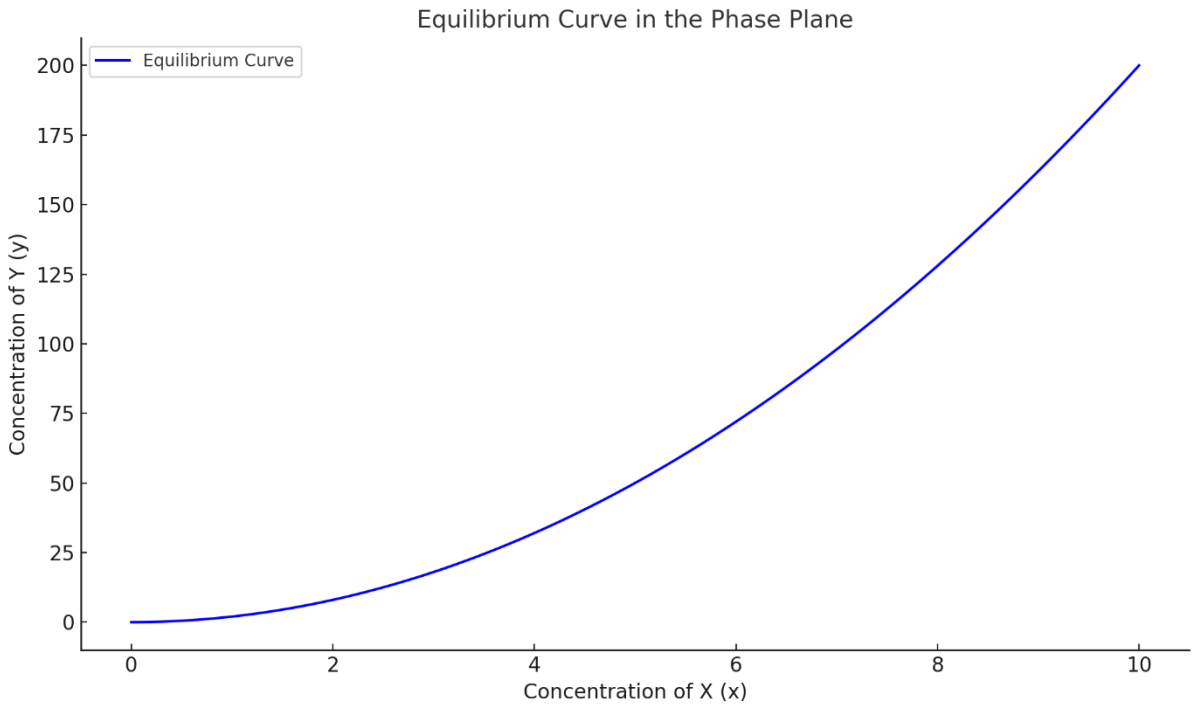
b) Write an R script that plots the possible equilbria in the xy phase plane. (1p) (Any values of *k* and μ will do)

```
# Parameters

k <- 1      # Reaction rate constant

mu <- 0.5    # Disintegration rate constant


# Equilibrium relationship: y = k * x^2 / mu

x_values <- seq(0, 10, length.out = 100)

y_values <- k * x_values^2 / mu


# Plot the equilibrium curve in the phase plane

plot(x_values, y_values, type = "l", col = "blue",

   xlab = "Concentration of X (x)", ylab = "Concentration of Y (y)",

   main = "Equilibrium Curve in the Phase Plane")

grid()
```

ChatGPT sade:

