Lecture 5
Unregulated population growth
continued and summarized

Dr. Ido Filin
ifilin@univ.haifa.ac.il

19 November 2012

Outline

Defining our own functions

1. Create a new script: `ThirdPopModel.r`
2. Write the following command lines into the script file:
    1. `genNum <- 32`
    2. `Ninitial <- 1`
    3. `lambda <- 2`
    4. `popGrowth <- function( popSize, growthParam ) { newVal <- growthParam * popSize; return(newVal) }`
    5. `N <- numeric(genNum)`
    6. `Time <- 24 * ( 0 :  ( genNum - 1 ) )`
    7. `N[1] <- Ninitial`
    8. `for ( index in 2:genNum ) { N[index] <- popGrowth( N[index-1], lambda ) }`
    9. `print(N)`
    10. `plot( Time, N, xlab = "Time[hours]" )`
3. Save and run the script.

Defining our own functions

- A function declaration has the following structure

<function name> `<-` `function(`*<list of argument names>*`)`

`{`

<list of R commands>

```
...
return(...)
}
```

- Note that this is similar to assignment into variables.

- Example: `funfun <- function( numarg, textarg ) { print(textarg); val <- numarg^2 + numarg; return(val) }`

  Test it by typing

```
print( 1.5 * funfun( 4, "Learning R is fun!" ) )
```

Default values of arguments

- We can define default values for arguments.
- We do it using the **=** sign within the function declaration.
- Example:

```
funfun <- function( numarg = 5, textarg = "**
Default text **" )
{ print(textarg); val <- numarg^2 + numarg;
return(val) }
```

- Test it by typing
  1. `funfun( 4, "Learning R is fun!" )`
  2. `funfun( 4 )`
  3. `funfun()`
- If we want to change only the second argument
  1. `funfun( , "R is fun!" )`
  2. `funfun( textarg = "Good morning" )`

Default values of arguments

- Or input them in a different order
  1. `funfun( textarg = "fun fun fun!", numarg = 3)`
- We can use the explicit names of the arguments, as defined in the function declaration, when setting values of arguments during a function call.
- In that case, we don't need to observe the original order of the arguments.
- Example: `foo <- function(x, y, z) {...}`
  The function call `foo( z = 3, x = 1, y = 2 )`
  is identical to the call `foo( 1, 2, 3 )`
  but different than `foo( 3, 1, 2 )`.
- We have already seen this syntax with the `plot` function.

```
plot( x, y, xlab = ..., type = ..., ylab = ...)
```

Default values of arguments

1. Change `ThirdPopModel.r` as follows.
   1. `genNum <- 32`
   2. `Ninitial <- 1`
   3. `lambda <- 3`
   4. `popGrowth <- function(popSize = 1, growthParam = 2)`
      `{ newVal <- growthParam * popSize; return(newVal) }`
   5. `N <- numeric(genNum)`
   6. `Time <- 24 * ( 0 :  ( genNum - 1 ) )`
   7. `N[1] <- Ninitial`
   8. `for ( index in 2:genNum )`
      `{ N[index] <- popGrowth( popSize = N[index-1] ) }`
   9. `print(N)`
   10. `plot( Time, N, xlab = "Time[hours]" )`
2. Save and run the script.
3. What is the finite rate of increase of this population?

Outline

Erros, readability and comments

1. Read errors carefully – they guide you to the type and location of the problem.

2. Choose meaningful names for variables.

```
plot(time,popSize)
```

is clearer to read and understand, compared to:

```
plot(T,N)
```

3. Similarly, use spaces.

```
time <- 0:4 ; popSize <- c( 10, 20, 40, 80, 160 )
plot( time, popSize, xlab = "Time[hours]", ylab =
"Population size", type = "b" )
```

is clearer to read than:

```
time<-0:4;popSize<-c(10,20,40,80,160)
plot(time,popSize,xlab="Time[hours]",ylab="Population
size",type="b")
```

Erros, readability and comments

1. Similarly, write each command in a new line, separate different sections of the program with a blank line, and indent!
2. For example, compare:

```
 PopGrowth <- function( popSize, growthParam )
{ newPopSize <- popSize * growthParam ; return(
newPopSize ) }
genNum <- 10; Ninitial <- 2
N <- numeric(genNum) N[1] <- Ninitial
for ( index in 2:genNum )
{ N[index] = PopGrowth( N, 2 ) }
```

