生物資源解析学演習 Lecture-1: Handout

Toshihide Kitakado

2019 年 *10* 月 *1* 日

目次

1 R 事はじめ

1.1 Pella-Tomlinson 余剰生産モデル (確率変動なし)

1.1.1 モデルの定義

$$P_{t+1} = P_t + rP_t\left\{1 - \left(\frac{P_t}{K}\right)^z\right\} - C_t$$

1.1.2 初歩的に漸化式を利用して個体群動態を計算

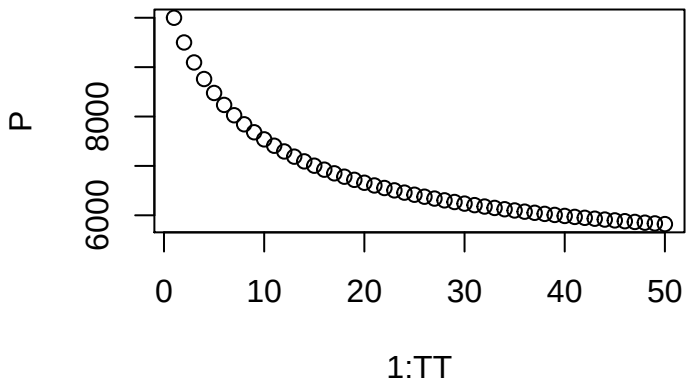
```
r <- 0.2
K <- 10000
z <- 1
TT <- 50
```

```
Catch <- rep(500, TT)

TT1 <- TT-1
P <- numeric(TT)
P[1] <- K
for(t in 1:TT1)
{
 tmp <- P[t] + r*P[t]*(1-(P[t]/K)^z) - Catch[t]
 P[t+1] <- max(tmp, 0.001)
}

print(P, digits=0)
```

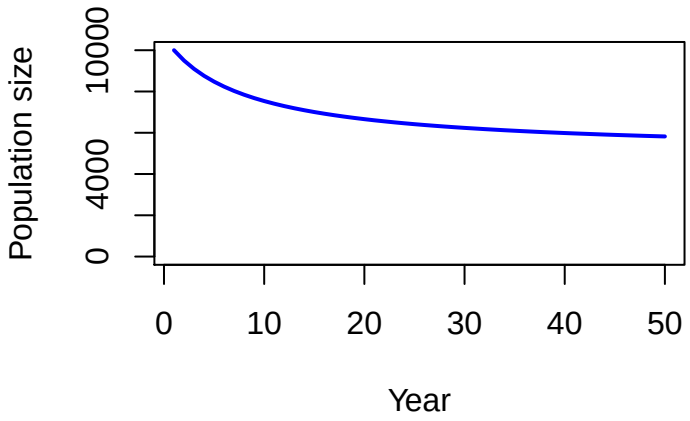
```
##  [1]10000 9500 9095 8760 8477 8235 8026 7843 7681 7537 7409 7293 7187 7092
## [15] 7004 6924 6850 6781 6718 6659 6604 6552 6504 6459 6416 6376 6338 6303
## [29] 6269 6236 6206 6177 6149 6123 6097 6073 6050 6028 6007 5987 5967 5949
## [43] 5931 5913 5897 5881 5865 5850 5836 5822
```

1.1.3 計算結果の図示

```
plot(1:TT, P)
```



```
plot(1:TT, P, type="l", lwd=2, col="blue", xlab="Year", ylab="Population size", ylim=c(0,K))
```



1.1.4　もう少し進んだ方法：個体群動態計算用関数 PDM.PT を作成

```
#Pella-Tomlinson Production model (without stochastic process error)
PDM.PT<-function(r, K, z, P1, TT, Catch)
{
 TT1 <- TT-1
 P <- numeric(TT)
 P[1] <- P1
 for(t in 1:TT1)
 {
  tmp <- P[t] + r*P[t]*(1-(P[t]/K)^z) - Catch[t]
  P[t+1] <- max(tmp, 0.001)
 }
 return(P)
}
```

