Ecología de poblaciones silvestres

David Martínez Cascante

2018-02-22

Índice general

Capítulo 1

Introducción

La ecología de poblaciones se centra en el estudio de la dinámica de las poblaciones (su crecimiento e interacción con otras poblaciones), y en las interacciones de éstas con el ambiente. La ecología de poblaciones (también llamada **dinámica de poblaciones**) es un campo con un componente matemático y estadístico fuerte, y de gran importancia para la gestión de vida silvestre.

Algunas de las aplicaciones más importantes de esta disciplina, están relacionadas al cálculo de la viabilidad de poblaciones, al cálculo de tasas de extracción, e incluso a la creación de áreas protegidas dedicadas a proteger el ciclo de vida, o parte de éste, en determinadas especies.

El *Análisis de Viabilidad de Poblaciones*, es un ejemplo de una de las aplicaciones de la ecología de poblaciones para la gestión de vida silvestre. Este modelo predice el riesgo de que una población se extinga en una determinada cantidad de años. De esta manera, los gestores pueden modelar diferentes escenarios, cada cual con un conjunto específico de acciones de manejo, y decidir cuál de éstos es más efectivo en la conservación o manejo de la especie.

La creación de santuarios de pesca, por ejemplo, se fundamenta en el concepto de *Biogeografía de Islas* (REF) que también es parte de la ecología de poblaciones. Los santuarios de pesca funcionan como *fuentes*, es decir, zonas donde el crecimiento poblacional es positivo y existe migración de individuos. Éstos individuos, que se producen en exceso, migrarán hacia zonas de pesca, o extracción, para sostener actividades económicas. De esta manera, se garantiza la extracción sostenible en las zonas aledañas.

Algunos modelos importantes, como el modelo **bioeconómico**, que buscan la mayor rentabilidad económica por la extracción de una especie (Grafton *et al.*, 2006), están basados en

modelos de crecimiento derivados de la dinámica de poblaciones. Este modelo estima la cantidad de esfuerzo extractivo que debe aplicarse a una especie, para mantener una rentabilidad positiva, y mantener un tamaño poblacional que garantice la continuidad de las poblaciones aprovechadas.

La dinámica de poblaciones es una de las ramas de la biología con un componente matemático y estadístico más fuertes. El desarrollo teórico de los modelos implica conocimiento de planteamiento y resolución de *ecuaciones diferenciales*. En la práctica, muchos problemas se plantean como ecuaciones diferenciales, pero no tienen solución analítica, por lo cual se requiere de conocimiento sobre *métodos numéricos*, *programación* o uso de lenguajes de programación. La mayoría de profesionales, no son desarrolladores teóricos, pero deben saber, al menos, sobre el uso de herramientas de análisis para esta disciplina.

Si el investigador conoce las herramientas de análisis, y quiere ponerlas en práctica, entonces requiere de conocimientos en *diseño experimental y muestreal*; así como, *técnicas de muestreo* para conseguir los datos. Pero la limitación más fuerte, es el financiamiento requerido; ya que, la mayoría de los análisis tienen fuertes requerimientos de datos, y series de tiempo bastante amplias.

Capítulo 2

Modelos de crecimiento

HACER: conceptos de producción en exceso, de Darwin, y lucha por la existencia.

La evolución por selección natural implica que en una población que enfrente presiones para subsistir, existirán individuos mejor adaptados que otros. Algunos vivirán lo suficiente para reproducirse y otros no; además, dentro de aquellos que se reproduzcan, los más exitosos lo harán más frecuentemente, o con mayor descendencia. Este concepto implica que en una población debe haber suficiente variabilidad genética, que se refleje en un desempeño diferente en la reproducción, y que no todos los organismos vivirán lo suficiente para dejar descendencia o reemplazarse a sí mismos. Esto quiere decir, que las poblaciones deben de reproducirse y dejar un *exceso de decendencia*, para poder amortiguar el efecto sobre la reproducción de aquellos organismos que no logren reproducirse con éxito.

De esta manera, la sobre-producción de organismos es un requisito para que una población subsista en un intervalo prolongado de tiempo. Y la sobre-producción implica que las poblaciones tienen el potencial de *crecer*. La disciplina de la ecología de poblaciones, entonces, ha enfocado esfuerzos en modelar el crecimiento poblacional usando funciones matemáticas. Veremos las más básicas de ellas, con el objetivo de entender el origen y desarrollo de estos modelos.

El crecimiento en dinámica de poblaciones, está enfocado en la población, y no en el individuo. Algunos aspectos fisiológicos, e individuales, pueden ser importantes a la hora de modelar el crecimiento poblacional, y estos pueden ser incluidos como parámetros del modelo; pero, en general, el interés se centra en la estimación de la cantidad de individuos (o la biomasa) que conforma una población, y cómo cambia esta cantidad con respecto al tiempo.

El objetivo de los modelos de crecimiento, es obtener una función del tamaño de la po-

blación con respecto al tiempo. Existen dos aproximaciones principales para obtener esta función: la exponencial y la geométrica. El crecimiento exponencial se mide en cualquier momento en el tiempo, mientras que el crecimiento geométrico se mide a intervalos discretos. Es decir, ambos miden el crecimiento poblacional, pero una aproximación lo hace en intervalos continuos y la otra en intervalos discretos.

Las otra gran categoría de modelos de crecimiento tienen que ver con la dependencia de la densidad de población. Por ejemplo, una población con suficiente espacio y recursos, puede considerarse *denso-independiente*, mientras que una población que está en permanente competencia intraespecífica por la adquisición de espacio y recursos, tiene un crecimiento denso-dependiente.

