

Population growth

Daijiang Li

Contents

Exponential growth	1
Logistic growth	3
Matrix model	15

((use this lecture to discuss the concepts of stable and unstable equilibrium))

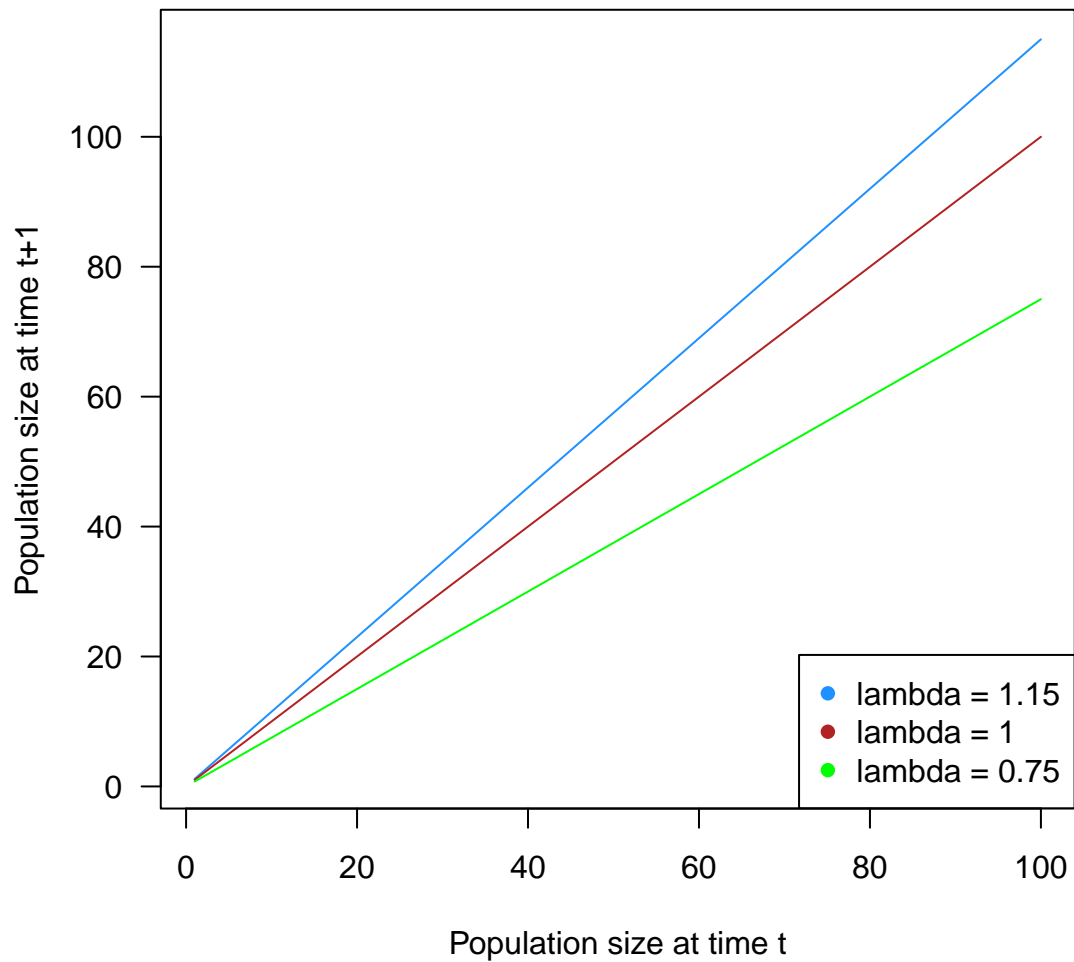
Exponential growth

```
expoGrowth <- function(n, lambda){
  n * lambda
}

n0 <- 1:100

# effect of growth rate
plot(x = n0, y = expoGrowth(n0, lambda = 1.15), type = 'l', las = 1,
     xlab = 'Population size at time t',
     ylab = 'Population size at time t+1', col = 'dodgerblue'
)
lines(n0, expoGrowth(n0, lambda = 1), col = 'firebrick')
lines(n0, expoGrowth(n0, lambda = 0.75), col = 'green')

legend('bottomright',
     paste(expression(lambda), c('= 1.15', '= 1', '= 0.75')),
     pch = 16, col = c('dodgerblue', 'firebrick', 'green'))
)
```

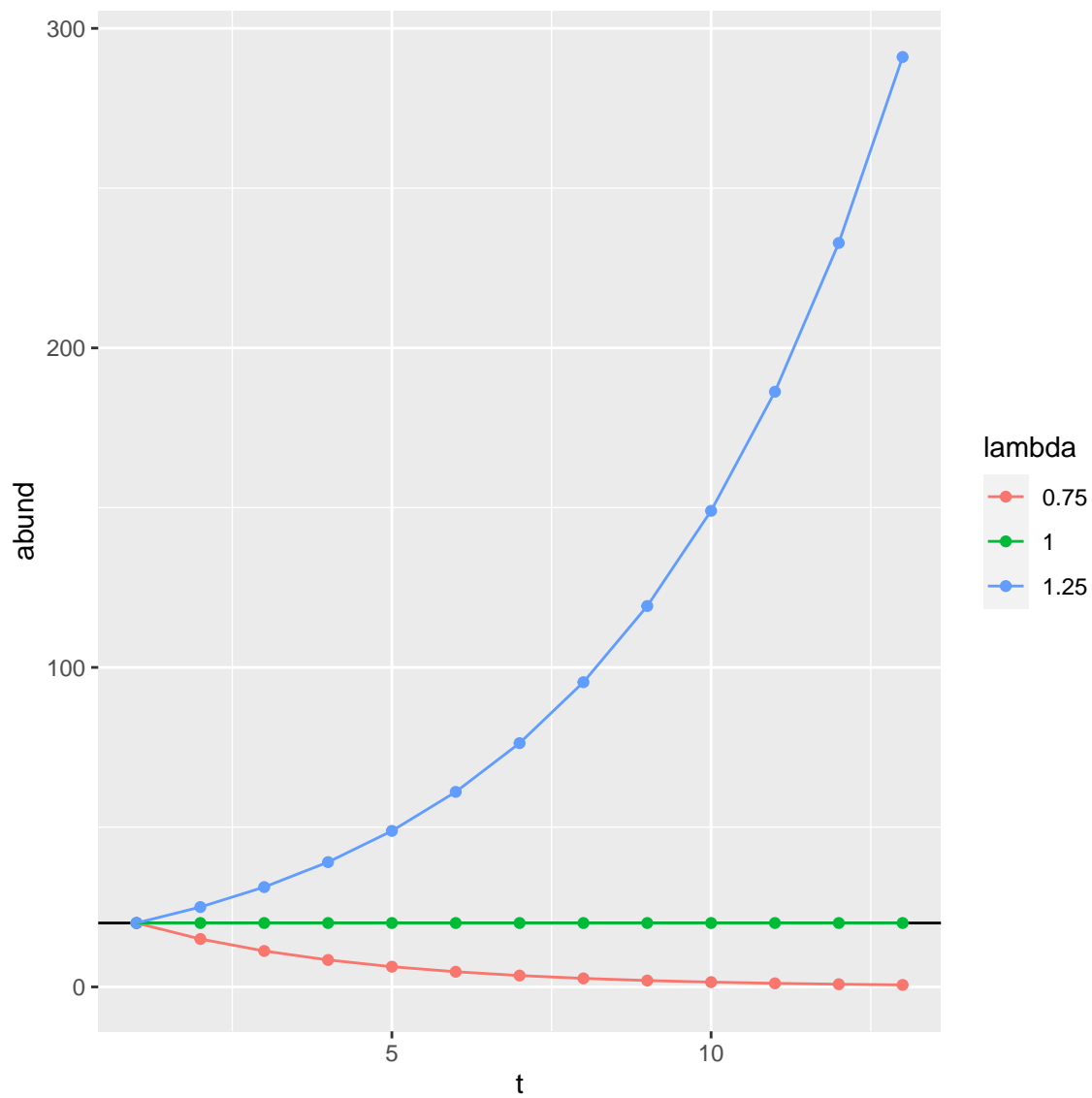


```
# exponential growth
expoDynamics <- function(n, lambda, steps = 100){
  ret <- c()
  ret[1] <- n
  for(i in 1:steps){
    ret[i+1] <- expoGrowth(ret[i], lambda)
  }
  return(ret)
}
```

```
library(ggplot2)
t <- 12
n0 <- 20
d <- data.frame(t = rep(1:(t + 1), 3),
  lambda = as.factor(rep(c(0.75, 1, 1.25), each = t + 1)),
  abund = c(
    expoDynamics(n0, lambda = 0.75, steps = t),
    expoDynamics(n0, lambda = 1, steps = t),
    expoDynamics(n0, lambda = 1.25, steps = t)
  ))

ggplot(data = d, aes(x = t, y = abund, color = lambda, group = lambda)) +
  geom_hline(yintercept = n0) +
```

```
geom_point() +  
geom_line()
```