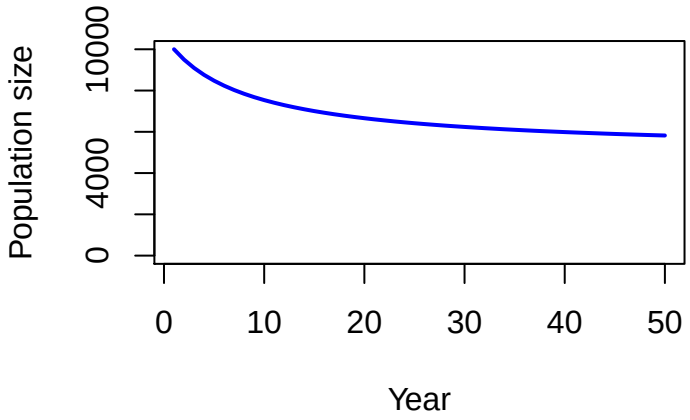
1.1.5　新しく作った個体群動態計算用関数 PDM.PT を実行

```
Res <- PDM.PT(r=0.2, K=10^4, z=1, P1=10^4, TT=50, Catch=rep(500,TT))
print(Res, digits=0)
```

```
##  [1]10000 9500 9095 8760 8477 8235 8026 7843 7681 7537 7409 7293 7187 7092
## [15] 7004 6924 6850 6781 6718 6659 6604 6552 6504 6459 6416 6376 6338 6303
```

```
## [29] 6269 6236 6206 6177 6149 6123 6097 6073 6050 6028 6007 5987 5967 5949
## [43] 5931 5913 5897 5881 5865 5850 5836 5822
```

```r
plot(1:TT, Res, type="l", lwd=2, col="blue", xlab="Year", ylab="Population size", ylim=c(0,K))
```



1.2 Pella-Tomlinson 余剰生産モデル (確率変動あり)

1.2.1 モデルの定義

$$P_{t+1} = \left[P_t + rP_t \left\{ 1 - \left(\frac{P_t}{K} \right)^z \right\} - C_t \right] * \exp(\varepsilon_t), \quad \varepsilon_t \sim N(0, \sigma^2)$$

1.2.2 さっき作った個体群動態計算用関数 PDM.PT を書き換え（上書き更新）

```r
#Pella-Tomlinson Production model (with stochastic process error)
PDM.PT<-function(r, K, z, P1, TT, Catch, sigma=0)
{
 TT1 <- TT-1
 P <- numeric(TT)
 Epsilon <- rnorm(n=TT, mean=0, sd=sigma)
 P[1] <- P1
 for(t in 1:TT1)
 {
  tmp <- P[t] + r*P[t]*(1-(P[t]/K)^z) - Catch[t]
  tmp <- tmp*exp(Epsilon[t])
```

```
  P[t+1] <- max(tmp, 0.001)
 }
 return(P)
}
```

1.2.3 再定義した PDM.PT を実行

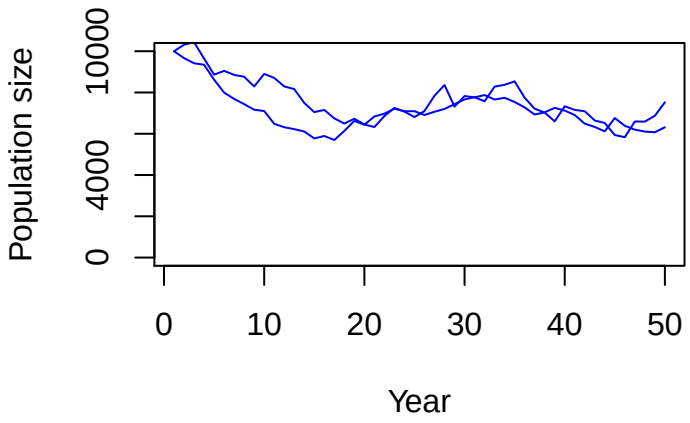
```
K <- 10^4
TT <- 50

Res <- PDM.PT(r=0.2, K=K, z=1, P1=K, TT=50, Catch=rep(500,TT), sigma=0.05)
print(Res, digits=0)
```

```
##  [1]1.000000e+041.031626e+041.043654e+049.644093e+038.863482e+039.048655e+038.853838e+038.761868e
## [15]7.049893e+037.153212e+036.747508e+036.495007e+036.729025e+036.449974e+036.325127e+036.858247e
## [29]7.316171e+037.829378e+037.762402e+037.876090e+037.660499e+037.741160e+037.540567e+037.276958e
## [43]6.330595e+036.116989e+036.760985e+036.386130e+036.198847e+036.106194e+036.072833e+036.314825e
```

```
plot(1:TT, Res, type="l", col="blue", xlab="Year", ylab="Population size", ylim=c(0,K))

Res <- PDM.PT(r=0.2, K=K, z=1, P1=K, TT=50, Catch=rep(500,TT), sigma=0.05)
print(Res, digits=0)
```