2.1. Crecimiento denso-independiente

2.1.1. Crecimiento geométrico

Nuestra variable de interés es el tamaño poblacional, N. Queremos conocer el crecimiento poblacional desde año 0 ($t = 0$) hasta el año 1 ($t = 1$). Entonces, podemos restar $N_1 - N_0$ para encontrar dicho crecimiento, al que llamaremos ΔN ("Delta N"). De manera similar, podemos encontrar el crecimiento de la población en cualquier sub-intervalo de tiempo. Por ejemplo, si queremos conocer el crecimiento en el periodo $t = 1$ y $t = 0.5$, entonces nombramos este intervalo como Δt, y obtenemos el dato al dividir $\Delta N / \Delta t$. Esta razón corresponde a la *tasa de crecimiento*.

Una primer idea de cómo modelar la tasa de crecimiento, es pensar en que ésta equivale a la diferencia entre las *entradas* a la población (B, natalidad e inmigración) menos las *salidas* de la población (D, mortalidad y emigración):

$$\frac{\Delta N}{\Delta t} = B - D$$

Para conocer la tasa de crecimiento *per cápita*, dividimos la ecuación anterior por N:

$$\frac{\frac{\Delta N}{\Delta t}}{N} = \frac{D - B}{N}$$

Si la tasa de crecimiento per cápita es mayor a cero, entonces la población crece. Si es igual a cero, la población se mantiene estable. Si es menor a cero, la población decrece. Si asu-

mimos que la diferencia entra las entradas de la población y sus salidas son *constantes*, podemos arreglar la expresión anterior como $\frac{E-S}{N} = \mathrm{R_m}$; con lo que obtenemos la forma familiar de la tasa de crecimiento:

$$\frac{\Delta N}{\Delta t} = \mathrm{R_m} N \tag{2.1}$$

Sin embargo, la ecuación (2.1) aún no está en función del tiempo, que es el objetivo que se busca. Primero empecemos por predecir La población en el año uno (N_1) en función del tamaño de población inicial (N_0). Sabemos que N_1 será igual a N_0 *más* el crecimiento poblacional durante ese intervalo de tiempo. Es decir:

$$N_1 = N_0 + \frac{\Delta N}{\Delta t}$$

Y por la ecuación (2.1), substituyendo $N = N_0$, se tiene la relación:

$$\begin{aligned} N_1 &= N_0 + \mathrm{R_m} N_0 \\ &= N_0 \left(1 + \mathrm{R_m}\right) \\ &= N_0 \lambda \end{aligned} \tag{2.2}$$

Por tanto, el tamaño de población en el año uno, es igual al tamaño de población en el año cero, más el producto de la tasa de crecimiento per cápita por el tamaño de población en el año cero. Los arreglos posteriores, muestran que N_1 depende de N_0 y una constante $\lambda = 1 + \mathrm{R_m}$, la cual representa la *tasa de multiplicación*. Entonces, la población crece cuando $\lambda > 1$, se mantiene estable si $\lambda = 1$, y decrece si $\lambda < 1$.

Ahora, podemos obtener N_2 al saber que $N_2 = N_1 \lambda$. Observamos que $N_1 = N_0 \lambda$; por tanto, sustituimos el valor de N_1 para acabar con $N_2 = N_0 \lambda \lambda = N_0 \lambda^2$. Si proseguimos de esta manera, concluimos que:

$$N_t = N_0 \lambda^t \tag{2.3}$$

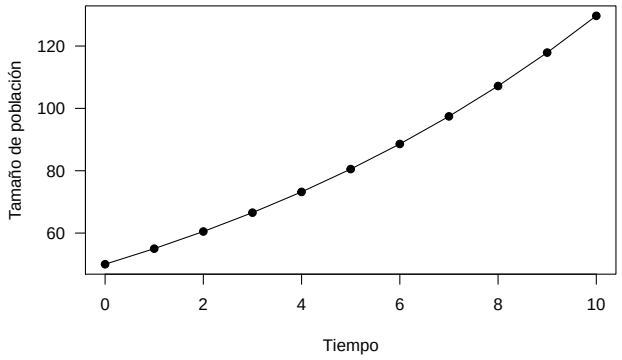
Con lo que finalmente se logra el objetivo de tener una función del tamaño poblacional en relación al tiempo.

2.1.1.1. Ejemplos

Ejemplo 1 *Graficar la ecuación* (2.3)

Ahora que tenemos una relación del tamaño poblacional con el tiempo, podemos crear una función sencilla para observar su comportamiento.

```
plotGeomGrowth <- function(N0, lambda, t){
    vectorTiempo <- 0:t
    vectorPoblacion <- N0*lambda^vectorTiempo
    plot(vectorTiempo, vectorPoblacion,
         type = "p", xlab = "Tiempo", ylab = "Tamaño de población",
         las = 1, pch = 21, bg = 1)
    lines(vectorTiempo, vectorPoblacion)
}

plotGeomGrowth(50, 1.1, 10)
```

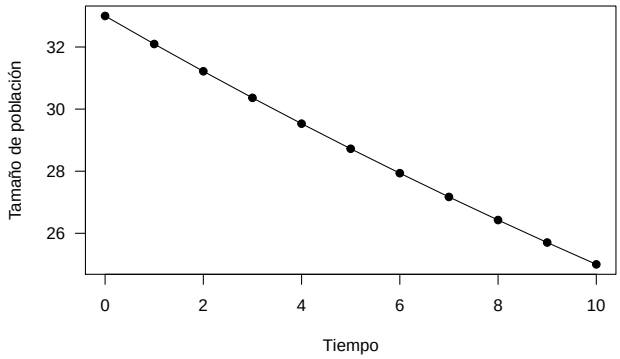


Ejemplo 2 *¿Cuál es el λ de una población que cuenta con 33 individuos en el año 0 ($t = 0$), y que tras 10 años cuenta con 25 individuos? Grafique la curva de crecimiento.*

Al despejar la ecuación (2.3) para λ se tiene

$$\lambda = \left(\frac{N_t}{N_0}\right)^{\frac{1}{t}}$$

Substituyendo los valores correspondientes se tiene que $\lambda = 0.973$. Luego, usando la función creada en el ejemplo 1, y el recién calculado lambda, se grafica la curva de crecimiento.