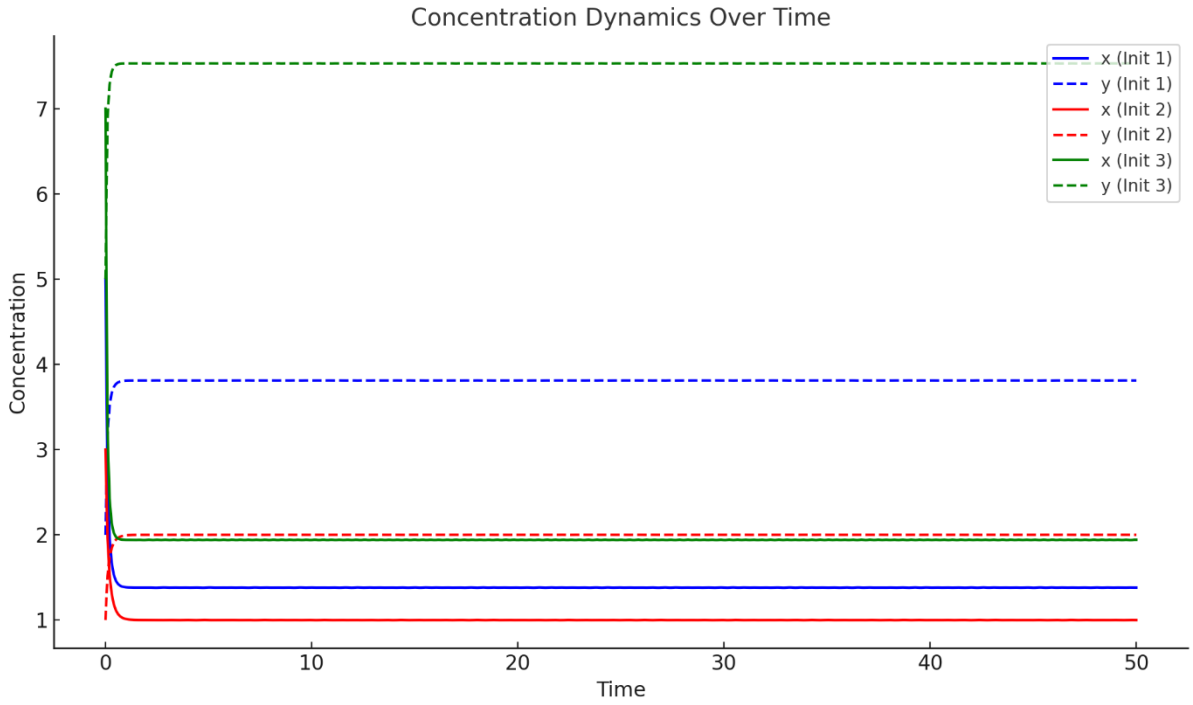
ChatGPT

Equilibrium Curve in the Phase Plane

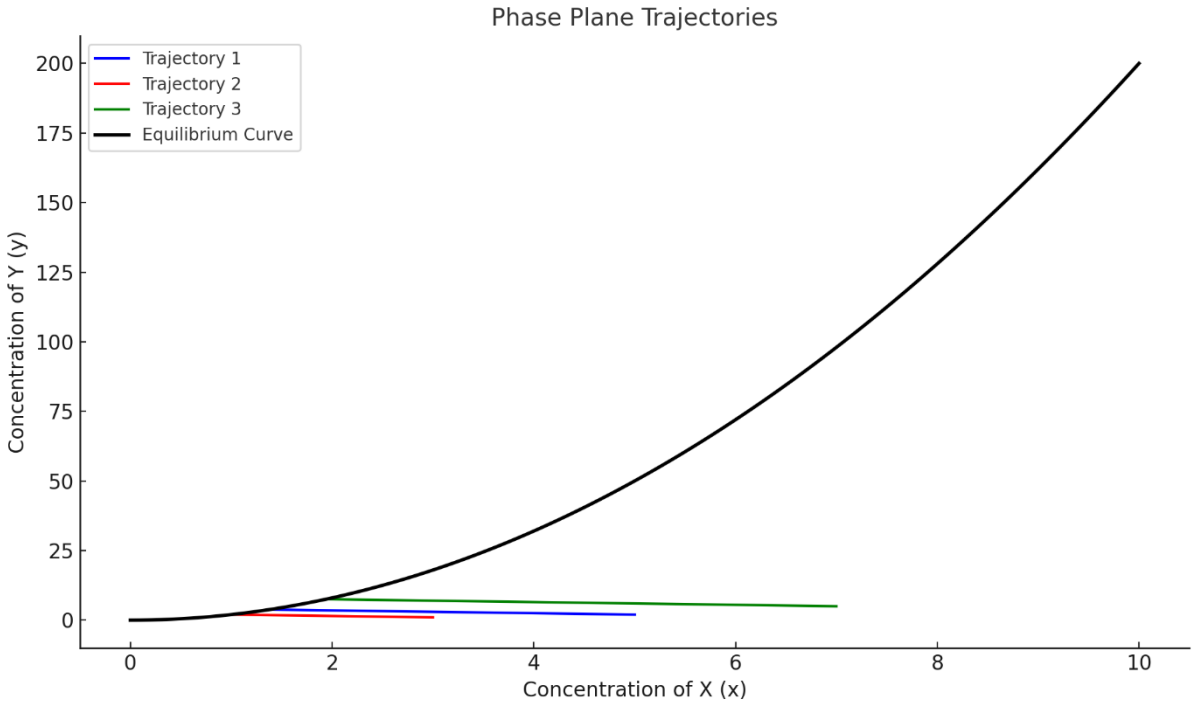


c) Write another script that simulates the differential equations above, using a few different initial conditions, and plots the result in two ways: *i*) as *x* and *y* vs. time and *ii*) in the phase plane together with the solution in b). (2p)

