Funciones matemáticas, de crecimiento y otras.

Uno de los temas más interesantes de la silvicultura y de la Ecología en general es el estudio del crecimiento, para lo cual se debe dominar todo lo relativo a su modelación y manejo de funciones. Iniciamos con la nomenclatura sobre crecimiento y luego se mostratrán algunas funciones importantes para su estimación.

Función de crecimiento. Describe el comportamiento del tamaño de un organismo o población con la edad (variaciones de crecimiento). El crecimiento es el resultado de interacción entre dos factores opuestos: una tendencia hacia un crecimiento casi ilimitado dependiente del potencial biótico de los individuos, actividad fotosintética, absorción de nutrientes, procesos catabólicos y anabólicos y por otra parte las condiciones de un entorno que limita por competencia con otros individuos, escasez de recursos, autolimitaciones, etc.

La representación de la evolución del tamaño de algún parámetro dasométrico (diámetro, altura, área basal, volumen, etc.) o de una masa se hace por medio de una función, generalmente una sigmoide derivada de los factores opuestos mencionados y con elementos resaltantes como un punto de inflexión y una asíntota horizontal. Figura 1.

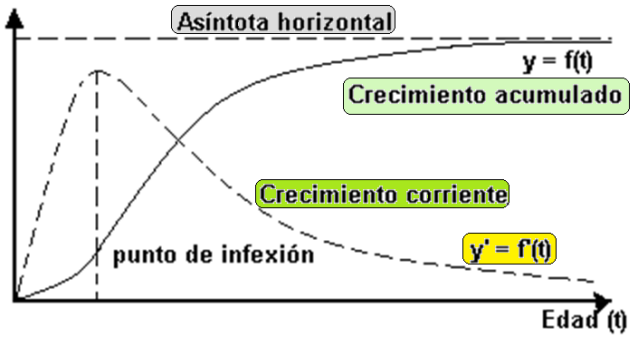


Figura. Elementos de una sigmoide de crecimiento y su primera derivada

En este curso ya saben crear $y = f(t)$ en diversos aspectos y se mostrarán otras propuestas para llegar a otras funciones de crecimiento. A la primera derivada de la curva de crecimiento, y'= f'(t), se le conoce como crecimiento corriente, osea: $\frac{\partial y}{\partial t} = y' = f'(t)$

Se puede observar que en las primeras edades pesan más los elementos del factor positivo que dispara el crecimiento dando lugar a una curva cóncava hacia arriba, pero a medida que aumentan las restricciones se produce incluso un cambio en la curvatura un punto de inflexión que corresponde con el máximo de la curva de crecimiento corriente. A partir del punto de inflexión, como siguen aumentando las restricciones el crecimiento se va haciendo más lento hasta que se estabiliza, lo que corresponde con la asíntota horizontal.

La sigmoide representa la evolución del tamaño de cualquier variable con el tiempo, pero el verdadero crecimiento, lo representa la derivada de la anterior, lo que ha dado a lugar de confusiones de terminología:

Los ingleses hablan de *growth function* (función de crecimiento) al referirse a la forma integral e *increment function* (función de incremento) a su derivada.

Otra literatura habla de crecimiento corriente en vez de función de incremento o de incremento corriente. Entre nosotros se habla de "función de crecimiento" al referirse a la integral y de "crecimiento corriente" al hacerlo con la derivada de la anterior.

Son muchas las aplicaciones de estos conceptos, fuera de conocer la evolución del crecimiento, esta función aplicada a la altura de los árboles dominantes permite conocer los índices de sitio (calidad) de un lugar, permiten la elaboración de tablas de producción, encontrar los valores máximos de los crecimientos promedio y corriente, determinar diversos turnos y hasta lograr proyecciones en el tiempo. Para ello las características de una buena función de estas debe tener:
- Un punto de inflexión
- Una asíntota horizontal, a veces dos
- Un comportamiento lógico matemáticamente hablando, y
- Un comportamiento biológico

Modelos

Existe una gama alta de modelos, más de 80 que muchos autores agrupan en 8 grandes grupos y aun en solo dos, pero la clasificación más interesante se basa en la forma diferencial o sea cuando en la ecuación

$$y' = dy/dt$$

y' es la variable dependiente y la independiente, el tiempo "t" (ej: la edad del árbol) para un crecimiento acumulado "y". Cuando se expresan así los modelos de crecimiento aparece una clara interpretación biológica puesto que pueden ser descompuestos en los dos factores opuestos ya mencionados, cuyo efecto combinado determina la pauta de crecimiento. Con base en ello (Zeide, 1993) propuso dos categorías al asumir que el factor positivo del crecimiento es proporcional al tamaño acumulado "y" elevado a una cierta potencia, mientras que el negativo se asocia con la edad elevada también a una potencia, dando lugar a los modelos de decrecimiento potencial: "DP"

$$DP: \frac{dy}{dt} = ky^p t^q$$

o con la edad como exponente o a los modelos de deslizamiento exponencial o "DE", así:

$$DE: \frac{dy}{dt} = ky^p e^{qt}$$

En los cuales k>0, p>0, q<0 son los parámetros específicos de cada uno de los modelos. Entonces la clave del estudio del crecimiento la tiene las fortalezas en el manejo de las funciones.

Conceptos claves. Se recuerdan algunos al trabajar con funciones:
- Cualquier variable o número a la potencia cero es igual a 1: $x^0 = 1$.
- Uno elevado a cualquier potencia es 1: $1^x = 1$.
- Infinito más 1 es infinito. $\infty + 1 = \infty$
- Uno sobre infinito es cero (el reciproco de infinito, $\infty^{-1} = 0 = \frac{1}{\infty} = 0$).

• Un número mayor que 1 elevado a infinito es infinito: $1.2^{\infty} = \infty$.

• Una fracción (ej: 0.99) elevada a infinito es cero: $0.99^{\infty} = 0$.

• Potencias negativas son recíprocos: $x^{-b} = \frac{1}{x^b}$.

• Potencias fraccionarias son raices: $x^{\frac{1}{3}} = \sqrt[3]{x}$.

• La base de los logaritmos naturales, e, es 2.718 28.

• Una muy utilizada: $e^{-\infty} = \frac{1}{e^{\infty}} = 0$.

R tiene diversas librerías con procesos incorporados para las funciones logarítmicas, de probabilidades y trigonométricas.

Funciones logarítmicas. La función logarítmica está dada por:

$$y = a\ln(bx)$$

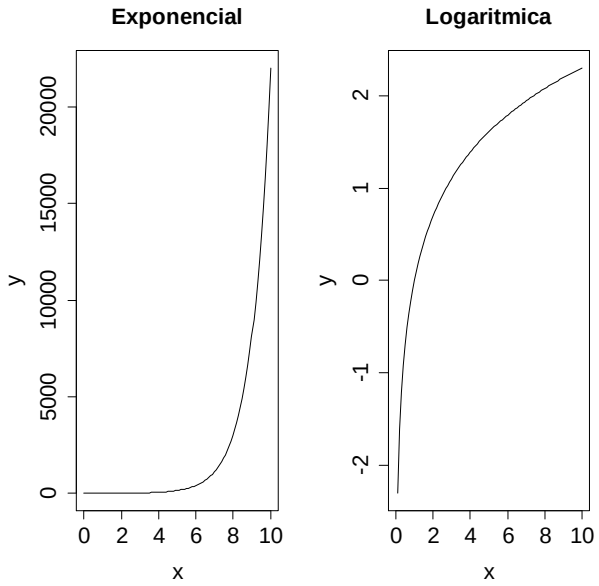
La función exponencial, en la cual la variable respuesta *y* es el antilogaritmo de la variable continua explicatoria *x*, es dada por:

$$y = ae^{bx}$$

Las dos anteriores funciones son suavizadas y dibujadas en R por medio de series, por ejemplo, con 100 o más valores regularmente espaciados, a décimos (1/10), entre min(x) y max(x):
x<-seq(0,10,0.1)

En R la función **exponencial** es ***exp*** y la función para **logaritmo natural** (ln) es ***log***. Sea a=b=1. La grafica de las funciones exponencial y logaritmica se escribe así:

```
y<-exp(x)
par(mfrow=c(1,2))
plot(y~x,type="l",main="Exponencial",cex.axis=1.5,cex.lab=1.5,cex.main=1.5)
y<-log(x)
plot(y~x,type="l",main="Logaritmica", cex.axis=1.5,cex.lab=1.5,cex.main=1.5)
```

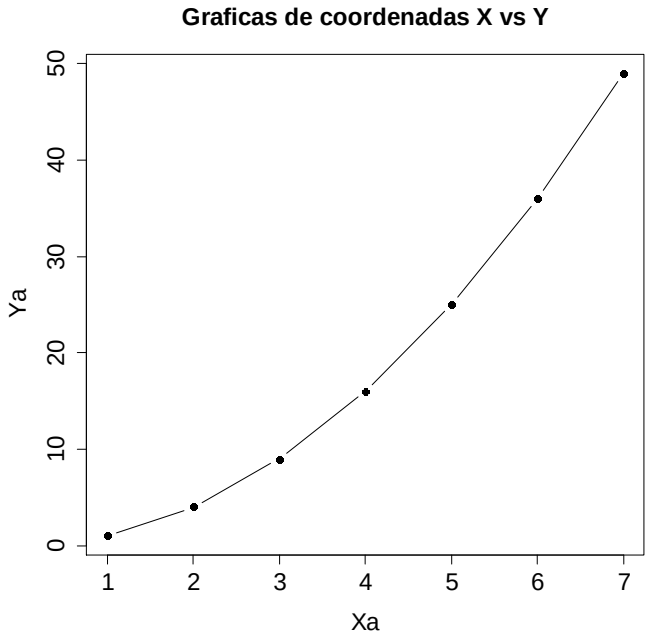


Notará que la función **plot** puede usarse en forma alternativa, bien sea especificando las coordenadas cartesianas de la línea usando **plot(x,y)** en vez de la fórmula modelada **plot(y~x)**. Las dos funciones anteriores son muy útiles en la modelación de procesos exponenciales de crecimiento y declinación Ejemplo sean las dos variables siguientes:

Xa<-c(1, 2, 3, 4, 5, 6, 7)

Ya<-c(1, 4, 9, 16, 25, 36, 49)

plot(Xa,Ya, main="Graficas de coordenadas X vs Y",pch=16,type="b", cex.axis=1.5, cex.lab=1.5, cex.main=1.5**)**



Funciones potenciales. Hay una importante familia de dos funciones matemáticas biparamétricas de la forma:

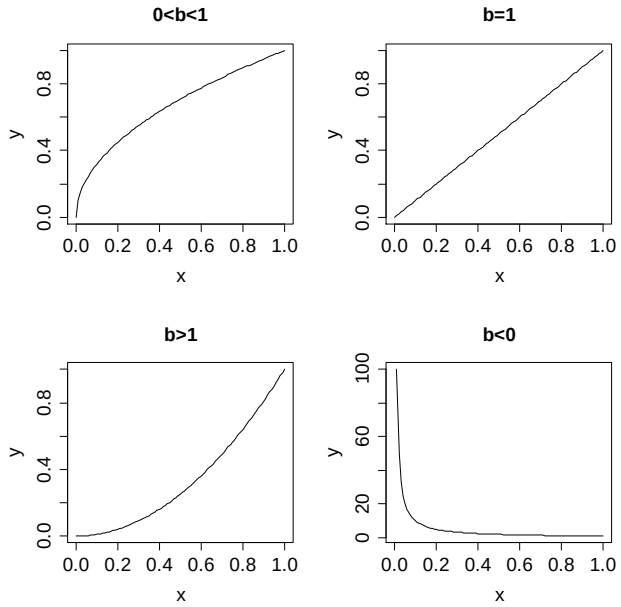
$$y = ax^b$$

conocidas como **formas potenciales** (o leyes de potencia).