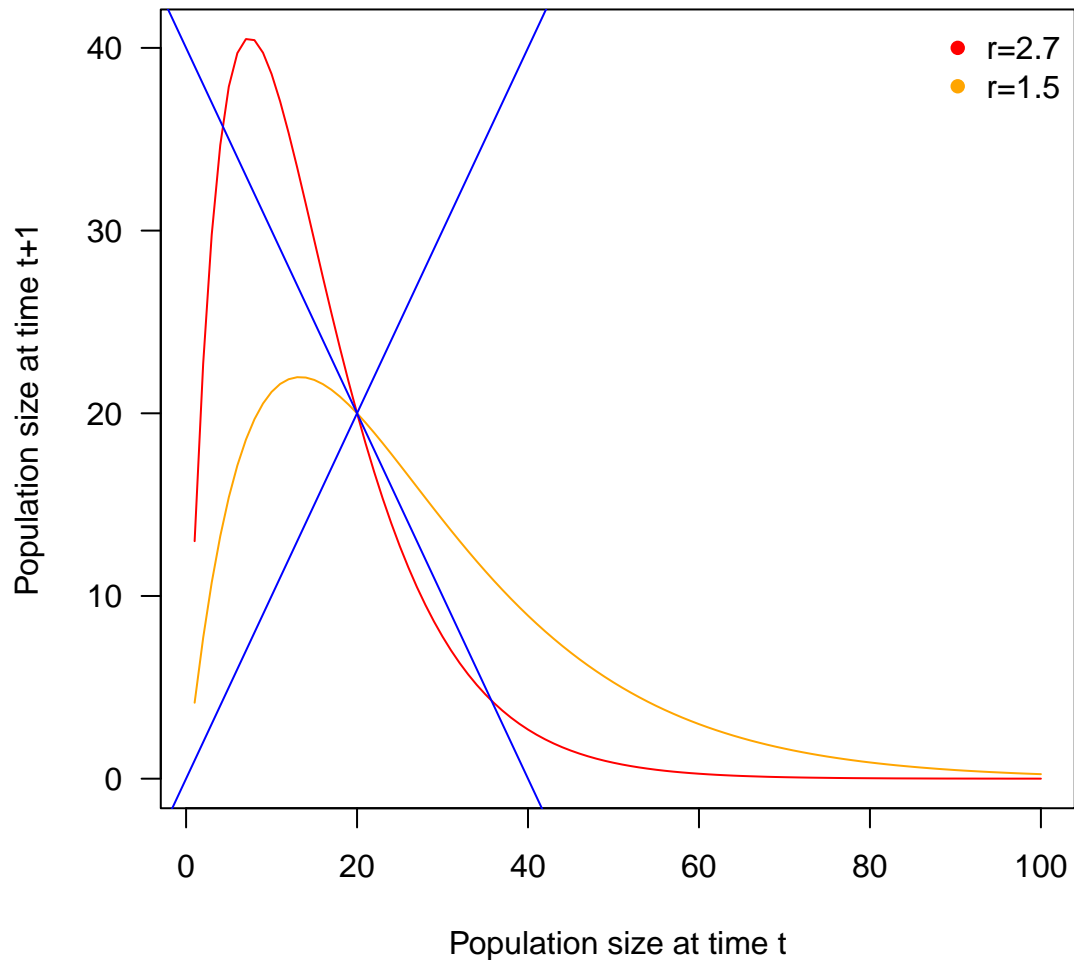
Logistic growth

```
n0 <- 1:100  
k <- 20  
r <- 0.5  
  
logisticGrowth <- function(n, r, k) {  
  n * exp(r * (1 - (n / k)))  
}  
  
colz <- c(grey(0.1, 0.9), 'dodgerblue', 'firebrick', 'forestgreen')  
#effect of growth rate
```

```

plot(n0, logisticGrowth(n = n0, r = 2.7, k = 20),
     type='l', las=1,
     xlab='Population size at time t',
     ylab='Population size at time t+1',
     col="red")
lines(n0, logisticGrowth(n = n0, r = 1.5, k = 20), col = "orange")
abline(a = 0, b = 1, col = "blue")
abline(a = 40, b = -1, col = "blue")
legend('topright', bty='n', c('r=2.7', 'r=1.5'),
      pch = 16, col = c("red", "orange"))