Erros, readability and comments

2. ….. with the following

```
PopGrowth <- function( popSize, growthParam )
{
   newPopSize <- popSize * growthParam
   return( newPopSize )
}

genNum <- 10
Ninitial <- 2

N <- numeric(genNum)
N[1] <- Ninitial

for ( index in 2:genNum )
{
    N[index] = PopGrowth( N, 2 )
}
```

Erros, readability and comments

3. ..... and with comments

```
PopGrowth <- function( popSize, growthParam )
{
  # New population size using geometric model.
  newPopSize <- popSize * growthParam
  return( newPopSize )
}

# Parameters.
genNum <- 10
Ninitial <- 2

# Initialization of variables.
N <- numeric(genNum)
N[1] <- Ninitial

# For-loop to calculate population trajectory.
for ( index in 2:genNum )
{
```

Outline

Geometric model of population growth

Recursion relation/Difference equation of geometric growth

$$N_{t+1} = \lambda\ N_t \ \Leftrightarrow\ \Delta N = (\lambda - 1)N_t$$

- λ is called **finite rate of increase**.
- It is the **average**/**mean** **per-capita** multiplication factor per one time-step.
- Average in the sense that some individuals contribute negative growth (die), some contribute positive growth (reproduce), some contribute zero growth (survive but do not reproduce). λ represents a weighted average of these different contributions.
- Classification of geometric growth:
  - $\lambda > 1 \Rightarrow \Delta N > 0$, population grows.
  - $\lambda = 1 \Rightarrow \Delta N = 0$, population size unchanged.
  - $0 \leqslant \lambda < 1 \Rightarrow \Delta N < 0$, population declines.

The net reproductive rate, R_0

- A second important parameter is the **net reproductive rate**, R_0, which is the **expected** lifetime reproductive output of a female.
- Example: for unicellulars, when time between divisions represents lifetime,
  - if there is no mortality, $R_0 = 2$.
  - if there is 25% mortality between divisions, $R_0 = 0.25 * 0 + 0.75 * 2 = 1.5$
  - if there is 50% mortality between divisions, $R_0 = 0.5 * 0 + 0.5 * 2 = 1$
  - etc.
- If we measure population growth in time-steps of 1 lifetime, $\lambda = R_0$.
- If in time-steps of 2 lifetimes, $\lambda = R_0^2$.
- If in time-steps of 3 lifetimes, $\lambda = R_0^3$.

The net reproductive rate, R_0

- Generalizing:

  If T is lifetime / generation time,
  and τ is the time-step of the model,
  The relationship between λ and R_0 is

$$\boxed{\lambda = R_0^{(\tau/T)}}$$

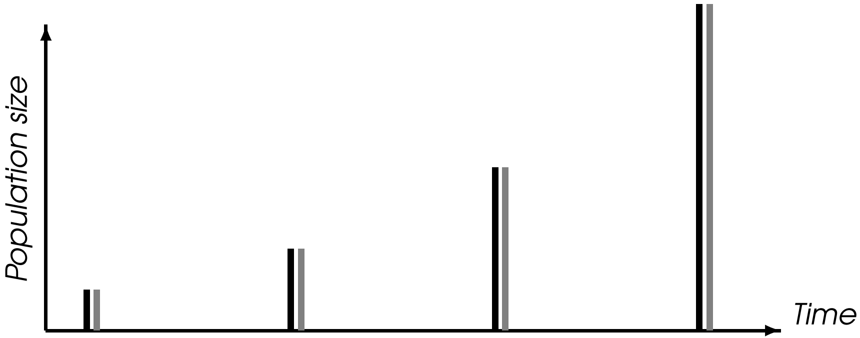
- So, for example, if we measure population growth in time-steps of 10 generation times, $\tau = 10T$, then $\lambda = R_0^{10}$.

- But what about when $\tau < T$ we measure population growth in time-steps shorter than lifetime / generation time.

- Does this general relation still apply?

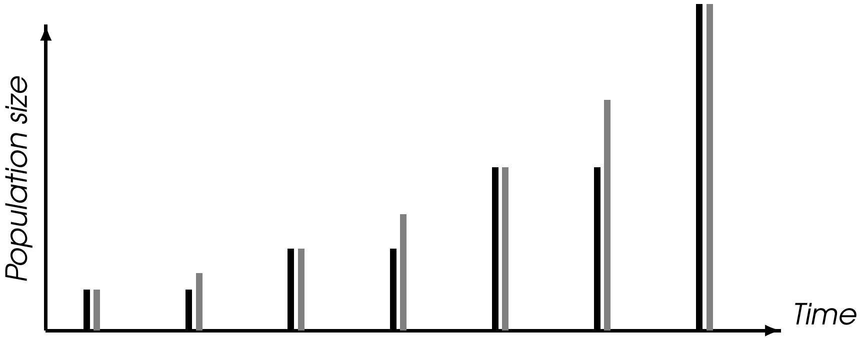
Synchronous and asynchronous reproduction

Synchronous reproduction: $\tau = T$.



