ライフゲームによる食物連鎖のシミュレーション

班員　入口 崇紀、川上 拓海、小林 嵩門、高村 晃拓
担当教員　福光 英徳

キーワード：食物連鎖、ライフゲーム、シミュレーション

While the food chain is too complicated to simulate, we made a model limiting the number of species to 3. We compared the result of the simulation on the model with that of the equation we made based on the Lotka-Volterra equation.

１　はじめに

ライフゲームとは、決められたルールによって生物の誕生、淘汰などを簡易的に行えるシミュレーションである。本研究では３種類に限定した生物のシミュレーションを行うためにライフゲームのルールの拡張を試みた。

２　方法

(1) ライフゲームのルールの拡張

ライフゲームに被食者・一次捕食者・二次捕食者とエネルギーのルールを定義した「Life_Plus」というシミュレーションを作成し、実行するためにGollyというソフトを用いた。

《Life_Plusのルール》

〈被食者のルール〉

・エネルギー値は常に１。
・周りの８セル中の生存セルの個体数が０～１，４～８体なら死亡する。
・周りの８セル中に一次捕食者が１つでもあると死亡する。
◎周りの８セル中に二次捕食者が１つでもあると死亡する。

〈一次捕食者のルール〉

・エネルギー値は０～３。
・周りの８セル中の４～８セルが一次捕食者なら死亡する。
◎周りの８セル中に二次捕食者が１つでもあったら死亡する。
・エネルギー値が１世代ごとに１減る。
・エネルギー値が０になると死亡する。
・周りの８セル中の被食者の個体数だけエネルギー値が増加する。

〈二次捕食者のルール〉

◎エネルギー値は０～５。
◎周りの８セル中の４～８セルが二次捕食者なら死亡する。
◎エネルギー値が１世代ごとに１減る。
◎エネルギー値が０になると死亡する。
◎周りの８セル中の被食者と一次捕食者の合計エネルギー値の分だけエネルギー値が増加する。

〈死亡セルのルール〉

・エネルギー値は常に０。
・周りの８セル中３セルが被食者なら被食者になる。最優先される。
・周りの８セル中３セルが一次捕食者ならエネルギー値が２の一次捕食者になる。優先される。
◎周りの８セル中３セルが二次捕食者ならエネルギー値が２の二次捕食者になる。

(2)Life_Plusの２種での有効性の検証

３種は複雑であることから、最初に被食者と一次捕食者の２種でのシミュレーションを行った。２種のシミュレーションについて、以下のとおり２つの条件で行った（表１）。なお、このシミュレーションではルールの◎の部分は使用しない。

表１：シミュレーションの条件

	実行範囲	死亡	被食者	一次捕食者
1	60×60	4	4	1
2	60×60	4	1	4

エクセルで、Life_Plusのランダムなパターンを生成するプログラムと、Life_Plusを実行するプログラムを作成し、データ収集をできるようにした。それを用いて被食者と一次捕食者だけのパターンのシミュレーション結果を収集した。

次にロジスティック方程式(式１)とロトカ・ヴォルテラ方程式(式２)を混合した方程式(式３)での計算結果を求めた。

式１：ロジスティック方程式

$$\mathrm{N(t)} = \frac{N_o K e^{rt}}{K - N_0 + N_0 e^{rt}}$$

式２：ロトカ・ヴォルテラ方程式

$$\begin{cases} \dfrac{\mathrm{d}N_1}{\mathrm{d}t} = C_1 N_1 - C_1 N_1 N_2 \\ \dfrac{\mathrm{d}N_2}{\mathrm{d}t} = C_3 N_1 N_2 - C_4 N_2 \end{cases}$$