```



```

#effect of carrying capacity
plot(n0, logisticGrowth(n0,1, 50),
     type='l', las=1, ylim=c(0,100),
     xlab='Population size at time t',
     ylab='Population size at time t+1',
     col=colz[2])
lines(n0, logisticGrowth(n0,1,100),
     col=colz[1])
lines(n0, logisticGrowth(n0,1,10),
     col=colz[3])
legend('topright', bty='n',

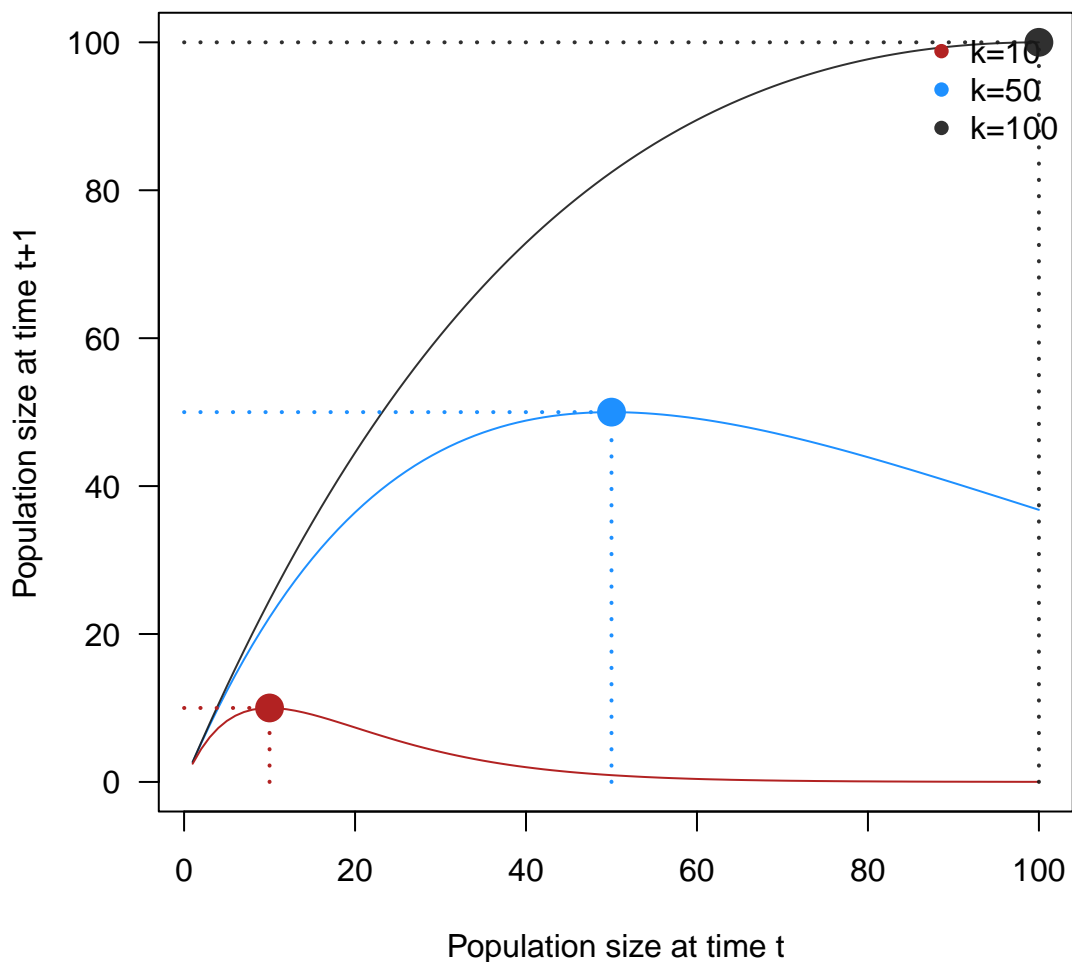
```

```
c('k=10', 'k=50', 'k=100'),
pch=16, col=colz[c(3,2,1)])
```

#Look at the peaks of the growth (where is the maximum population size here?)

```
plotSegs <- function(kx,ky, color){
  segments(x0=0,x1=kx, y0=ky,y1=ky, col=color, lwd=2, lty=3)
  segments(x0=kx,x1=kx, y0=0,y1=ky, col=color, lwd=2, lty=3)
  points(kx,ky, pch=16, cex=2, col=color)
}
```

```
plotSegs(10,10, color=colz[3])
plotSegs(50,50, color=colz[2])
plotSegs(100,100, color=colz[1])
```



But why is this? What situations would cause this to not be the case?

```
plot(n0, logisticGrowth(n0,1.5, 50),
     type='l', las=1, ylim=c(0,150),
     xlab='Population size at time t',
```

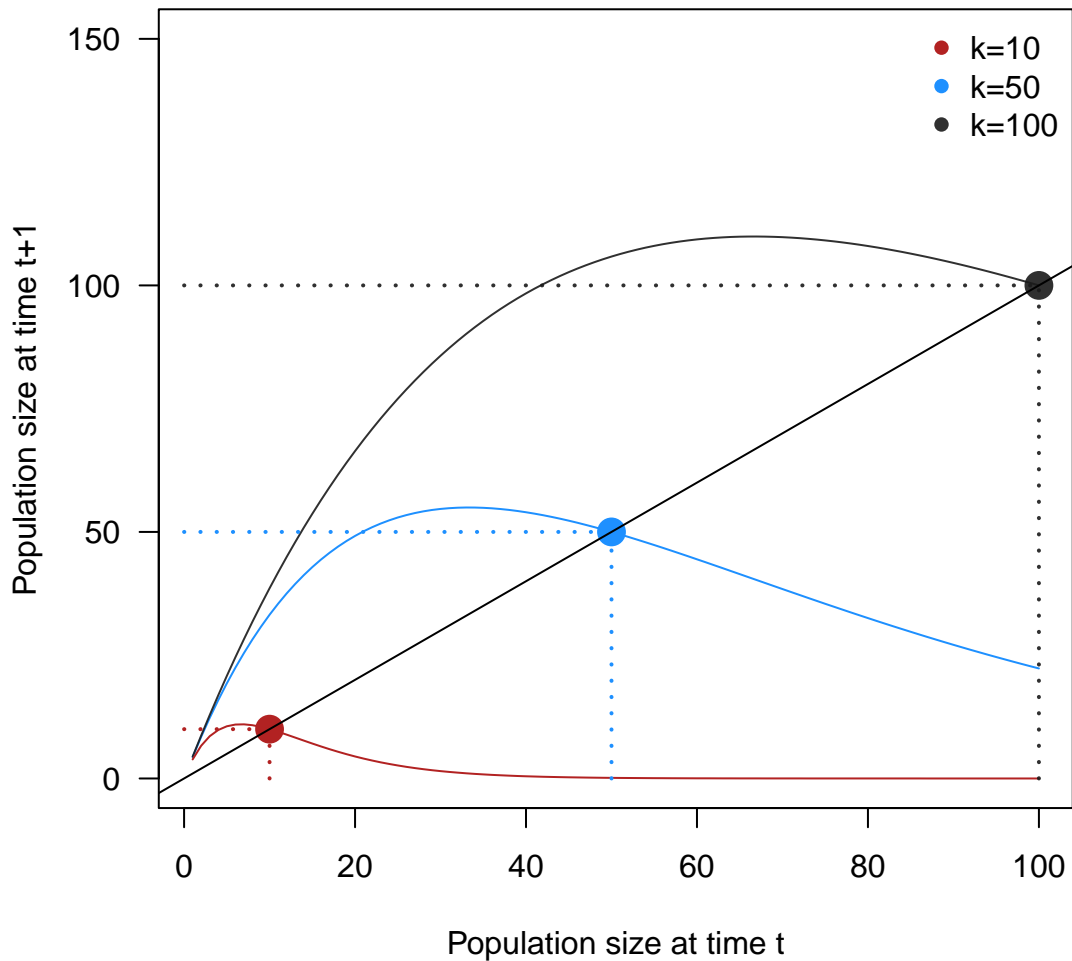
```

    ylab='Population size at time t+1',
    col=colz[2])
lines(n0, logisticGrowth(n0,1.5,100),
    col=colz[1])
lines(n0, logisticGrowth(n0,1.5,10),
    col=colz[3])
legend('topright', bty='n',
    c('k=10', 'k=50', 'k=100'),
    pch=16, col=colz[c(3,2,1)])

#Look at the peaks of the growth (where is the maximum population size here?)
plotSegs(10,10, color=colz[3])
plotSegs(50,50, color=colz[2])
plotSegs(100,100, color=colz[1])

# this line intersects points where population change from t to t+1 is 0. These are equilibrium.
abline(a=0,b=1)