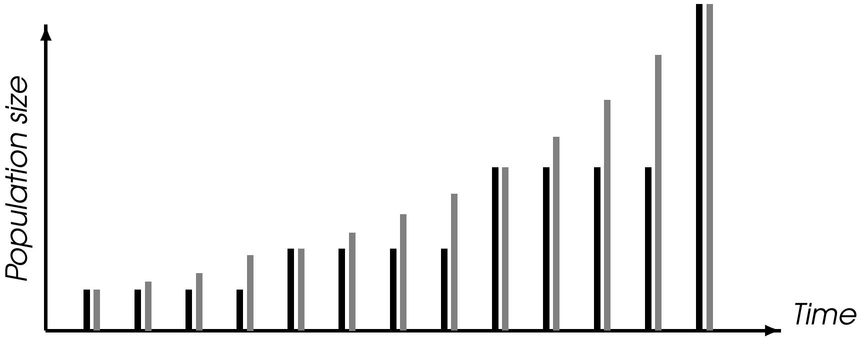
Synchronous and asynchronous reproduction

Synchronous reproduction: $\tau = T/2$.



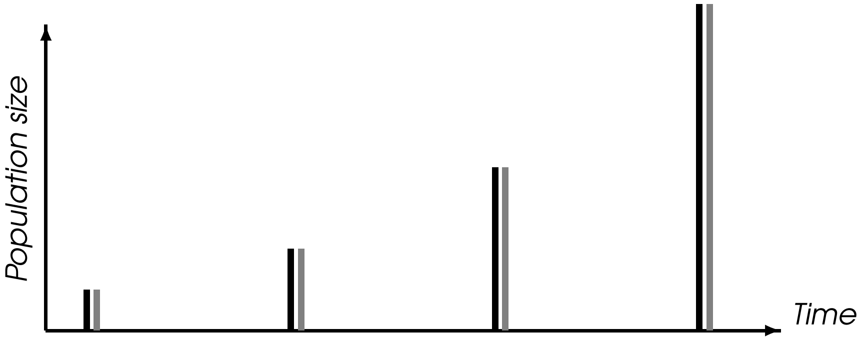
Synchronous and asynchronous reproduction

Synchronous reproduction: $\tau = T/4$.



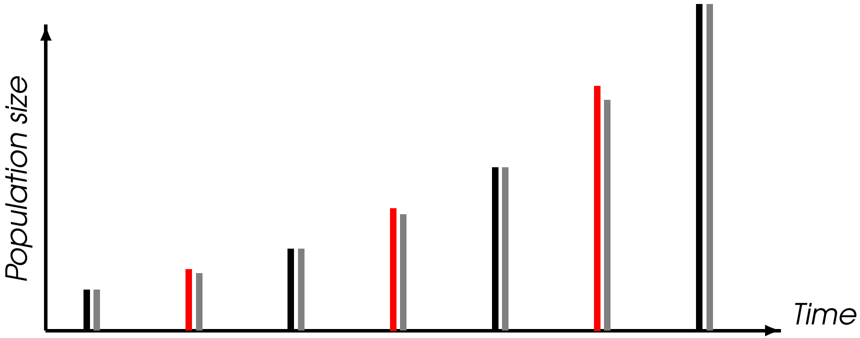
Synchronous and asynchronous reproduction

But if we break the synchrony in reproduction.
For example, half the cells reproduce at noon and half at midnight.
Asynchronous reproduction: $\tau = T$.