Concentration Dynamics Over Time



Phase Plane Trajectories



Introgression (6p)

The process of 'foreign' genes entering, and becoming established in, a population is called introgression. For example, one can follow the introgression of genes across hybrid zones, where genes can travel from one population (or species) to another. A famous example is the finding that all humans with ancestors from outside Africa have a small proportion of Neanderthal genes, probably a result of hybridization and introgression that happened a long time ago. We shall now study the process of introgression, keeping things as simple as possible.

Consider a haploid population of n individuals (n is fixed). Each individual carries a genome of L genes. The genes can be of two types, the old type and the 'new' type. If we represent the old type with zeros and the new type with ones, an individual that carries some old and some new genes could be represented by a sequence like 0110 (for the case $L = 4$).

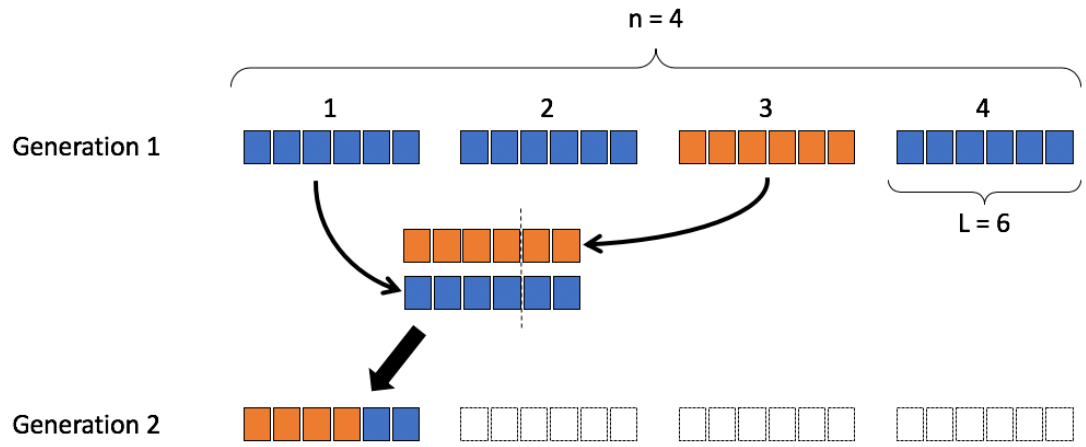
Each generation, the individuals mate randomly and the offspring is formed through recombination. More precisely (see figure):

Start the simulation (generation 1) with a single individual with a completely foreign genome (orange in the figure), and all other individuals of the old type.

For each individual of the next generation:

i. choose two parents randomly from the parent generation
ii. pair up their genomes and choose a random point of gene crossover (dashed line in figure).
iii. put the recombined offspring in the next generation
iv. repeat until the new population is full

The new population then replaces the old one and the whole procedure repeats. There is no selection for or against the new genes.



a) If you would write a program to simulate this process, what would be an appropriate representation of the population, describing the current 'system state'? **(1p)**

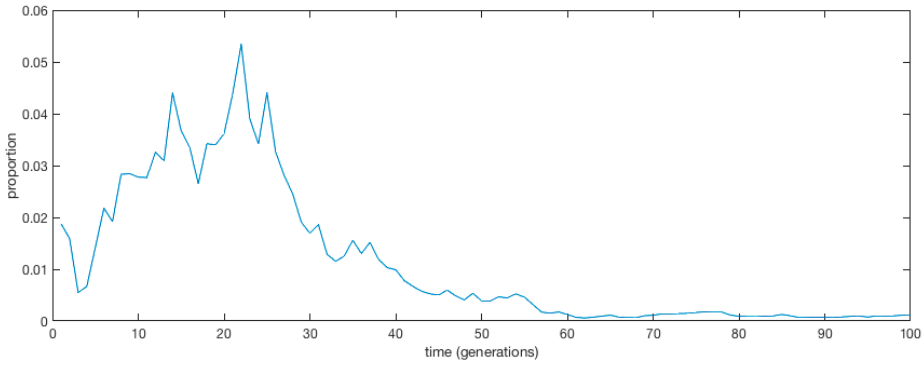
Each individual could be a vector of zeros and ones, representing old type and new type genes, respectively. The vectors could be arranged in a matrix, such that each row represents an individual, for example. There are many other alternatives, using lists, for example.