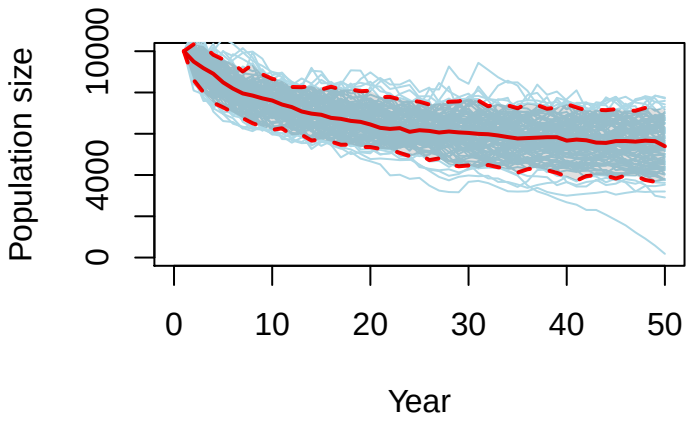
```
##  [1]10000 9666 9416 9345 8619 7994 7689 7434 7165 7101 6482 6316 6223 6108
## [15] 5773 5893 5697 6138 6624 6437 6838 6974 7216 7090 7094 6910 7065 7202
## [29] 7435 7665 7769 7574 8285 8372 8540 7745 7215 7025 6600 7331 7156 7091
## [43] 6642 6521 5937 5834 6594 6589 6872 7525
```

```
points(1:TT, Res, type="l", col="blue")
```



1.2.4 PDM.PT を繰り返し実行して結果を図示

```
Nsim <- 100
Pmat <- array(NA, c(Nsim, TT))
plot(0, type="n", xlab="Year", ylab="Population size", xlim=c(0,TT),  ylim=c(0,K))

for(i in 1:Nsim)
{
 Pmat[i,] <- PDM.PT(0.2, 10^4, 1, 10^4, 50, rep(500,TT), 0.05)
 points(1:TT, Pmat[i,], type="l", col="lightblue")
}
P.med <- apply(Pmat, 2, median);
P.L5per <- apply(Pmat, 2, quantile, 0.05)
P.U5per <- apply(Pmat, 2, quantile, 0.95)
points(1:TT, P.med, type="l", lwd=2, col="red")
points(1:TT, P.L5per, type="l", lty=2, lwd=2, col="red")
points(1:TT, P.U5per, type="l", lty=2, lwd=2, col="red")

polygon(
  c(1:TT, TT:1), c(P.L5per, rev(P.U5per)),
col = "#00000020", border = NA)
```



1.2.5 上記を関数化

```
PDM.PT.sim <- function(r, K, z, P1, TT, Catch, sigma=0, Nsim)
{
 Pmat <- array(NA, c(Nsim, TT))
 plot(0, type="n", xlab="Year", ylab="Population size",
      xlim=c(0,TT), ylim=c(0,K), xaxs="i", yaxs="i")

 for(i in 1:Nsim)
 {
  Pmat[i,] <- PDM.PT(0.2, 10^4, 1, 10^4, 50, rep(500,TT), 0.05)
  points(1:TT, Pmat[i,], type="l", col="lightblue")
 }
 P.med <- apply(Pmat, 2, median);
 P.L5per <- apply(Pmat, 2, quantile, 0.05)
 P.U5per <- apply(Pmat, 2, quantile, 0.95)
 points(1:TT, P.med, type="l", lwd=2, col="red")
 points(1:TT, P.L5per, type="l", lty=2, lwd=2, col="red")
 points(1:TT, P.U5per, type="l", lty=2, lwd=2, col="red")

 polygon( c(1:TT, TT:1), c(P.L5per, rev(P.U5per)), col = "#00000020", border = NA)

 rect(xleft=-1, ybottom=0, xright=TT, ytop=0.3*K, lwd=0, col=rgb(1,0,0,alpha=0.2))
 rect(xleft=-1, ybottom=0.3*K, xright=TT, ytop=0.5*K, lwd=0, col=rgb(1,1,0,alpha=0.2))
```

```
 rect(xleft=-1, ybottom=0.5*K, xright=TT, ytop=K, lwd=0, col=rgb(0,1,0,alpha=0.2))

 return(Pmat)
}
```

1.2.6 実行！

```
Res <- PDM.PT.sim(0.2, 10^4, 1, 10^4, 50, rep(500,TT), 0.05, Nsim=1000)
```