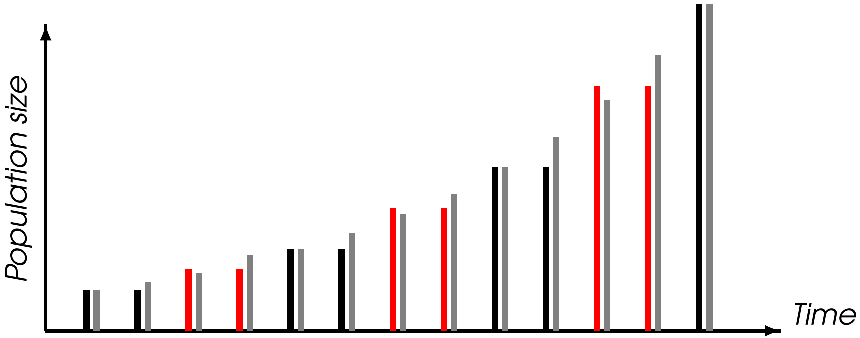
Synchronous and asynchronous reproduction

Asynchronous reproduction: $\tau = T/2$.



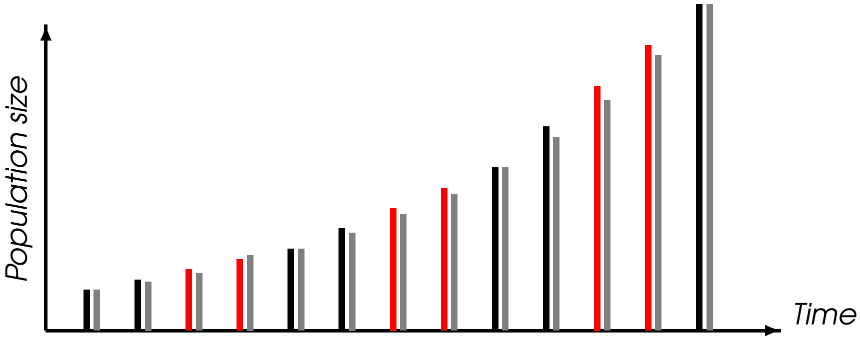
Synchronous and asynchronous reproduction

Asynchronous reproduction: $\tau = T/4$.



Synchronous and asynchronous reproduction

Even more asynchronous reproduction:
four "subpopulations", based on timing of division.



Outline

Exponential model of population growth

In the limit of completely asynchronous reproduction – reproduction events are continuously distributed over time – we obtain the **exponential growth** model.

- The population growth trajectory is given by
$$N(t) = N_0 e^{rt}$$
- The population growth rate is given by the differential equation
$$dN/dt = rN$$
- r is called **instantaneous rate of increase**.
- r has units of rate or 1/time (i.e., min $^{-1}$, day $^{-1}$, year $^{-1}$, etc.) – unlike λ and R_0, which are pure numbers.

Exponential model of population growth

In the limit of completely asynchronous reproduction – reproduction events are continuously distributed over time – we obtain the **exponential growth** model.

- The three parameters are related according to

$$r = \frac{\ln \lambda}{\tau} = \frac{\ln R_0}{T}$$

- Classification of geometric growth:
  - $r > 0 \Rightarrow dN/dt > 0$, population grows.
  - $r = 0 \Rightarrow dN/dt = 0$, population size unchanged.
  - $r < 0 \Rightarrow dN/dt < 0$, population declines.

Outline

The BIDE model

- A fundamental ecological fact of life:

$$N_{t+1} = N_t + B + I - D - E$$

- Change in population size is the sum of positive contributions (birth and immigration) and negative contributions (death and emigration).
- While demonstrating those contributions in a compact form, this model says very little else.
- If B, I, D, E are fixed constants, we get the arithmetic model $\rightarrow$ not very useful.
- If B, I, D, E depend on population size (as in the geometric model), the above expression hardly displays this dependence $\rightarrow$ again, not very useful.

Birth, death and migration rates

- We can modify the BIDE model:

$$N_{t+1} = N_t + bN_t - dN_t - eN_t + I = (1 + b - d - e)N_t + I$$

- In this case, b, d, e are the (per-capita) birth, death and emigration rates.
- They are measured with respect to the original population at time t.
- To begin, let us ignore migration, i.e., $e = 0$, $I = 0$.
- If at time $t + 1$ (i.e., after a period of duration τ), all original individuals (N_t) have died, we have: $d = 1$, $b = R_0$, and so $\lambda = 1 + R_0 - 1 = R_0$.
- If at time $t + 1$ three generations have passed (original individuals, daughters and granddaughters are already dead), we have $d = 1$, $b = R_0^3$, and so $\lambda = 1 + R_0^3 - 1 = R_0^3$.