Dependiendo del valor de la potencia, *b*, esta relación puede tomar una de *y* formas. En el caso trivial de **b = 0** la funcion es ***y = a*** (una perfecta línea recta horizontal).

Las 4 formas más interesantes son:

```
par(mfrow=c(2,2))
x<-seq(0,1,0.01)
y<-x^0.5
plot(x,y,type="l",main="0<b<1", cex.axis=1.5, cex.lab=1.5, cex.main=1.5)

y<-x
plot(x,y,type="l",main="b=1", cex.axis=1.5, cex.lab=1.5, cex.main=1.5)
y<-x^2

plot(x,y,type="l",main="b>1", cex.axis=1.5, cex.lab=1.5, cex.main=1.5)
y<-1/x

plot(x,y,type="l",main="b<0", cex.axis=1.5, cex.lab=1.5, cex.main=1.5)
```



Los parámetros ***a*** y ***b*** son fáciles de estimar con los datos, al linealizar, por medio de la **transformación log-log**,

$$\log(y) = \log(ax^b) = \log a + b\log(x)$$

que, en la escala bilogarítmica, da el intercepto como ***log(a)*** y, ***b*** la pendiente. Estas son llamadas a menudo **relaciones alométricas** pues si b≠1 la proporción de *x* que se convierte en *y* varía con ***x***.

Esta forma, tiene importantes aplicaciones en entomología ecológica en un amplio rango de análisis estadísticos conocidos como las leyes de potencia de Taylor, que relacionan las medias y las varianzas de una muestra.

Recuerde que, usualmente en estadística elemental, la varianza se asume como una constante (o sea, la varianza no depende de la media). Sin embargo, en muchos datos de campo, Taylor encontró que la varianza se incrementaba con la media de acuerdo con esta ley, de tal forma que en escalas o ejes log-log los datos de muchos sistemas caían por encima de una línea a través del origen con pendiente = 1 (cuando, en el patrón mostrado por unos datos con una distribución de Poisson, la varianza es igual a la media) y por debajo de una línea a través del origen con una potencia de 2.

La ley de potencias de Taylor establece que, para un sistema particular:

- El logaritmo de la varianza (log(varianza)) es una función lineal del logaritmo de la media (log(media));
- La dispersión alrededor de esta línea recta es pequeña;
- La pendiente de la regresión log(varianza) vs log(media) es mayor que 1 y menor que 2;
- Los valores de los parámetros de la regresión log-log son características fundamentales del sistema.

Funciones polinomiales. Son aquellas en la cual la variable explicatoria x aparece varias veces cada una de ellas elevada a una potencia diferente. Son útiles para describir curvas arqueadas, inflexiones o puntos de máxima locales, similar al siguiente ejemplo con:

```
x<-seq(0,10,0.1)
y1<-2+5*x-0.2*x^2
```

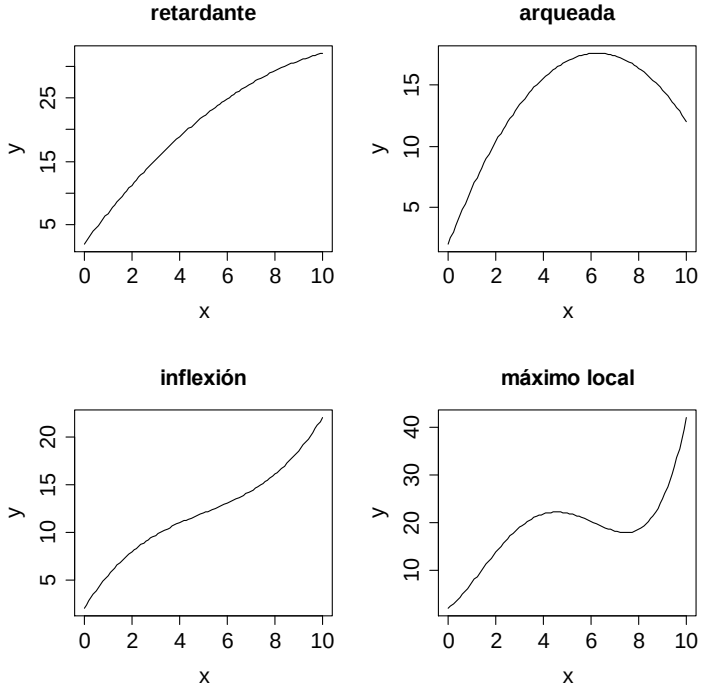
La gráfica izquierda superior muestra una función positiva desacelerada modelada por un polinomio cuadrático.

Haciendo negativo el coeficiente del término x^2 mayor se produce una curva con una joroba como en la parte superior derecha de la gráfica:
y2<-2+5*x-0.4*x^2

Las polinomiales cúbicas pueden mostrar puntos de inflexión como en la parte inferior de la gráfica:
y3<-2+4*x-0.6*x^2+0.04*x^3

Finalmente, polinomiales de orden 4 son capaces de producir curvas con máximos locales, como en la parte inferior derecha de la gráfica:
y4<-2+4*x+2*x^2-0.6*x^3+0.04*x^4

```
par(mfrow=c(2,2))
plot(x,y1,type="l",ylab="y",main="retardante", cex.axis=1.5, cex.lab=1.5, cex.main=1.5)
plot(x,y2,type="l",ylab="y",main="arqueada", cex.axis=1.5, cex.lab=1.5, cex.main=1.5)
plot(x,y3,type="l",ylab="y",main="inflexión", cex.axis=1.5, cex.lab=1.5, cex.main=1.5)
plot(x,y4,type="l",ylab="y",main="máximo local", cex.axis=1.5, cex.lab=1.5, cex.main=1.5)
```



Entonces se tiene una amplia gama de posibles modelos, dependieno de nuestros datos.

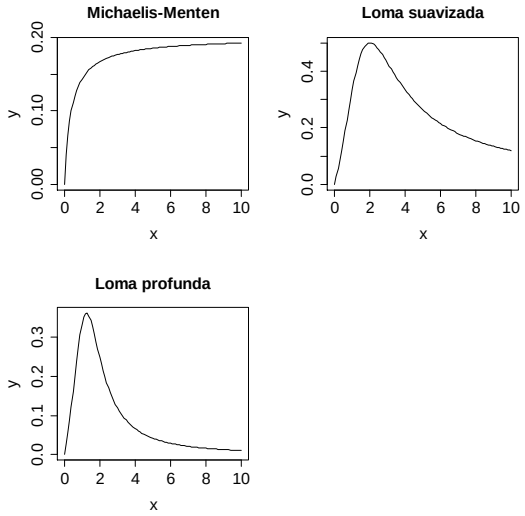
Polinomiales inversas. Constituyen una clase importante de funciones ideales para ajustar modelos lineales generalizados con errores tipo *función gamma* y función de encadenamiento como:

$$\frac{1}{y} = a + bx + cx^2 + dx^3 \cdots + zx^n$$

Dependiendo del orden de las polinomiales (máxima potencia) y de los signos de los parámetros se producen varios tipos de formas:
par(mfrow=c(2,2))
y1<-x/(2+5*x)

y2<-1/(x-2+4/x)
y3<-1/(x^2-2+4/x)

```
plot(x,y1,type="l",ylab="y",main="Michaelis-Menten",cex.axis=1.5, cex.lab=1.5, cex.main=1.5)
plot(x,y2,type="l",ylab="y",main="Loma suavizada", cex.axis=1.5, cex.lab=1.5, cex.main=1.5)
plot(x,y3,type="l",ylab="y",main="Loma profunda", cex.axis=1.5, cex.lab=1.5, cex.main=1.5)
```



Hay dos formas de parametrizar la ecuación Michaelis–Menten:

$$y = \frac{ax}{1+bx}; \quad y \quad y = \frac{x}{c+dx}$$

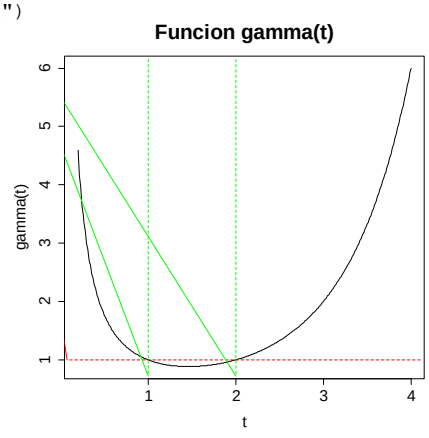
En el primer caso, el valor asintótico de *y* es a/b y en el segundo 1/d.

Función Gamma. Ya vista en distribuciones diamétricas y, acá en su contexto más matemático. La función gamma **Γ(t)** es una extensión de la función factorial, ***t*!**, a números reales positivos:

$$\Gamma(t) = \int_0^\infty x^{t-1} e^{-x} dx$$

Luce similar a esto:

```
t<-seq(0.2,4,0.01)
plot(t,gamma(t),type="l",main="Funcion gamma(t)", cex.axis=1.5, cex.lab=1.5, cex.main=2)
abline(h=1,lty=2,col="red")
abline(v=1,lty=2,col="green")
abline(v=2,lty=2,col ="green")
```



Note que $\Gamma(t)$ es igual a 1 en ambos valores de t=1 y t=2. Para valores enteros de t: $\Gamma(t+1)=t!$.

Funciones asintóticas. En la mayoría de casos la función asintótica más usada es:

$$y=\frac{ax}{1+bx}$$

la cual cambia de nombre de acuerdo con la disciplina que la usa. Por ejemplo, en bioquímica es llamada Michaelis–Menten, ya mostrada antes, y muestra la tasa de reacción como función de la concentración enzimática; en ecología se la conoce como la ecuación de disco de Holling y muestra la tasa de consumo de predadores como función de la densidad de presas. La función pasa por el origen y asciende con los retornos decrecientes hasta un valor asintótico al cual el incremento de *x* no produce más incrementos en *y*.