2.1.1.2. Ejercicios

Ejercicio 1 *Grafique la ecuación* (2.1)*. En el eje **y** la tasa de crecimiento y en el eje **x** el tamaño poblacional. Utilice tres valores de* $\mathrm{R_m}$*, uno positivo, uno igual a cero y otro negativo. El tamaño inicial de la población es de 50 individuos.* $\mathrm{R_m} \in [-1, 1]$*, y N* $\in [0, 50]$*. Cuál es la representación gráfica de* $\mathrm{R_m}$ *en el gráfico.*

Ejercicio 2 *Grafique la ecuación* (2.3)*. Utilice tres valores de λ, uno mayor a uno, otro igual a uno, y el último menor a 1, pero mayor a cero. El tamaño inicial de la población es de 50 individuos.*

Ejercicio 3 ***PICANTE*** *Todo libro de lógica matemática debe contener los métodos de demostración más comunes. Utilice el metodo de **inducción matemática** para demostrar que la ecuación* (2.3) *es válida para todo n* $\in \mathbb{N}$ *(números naturales). **5 % sobre la nota, dividido entre el número de estudiantes que respondan el ejericio**.*

Ejercicio 4 *Si inoculo una población de bacterias en un medio de cultivo con suficiente espacio y nutrientes, con un estimado de 1* $\times$ 10^6 *individuos, y tras tres horas, se estima una población de 3.5* $\times$ 10^6 *individuos, ¿Qué valor tiene lambda? NOTA. En este caso, t representa una hora.*

Ejercicio 5 *Un cultivo de células dobla su tamaño poblacional en 15 minutos (*$\lambda = 2$*). Si se empieza con 1000 células, ¿cuántas de ellas existen tras 3 horas?*

2.1.2. Crecimiento exponencial

En la sección anterior, se trabajó con intervalos de tiempo discretos. Pero si queremos conocer el tamaño poblacional en cualquier momento del tiempo, debemos trabajar con inter-

valos infinitamente pequeños. Esto quiere decir que la ecuación (2.3) se escribe en su forma continua:

$$\frac{dN}{dt} = \mathrm{r_m} N \tag{2.4}$$

La ecuación (2.4) es una *ecuación diferencial de primer orden*[1]. Este tipo particular de ecuaciones diferenciales tienen una solución analítica. Para este caso, se puede utilizar el método de separación de variables, para obtener la siguiente expresión del tamaño poblacional con respecto al tiempo (ver ejemplo 3):

$$N_t = N_0 e^{\mathrm{r_m} t} \tag{2.5}$$

En la expresión anterior, $\mathrm{r_m}$ es la *tasa instantánea de crecimiento*, también conocida como la *tasa intrínseca de crecimiento natural*, o el parámetro de Malthus por Thomas Malthus. Este parámetro equivale a la diferencia entre la tasa intrínseca de nacimiento y la tasa intrínseca de mortalidad ($b - d$). La tasa intrínseca está relacionada con la tasa de multiplicación de la siguiente forma:

$$\lambda = e^{\mathrm{r_m}}$$
$$\mathrm{r_m} = \ln \lambda$$

El parámetro $\mathrm{r_m}$ tiene aplicaciones interesantes. Una de ellas es su facilidad para utilizarse en diferentes escalas de tiempo. Por ejemplo, si $\mathrm{r_m} = 0.1$ por día, y queremos escalarlo a escala semanal, procedemos a multiplicar $\mathrm{r_m} = 0.1 \times 7 = 0.7$. Al hacer esta transformación, se debe tener en cuenta la escala de tiempo con la que se interpretan y presentan los resultados.

2.1.2.1. Ejemplos

Ejemplo 3 *Como obtener la ecuación de crecimiento exponencial* (2.5) *de la ecuación diferencial* (2.4)*.*

[1] una ecuación diferencial existe cuando en la ecuación, la incógnita depende de su derivada. En este caso, si queremos despejar *N*, observamos que su derivada se encuentra en la expresión resultante

El método de separación de variables consiste en dejar todos los términos de la incógnita de un lado, y los términos de la variable independiente (t) del otro lado de la igualdad (Barrantes Campos, 2015). Entonces:

$$\frac{1}{N} \times \frac{dN}{dt} = \mathrm{r_m}$$

Luego se integra ambos lados con respecto de la variable independiente:

$$\int \left(\frac{1}{N} \times \frac{dN}{dt} \right) \, dt = \int \mathrm{r_m} \, dt$$

Observe que del lado izquierdo los diferenciales se cancelan:

$$\int \frac{dN}{N} = \mathrm{r_m} t + c$$
$$\ln N = \mathrm{r_m} t + c$$

Se despeja N, y se obtiene $N = Ce^{\mathrm{r_m} t}$. Luego, cuando $N = N_0$ entonces $t = 0$; por lo que la expresión se simplifica a $N_0 = Ce^0 = C$. Dando como resultado la expresión

$$N = N_0 e^{\mathrm{r_m} t}$$

Ejemplo 4 *De acuerdo con Illman* et al. *(2000) un gramo de* Chlorella emersonii *puede contener 29 kJ g^{-1} (energía por gramo). Si la tasa intrínseca es de 0.99 g d^{-1}, ¿cuántos gramos de* Chlorella *necesito para producir 5000 kJ? ¿Cuál es el tiempo de producción? Asuma un crecimiento exponencial, y un inóculo inicial con $N_0 = 1 \mu\mathrm{g}$ de* Chlorella*.*