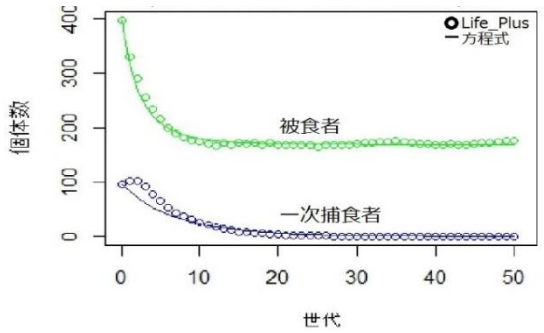
式３：混合方程式

$$\begin{cases} \dfrac{\mathrm{d}N_1}{\mathrm{d}t} = C_1 N_1 \left(1 - \dfrac{N_1}{K}\right) - \dfrac{C_2 N_1 N_2}{1 + hN_1} \\ \dfrac{\mathrm{d}N_2}{\mathrm{d}t} = \dfrac{c_3 N_1 N_2}{1 + hN_1} - c_4 N_2 \end{cases}$$

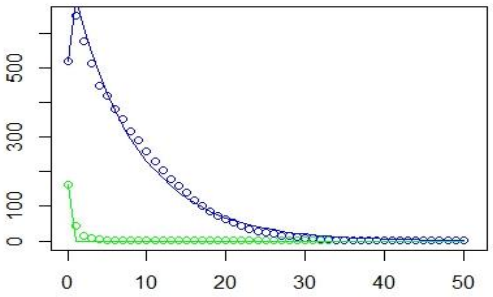
シミュレーションと計算の結果が一致すれば、２種での有効性が示せると考えられる。そのため、結果が一致する式３の係数があるかどうかR-studioで勾配法(ソースコード１)を用いて調べた。

その結果をグラフで示した。

なお、グラフの○がシミュレーション結果、曲線が計算結果を示し、緑が被食者、青が一次捕食者のデータを示している。



シミュレーション１と方程式



シミュレーション２と方程式

グラフがほぼ一致することから、Life_Plusは被食･捕食関係にある２種の生物のシミュレーションとして有効であるといえる。

(3)Life_Plusの３種での有効性の検証

Life_Plusを用いた被食者と一次捕食者、二次捕食者の３種でのシミュレーションを行い、結果を収集した。

計算については、式３を３種に拡張した方程式(式４)を考案し、計算結果を求めた。

式４：式３を３種に拡張した方程式

$$\begin{cases} \dfrac{\mathrm{d}N_1}{\mathrm{d}t} = C_1 N_1 \left(1 - \dfrac{N_1}{K}\right) - \dfrac{C_2 N_1 N_2}{1 + h_1 N_1} - \dfrac{C_3 N_1 N_3}{1 + h_2 N_1} \\ \dfrac{\mathrm{d}N_2}{\mathrm{d}t} = \dfrac{c_4 N_1 N_2}{1 + h_1 N_1} - \dfrac{c_5 N_2 N_3}{1 + h_2 N_2} - c_6 N_2 \\ \dfrac{\mathrm{d}N_3}{\mathrm{d}t} = \dfrac{c_7 N_1 N_3}{1 + h_2 N_1} + \dfrac{c_8 N_2 N_3}{1 + h_2 N_2} - c_9 N_3 \end{cases}$$