```



Alright. So now we can look at the actual dynamics across many generations.

```

logisticDynamics <- function(n, r, k, steps = 100){
  ret <- c()
  ret[1] <- n
  if(length(r) == 1){
    r <- rep(r, steps)
  }
  for(i in 1:(steps-1)){
    ret[i+1] <- logisticGrowth(ret[i], r[i], k)
  }
  return(ret)
}

```

```

stps <- 20

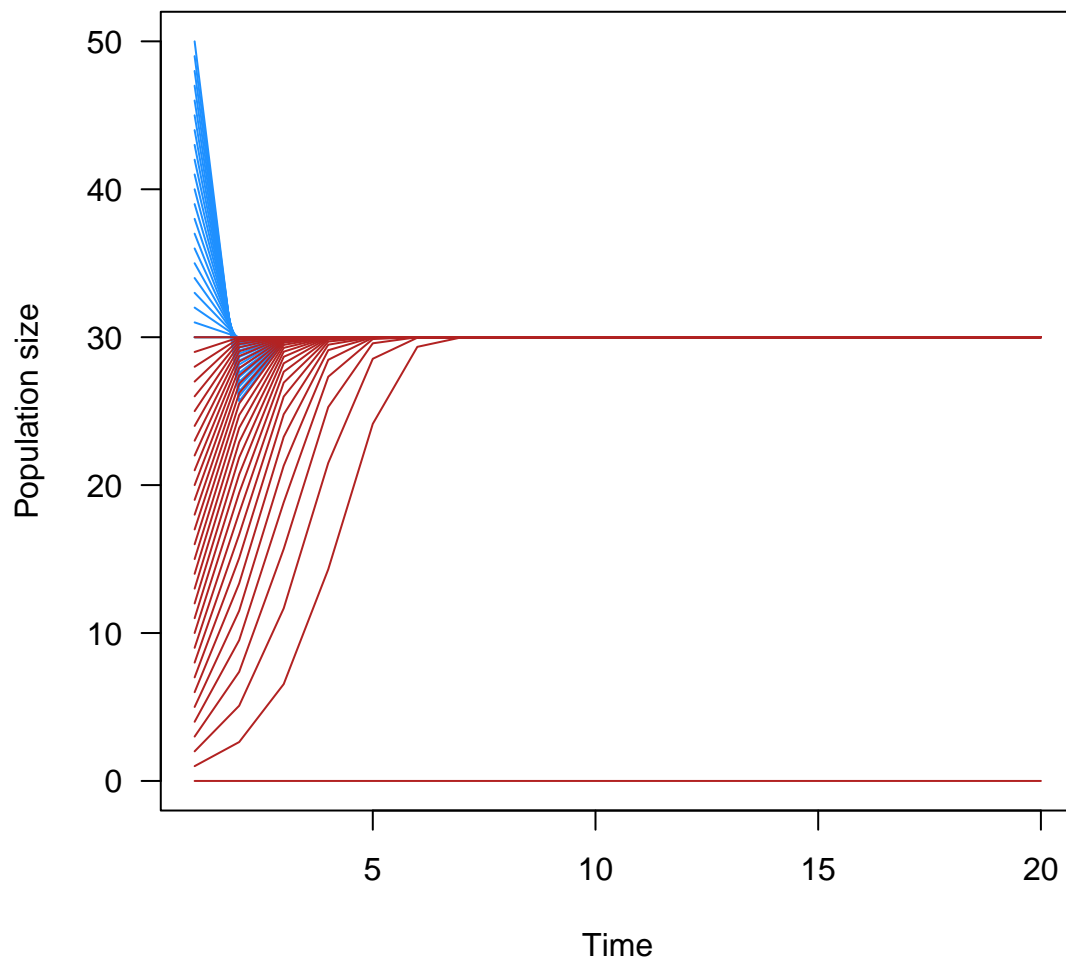
plot(1:stps,
     logisticDynamics(n=30, r=1, k=30, steps=stps),
     type='l', las=1, ylim=c(0,50),
     xlab='Time',
     ylab='Population size',
     col=1)

#sapply(seq(1,25,by=1), function(x){
# lines(logisticDynamics(n=x, r=1, k=30, steps=stps), col='firebrick')
#})

for(x in seq(30, 50,by=1)){
  lines(logisticDynamics(n=x, r=1, k=30, steps=stps), col='dodgerblue')
}

for(x in seq(0,30,by=1)){
  lines(logisticDynamics(n=x, r=1, k=30, steps=stps), col='firebrick')
}

```



What if growth rate is not 1?