Exponencial biparamétrica. Otra función asintótica muy común es la exponencial asintótica:

$$y=a(1-e^{-bx})$$

biparamétrica y, cualquiera de las dos anteriores podría describir en muchos casos unos datos igualmente bien. Veamos el comportamiento de estas funciones en sus límites, iniciando con la asintótica exponencial en x = 0:

$$y=a(1-e^{-bx})=a(1-e^{-b*0})==a(1-e^{0})=a(1-1)=0$$

o sea que pasa por el origen. Para x=∞,

$$y=a(1-e^{-b\infty})=a(1-e^{-\infty})==a(1-0)=a$$

muestra que es asintótica con valor *a*.

Para la *Michaelis–Menten* el hecho de que para x=∞, y=∞/∞ hay que acudir a la regla de l'Hospital: en la cual para una razón entre infinitos, obliga a buscar la relación de las derivadas para obtener el comportamiento al límite. Entonces para *x=0* el límite es fácil:

$$y=\frac{ax}{1+bx}=\frac{a0}{1+b0}=\frac{0}{1+0}=\frac{0}{1}=0$$

Para x=∞, y=∞/(1+ ∞)= ∞/∞. Para el numerador *ax*, su derivada es a, y para el denominador su derivada es b, por lo cual la razón de las derivadas es ***a/b***, valor asintótico de esta ecuación.

Estimación de parámetros en funciones asintóticas.

El modelo asintótico exponencial no es linealizable, así que hay que acudir a la regresión no lineal minimocuadrática. Una de las ventajas del modelo Michaelis–Menten es su fácil linealización por medio de la transformación reciproca:

$$\frac{1}{y}=\frac{1+bx}{ax}=\frac{1}{ax}+\frac{bx}{ax}=\frac{1}{ax}+\frac{b}{a}$$

Para la cual si hacemos $y'=\frac{1}{y}$ y $x'=\frac{1}{x}$, encontramos ul modelo lineal, fácil de estimarle sus parámetros: *y'=b/a+1/ax'*: intercepto (*b/a*), pendiente *1/a*. Basta entonces correr una regresión lineal con los recíprocos de las variables y detransformar al final. Supongamos que nuestro grafico pasa por los puntos ((0.2, 44.44) y (0.6, 70.59). ¿Cómo obtener los valores de los parámetros a y b?

Primero calculamos los 4 recíprocos. La pendiente de la función linealizada es el cambio en *1/y* dividido por el cambio en *1/x*:
(1/44.44 - 1/70.59)/(1/0.2 - 1/0.6)
[1] 0.002500781
entonces a =1/0.002500781 = 400

Ahora rearreglamos la ecuación y usamos uno de los puntos, por ejemplo: (x =0.2, y=44.44) para obtener el valor de *b*:

$$b = \frac{1}{x}\left(\frac{ax}{y} - 1\right) = \frac{1}{0.2}\left(\frac{400 \times 0.2}{44.44} - 1\right) = 4$$

Funciones sigmoidales (forma de S). La más simple de ellas es la logística biparamétrica en la cual para 0≤ y ≤1, la escribimos como:

$$y = \frac{e^{a+bx}}{1+e^{a+bx}}$$

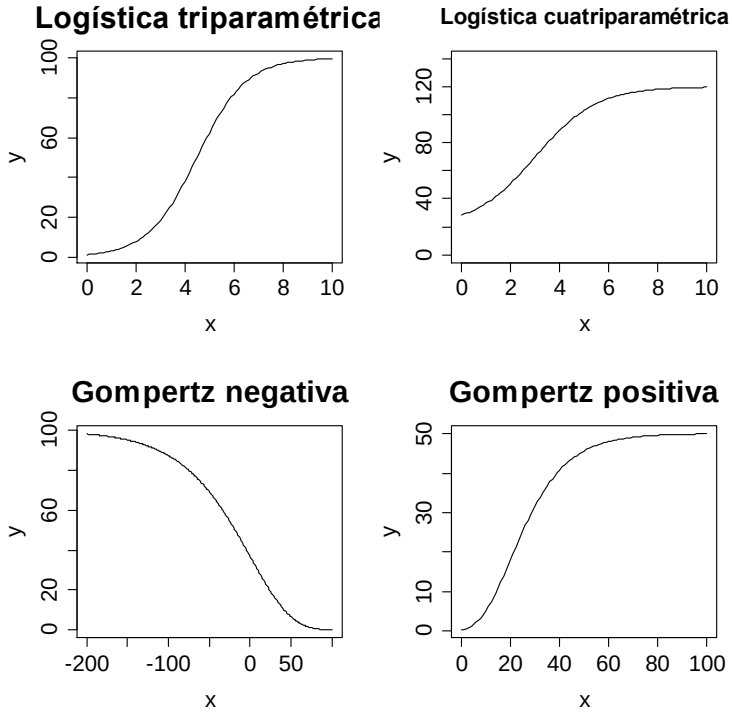
la cual es importante para el ajuste de modelos lineales generalizados para datos de proporciones, como se mostró ya, ampliamente.

La función triparamétrica logística permite que *y* varíe en alguna escala:

$$y = \frac{a}{1+be^{-cx}}$$

El ***intercepto*** es ***a/(1+b)***, el valor asintótico es ***a*** y la pendiente inicial es medida por ***c***. Por ejemplo, sea una curva con parámetros 100, 90 y 1.0:

```
par(mfrow=c(2,2))
x<-seq(0,10,0.1)
y<-100/(1+90*exp(-1*x))
plot(x,y,type="l",main="Logística triparamétrica", cex.axis=1.5, cex.lab=1.5, cex.main=2)
```



La función logística cuatriparamétrica tiene asíntotas a izquierda –(a) y a la derecha (b), como extremos del eje x y escala (c) y, la respuesta para x alrededor del medio punto (d) donde la curva tiene su inflexión:

$$y = a + \frac{b-a}{1+e^{c(d-x)}}$$

Hagamos a=20, b=120, c=0.8 and d=3, la función será:

```
y<-20+100/(1+exp(0.8*(3-x)))
plot(x,y,ylim=c(0,140),type="l",main="Logística  cuatriparamétrica",cex.axis=1.5,  cex.lab=
1.5,cex.main=1.5)
```

Curvas sigmoides negativas tienen el parámetro c<0 como en la función:

$$y = 20 + \frac{100}{1+e^{-0.8\times(3-x)}}$$

Función de Gompertz. Una curva asimétrica ***S*** muy usada en demografía, datos forestales y trabajos de seguros de vida es el **modelo de crecimiento de Gompertz**:

$$y = ae^{be^{cx}}$$

La forma de la función depende de los signos de los parámetros b y c. Para una **sigmoide negativa**, b es negativo (acá −1) y c es positivo (acá +0.02). Grafica anterior.

```
x<- -200:100
y<-100*exp(-exp(0.02*x))
plot(x,y,type="l",main="Gompertz negativa", cex.axis=1.5, cex.lab=1.5, cex.main=2)
```

para una **sigmoide positiva**, ambos parámetros son negativos:

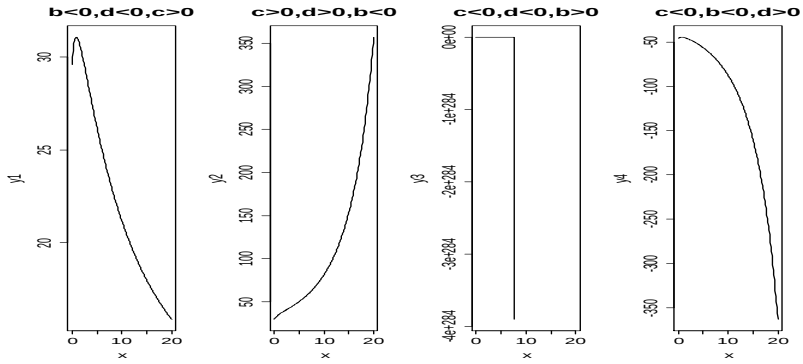
```
x<- 0:100
y<- 50*exp(-5*exp(-0.08*x))
plot(x,y,type="l",main="Gompertz positiva", cex.axis=1.5, cex.lab=1.5, cex.main=2)
```

Modelo Biexponencial. Es una útil función no lineal, tetraparamétrica, configurada por la suma de dos funciones exponenciales de x:

$$y = ae^{bx} + ce^{dx}$$

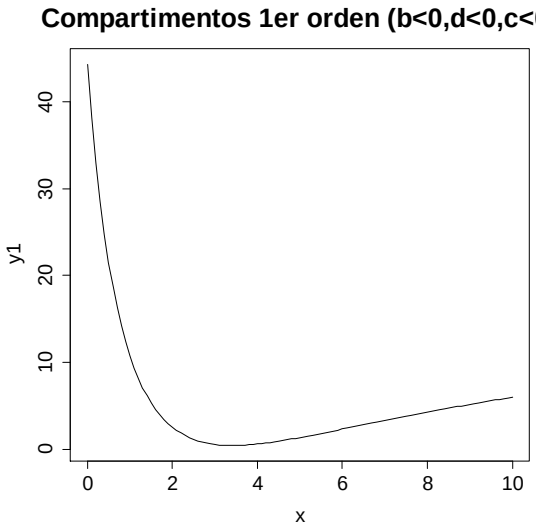
Varias formas dependen de los signos de los parámetros *b, c* y d:

```
par(mfrow=c(2,2))
x<-seq(0,20,0.1)
y1<- -3.0196*exp(exp(-0.842*x))+ 13.9*exp(exp(-0.059*x))
y2<- -3.0196*exp(exp(-0.842*x))+13.9*exp(exp(0.059*x))
y3<- -3.0196*exp(exp(0.842*x))-13.9*exp(exp(-0.059*x))
y4<- -3.0196*exp(exp(-0.842*x))-13.9*exp(exp(0.059*x))
plot(x,y1,type="l",ylab="y1",main="b<0,d<0,c>0", cex.axis=1.5, cex.lab=1.5, cex.main=2)
plot(x,y2,type="l",ylab="y2",main="c>0,d>0,b<0", cex.axis=1.5, cex.lab=1.5, cex.main=2)
plot(x,y3,type="l",ylab="y3",main="c<0,d<0,b>0", cex.axis=1.5, cex.lab=1.5, cex.main=2)
plot(x,y4,type="l",ylab="y4",main="c<0,b<0,d>0", cex.axis=1.5, cex.lab=1.5, cex.main=2)
```



cuando *b, c y* d son todos negativas esta función es conocida como el **modelo de compartimientos de primer orden** en el cual una droga administrada al tiempo 0 pasa a través del sistema con su dinámica afectada por eliminación, absorción y residual. Para verla sea,

```
par(mfrow=c(1,1))
x<-seq(0,10,0.1)
y1<- 30.196*exp(exp(-0.842*x))-13.9*exp(exp(-0.059*x))
plot(x,y1,type="l",ylab="y1",main="Comparrtimentos 1er orden (b<0,d<0,c<0)", cex.axis=1.5,
cex.lab=1.5, cex.main=2)
```



Transformaciones de las variables: respuesta y variables explicatorias. Las transformaciones más comunes para linealizar las relaciones entre *y* y *x* son:

- log(y) vs x para relaciones tipo exponencial;
- log(y) vs log(x) para funciones potenciales;
- exp(y) vs x para relaciones logarítmicas;
- 1/y vs 1/x para relaciones asintóticas;
- log(p/(1−p)) vs x para datos en proporciones.