シミュレーションと計算の結果を比較したところ、２種のときほどの一致が見られなかった。

このことから、Life_Plusの有効性については、不明である。今後の検討が必要である。

３　結論

Life_Plusは２種でのシミュレーションについては有効であると考えられるが、３種では更なる検討が必要である。

４　参考文献

巌佐　庸（1998）数理生物学入門－生物社会のダイナミックスを探る.共立出版株式会社

Golly Game of Life Home Page（2017）
http://golly.sourceforge.net/

ライフゲーム－Wikipedia（2017）
https://ja.wikipedia.org/wiki/ライフゲーム

ソースコード１：２種での検証に用いた
※使用するcsvファイルの名前は
“ファイル名ｍ”（ｍ＝１，２，３，・・・）
の形にすること
※ページの都合により一部改行がおかしくなっている。

```
setwd("ファイルのある場所")
#読み込むファイルの数を指定
x =
#世代数
nt =
#読み込むファイルの名前
name <- "名前を入れる"
for(i in 1:x){
  s        <-        paste("table",i,"        <-
read.csv('",name,i,".csv')",sep = "")
  eval(parse(text = s))
}

for(i in 1:x){
  sp  <-  paste("p",i,"  <-  ","c(table",i,"$被食者)",sep = "")
  st <- paste("pp",i," <- ","c(table",i,"$一次捕食者合計)",sep = "")
  eval(parse(text =sp))
  eval(parse(text =st))
}
d1 <- p1[1]
d2 <- pp1[1]
for(i in 2:x){
  sd1 <- paste("d1"," <- c(d1",",p",i,"[1])",sep = "")
  sd2 <- paste("d2"," <- c(d2",",pp",i,"[1])",sep = "")
  eval(parse(text = sd1))
  eval(parse(text = sd2))
}
p <- mean(d1)
pp <- mean(d2)
for(i in 2:nt){
  se1 <- paste("d1"," <- c(p1[",i,"])",sep = "")
  se2 <- paste("d2"," <- c(pp1[",i,"])",sep = "")
  eval(parse(text = se1))
  eval(parse(text = se2))
  for(ii in 2:x){
    sd1        <-        paste("d1","        <-
c(d1",",p",ii,"[",i,"])",sep = "")
    sd2        <-        paste("d2","        <-
c(d2",",pp",ii,"[",i,"])",sep = "")
    eval(parse(text = sd1))
    eval(parse(text = sd2))
  }
  p <- c(p,mean(d1))
  pp <- c(pp,mean(d2))
}
p <- c(p,p[nt]); pp <- c(pp,pp[nt])
t <- seq(0,nt,1)

n <- p[1]
e <- exp(1)
rp <- c(p,pp)

f <- c(a=1,b=0.6,c=1,d=0.2,k=200,h=0.2)
f1 <- f; f2 <- f

change <- c(1,0.1,0.01,0.001)

for(kurikaesi in 1:6){
for(I in 1:6){
  for(nc in change){
    library(deSolve)
    parameters <- f
    initial <- c(x= p[1],y= pp[1])
    times <- seq(0, nt, 1)
    rotoka <- function(t, state, parameters){
      with(as.list(c(state, parameters)),{
        dx <- a*x*(1-(x/k))-(b*x*y/(1+h*x))
        dy <- (c*x*y/(1+h*x))-d*y
        list(c(dx, dy))
      })
    }
    out0 <- ode(y = initial, times = times, func
= rotoka, parms = parameters)
    op0 <- c(out0[,2]); opp0 <- c(out0[,3])
    outp0 <- abs(p - op0); outpp0 <- abs(pp -
opp0)
    if(length(op0) < nt + 1){
      l0 <- length(op0) - 2
        library(deSolve)
        parameters <- f
        initial <- c(x= p[1],y= pp[1])
        times <- seq(0, l0, 1)
        rotoka     <-     function(t,     state,
parameters){
          with(as.list(c(state, parameters)),{
```

```
            dx <- a*x*(1-(x/k))-(b*x*y/(1+h*x))
            dy <- (c*x*y/(1+h*x))-d*y
            list(c(dx, dy))
          })
        }
        out0 <- ode(y = initial, times = times,
func = rotoka, parms = parameters)
        op0 <- c(out0[,2]); opp0 <- c(out0[,3])
        hp <- p[1:10]; hpp <- pp[1:10]
        outp0 <- abs(hp - op0); outpp0 <- abs(hpp
- opp0)
    }

    g0 <- sum(outp0,outpp0)#初期値の誤差を出す
    gosa1 <- -1*g0; gosa2 <- g0; gosa3 <- g0; G
<- abs(gosa3 - gosa1)

    while(G > 0.0001){
      f1[I] <- f1[I] + nc
      library(deSolve)
      parameters <- f1
      initial <- c(x= p[1],y= pp[1])
