Exercises2

Yongqi WANG, Hangjia ZHAO

Problem 1

a)

Since there 20 unique amino acids, there exist 20^{20} unique amino acid sequences for a sequence of $L = 20$.

b)

Codons consist of 3 DNA sequences is responsible for a single amino acid sequence, the DNA sequence encoding the amino acids sequence is of length of $20 \times 3 = 60$ and has 60^4 uniques sequences.

Problem 2

a)

Let x, y be two binary sequences of length L, $P(z_i = 1) = P(z_i = 1) = \frac{1}{2}, z \in \{x, y\}, i \in \{0, 1, \ldots, L\}$.

Let a, b be two DNA sequences of length L, $P(c_i = A) = P(c_i = T) = P(c_i = C) = P(c_i = G) = \frac{1}{4}, c \in \{a, b\}, i \in \{0, 1, \ldots, L\}$

Binary:

$$E(d_H(x, y)) = E(\sum_{i=1}^{L} 1_{x_i \neq y_i}) = \sum_{i=1}^{L} E(1_{x_i \neq y_i}) = L \cdot P(x_i \neq y_i) = \frac{L}{2}$$

Random DNA:

$$E(d_H(a, b)) = E(\sum_{i=1}^{L} 1_{a_i \neq b_i}) = \sum_{i=1}^{L} E(1_{x_i \neq y_i}) = L \cdot P(a_i \neq b_i) = \frac{3L}{4}$$

b)

Binary: $\binom{L}{K}$ of sequences at a Hamming distance K since the size of $\mathcal{A}$ is 2, $\binom{L}{2}$ sequences at a Hamming distance of 2.

Random DNA: $\binom{L}{K} \cdot 3^K$ because of the increase size in the alphabet $\mathcal{A}$ to 4.

Problem 3

a)

$$W = \begin{bmatrix} f_0 q_{00} & f_1 q_{10} \\ f_0 q_{01} & f_1 q_{11} \end{bmatrix} = \begin{bmatrix} f_0 q & 1 - q \\ f_0(1 - q) & q \end{bmatrix}$$

The eigenvalues of the above matrix by solving $det(W - \lambda I) = 0$:

$$\lambda_{1,2} = \frac{f_0(q + 1) \pm \sqrt{(f_0(q + 1))^2 - 4 f_0(2q - 1)}}{2}$$

b)

Perron-Frobenius theorem states that

c)

Assumed $f_0 = 1.5 > 1, q = 0.9$

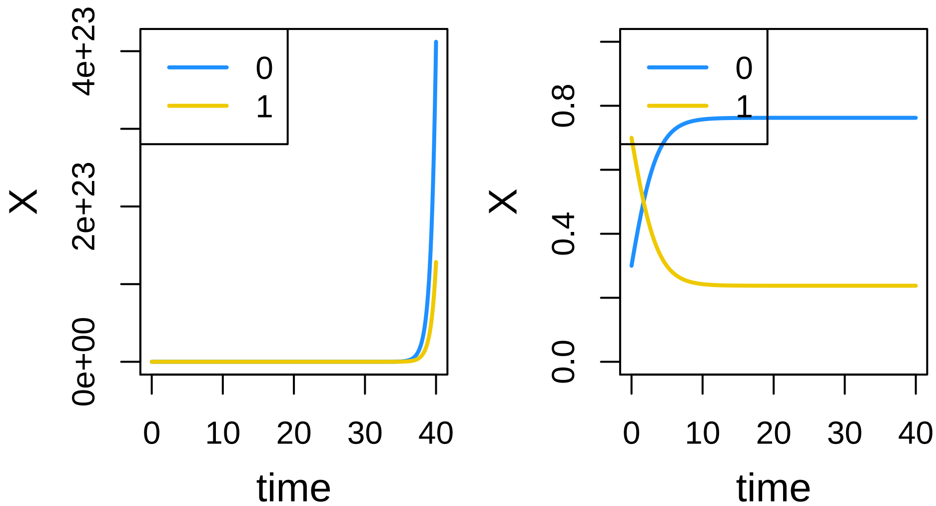
```r
library(matlib)
f0 = 1.5
q = 0.9
parms <- c(f0 = f0, f1 = 1, q_00 = q, q_01 = 1-q, q_10 = 1-q, q_11 = q)
times <- c(0:1000)/25
initconds <- c(a = 0.3, b = 0.7)
M = matrix(c(parms["f0"] * parms["q_00"], parms["f0"] * parms["q_01"],
             parms["f1"] * parms["q_10"], parms["f1"] * parms["q_11"]),
           2, 2)
eig = eigen(M)
#M
#eig

x_star = eig$vectors[, 1]/sum(eig$vectors[, 1])
x_star

sum(x_star * c(parms["f0"], parms["f1"]))

ivp = solve(eig$vectors, initconds)
Xa = ivp[1] * eig$vectors[1, 1] * exp(eig$values[1] * times) +
ivp[2] * eig$vectors[1, 2] * exp(eig$values[2] * times)
Xb = ivp[1] * eig$vectors[2, 1] * exp(eig$values[1] * times) +
ivp[2] * eig$vectors[2, 2] * exp(eig$values[2] * times)

par(mar = c(4, 4, 2, 0.5)) # margin size
par(mgp = c(2.5, 1, 0)) # axis localtion
par(cex.lab = 1.25) # size of y axis label
par(mfrow=c(1,2)) # 1x2 fig
plot(times, Xa, xlab = "time", ylab = expression(X), main = "",
     col = "dodgerblue", type = "l", lwd = 2)
lines(times, Xb, col = "gold2", lwd = 2)
legend("topleft", legend=c("0", "1"),
       col=c("dodgerblue", "gold2"), lty=1, lwd = 2)
plot(times, Xa/(Xa + Xb), ylim = c(0, 1), xlab = "time", ylab = expression(X),
     main = "", col = "dodgerblue", type = "l", lwd = 2)
lines(times, Xb/(Xa + Xb), col = "gold2", lwd = 2)
legend("topleft", legend=c("0", "1"),
       col=c("dodgerblue", "gold2"), lty=1, lwd = 2)
```



d)

By b) we know that the equilibrium point for $f_0 = f_1 = 1$ would be

$$\lambda = \frac{(q+1) + \sqrt{(q+1)^2 - 4(2q-1)}}{2}$$

e)

$$\lambda = \frac{2f_0 + \sqrt{4f_0^2 - 4f_0}}{2}$$

Code is available on github repo: (https://github.com/wyq977/evolutionary-dynamics-2019)