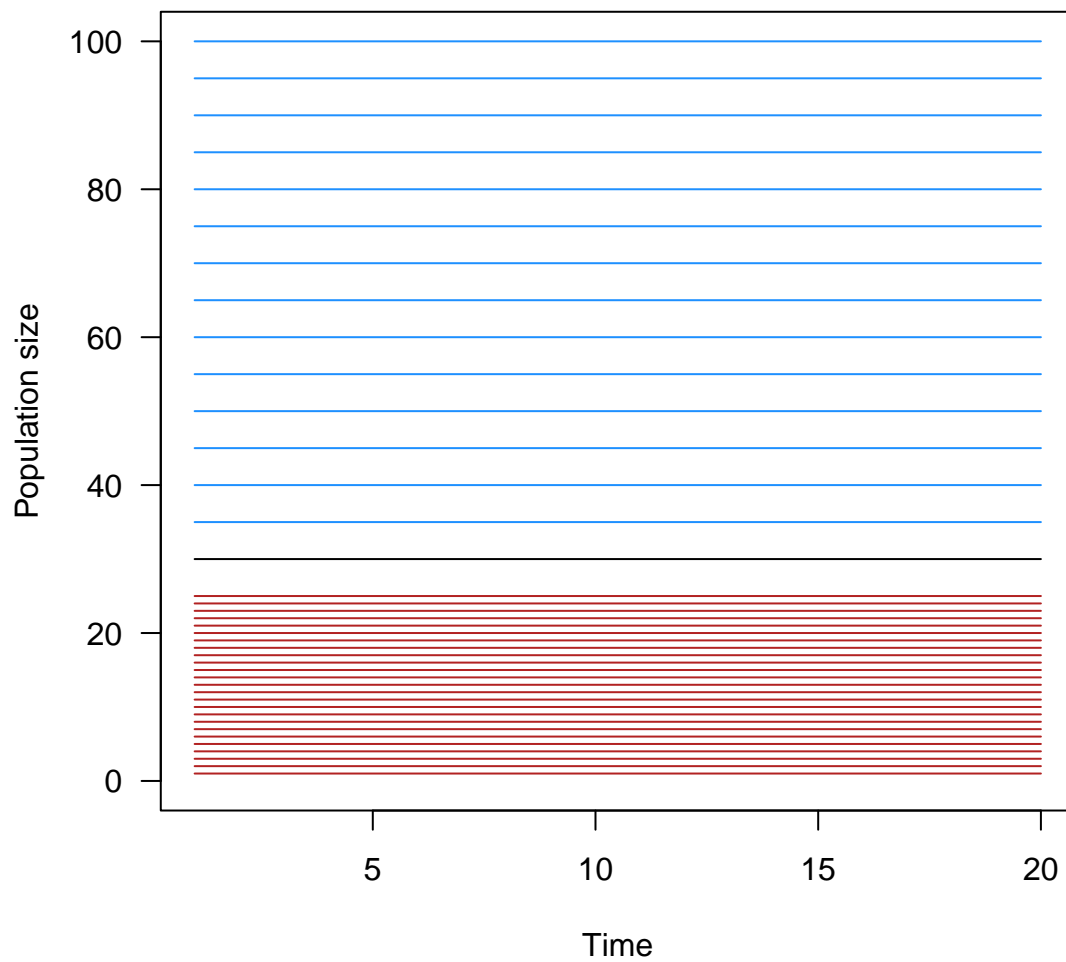
$r=0$

```
steps <- 20

plot(1:steps,
     logisticDynamics(n=30, r=0, k=30, steps=steps),
     type='l', las=1, ylim=c(0,100),
     xlab='Time',
     ylab='Population size',
     col=1)

for(x in seq(1,25,by=1)){
  lines(logisticDynamics(n=x, r=0, k=30, steps=steps), col='firebrick')
}

for(x in seq(35,100,by=5)){
  lines(logisticDynamics(n=x, r=0, k=30, steps=steps), col='dodgerblue')
}
```

$r=1.5$

```

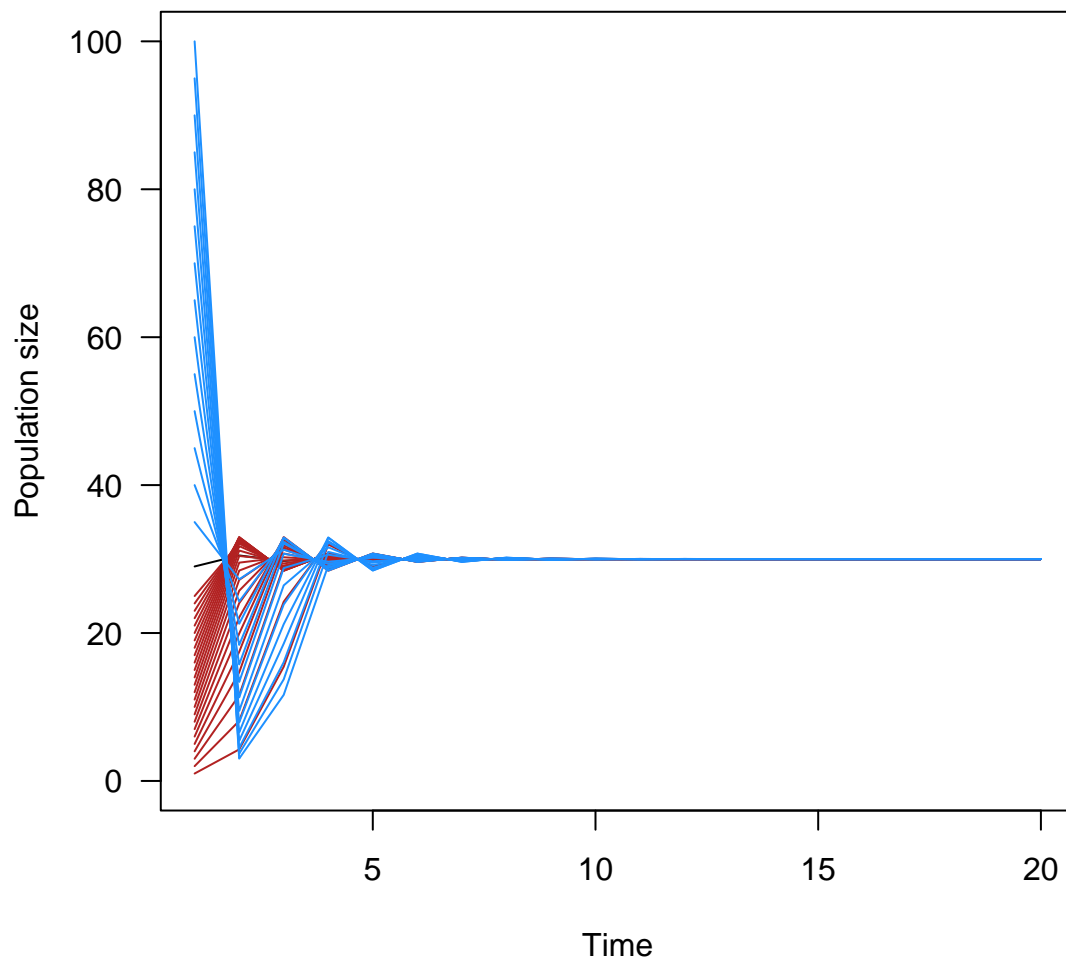
steps <- 20

plot(1:steps,
     logisticDynamics(n=29, r=1.5, k=30, steps=steps),
     type='l', las=1, ylim=c(0,100),
     xlab='Time',
     ylab='Population size',
     col=1)

for (x in seq(1,25,by=1)){
  lines(logisticDynamics(n=x, r=1.5, k=30, steps=steps), col='firebrick')
}

for(x in seq(35, 100, by = 5)){
  lines(logisticDynamics(n = x, r = 1.5, k = 30, steps = steps), col = 'dodgerblue')
}

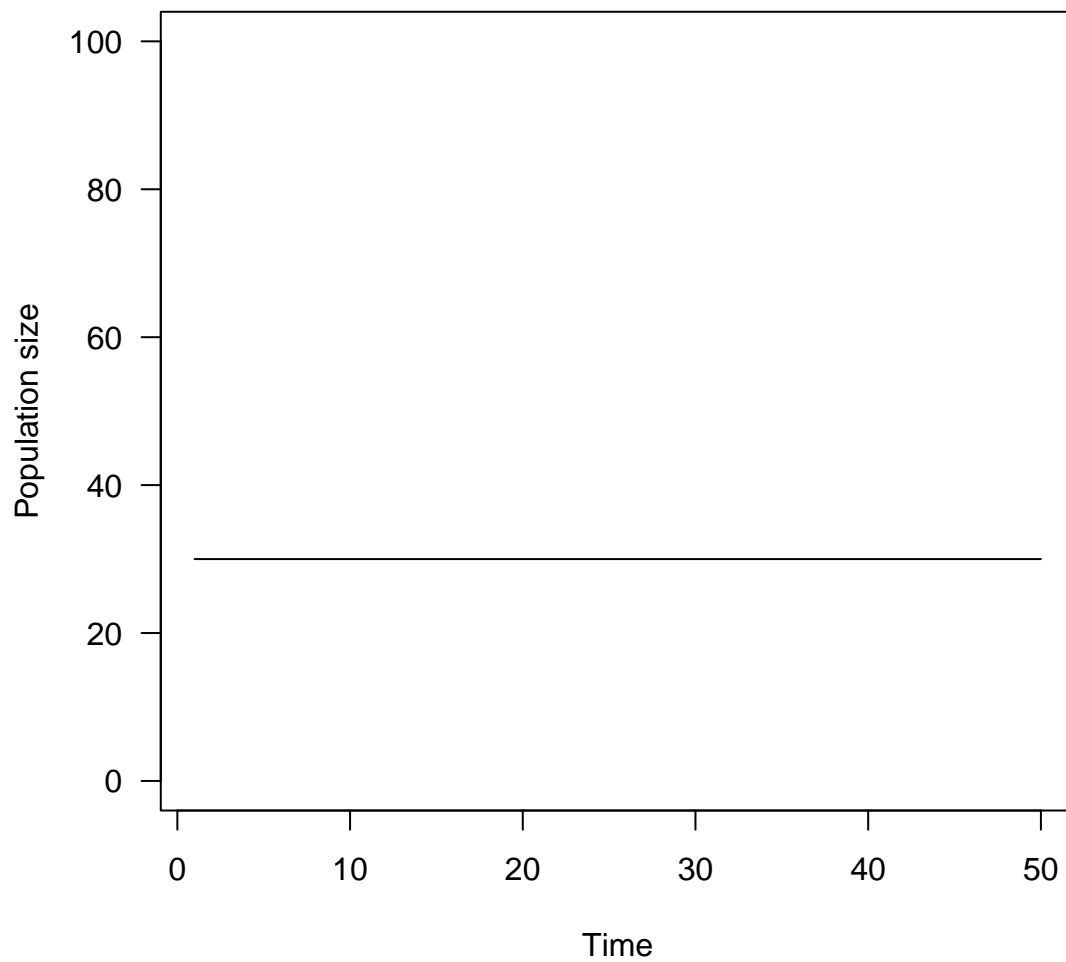
```