b) Write a *function* in *R* that takes two parent genomes as input and returns a single recombined offspring genome, following the procedure of recombination described above. **(2p)**

```
recombine <- function(genome1, genome2) {
  L <- length(genome1)
  crossover_point <- sample.int(L-1,1)
  new_genome <- genome1
  new_genome[(crossover_point+1):L] <- genome2[(crossover_point+1):L]
  return(new_genome)
}
```

c) Write a script in R that uses the above function and simulates the population for 100 generations. For your own convenience, write it such that you can easily change the population size *n* and the genome size *L*. **(2p)**

```
# run introgression model
n <- 100
L <- 100
tmax <- 100
source('recombine.R')
# generate starting population
pop <- matrix(0,nrow=n,ncol=L)
pop[1,] <- 1
# saved proportions of new genes:
prop_new <- rep(0,tmax)
# run for tmax generations
for (t in 1:tmax) {
  # Each generation:
  # empty new generation:
  new_pop <- matrix(0,nrow=n,ncol=L)
  for (oi in 1:n) {
    # choose parents randomly
    parents <- sample.int(n,2,replace=FALSE)
    # recombine their genomes to an offspring
    offspring <- recombine(pop[parents[1],], pop[parents[2],])
    # insert offspring in next generation
    new_pop[oi,] <- offspring
    #alternative: new_pop <- rbind(new_pop, offspring)
    # until complete next generation
  }
  pop <- new_pop
  # store current proportion of new genes:
  prop_new[t] <- sum(pop)/(n*L)
}
plot(1:tmax, prop_new,type='l')
```



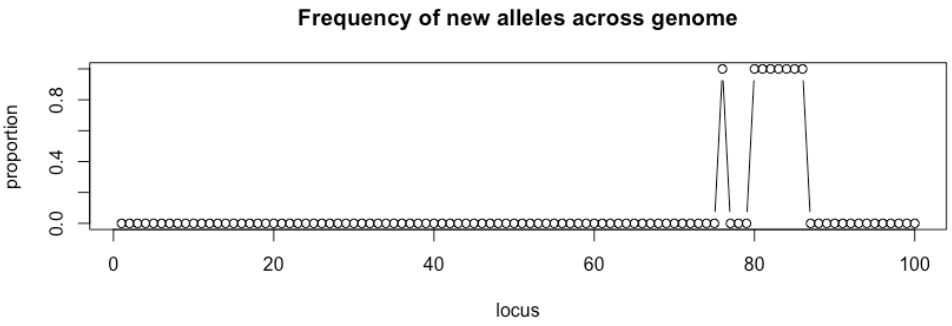
e) *Extra (no points!)*: After a long time, say 1000 generations, most new genes (out of the original *L*) have either gone extinct or come to fixation. The population now has a mixed genome of old and new genes (like non-African humans). Add a few lines to your script to also plot the frequency of new genes at each locus of the final population. You may find short sequences of introgressed genes spread out over the genome, but usually they occur together, side-by-side. Can you think of why? (0p)

Nearby loci are often inherited together, due to how the recombination works. Consequently, they also become fixated together, with high probability.

New code without comments (only a couple of lines (red) are different).

```
n <- 100
L <- 100
tmax <- 1000
source('recombine.R')
pop <- matrix(0,nrow=n,ncol=L)
pop[1,] <- 1
prop_new <- rep(0,tmax)
for (t in 1:tmax) {
  new_pop <- matrix(0,nrow=n,ncol=L)
  for (oi in 1:n) {
    parents <- sample.int(n,2,replace=FALSE)
    offspring <- recombine(pop[parents[1],], pop[parents[2],])
    new_pop[oi,] <- offspring
  }
  pop <- new_pop
  prop_new[t] <- sum(pop)/(n*L)
}
frequencies <- colSums(pop)/n
plot(1:n, frequencies, type='b', xlab='locus',
ylab='proportion',main='Frequency of new alleles across genome')
```

Sample output:


Frequency of new alleles across genome
proportion
0.0
0.4
0.8
0
20
40
60
80
100
locus