Otras transformaciones para estabilizar varianzas son:

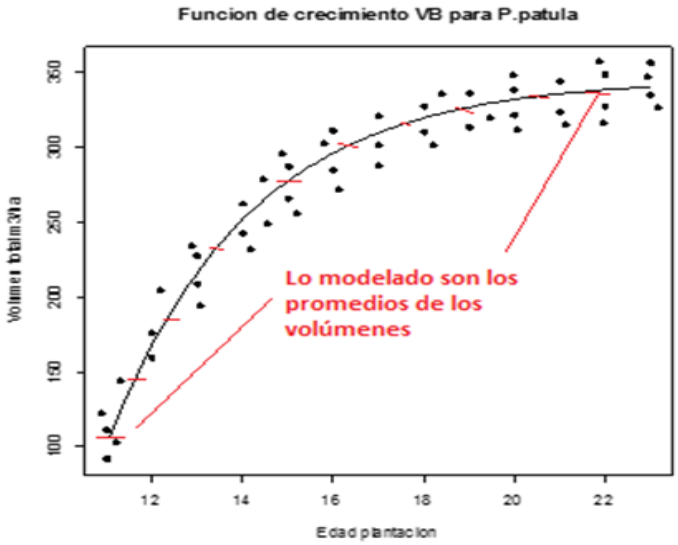
- $\sqrt{y}$ para estabilizar varianzas de datos de conteos
- *arcsin(y)* para estabilizar varianzas de datos en porcentajes.

Como caso muy iportante se hará énfasis en la siguiente función.

Función de crecimiento de Von Bertalanffy.

Es una de las populares para el estudio del crecimiento, de algunos organismos vivos dentro de una población, compuesto de las ganancias promedias de una dimensión menos las pérdidas o gastos para crecer, contra la edad o tiempo transcurrido.

El crecimiento de organismos individuales o en parcelas, como el caso que nos ocupará luego, (volumen promedio con corteza de madera de P. patula, en dos sitios de Piedras Blancas) se logra mediante una función, entre el crecimiento promedio del individuo (en este caso crecimiento promedio/ha) tratada, como un incremento de una longitud dada (vha) con el incremento en la edad (t), algo así:



Para ello una de las propuestas más famosas es la función de crecimiento de von Bertalanffy (VB), según (del Valle, 1986), el primero en formular una teoría sobre el crecimiento orgánico, basada en principios biológicos, desde 1938. Para este autor "el crecimiento se basa en la acción encontrada de procesos anabólicos y catabólicos, un organismo crece cuando la formación sobrepasa a la degradación, y se detiene cuando se equilibran ambos procesos". Para ello propuso la ecuación b mostrada más adelante $\frac{dW}{dt} = nW^m - rW$.

Modelo y Parámetros. Hay muchas versiones de la original de von Bertalanffy (1939), muy trabajadas en Colombia por (del Valle, 1986), como la a), conocida para el sector forestal o como la siguiente, b) muy usada en ecología de especies animales:

$$a)\ \frac{dL}{dt} = aL^b - cL;\ \text{sector forestal}$$

$$b)\ E[L\,|\,t]\ =\ L_\infty\left(1-e^{-k(t-t_0)}\right)\ esp.\ animales$$

En que:
E[L|t]: longitud promedia esperada en el tiempo t,
L_∞ : Longitud promedia asintótica,
k: coeficiente de la tasa de crecimiento (con unidades 1/t); y
t_0: artificio de modelación para representar el tiempo en que la longitud promedia es cero y que *a pesar de esta definición, no es un parámetro biológico, sino un valor para ajustar o corregir el modelo al tamaño inicial del individuo, pues muchos modelos no pasan por el origen.

Todos estos parámetros han sido interpretados de múltiples formas, pero se deja constancia de lo siguiente:

L_∞ : no es la máxima longitud sino una asíntota para la longitud promedia a un tiempo dado, porque es posible encontrar individuos mayores a ella, como sucede con muchos promedios.
k no es una tasa de crecimiento, como se vería al despejarla de la ecuación mostrada

$$k \;=\; log\frac{\left(\frac{L_\infty - E\left[L\,|\,t\right]}{L_\infty}\right)}{t - t_o}$$

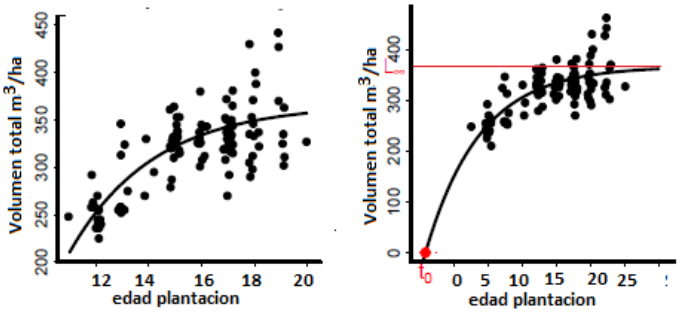
cuyas unidades son t^{-1}, diferente al cambio de L para una unidad del tiempo, como en la función clásica del crecimiento. Dos interpretaciones serían posibles para k:

a- Medida de la tasa exponencial de aproximación al tamaño asintótico.

b- Por medio del valor e^{-k}, la fracción fija por la cual debe multiplicarse el incremento anual cada año. Esta podría verse como:

$$E\left[L\,|\,(t\;=i+2)\right] - E\left[L\,|\,(t=i+1)\right] \;=\; e^{-k}\left(E\left[L\,|\,(t\;=i+1)\right] \;- E\left[L\,|\,(t\;=i)\right]\right)$$

En que, como k > 0, la cantidad e^{-k} es un decimal, por lo cual cada incremento posterior del crecimiento es una fracción constante del incremento corriente, lo que indica que el crecimiento se hace menor a medida que pasa el tiempo en los individuos más viejos. Y, como debe suponerse L_∞, k y t_0 son altamente relacionados como se vería en unos datos de crecimiento del volumen total $^{m^3}/_{ha}$ en una plantación.



Ajuste del modelo VB (MVB) en R. En R se requieren algunas funciones localizadas en los paquetes FSA, FSAdata, y nlstools, para lo cual se requieren abiertas las librerías

```
library(FSA)
library(FSAdata)
library(nlstools)
library(FSATeach)
```

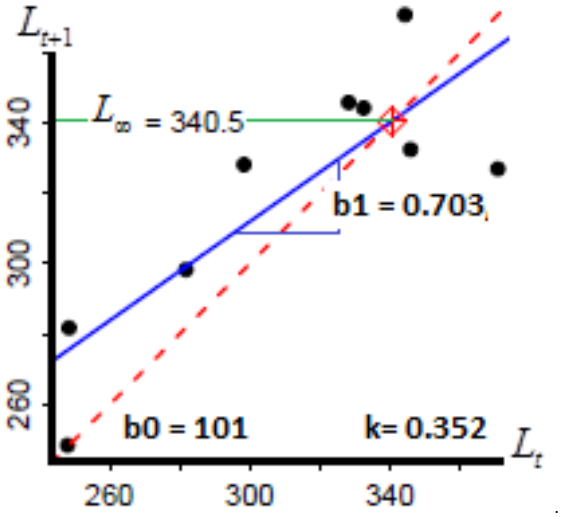
Como la mayoría de modelos de crecimiento requiere métodos no lineales para estimar sus parámetros fijando el modelo y sus valores de partida con la función nls().

Métodos gráficos para obtener valores de partida.

Se expondrán dos métodos para obtener los valores de partida llamados valores gráficos de partida de Ford (1933) y Walford (1946), para la estimación de L_∞ y *k* para las longitudes promedias a unas edades dadas de los individuos. F y W, sostienen que una gráfica de L_{t+1} versus L_t para la longitud escogida de los datos en un MVB se ajusta a una línea recta con pendiente igual a e^{-k} y, adicional a ello, que el punto en que la relación lineal entre L_{t+1} versus L_t intercepta la línea 1:1 (45°) da un buen estimado para L_∞ como se aprecia en la siguiente figura. La manipulación algebraica de la gráfica Ford-Walford, muestra las siguientes ecuaciones para lograr valores de partida aceptables

para $\tilde{L}_\infty$ y $\tilde{k}$ para la regresión no lineal:

$$\tilde{k} = -\ln(b_1) \text{ y } \tilde{L}_\infty = \frac{b_0}{1-b_1}$$



Gráfica de Ford_Waldford

Con la línea ajustada entre L_{t+1} versus L_t mostrada en la gráfica. Para L_∞ de partida y pares cartesianos edad y longitud promedia de la variable escogida $(\dot{t}, \bar{L}_{\dot{t}})$ el MVB típico puede ser rearreglado para resolver t_0 así:

$$\tilde{t}_0 = \dot{t} * LN\left(\frac{\tilde{L}_\infty - \bar{L}_{\dot{t}}}{\tilde{L}_\infty}\right)$$

Otra alternativa, para obtener $\tilde{t}_0$, es ajustar una polinomial de segundo grado entre las longitudes promedias de los datos y el tiempo y, la raíz que pase más cercana a cero dará un buen estimado.

Afortunadamente estos métodos se encuentran en R en el paquete *vbStarts()*, cuyo primer argumento es una fórmula de la forma variable~tiempo. El argumento *data=* debe darse. Por defecto se usa la regresión polinómica para el valor de partida para t_0 usando el argumento meth0="yngAge".