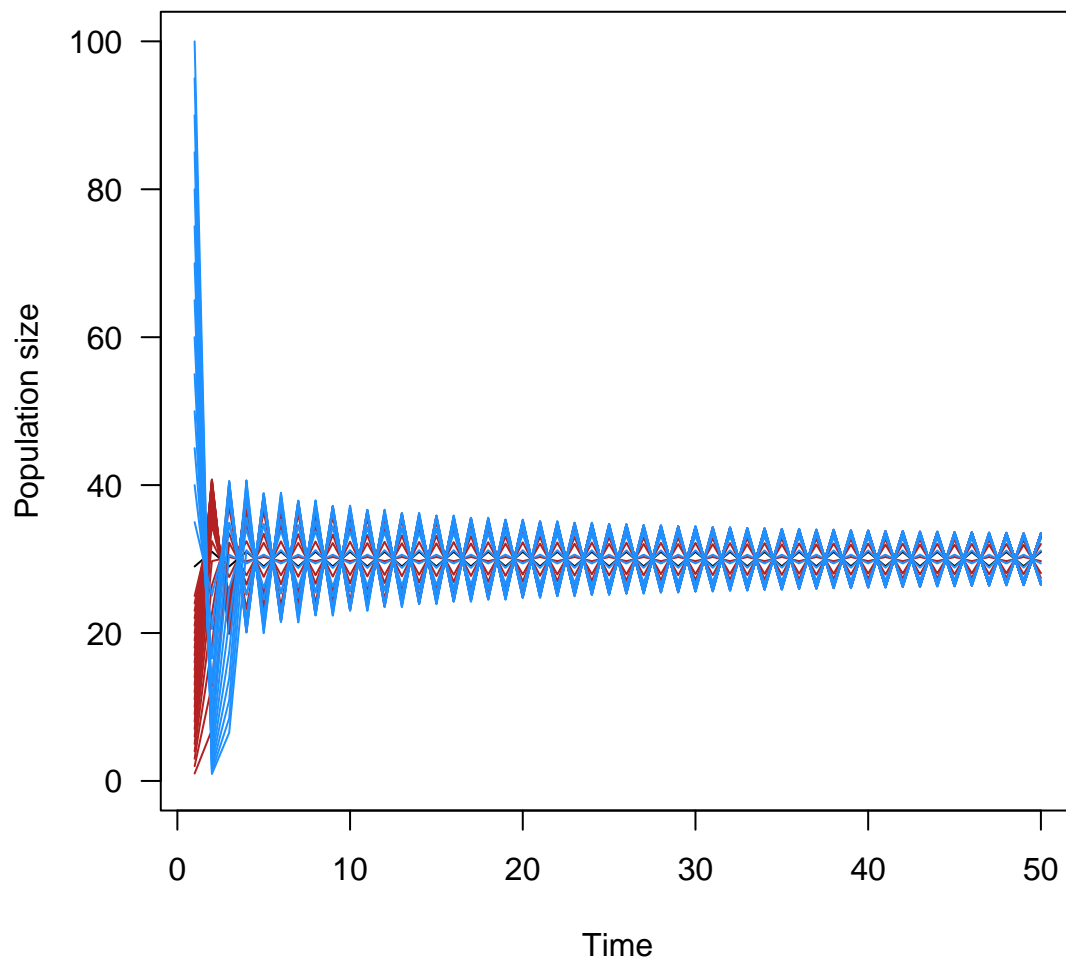
$r=2$

```
steps <- 50

plot(1:steps,
     logisticDynamics(n=30, r=2, k=30, steps=steps),
     type='l', las=1, ylim=c(0,100),
     xlab='Time',
     ylab='Population size',
     col=1)
```



```
plot(1:stps,  
     logisticDynamics(n=29, r=2, k=30, steps=stps),  
     type='l', las=1,ylim=c(0,100),  
     xlab='Time',  
     ylab='Population size',  
     col=1)  
  
for(x in seq(1,25,by=1)){  
  lines(logisticDynamics(n=x, r=2, k=30, steps=stps), col='firebrick')  
}  
  
for(x in seq(35,100,by=5)){  
  lines(logisticDynamics(n=x, r=2, k=30, steps=stps), col='dodgerblue')  
}
```