En este caso, pensamos en el tamaño poblacional como biomasa, en lugar del número de individuos. El primer paso es calcular N para producir la cantidad deseada de energía, lo cual resolvemos con una simple conversión para obtener:

$$N = \frac{1\mathrm{g}}{29\cancel{\mathrm{kJ}}} \times 5000\cancel{\mathrm{kJ}} = 172.4138\mathrm{g}$$

Luego, despejamos t de la ecuación (2.5):

$$t = \ln \left(\frac{N}{N_0} \right) {\mathrm{r_m}}^{-1}$$

Se hacen las sustituciones correspondientes: $\mathrm{r_m} = 0.99$, $N_0 = 1 \times 10^{-6}\,\mathrm{g}$, $N = 172.414\,\mathrm{g}$, y se obtiene que el tiempo necesario para obtener una biomasa equivalente a una energía de 5000 kilojoule es $t = 19.2$ d.

2.1.2.2. Ejercicios

Ejercicio 6 *Si $\lambda = 1.027$ por semana. Escale λ de semanas a meses (1 mes = 4 semanas). Utilice la relación de $\lambda = e^{\mathrm{r_m}}$.*

Ejercicio 7 *Para los siguentes escenarios de la ecuación* (2.5)*:*

- $\mathrm{r_m}$ *negativo.*
- $\mathrm{r_m}$ *igual a cero.*
- $\mathrm{r_m}$ *positivo.*

Obtenga el límite:

$$\lim_{t\to\infty} N(t)$$

Dé una interpretación de los resultados, en términos de la población.

Ejercicio 8 *Demuestre, utilizando un razonamiento deductivo, que si $\mathrm{R_m} < 0$, la población decrece. Puede usar los resultados del ejercicio 7.*

Ejercicio 9 *Analice biológicamente el significado del resultado del ejercicio 7, cuando $\mathrm{r_m}$ es positivo.*

2.2. Crecimiento denso-dependiente

2.2.1. Matrices

COMADRE COMADRE DATA Ojo a la guia de usuario

2.3. Otras fuentes bibliográficas

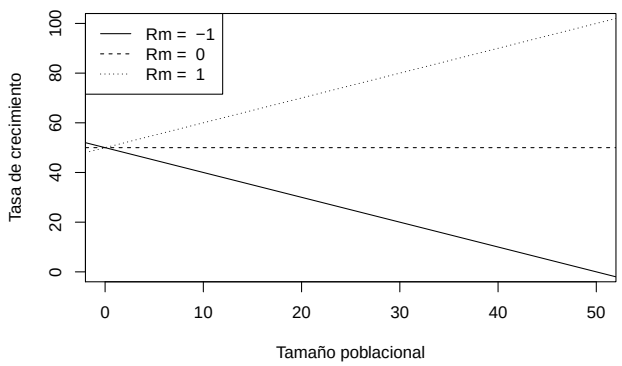
Esta sección está basada en los capítulos 4 y 5 de Neal (2004). En la sección 2.3 de Berryman y Kindlmann (2008), se desarrollan los mismos modelos básicos vistos aquí; en este mismo libro se presenta una buena introducción sobre la ecología de poblaciones como *sistemas*, en el capítulo 1.

Capítulo 3

Soluciones a los ejercicios

Ejercicio 1

```
plot(0,0,xlim = c(0,50),
     ylim = c(0,100) ,
     type = 'n',xlab = 'Tamaño poblacional',
     ylab = "Tasa de crecimiento")

val=numeric()

for(Rm in c(-1,0,1)){
  abline(50,Rm,lty=Rm+2)
  val=append(val,Rm)
}
legend("topleft",lty= 1:3,legend = paste('Rm = ',val))
```



R_m representa la pendiente de la recta.

Ejercicio 5

En 3 horas existen 12 periodos de 15 minutos ($t = 12$). Entonces aplicamos la ecuación (2.3):

$$100 \times 2^{12} = 409\,600 \text{ células}$$

Ejercicio 6

Obtener $r_m = \ln \lambda$; luego multiplicar $r_{mes} = r_m \times 4$ para obtener la escala a meses. Finalmente, transformar $\lambda_{mes} = e^{r_{mes}}$.

Ejercicio 8

Asumimos que r_m es cualquier constante positiva ($r_m \in \mathbb{R}^+$). Entonces $-1 \times r_m \in \mathbb{R}^-$.

Luego, tomamos el límite:

$$\lim_{t \to \infty} N_0 e^{-r_m t}$$

Que equivale a:

$$\lim_{t \to \infty} \frac{N_0}{e^{r_m t}}$$

Vemos que el denominador de la expresión anterior es un número que crecerá infinitamente. Si reemplazamos $e^{r_m t}$ por x, cuando $t \to \infty \Rightarrow x \to \infty$. Y quedamos con la expresión:

$$\lim_{x \to \infty} \frac{N_0}{x} = 0$$

Porque cuando $x \to \infty \Rightarrow x \gg N_0$. Es decir, cuando x se vuelve infinito, es mucho más grande que N_0, por tanto, el cociente tiende a cero, cuando x tiende a infinito.

Poblacionalmente, esto significa, que si una población mantiene una tasa intrínseca negativa, por un periodo de tiempo suficientemente largo, sufrirá un evento de extinción.

Índice alfabético

Apéndice A

Métodos numéricos para ecología de poblaciones

A.1. Simulación de ecuaciones diferenciales

Tomaremos el ejemplo de la ecuación (2.4), para mostrar un método para encontrar el tamaño de población, sin tener que utilizar cálculo (Barrantes Campos, 2015). Este es el *método de Euler*, que se explicará mediante un ejemplo.