      times <- seq(0, nt, 1)
      rotoka <- function(t, state, parameters){
        with(as.list(c(state, parameters)),{
          dx <- a*x*(1-(x/k))-(b*x*y/(1+h*x))
          dy <- (c*x*y/(1+h*x))-d*y
          list(c(dx, dy))
        })
      }
      out1 <- ode(y = initial, times = times,
func = rotoka, parms = parameters)
      op1 <- c(out1[,2]); opp1 <- c(out1[,3])
      l1 <- length(op1)
      if(l1 < nt + 1){
        l1 <- l1 -2
        library(deSolve)
        parameters <- f1
        initial <- c(x= p[1],y= pp[1])
        times <- seq(0, l1, 1)
        rotoka     <-     function(t,     state,
parameters){
          with(as.list(c(state, parameters)),{
            dx <- a*x*(1-(x/k))-(b*x*y/(1+h*x))
            dy <- (c*x*y/(1+h*x))-d*y
            list(c(dx, dy))
          })
        }
        out1 <- ode(y = initial, times = times,
func = rotoka, parms = parameters)
        op1 <- c(out1[,2]); opp1 <- c(out1[,3])
      }

      f2[I] <- abs(f2[I] - nc)
      library(deSolve)
      parameters <- f2
      initial <- c(x= p[1],y= pp[1])
      times <- seq(0, nt, 1)
      rotoka <- function(t, state, parameters){
        with(as.list(c(state, parameters)),{
          dx <- a*x*(1-(x/k))-(b*x*y/(1+h*x))
          dy <- (c*x*y/(1+h*x))-d*y
          list(c(dx, dy))
        })
      }
      out2 <- ode(y = initial, times = times,
func = rotoka, parms = parameters)
      op2 <- c(out2[,2]); opp2 <- c(out2[,3])

      l2 <- length(op2)
      if(l2 < nt + 1){
        l2 <- l2 - 2
        library(deSolve)
        parameters <- f2
        initial <- c(x= p[1],y= pp[1])
        times <- seq(0, l2, 1)
        rotoka     <-     function(t,     state,
parameters){
          with(as.list(c(state, parameters)),{
            dx <- a*x*(1-(x/k))-(b*x*y/(1+h*x))
            dy <- (c*x*y/(1+h*x))-d*y
            list(c(dx, dy))
          })
        }
        out2 <- ode(y = initial, times = times,
func = rotoka, parms = parameters)
        op2 <- c(out2[,2]); opp2 <- c(out2[,3])
      }

      if(l1 <= l2){
        hp <- p[1:l1]; hpp <- pp[1:l1]; op2 <-
op2[1:l1]; opp2 <- opp2[1:l1]
      }else{
        hp <- p[1:l2]; hpp <- pp[1:l2]; op1 <-
op1[1:l2]; opp1 <- opp1[1:l2]
      }

      outp1 <- abs(hp - op1); outpp1 <- abs(hpp
- opp1); outp2 <- abs(hp - op2); outpp2 <- abs(hpp
- opp2)
      g1 <- sum(outp1,outpp1)
      g2 <- sum(outp2,outpp2)

      gosa3 <- gosa2
      gosa2 <- gosa1

      if(abs(g1 - g2) < 0.0001){
          gosa1 <- g0
          out <- out0
          f1 <- f; f2 <- f
      }else if(g1 > g2){
        f <- f2; f1 <- f2
        gosa1 <- g2
        out <- out2
      }else{
        f <- f1; f2 <- f1
        gosa1 <- g1
        out <- out1
      }
      G <- abs(gosa3 - gosa1)
    }
  }
}
}
popp <- c(out[,2])
pop1p <- c(out[,3])
len <- 0:length(popp)
# 散布図
plot(t, p, xlim = range(len), ylim = range(rp),lwd
= 1, xlab = "世代", ylab = "個体数",col
="green",pch = 1)
points(t, pp, xlim = range(len), ylim =
range(rp),lwd = 1,col ="blue",pch = 1)
points(times,popp, xlim = range(len), ylim =
range(rp),lwd = 1,col ="green",type = "l")
points(times,pop1p, xlim = range(len), ylim =
range(rp),lwd = 1,col ="blue",type = "l"
```