(between 3 and 3.449 – oscillates between 2 values)

$r=3.2$

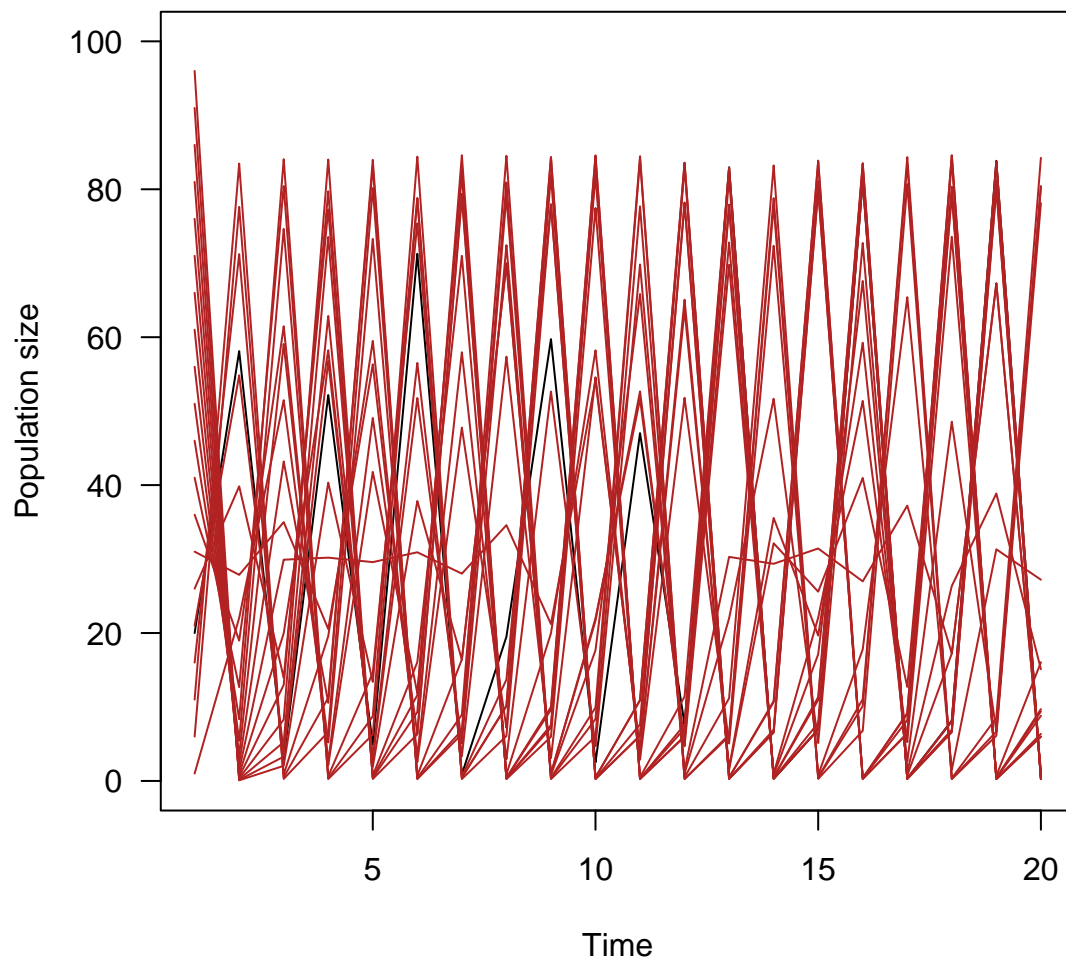
```

steps <- 20

plot(1:steps,
     logisticDynamics(n=20, r=3.2, k=30, steps=steps),
     type='l', las=1, ylim=c(0,100),
     xlab='Time',
     ylab='Population size',
     col=1)

for(x in seq(1,100,by=5)){
  lines(logisticDynamics(n=x, r=3.2, k=30, steps=steps), col='firebrick')
}

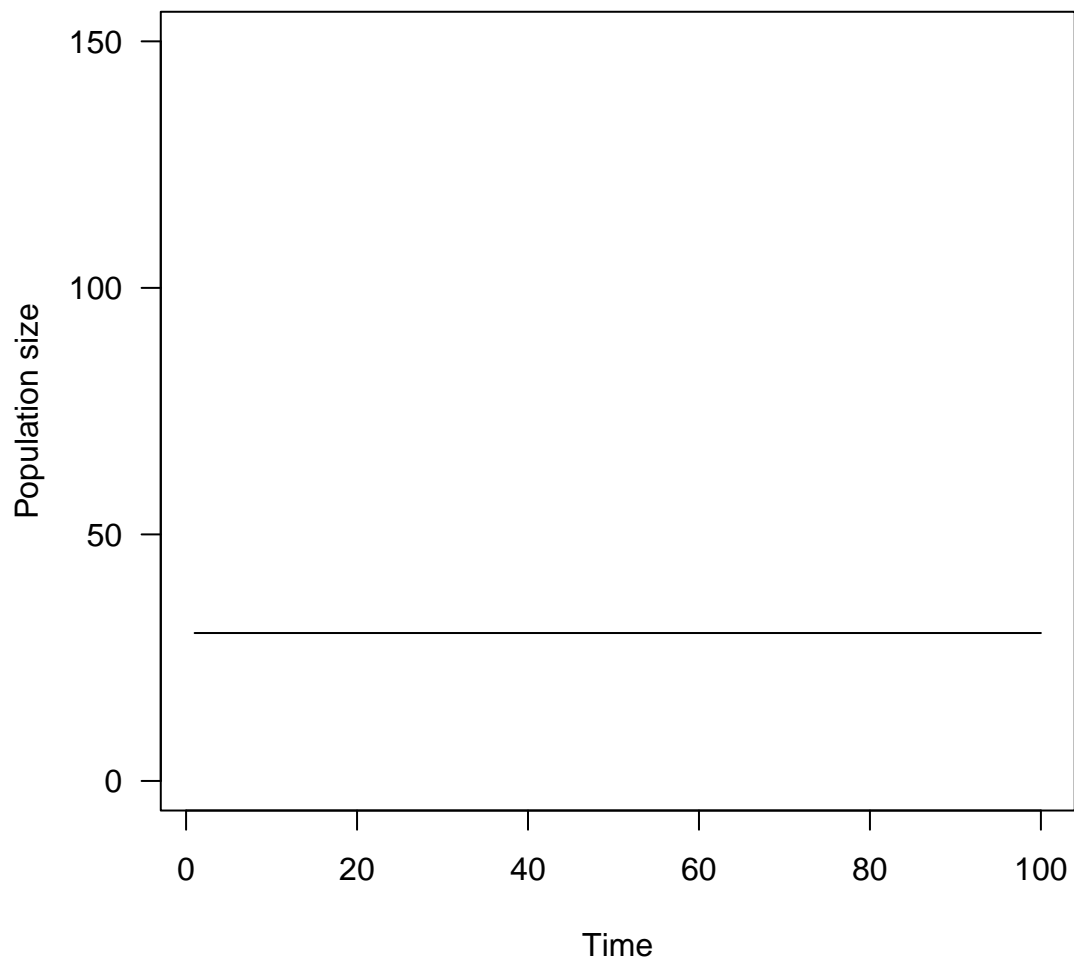
```