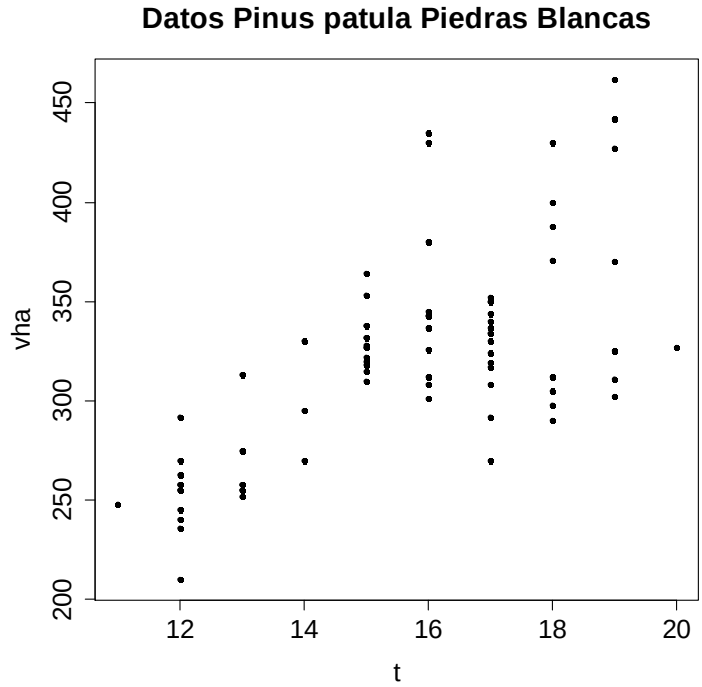
Finalmente existe la opción grafica para MVB utilizando los valores de partida impuestos a los datos observados por medio de plot=TRUE. Se usará una base de datos de Piedras Blancas con volúmenes promedios/ha en parcelas temporales:

	t	vha		t	vha		t	vha		t	vha
1	11	248	13	13	252	25	15	332	37	16	301
2	12	210	14	13	255	26	15	328	38	16	308
3	12	245	15	13	258	27	15	327	39	16	312
4	12	255	16	13	275	28	15	322	40	16	337
5	12	258	17	13	313	29	15	320	41	16	343
6	12	263	18	13	255	30	15	318	42	16	345
7	12	270	19	13	258	31	15	315	43	16	380
8	12	292	20	14	330	32	15	310	44	16	430

9	12	236	21	14	295	33	15	327	45	16	435
10	12	258	22	14	270	34	15	338	46	16	343
11	12	255	23	15	364	35	15	353	47	16	326
12	12	240	24	15	338	36	15	332	48	17	270
	t	vha		t	vha		t	vha			
49	17	292	61	17	334	73	19	427			
50	17	308	62	17	350	74	19	370			
51	17	334	63	18	430	75	19	325			
52	17	337	64	18	400	76	19	311			
53	17	340	65	18	388	77	19	302			
54	17	344	66	18	371	78	20	327			
55	17	350	67	18	312						
56	17	352	68	18	305						
57	17	317	69	18	298						
58	17	319	70	18	290						
59	17	324	71	19	462						
60	17	330	72	19	442						

#Ejemplo 1 von Bertalanffy, solo para conocer ciertas órdenes en R y manejo de los datos
pppb<-read.table("clipboard")
attach(pppb)
names(pppb)
[1] "t" "vha"
Lo primero que hacemos es graficar los datos crudos de promedios para ver su comportamiento:

plot(t,vha,main="Base de datos P.patula Piedras Blancas", pch=20, cex.lab=1.5, cex.axis=1.5, cex.main=1.7)



A pesar que se ven grandes distorsiones, lo cual se debe a la falta de más información que pudiera estratificar estos datos, se inicia el proceso de modelación, para lo cual se requiere de la:
library(FSA)

FSA v0.8.13. See citation('FSA') if used in publication.
Run fishR() for related website and fishR('IFAR') for related book.

```
valpart <- vbStarts(vha~t,data=pppb)#Valores de partida iniciales
unlist(valpart)#opcion sintetica para ver valpart
       Linf           K          t0
339.6830971   0.3670235   7.4316319
```

Luego creamos un vector (objeto R) con los valores de partida obtenidos para el modelo de VB:

```
vpartVB <- list(Linf=339, K=0.37, t0=7.4)#Valores de partida para modelo vB
vpartVB
$Linf
[1] 339
$K
[1] 0.37
$t0
[1] 7.4
```

Con los anteriores creamos el objeto R con el modelo volumétrico por hectárea VB:

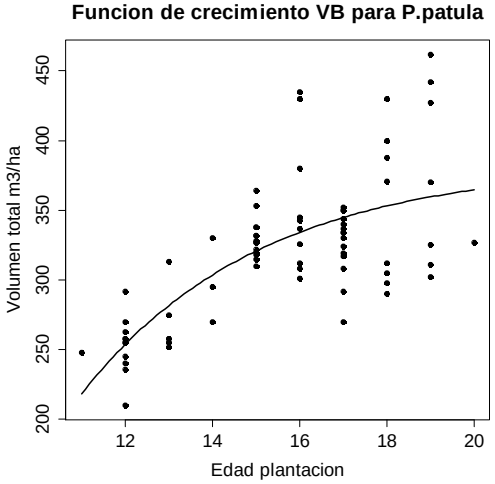
```
vbpppb<- vha~Linf*(1-exp(-K*(t-t0)))#modelo VB para P. patula Piedras Blancas
vbpppb
vha ~ Linf * (1 - exp(-K * (t - t0)))
```

Y, con este, creamos el objeto con el modelo ajustado, con base en los valores de partida:

```
ajvbpppb<- nls(vbpppb,data=pppb,start=vpartVB)
ajvbpppb
Nonlinear regression model
  model: vha ~ Linf * (1 - exp(-K * (t - t0)))
   data: pppb
    Linf        K       t0
383.3017   0.2426   7.5287
 residual sum-of-squares: 102835

Number of iterations to convergence: 6
Achieved convergence tolerance: 5.156e-06
```

Graficamos el modelo ajustado para mirar su comportamiento frente a los datos.

```
fitPlot(ajvbpppb,xlab="Edad plantacion",ylab="Volumen total m3/ha",main="Funcion de
crecimiento VB para P.patula",,cex.lab=1.5,cex.axis=1.5,cex.main=1.7)
```

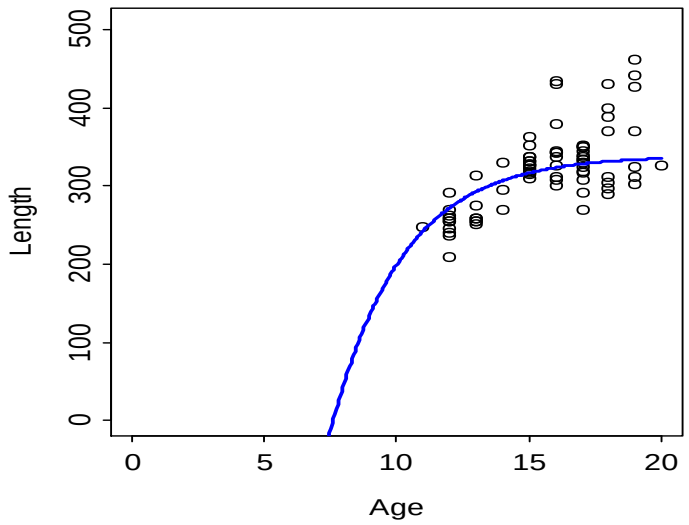


Podemos ver gran cantidad de información del modelo con el paquete nls()y la siguiente función

```
overview(ajvbpppb)
```

```
------
Formula: vha ~ Linf * (1 - exp(-K * (t - t0)))

Parameters:
     Estimate Std. Error t value Pr(>|t|)
Linf 383.3017    33.2881  11.515  < 2e-16 ***
K      0.2426     0.1218   1.992     0.05 *
t0     7.5287     1.7113   4.399 3.54e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 37.03 on 75 degrees of freedom

Number of iterations to convergence: 6
Achieved convergence tolerance: 5.156e-06

------
Residual sum of squares: 103000

------
t-based confidence interval:
             2.5%       97.5%
Linf 3.169884e+02 449.6149261
K    6.607935e-06   0.4851637
t0   4.119620e+00  10.9378034

------
Correlation matrix:
          Linf          K         t0
Linf  1.0000000 -0.9672082 -0.8926723
K    -0.9672082  1.0000000  0.9723847
t0   -0.8926723  0.9723847  1.0000000
```

growthModelSim("vbTypical",vha~t,data=ppat)#permite ajustar los parametros.
Loading required package: tkrplot



A parte de los parámetros, se ve la alta correlación entre ellos. La función overview() también entrega los límites de confianza, pero basados en la distribución normal, lo cual puede no ser lo más correcto, pero vale como aproximación.

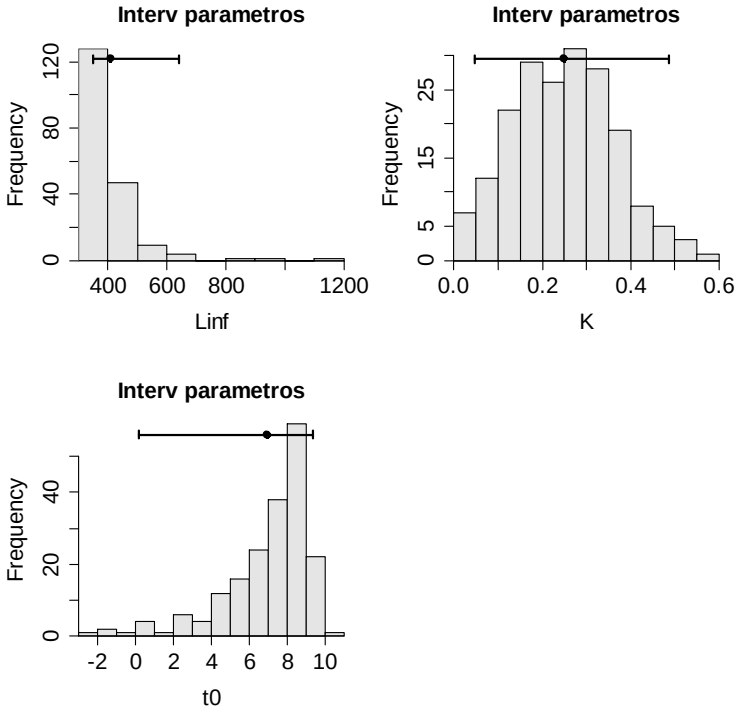
```
svTypical <- list(Linf=383,K=0.24,t0=8)
> svTypical
$Linf
[1] 383
$K
```

```
[1] 0.24
$t0
[1] 8
```

Existen métodos más sofisticados para ello como los bootstrap., cuya teoría escapa en este momento, pero se muestra para ver sus resultados

```
bootpppb<- nlsBoot(ajvbpppb,niter=200) # estimado bootstrap para los Limites de los parámetros
del modelo
confint(bootpppb,plot=TRUE)

bootpppb<- nlsBoot(ajvbpppb,niter=200) # estimado bootstrap para los Limites de los parámetros
del modelo
Warning message:
In nlsBoot(ajvbpppb, niter = 200) :
  The fit did not converge 9 times during bootstrapping

confint(bootpppb,plot=TRUE,cex.lab=1.5,cex.axis=1.5,cex.main=1.5,main="Interv parametros")
          95% LCI     95% UCI
Linf 349.64279984 641.9914633
K      0.04715112   0.4864491
t0     0.17252908   9.3973762
```

Como se ve de los tres parámetros, de pronto k tiende a una normal.