Una derivada implica un cambio infinitesimal de una variable en relación a otra. Por ejemplo, el cambio en el número de individuos de una población en un momento pequeñísimo de tiempo, puede representarse como la diferencia de la población, entre la duración de ese pequeño intervalo de tiempo:

$$\frac{dN}{dt} \approx \frac{\Delta N}{\Delta t} = \frac{N_t - N_{t-\Delta t}}{\Delta t}$$

Si sustituimos en la ecuación (2.5), tenemos:

$$\frac{N_t - N_{t-\Delta t}}{\Delta t} = r_m N$$

Arreglando la expresión anterior, podemos despejar en terminos de N_t:

$$N_t = N_{t-\Delta t} + r_m N_{t-\Delta t} \Delta t$$

Hay que resaltar que esta **no** es una solución exacta; sino, una aproximación. Entre más pequeño se haga Δt, más se aproximará el resultado, al valor exacto dado por (2.5). En casos donde no existe una solución analítica, o simplemente, no es sencillo resolver la ecuación, siempre se puede recurrir a los métodos numéricos, para tener una idea de la solución real.

Para programar este sencillo ejemplo, necesitamos varios pasos:

- Definir un valor inicial de la población, y el valor de t en el cuál queremos conocer el tamaño de población.
- Definir el Valor de r_m.
- Establecer un criterio para guardar el valor de N_t, cada cierto lapso de tiempo. (Para no crear un objeto virtual innecesariamente grande)
- Crear un objeto para guardar el tamaño de la población, y los puntos de tiempo a los que está asociada.
- Definir el tamaño de Δt, y calcular el número de iteraciones necesarias hasta llegar al final del periodo de tiempo de interés.
- Crear un bucle, y ejecutar iteractivamente la integración de Euler.
- Definir un criterio para detener el algoritmo.

El siguiente algoritmo generaliza todas las funciones dependientes de $N_{t-\Delta t}$.

$$N_t = N_{t-\Delta t} + f(N_{t-\Delta t}, \mathbf{c})\,\Delta t$$

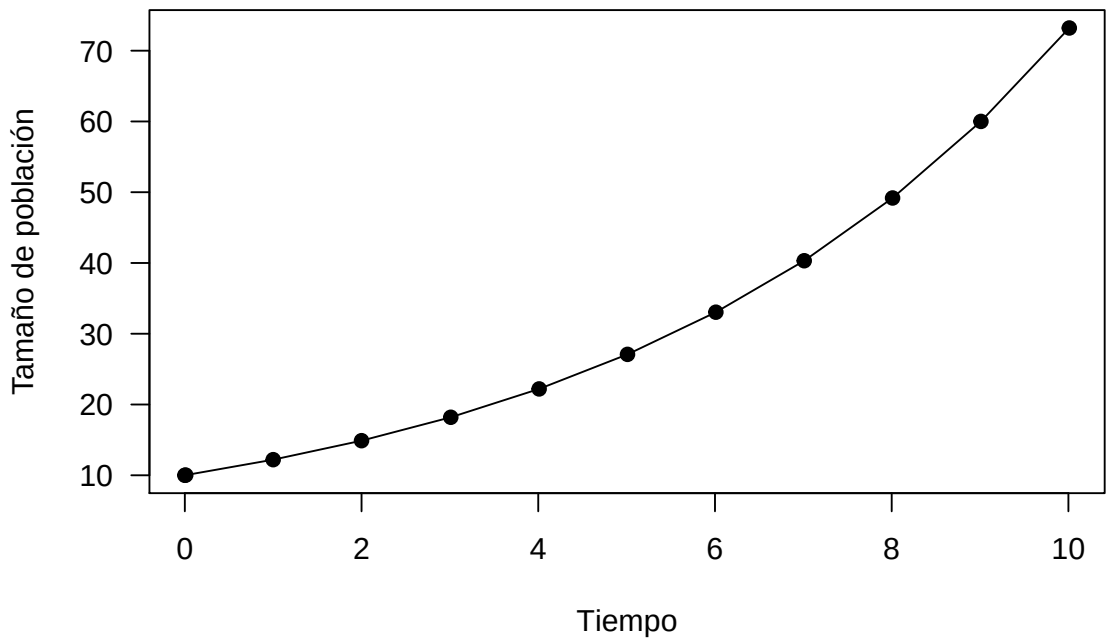
Donde $\mathbf{c}$ son constantes.

```
euler <- function(fooName, valInic, tiempoParar, deltaT,guardarCada,...){
  arg <- list(...)
  fn <- get(fooName)

  #Encuentra los argumentos provistos
  argName <- match.arg(names(arg),#arg provistos
                       formalArgs(fn),#arg existentes
                       several.ok = TRUE)
  #Nombra la lista con los nombres de los argumentos provistos
  names(arg) <- argName
```