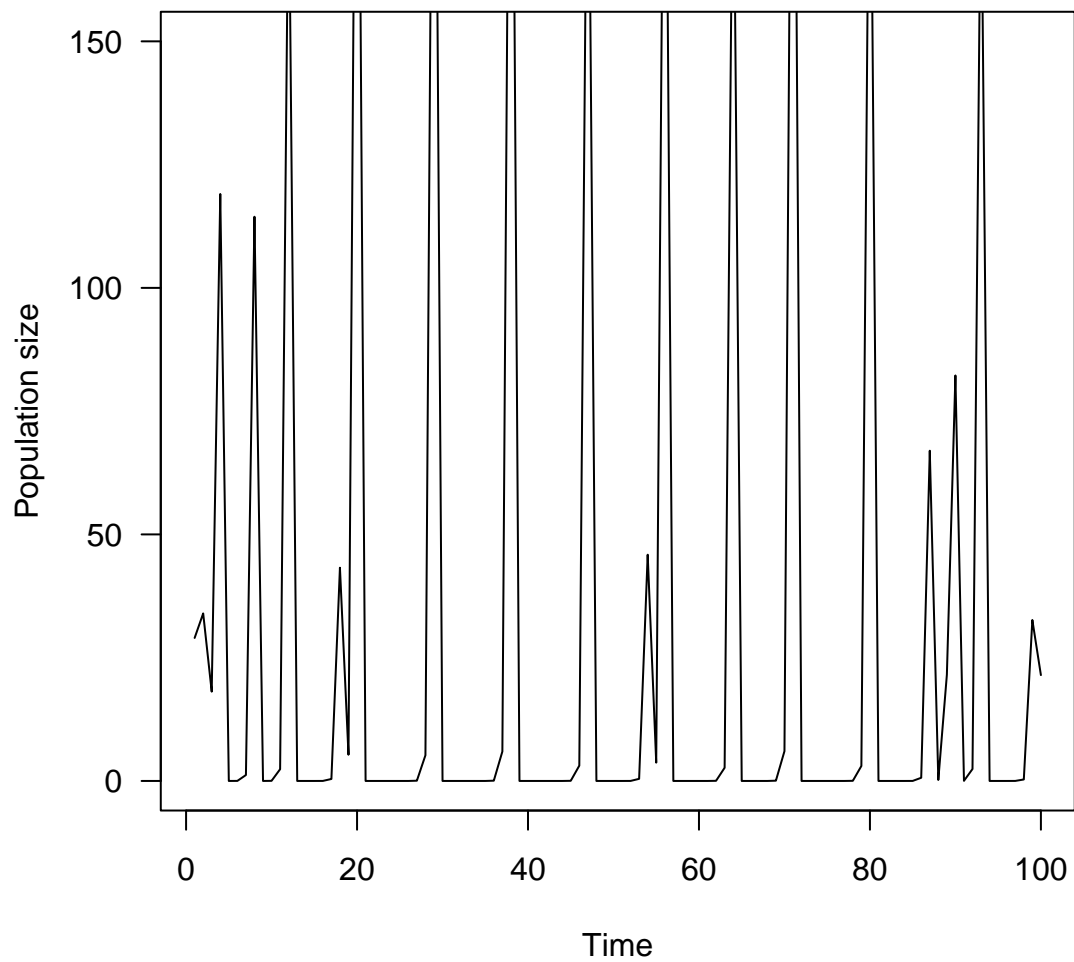
(onset of chaos) $r > 3.56$

```
steps <- 100

plot(1:steps,
     logisticDynamics(n=30, r=4.75, k=30, steps=steps),
     type='l', las=1, ylim=c(0,150),
     xlab='Time',
     ylab='Population size',
     col=1)
```



```
plot(1:steps,  
     logisticDynamics(n=29, r=4.75, k=30, steps=steps),  
     type='l', las=1, ylim=c(0,150),  
     xlab='Time',  
     ylab='Population size',  
     col=1)
```



Matrix model

First, build a simple projection matrix

```
projMatrix <- matrix(
  c(0.2, 1, 0.25,
    0.3, 0, 0,
    0, 0.3, 0.6)
  ,
  nrow = 3,
  ncol = 3,
  byrow = TRUE
)

abund0 <- matrix(c(20, 20, 20), ncol = 1)
```

Simulate one generation into the future.

```
abund1 <- projMatrix %*% abund0
```

Simulate one more generation

```
projMatrix %*% abund1
```

```
##      [,1]  
## [1,] 16.3  
## [2,]  8.7  
## [3,] 12.6
```

Simulate many generations to examine dynamics

```
getStageDynamics <- function(projMatrix, abund, steps=100){  
  ret <- matrix(0, ncol=3, nrow=steps+1)  
  ret[1,] <- abund  
  for(i in 1:steps){  
    ret[i+1, ] <- projMatrix %*% matrix(ret[i,],ncol=1)  
  }  
  return(ret)  
}
```

```
stageDynamics <- getStageDynamics(projMatrix, abund0, steps=50)  
  
plot(stageDynamics[,1], type='l', lwd=2,  
      col="red",  
      ylab='Abundance', xlab='Time')  
lines(stageDynamics[,2], lwd=2, col="blue")  
lines(stageDynamics[,3], lwd=2, col="orange")  
legend('topright', pch=16, col=c("red", "blue", "orange"),  
      c('Young', 'Middle', 'Old'), bty='n')
```