Los p-values para verificar la hipotesis: H0: K = 0.24 contra HA : K < 0.5 se obtiene mediante: >

```
htest(bootpppb,"K",0.24,"less")
  Ho Value  p value
K     0.24 0.513089
```

Por lo tanto, k es una buena estimación como se dio por el modelo.

Vistos los datos, se ve necesario se necesita una variable estratificadora para una base de datos similar la anterior.

Se propondrá un nuevo ejemplo con la variable índice de sitio como elemento diferenciador, para un mejor ajuste de los datos.

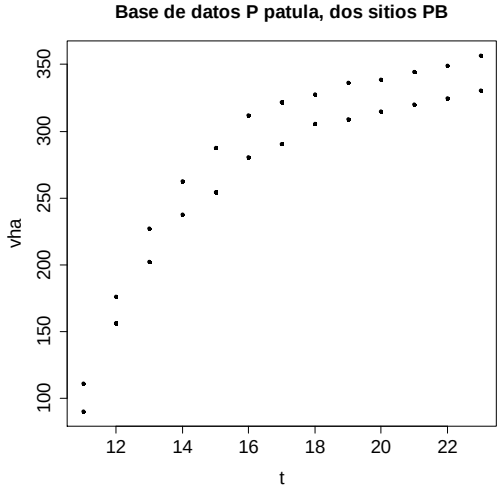
	t	vha	sit
1	11	90.3	2
2	12	156.4	2
3	13	202.5	2
4	14	237.6	2
5	15	254.8	2
6	16	280.6	2
7	17	290.7	2
8	18	305.4	2
9	19	309.1	2
10	20	314.6	2
11	21	319.8	2
12	22	324.4	2
13	23	330.3	2
14	11	111.4	1
15	12	176.1	1
16	13	227.4	1
17	14	262.9	1
18	15	287.3	1
19	16	311.8	1
20	17	321.6	1
21	18	327.6	1
22	19	336.4	1
23	20	338.7	1
24	21	344.3	1
25	22	348.9	1
26	23	356.8	1

library(FSA), library(FSAdata), library(nlstools), library(FSATeach)

```
pppb<-read.table("clipboard")
attach(pppb)
names(pppb)
[1] "t"   "vha" "sit"
```

Insistimos en lo primero que debemos hacer, graficar los datos crudos de promedios para ver su comportamiento.

```
plot(t,vha,main="Base de datos P patula, dos sitios PB",pch=20, cex.lab=1.5, cex.axis=1.5, cex.main=1.5)
```



Y, entonces a nivel intuitivo, deberemos hacer dos modelos de crecimiento, uno para cada sitio. Iniciamos nuevamente buscando los valores de partida. Pero iniciaremos con todos los datos.

```
valpart <- vbStarts(vha~t,data=pppb)#
```
muestra los valores calculados de partida por el método ya mencionado, pero para economizar espacio también con
```
unlist(valpart)
```

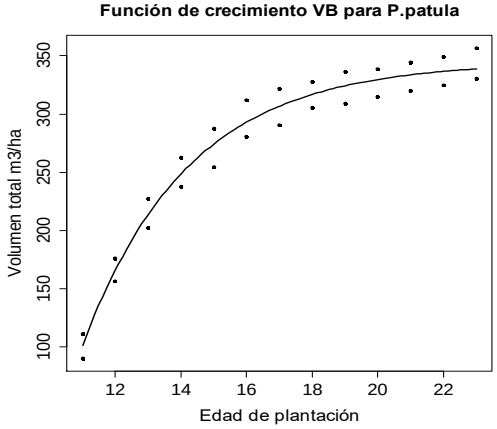
Creamos el vector (objeto R) con los valores de partida obtenidos para el modelo de VB:
```
vpartVB <- list(Linf=346, K=0.3, t0=9.9)#Para ser usados posteriormente.
```

Creamos el objeto (volumétrico) con el modelo VB:
```
vbpppb<- vha~Linf*(1-exp(-K*(t-t0)))#
```
modelo VB para P. patula Piedras Blancas

Creamos el objeto con el modelo ajustado, con base en los valores de partida:
```
ajvbpppb<- nls(vbpppb,data=pppb,start=vpartVB)

ajvbpppb
Nonlinear regression model
  model: vha ~ Linf * (1 - exp(-K * (t - t0)))
   data: pppb
    Linf        K       t0
344.5947   0.3101   9.8755
 residual sum-of-squares: 4525

Number of iterations to convergence: 3
Achieved convergence tolerance: 1.317e-06

summary(ajvbpppb)

Formula: vha ~ Linf * (1 - exp(-K * (t - t0)))

Parameters:
      Estimate Std. Error t value Pr(>|t|)
Linf 344.59471    6.93657  49.678  < 2e-16 ***
K      0.31006    0.03203   9.679 1.41e-09 ***
t0     9.87549    0.18969  52.062  < 2e-16 ***

Residual standard error: 14.03 on 23 degrees of freedom

Number of iterations to convergence: 3, Achieved convergence tolerance: 1.317e-06
```

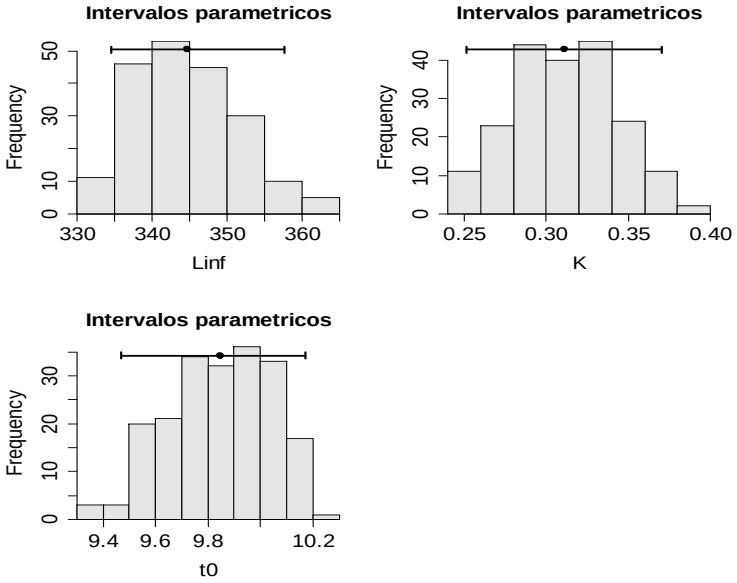
Graficamos el modelo ajustado para mirar su comportamiento frente a los datos.

```
fitPlot(ajvbpppb,xlab="Edad de plantación",ylab="Volumen total m3/ha",main="Función de crecimiento VB para P.patula", pch=20, cex.lab=1.5, cex.axis=1.5, cex.main=1.5)
```




```
overview(ajvbpppb)#para obtener mucha información adicional del modelo
```

```
overview(ajvbpppb)

------
Formula: vha ~ Linf * (1 - exp(-K * (t - t0)))

Parameters:
      Estimate Std. Error t value Pr(>|t|)
Linf 344.59471    6.93657  49.678  < 2e-16 ***
K      0.31006    0.03203   9.679 1.41e-09 ***
t0     9.87549    0.18969  52.062  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.03 on 23 degrees of freedom

Number of iterations to convergence: 3
Achieved convergence tolerance: 1.317e-06

------
Residual sum of squares: 4520

------
t-based confidence interval:
           2.5%       97.5%
Linf 330.2453196 358.9441040
K      0.2437915   0.3763245
t0     9.4830955  10.2678872

------
Correlation matrix:
          Linf          K         t0
Linf  1.0000000 -0.8342764 -0.5541656
K    -0.8342764  1.0000000  0.8405430
t0   -0.5541656  0.8405430  1.0000000

confint(bootpppb,plot=TRUE,main="Intervalos   parametricos",   cex.lab=1.5,   cex.axis=1.5,
cex.main=1.5)
         95% LCI     95% UCI
Linf 334.5875642 357.7236346
K      0.2514334   0.3706597
t0     9.4709314  10.1725063
```



Como se ve los tres parámetros tienden a una normal.

Los p-values para verificar la hipotesis: H0: K = 0.31 contra HA : K < 0.5, etc. se obtiene mediante:

```
htest(bootpppb,"K",0.31,"less")
  Ho Value p value
K     0.31   0.525
htest(bootpppb,"Linf",344.6,"less")
     Ho Value p value
Linf    344.6   0.465
htest(bootpppb,"t0",9.9,"less")
   Ho Value p value
t0      9.9   0.435
```

Por lo tanto, los 3 parámetros son buenos estimadores para el modelo.

Valores predichos. Los volúmenes predichos para otras edades se obtienen como ya se sabe con la funcion predict().