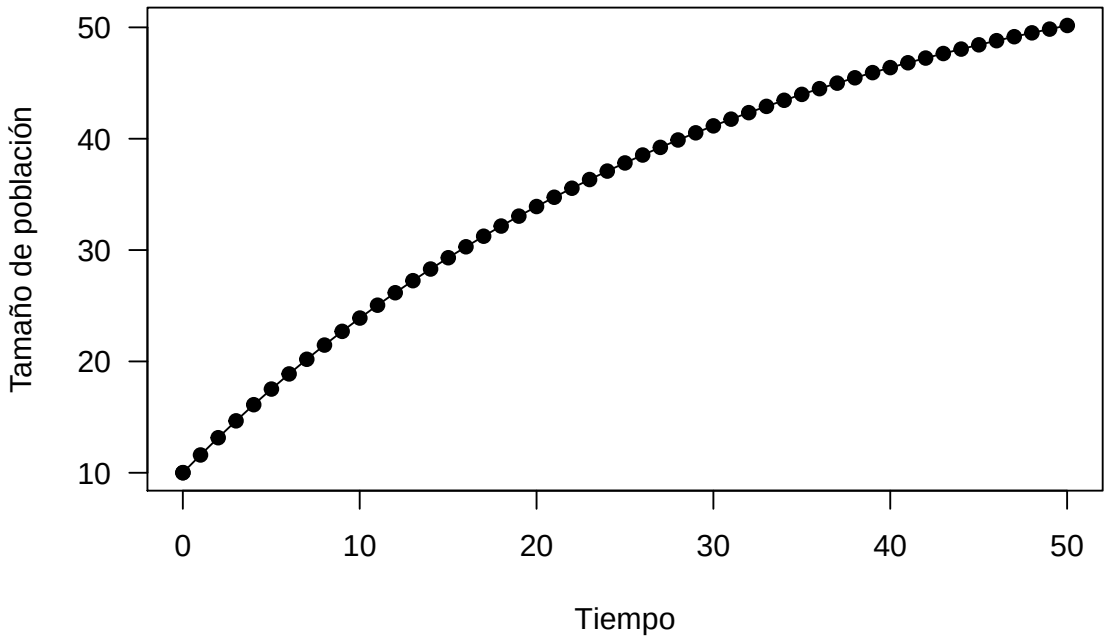
```
  val <- numeric()
  val[1] <- valInic

  valTmp <- numeric()
  valTmp <- val[1]

  #Completa la lista de argumentos con N[t-1]
  arg[[ (length(arg)+1) ]] <- valInic
  totalArg <- length(arg)
  #Escribe todos los nomres de los argumentos, para do.call
  names(arg) <- formalArgs(fn)#Encuetra los nombres de los argumentos

  tiempo <- numeric()

  tNow <- 0
  tiempo[1] <- 0

  while( tNow < tiempoParar ){
    valTmp <- valTmp + do.call(fn,args = arg)*deltaT
    tNow <- tNow + deltaT

    if( tNow%%guardarCada <=deltaT ){
      val <- append(x = val,values = valTmp)
      tiempo <- append(x = tiempo, values = tNow)
      arg[[totalArg]] <- valTmp
    }
  }

  return( list(
    poblacion = val,
    tiempo = tiempo,
    tNow = tNow,
    arg = arg
  ) )

}
```

Por ejemplo

```
diffG1 <- function(rm, N) N * rm
N0 <- 10

Resultados1 <- euler(fooName = "diffG1", valInic = N0, tiempoParar = 10, deltaT = 1/10
    guardarCada = 1, rm = 0.22)

diffG2 <- function(rm, Kmax, N) rm * (1 - N/Kmax)

N0 <- 10

Resultados2 <- euler(fooName = "diffG2", valInic = N0, tiempoParar = 50, deltaT = 1/10
    guardarCada = 1, rm = 1.92, Kmax = 60)

plot(Resultados1$tiempo, Resultados1$poblacion, type = "p", xlab = "Tiempo",
    ylab = "Tamaño de población", las = 1, pch = 21, bg = 1)
lines(Resultados1$tiempo, Resultados1$poblacion)
```



```
plot(Resultados2$tiempo, Resultados2$poblacion, type = "p", xlab = "Tiempo",
    ylab = "Tamaño de población", las = 1, pch = 21, bg = 1)
lines(Resultados2$tiempo, Resultados2$poblacion)
```



Apéndice B

Tutorial de R con RStudio

B.1. Crear un proyecto en *RStudio*

Crear un proyecto en **RStudio** para cualquier proyecto con **R**, es importante. Los proyectos organizan los documentos en una sola carpeta, y son fundamentales para el control de versión con un software como **Git**. Primero crearemos un proyecto sencillo para conocer la mecánica básica, y luego haremos un proyecto con un repositorio en línea, en **GitHub**.

El primer paso es ir a `file --> New Project`. También existen botones específicos para crear proyectos. Creamos una carpeta en una ubicación que nos permita tener derechos de administrador, e idealmente, fuera de cualquier carpeta de sincronización en línea. En caso de que quieran tener un respaldo en la nube, se recomienda pausar la sincronización mientras se trabaja en el proyecto.

RStudio nos guiará por los siguientes pasos:

- `Crear un Proyecto`: Escogemos que sea un nuevo directorio.
- `Tipo de Proyecto`: Escogemos *nuevo proyecto*.
- `Crear`: Escogemos la carpeta, y el nombre del proyecto. NO marcamos *crear repositorio con Git*.

Ahora, en **RStudio** vamos a `File --> New file --> R script`. Este archivo solo soporta código en **R**, con la gran ventaja de que colorea las funciones, variables y estructuras más comunes; lo cual, hace que el código sea más legible.

B.2. Funciones básicas en R

Ahora que hemos creado un proyecto, y tenemos un lienzo en blanco, empezamos por ver las funciones más elementales **R** puede ser utilizado de manera tan simple, como una calculadora. Contiene, por supuesto, todas las operaciones básicas como: adición, substracción, multipliación, división, potencias, y logaritmos.

```
1 + 1 # adición

1-1 # substracción

1*2 # multiplicación

1/2 # división

2^(8) #potencia

log(2) # logaritmo natural

log10(2) #logaritmo base 10

log(x = 2, base = <n>) #logaritmo base <n>, donde <n> se
                       #   reemplaza por cualquier número.
```

Notar que en el código, cualquier línea de texto precedida de # es un comentario, que no será evaluado por el computador.

También podemos mezclar operaciones de la forma que convenga. Siempre considerando las reglas de prioridad por paréntesis.

```
log( x= 1 / (1 + (5/2) ) , base = 2^(-1/2) )
```

Luego, como la mayoría de lenguajes de programación, podemos asignar valores a un objeto y utilizarlo después en otra operación:

```
a <- log10(4/3)

b <- a^2
```

```
c <- b^2 - a


c
```

Esto quiere decir que podemos crear un objeto con el operador <−, que pueden ser datos, o resultados de otras operaciones, para incluirlo en una nueva función. Por lo que la salida de una función puede ser la entrada de la próxima.