Birth, death and migration rates

- For time-steps shorter than generation time ($\tau < T$) we have already seen the difference between pure geometric growth and explicitly following occurrences of deaths and births.
- For short time-steps, b and d reflect only mean rates.
- Or, if death and birth occur asynchronously, randomly, and independently,
  - d represents probability to die during a single time-step (same for all individuals and at all times).
  - b represents the mean per-capita births that occurred in the population during that interval (again, same for all individuals and at all times).

Birth, death and migration rates

$$N_{t+1} = (1 + b - d - e)N_t + I$$

- When studying only a single population, emigration rate, e, has same effect as death rate, d. There is nothing significantly different in a model with emigration (e and d can combine into a single "effective death" rate).

- A population model with nonzero immigration, I, is no longer purely geometric, but represents a combination of geometric and arithmetic growth models.

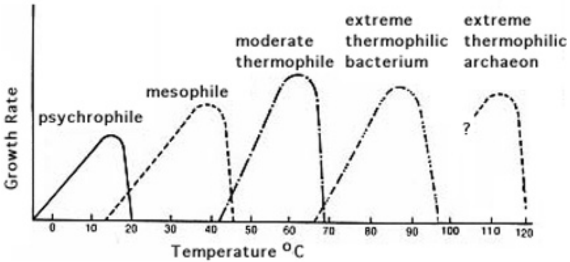
Birth, death and migration rates

- For the exponential growth model, b and d are "real" rates, i.e., have units of 1/time.
- For a very short time interval τ the total number of births in the population would be $bN\tau$, and the total number of deaths would be $dN\tau$.
- Or, the probability to die during the interval is $d\tau$, and the mean number of births per-capita is $b\tau$.
- In the exponential model $r = b - d$.
- d enforces a statistical distribution of lifetimes – where the life expectancy (mean lifetime) is $1/d$.
- R_0 is mean lifetime reproduction, therefore $R_0 = b/d$.

More detailed geometric and exponential models

r and λ may themselves be composed of more fundamental parameters:

- Give population age-.structure, R_0 and r are actually derived from age-dependent schedules of survival and reproduction.
- Temperature dependence, nutrient concentration, pH-dependence etc. For example: $\lambda = \lambda_0$ + $c *$ (Temperature) + $k *$ (Nutrient concentration)



Comparison between exponential and geometric models

	Geometric	**Exponential**
Time	discrete, $t = 0, 1, 2 \ldots$	continuous
Parameter	λ (no units)	r (1/time)
Dynamics	$N_t = \lambda N_t$ $\Delta N = (\lambda - 1)N_t$	$dN/dt = rN$
Trajectory	$N_t = N_0 \lambda^t$	$N(t) = N_0 e^{rt}$
Growing	$\lambda > 1$	$r > 0$
Stable	$\lambda = 1$	$r = 0$
Declining	$0 \leqslant \lambda < 1$	$r < 0$

Which model should I use?

1. The answer most strongly depends on the life-cycle of the organism.
   - Geometric model would fit an organism with synchronous reproduction, or with division of the life-cycle or annual cycle into distinct reproductive and non-reproductive phases (e.g., plants, many insects, landsnails etc.).
   - Exponential model would fit organisms that reproduce asynchronously and do not have distinct reproductive seasons (e.g., humans, tropical animals and plants, unicellulars in a beaker).

2. But also depends on the research goals and methodology.
   For example, if we want to model human population size at the end of each decade, a geometric model may serve better.

Which model should I use?

3. A related issue is that of **overlapping** vs. **nonoverlapping** generations.
   i.e., whether within a population we can find individuals belonging to different generations (different age- or year-classes).
   - Clearly, that depends on the organism: annual plants and insects have nonoverlapping generations, while perennial plants and humans have overlapping generations.
   - It is sometimes argued that geometric model fits the former, and exponential the latter.
   - However, perennial organisms may fit either geometric or exponential model:
     perennial plants fit a geometric model, because they reproduce only at specific times during the year.
     Humans, however, are better described by an exponential growth model.
   - It is the synchrony of reproduction (or lack of it) that is more important in determining the preferred model.

Too many names

- The net reproductive rate, R_0, has also been called in the literature
  - basic reproductive rate
  - basic reproductive number/ratio
- The finite rate of increase, λ, is sometimes called
  - net reproductive rate
  - fundamental net reproductive rate
  - fundamental per capita rate of increase
- The instantaneous rate of increase, r, is sometimes called
  - intrinsic rate of (natural) increase
- Also, you will also sometimes see the relation $\lambda = e^r$. This is meaningless because r has units. The implicit assumption here is that we measure time in units of τ.