Ejemplo: para calcular los volúmenes predichos para una edad de 15 años con sus correspondientes intervalos sería así:

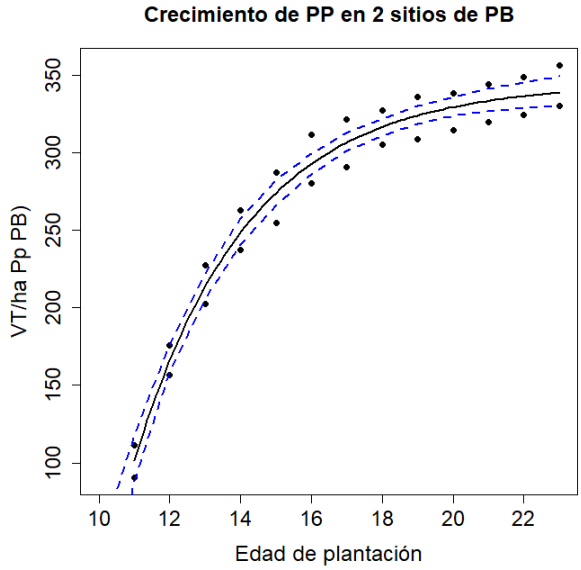
```
pred15 <- data.frame(t=15)
predict(ajvbpppb, pred15)
[1] 274.2453

pred1varios <- data.frame(t=c(10,15,20) )#Volumenes a los 10,15,20 años

predict(ajvbpppb, pred1varios)
[1]  13.04954 274.24533 329.66753

parboot<- bootpppb$coefboot # Solo se muestran los primeros pues contiene todos los datos de
la muestra bootstrp con sus limites
head(parboot)
         Linf         K        t0
[1,] 343.9411 0.3051373  9.718940
[2,] 337.8891 0.3598323 10.110726
[3,] 353.8371 0.2538830  9.432538
[4,] 345.4456 0.3325143 10.048812
[5,] 344.9828 0.3101872  9.792108
[6,] 350.6530 0.2991578  9.966406

valpb <- parboot[,"Linf"]*(1-exp(-parboot[,"K"]*(15-parboot[,"t0"]))) #Valores predichos boot
15

valpb
  [1]  275.2910  279.7174  267.7506  278.8596  276.3978  272.8673  272.1068  266.6535  277.5407
269.6978 272.1293 271.9169 273.1465 274.1644 279.8626,  268.0537 270.8077, etc.

quantile(valpb,c(0.025,0.975))#entre que valores estará el volumen a los 15 años
    2.5%    97.5%
266.6399 282.5840
```

Grafica de intervalos de confianza. Se puede crear una línea ajustada con los intervalos de confianza para los volúmenes promedios en un VB, primero haciendo los predichos para cada t en un rango de edades y, usar los parámetros para cada muestra *bootstrap*. Es posible, entonces, graficar los cuantiles entre 2.5% y 97.5%, como
sigue:

```
edades <- 10:23# edades para graficar
fitPlot(ajvbpppb,xlab="Edad        de        plantación",ylab="VT/ha        Pp        PB)",
xlim=range(edades),main="Crecimiento de PP en 2 sitios de PB", cex.lab=1.5, cex.axis=1.5,
cex.main=1.5)
```

Intervalos de confianza. Cálculos para obtener los valores más probables para los intervalos de confianza. Se empieza con unos vectores vacíos de longitud conocida (*numeric*)

```
LCI <- UCI <- numeric(length(edades))
for (i in 1:length(edades)) {
pv <- parboot [,"Linf"]*(1-exp(-parboot [,"K"]*(edades[i]- parboot [,"t0"])))
LCI[i] <- quantile(pv,0.025)
UCI[i] <- quantile(pv,0.975)
}
lines(UCI~edades,type="l",col="blue",lwd=2,lty=2)
lines(LCI~edades,type="l",col="blue",lwd=2,lty=2)
```



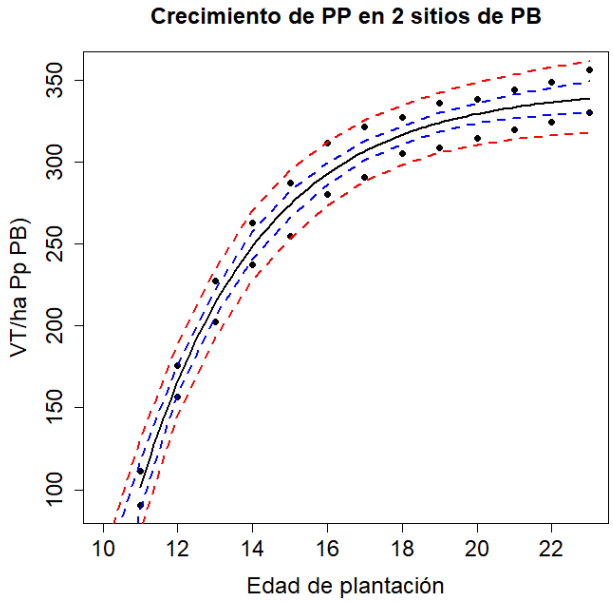
Límites de predicción. Se puede, incluso, adicionar unos límites de predicción aproximados para la gráfica de la línea ajustada, sumando y restando los errores estándar residuales de los datos en cada modelo *bootstrap* (los valores encontrados en la porción rse de los resultados del muestreo bootstrap, o sea en *bootpppb$rse*, del ejemplo usado), para las predicciones construidas con cada modelo, como se describió para los intervalos de confianza anteriores. Se Muestran

```
head(valpb)#Valores estimados muetreo bootstrap para t=15
[1] 275.2910 279.7174 267.7506 278.8596 276.3978 272.8673
str(valpb)
num [1:200] 278 277 272 276 279 ...#Como se ve es bastante largo

head(bootpppb$rse)
[1] 12.96768 13.31148 12.32361 13.29101 14.67155 12.57762
```

El cuantil 2.5% cuando se resta el error estándar residual se aproxima al límite al 95% inferior de predicción y lo mismo el cuantil 97.5% cuando se suma el error estándar residual(límite superior). El código para la gráfica con los intervalos de confianza para ambos casos, seria:

```
edades <- 10:23# ya creado
fitPlot(ajvbpppb,xlab="Edad de plantacción",ylab="VT/ha Pp PB)", xlim=range(edades),
main="Crecimiento de PP en 2 sitios de PB")
LCI <- UCI <- LPI <- UPI <- numeric(length(edades))
for (i in 1:length(edades)) {
pv <- parboot [,"Linf"]*(1-exp(-parboot [,"K"]*(edades[i]- parboot [,"t0"])))
LCI[i]<- quantile(pv,0.025)
UCI[i] <- quantile(pv,0.975)
LPI[i] <- quantile(pv- bootpppb$rse,0.025)
UPI[i] <- quantile(pv+ bootpppb$rse ,0.975)
}
lines(LPI~edades,type="l",col="red",lwd=2,lty=2)
```

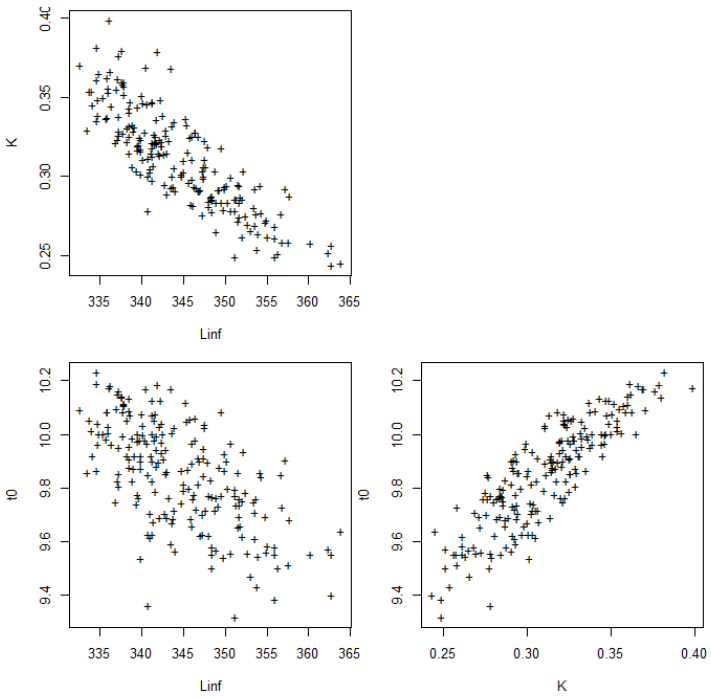
```
lines(UCI~edades,type="l",col="blue",lwd=2,lty=2)
lines(LCI~edades,type="l",col="blue",lwd=2,lty=2)
lines(UPI~edades,type="l",col="red",lwd=2,lty=2)
lines(LPI~edades,type="l",col="red",lwd=2,lty=2)
```



Gráfica de la línea ajustada para el modelo VB con intervalos *bootstrap* al 95% de confianza (líneas azules punteadas) y límites de confianza para las predicciones *bootstrap* al 95% (líneas rojas punteadas)

Por ultimo sometiendo el objeto nlsBoot() a la graficación se producen los gráficos de dispersión para los valores bootstrapp para cada par de parámetros:

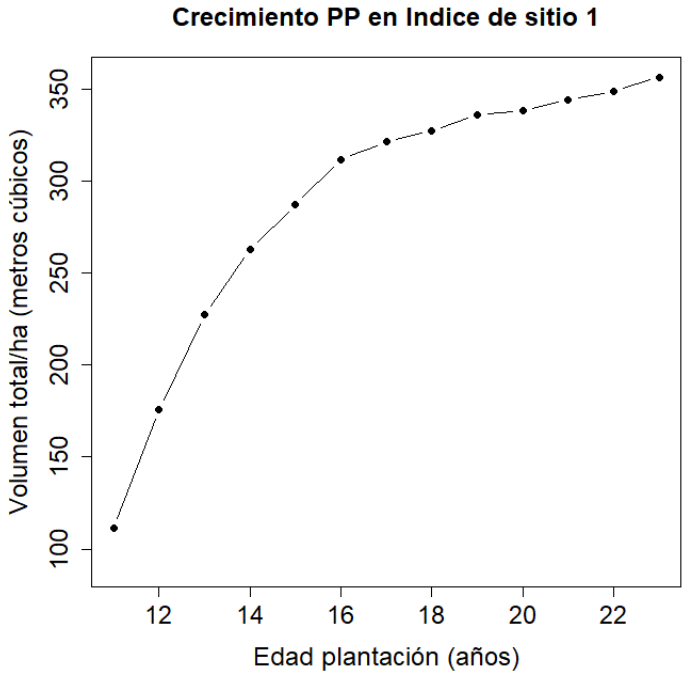
plot(bootpppb)



que ilustran la fuerte correlación entre ellos.