B.3. Estructuras de datos

R puede manejar objetos muy complejos; sin embargo, estos objetos generalmente se componen de partes muy sencillas. Revisaremos éstas partes sencillas, y luego crearemos un objeto más complejo.

Vectores

Un vector en programación, es una colección de *n* datos. A diferencia de los vectores en física, que señalan una magnitud asociada a una dirección. Podemos pensar que un vector en **R** equivale a una matriz de *n* filas, y solo una columna.

```
vect1 <- c(1,2,3,4,5,6,7,8,9,0) #es una concatenación de
# números, que se crea con la función 'c(...)'

vect2 <- rnorm(10) # son diez números al azar obtenidas
# de una distribución normal estándar
```

Los elementos de un vector pueden ser llamados utilizando un sub-índice. Éste inicia en 1, hasta *n*. Donde *n* es la cantidad de elementos en un vector. También es posible, llamar varios elementos a la vez, si el subíndice del vector es otro vector.

```
vect1[3]


vect1[c(3,4,5)]


vect1[3:5]
# 3:5 crea una secuencia de enteros, que incrementa en 1 a la vez.

vect1[rep(5,times=10)] # rep, es una función que repite un
```

```
#número un determinado número de veces.
```

Los vectores pueden ser datos en un archivo externo, o resultado de funciones u operaciones. A diferencia de otros lenguajes **R**, maneja vectores de una forma más intuitiva.

```
vec3 <- vec1 + vec2 # es un nuevo vector basado en la adición de los dos primeros.
```

Matrices

Las matrices son un arreglo de datos en dos dimensiones, es decir, filas y columnas. Por covención, cuando decimos que una matriz es de tamaño $f \times c$, nos referimos a que tiene f filas, y c columnas. Las filas siempre se nombran primero que las columnas.

```
(m1 <- matrix(1:9, ncol=3, byrow = T) )

(m2 <- matrix(1:9, ncol=3, byrow = F) )
```

Los elementos de una matriz se llaman por la combinación de filas y columnas a la que corresponde. Del ejemplo anterior, si quisiéramos obtener el elemento central de la matriz `m1`, lo llamamos así `m1[2,2]`.

Si quisiéramos llamar toda la primer columna, entonces escribimos `m1[,1]`. O la primer y tercer fila `m1[c(1,3),]`.

Las operaciones con matrices suelen ser más delicadas y existen operadores específicos para ellas.

Marco de datos o *Data frames*

Esta estructura es similar a una matriz, con la diferencia, que algunas de sus columnas pueden contener *factores*, y no solo valores numéricos. Estas son las estructuras con las que representamos un diseño experimental, por ejemplo:

```
##   Var1 Var2          z
## 1   T1    a  0.7564083
## 2   T2    a  0.5267798
## 3   T1    b -0.3691496
## 4   T2    b  0.1611349
## 5   T1    c  0.1978272
## 6   T2    c  1.7962886
```

Arreglos o *arrays*

Estos son matrices de 3 o más dimensiones. Por ejemplo, si tenemos una serie de fotografías con la misma resolución, en el mismo lugar, podemos representar los pixeles como una matriz $f \times c$, y el tiempo como una dimensión adicional. Si ponemos en rápida sucesión las matrices, tendremos un video o película.

Listas

Las listas son colecciones de cualquiera de los objetos anteriores (y otros que no hemos visto).

```
l1 <- list(
  vector = vect1,
  matriz = m1

)
```

Podemos llamar a los elementos de una lista, de dos formas: si conocemos el orden de los elementos de la lista, entonces, escribimos el índice dentro de dos pares de corchetes rectos: `l1[[1]]`, para llamar el vector y `l1[[2]]`, para llamar la matriz. Si conocemos los nombres de los elementos de la lista, usamos la siguiente forma: `l1$vector`, para el vector; y `l1$matriz`, para la matriz.

Una vez dentro del objeto de la lista, podemos llamar sus elementos de manera tradicional. Por ejemplo, `l1$matriz[2,2]`, para llamar el elemento central de la matriz *dentro de la lista*.

B.4. Funciones

Las funciones se representan por un nombre, seguido de un paréntesis redondo. Todo lo que esté dentro de ese paréntesis son sus argumentos. Para finalizar la función, cerramos con un paréntesis redondo derecho:

```
funcion(argumento1 = valor1,  argumento2 = valor 2)
```

Nosotros podemos crear nuestras propias funciones en R. El procedimiento es sencillo:

1. Llamamos a la función con un nombre, y declaramos que se trata de una función
2. Nombramos los argumentos de la función. Podemos asignar valores por defecto. Los argumentos deben ir entre paréntesis redondos.

3. Escribimos el cuerpo de la función entre paréntesis tipo llave {}. El cuerpo de la función debe terminar con *un solo* objeto que será retornado como salida del proceso.

Por ejemplo, si queremos hacer nuestra propia función para calcular un promedio. Primero debemos entender la fórmula subyacente:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

Es decir, sumamos todos los elementos de un vector de valores, y lo dividimos por el *tamaño*[1] del vector. Entonces, nuestro argumento será un vector, y nuestra salida, un valor único con el promedio.

```
promedio <- function(vectorX){

   sumaX <- sum(vectorX)

  n <- length(vectorX) #length(), es una función que calcula
                       #   el tamaño del vector.
  valor <- sumaX / n

  return(valor)

}

# Ahora probamos la función, tomando el promedio de un
#       vector de números normales con media igual a cero
#       Esperamos, que nuestro promedio sea un valor cercano
#         a cero.

valores <- rnorm(100)

(promedio(valores))
```

```
## [1] 0.1361579
```

[1] El tamaño del vector, se refiere al total de elementos que lo conforman

Apéndice C

Asignaciones

C.1. Tarea 01: ¡Hola mundo con *Rmarkdown*!

Objetivo: Verificar que el estudiante ha instalado, y maneja el ambiente de trabajo que se utilizará durante el curso.

Primero revisa los enlaces provistos en el wiki.

Actividades

- Haz un nuevo proyecto en **RStudio**, que se llame *Tarea01*. Ver pasos en sección B.1.
- En la consola de **R**, escribe `install.packages(rmarkdown)`, con todas las dependencias. O instala el paquete desde **RStudio** como se mostró en el wiki.
- En **RStudio** `File--> New File --> R Markdown`.
- Crea una sección principal que se llame *Información profesional*.
- Luego, crea una sección secundaria que se llame *Intereses*. Usa bullets para nombrar algunos intereses profesionales.
- Luego, crea una sección secundaria llamada *Experiencia Laboral*, si aplica. Nombra algunos trabajos relacionados con el curso de Ecología de Poblaciones.
- Crea una sección principal que se llame *Integración con R*
- Consigue algunos datos interesantes en internet. Deben ser datos para graficar, por tanto deben tener dos columnas, y varias filas. Puedes ir a Wolfram Alpha. Guarda los datos como un texto delimitado por comas (`.csv`).

- En **R** o **RStudio** corre el comando `?read.table`. Para correr un comando en **RStudio** apreta `Cntrl + R`.
- Crea un "*chunk*" de código. Esto se hace en **RStudio**, busca un botón en la barra especial de *rmarkdown* que diga `insert`, luego escoge `R`.
- Lee la tabla y asignala a un objeto:

```
datos <- read.table(<ruta_de_archivo_en_comillas>,
                    header = TRUE,
                    sep = ",")
```

- Grafica los datos en un nuevo "*chunk*". Usa el método que prefieras. Hay mucho material de cómo hacer gráficos en R. Por ahora, un gráfico básico es suficiente.
- Ahora, haz otra sección llamada *Bibliografía*. En un párrafo escribe una mini-revisión de algún tema que domines y del que dispongas referencias bibliográficas. Usa los mecanismos de citas de *rmarkdown*
  - Cita en texto con `@citationKey`
  - Cita en paréntesis con `[@citationKey]`

Importante: Para que las citas funcionen, debes agregar unas opciones en la *cabecera* del documento (*YAML header*):

```
bibliography: <tu_archivo_bib>.bib
csl: apa.csl
```

El archivo `apa.csl` se puede encontrar en google. Es un archivo de estilo APA, para dar formato a la bibliografía. Revisa el repositorio de CSL de Zotero, en busca de las revistas disponibles.

- Por último, corre el documento con el botón `knit`. Envía el documento `.Rmd` y el `.pdf` al profesor (`dawidh15@gmail.com`).

C.2. Tarea 02: Ejercicios de crecimiento

Se recomienda hacer primero todo en papel, y luego pasarlo en limpio usando *Rmarkdown*. LINKS para LatexMath

Ejercicio 1 *Resuelva el siguiente ejercicio de crecimiento exponencial*

En un laboratorio se cultiva una especie presa para un programa de reintroducción de una especie de pez. En el laboratorio, se inició un proyecto de mejora en la producción de la presa, y se ha diseñado un experimento para aumentar el valor nutricional de las presas.

Se cuenta con un presupuesto de 2×10^6 CRC para la producción de animales presa en el proyecto. Además, el diseño experimental requiere de 40 recipientes acondicionados con diferentes tratamientos. Las presas crecen con una tasa de crecimiento intrínseco de 0.098. Además, el inóculo inicial es de 1000 individuos por recipiente. Si se sabe que el costo de mantenimiento por organismo-día es de 0.5 CRC/(ind d):

¿Cuántos organismos por recipiente se pueden cultivar sin sobrepasar el dinero disponible? ¿Cuánto tiempo, en días, se necesitan para alcanzar esa cantidad?

Tips:

- Este es un problema de mínimos. Primero hay que buscar la función a minimizar. Luego, uno encuentra el valor apropiado del parámetro de interés cuando la función se minimiza.
- Para minimizar una diferencia, use el valor absoluto de la diferencia: $|x - y|$.
- Use la función `optim` o `optimize`. Una vez que tenga la función que desea minimizar escrita en *R* use este código:

```
out <- optim(par = 0,fn = <nombre_de_funcion_para_minimizar>,
       control = list(reltol=0.01),
       method = "Brent",
       lower = <numero>,
       upper = <numero>)
```

- Antes se recomienda buscar la ayuda de la función en la consola de **R**, al escribir `?optim`. Revise los ejemplos, y lea detalladamente la ayuda.

Bibliografía

Barrantes Campos, H. 2015. Introducción a las ecuaciones diferenciales. EUNED, San José, Costa Rica, 2 edición., 328 págs.

Berryman, A. A. y P. Kindlmann. 2008. Population systems: a general introduction. Springer, 2 edición., 222 págs.

Grafton, R. Q., J. Kirkley, T. Kompas y D. Squires. 2006. The Economics of Fishing and Fisheries Economics. *En* Economics for Fisheries Management, cap. 1. Ashgate, págs. 1–23.

Illman, A. M., A. H. Scragg y S. W. Shales. 2000. Increase in Chlorella strains calorific values when grown in low nitrogen medium. *Enzyme and Microbial Technology* 27(8): 631–635. URL http://www.sciencedirect.com/science/article/pii/S0141022900002660.

Neal, D. 2004. Introduction to population biology. Cambridge, Cambridge, UK, 393 págs.