Modelación por sitios. Se mirará lo que pasa por cada sitio, individualmente iniciando con sitio1.

```
pppb1<- subset(pppb,sit==1)
plot(pppb1$t, pppb1$vha, pch = 16, type="b", xlab ="Edad plantación (años)", ylab ="Volumen total/ha (metros cúbicos)", xlim=range(t), ylim=range(vha), main="Crecimiento PP en Indice de sitio 1", cex.lab=1.5, cex.axis=1.5, cex.main=1.5)
```



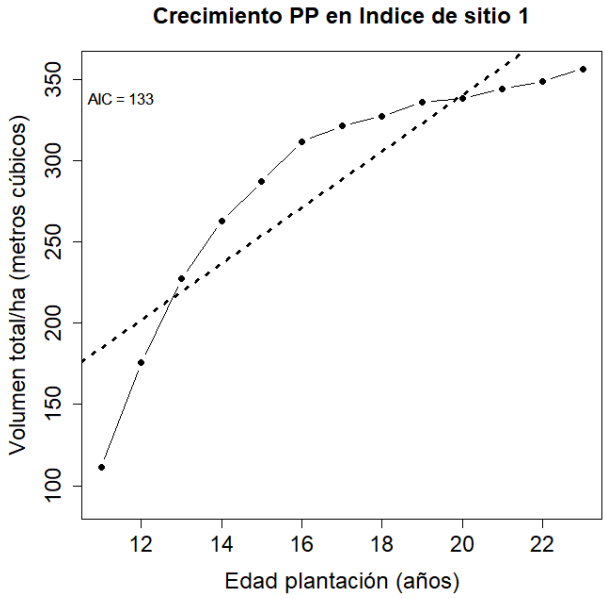
Al mirar la gráfica, sabemos que el modelo no debe ser lineal, pero sin embargo se corre como referencia.

```
ajlin1<-lm(pppb1$vha~pppb1$t)#como referencia
summary(ajlin1)
Call:
lm(formula = pppb1$vha ~ pppb1$t)

Residuals:
    Min      1Q  Median      3Q     Max
-73.459 -26.042   7.976  26.193  40.529

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   -5.247     45.095  -0.116    0.909
pppb1$t       17.282      2.591   6.671 3.51e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 34.95 on 11 degrees of freedom
Multiple R-squared:  0.8018,    Adjusted R-squared:  0.7838
F-statistic:  44.5 on 1 and 11 DF,  p-value: 3.51e-05
```

Se le calcula el AIC

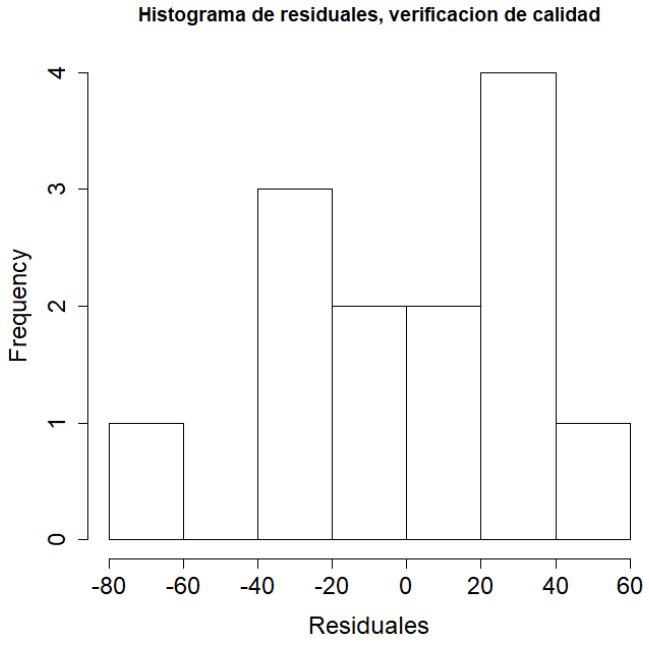
```
AICl1<-signif(AIC(ajlin1),digits=3)#se calcula para posteriores evaluaciones
> AICl1
[1] 133
```

```
plot(pppb1$t, pppb1$vha, pch = 16, type="b", xlab ="Edad plantación (años)", ylab ="Volumen total/ha (metros cúbicos)", xlim=range(t), ylim=range(vha), main="Crecimiento PP en Indice de sitio 1", cex.lab=1.5, cex.axis=1.5, cex.main=1.5)

abline(ajlin1, lwd=3, lty=3); legend (10, 350, paste("AIC =", AICl1), bty = 'n')
```



Suficiente para ver que presentaría desajustes en los residuales. Veamos el histograma de los residuales

```
hist(residuals(ajlin1), xlab='Residuales', main='Histograma de residuales, verificacion de calidad',cex.lab=1.5,cex.axis=1.5,cex.main=1.2)
```



A pesar de la aparente correlación lineal entre t y vha (r2 = 0.80), el ajuste predicho por el

modelo lineal (línea gruesa punteada, de forma Y=a+bX), severamente, sobre o subestima el vha en la mayoría de las clases. El histograma no muestra normalidad, por lo que el modelo simple lineal no es apropiado para esta base de datos y, se acudirá a modelar con la función de crecimiento no lineal de von Bertalanffy (VBGM) y, nuevamente usaremos la más común (ya usada): $E\left[L \mid t\right]$=$\mathrm{L}_t = L_\infty \left(1-e^{-k(t-t_0)}\right)$

en la cual
L_t es la longitud considerada en un tiempo t;
L_∞ es el promedio de la dimensión (v/ha) a su máximo tiempo,
k es el coeficiente de Brody (t−1−1); y
t_o el tiempo (edad) cuando el tamaño promedio es cero o su equivalente.

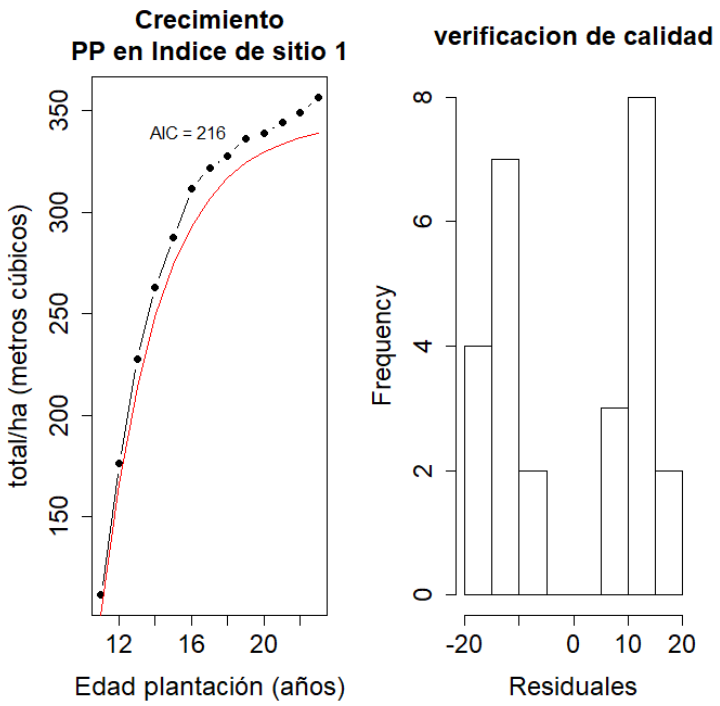
Para ajustar el VBGM en R debemos como ya se sabe especificar el modelo y asignar unos valores de partida para los parámetros como parte de todo modelo no lineal “nls”.
Los nombres asignados para los parámetros serán: Li = el máximo tamaño observado en los datos, k = un pequeño valor por ejemplo 0.2, y to, = 10 en este caso:

```
VB <- vha ~ Li*(1-exp(-k*(t-t0)))#modelo de VB
VBGM1<-nls(VB, start = list(Li = max(pppb1$vha), k=0.2, t0=10))
> VBGM1
Nonlinear regression model
  model: vha ~ Li * (1 - exp(-k * (t - t0)))
   data: parent.frame()
      Li         k        t0
344.5946   0.3101   9.8755
 residual sum-of-squares: 4525

Number of iterations to convergence: 4
Achieved convergence tolerance: 2.492e-06
> summary(VBGM1)

Formula: vha ~ Li * (1 - exp(-k * (t - t0)))

Parameters:
    Estimate Std. Error t value Pr(>|t|)
Li 344.59460    6.93656  49.678  < 2e-16 ***
k    0.31006    0.03203   9.679 1.41e-09 ***
t0   9.87549    0.18969  52.062  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.03 on 23 degrees of freedom

Number of iterations to convergence: 4
Achieved convergence tolerance: 2.492e-06
```

Nuevamente se ve que los errores estándar (SE) de L_∞ y k son relativamente pequeños con base en los estimados. La tabla muestra esto con la t de student (P < 0. 001) para ambos parámetros. Y, dasométricamente L_∞ y k significativamente >0.

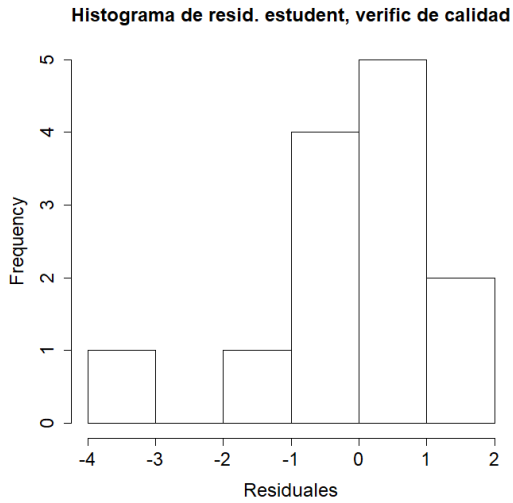
```
plot(pppb1$t, pppb1$vha, pch = 16, type="b", xlab ="Edad plantación (años)", ylab ="Volumen
total/ha (metros cúbicos)", xlim=range(pppb1$t), ylim=range(pppb1$vha), main="Crecimiento
PP en Indice de sitio 1", cex.lab=1.5,cex.axis=1.5,cex.main=1.5)
hist(residuals(VBGM1), xlab='Residuales', main='verificacion de calidad', cex.lab=1.5,
cex.axis=1.5,cex.main=1.5)
```

El modelo no lineal parece más exitoso para la moderación del crecimiento en el sitio1, aunque los residuales extremos son numerosos.



Veamos el comportamiento de los residuales estudentizados

```
library(MASS)
par(mfrow=c(1,1))
hist(studres(ajlin1), xlab='Residuales', main='Histograma de resid. estudent, verific de
calidad', cex.lab=1.5, cex.axis=1.5,cex.main=1.5)
```

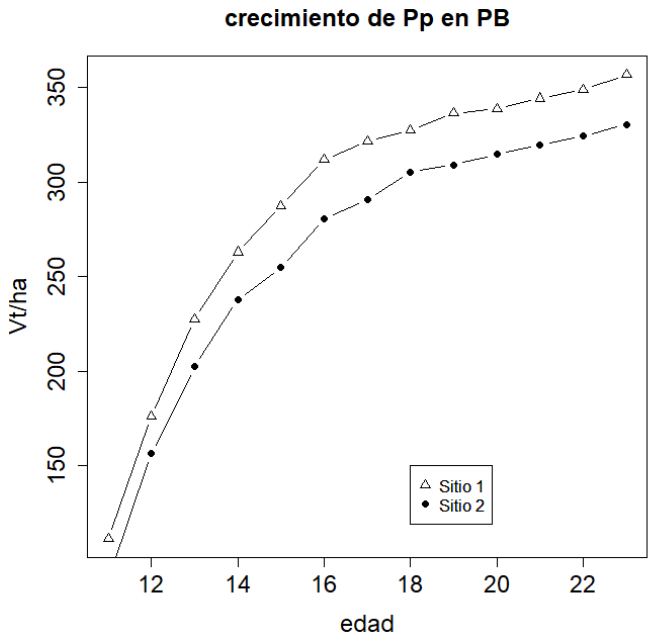


El histograma estudentizado parece mostrar algunas observaciones remotas, pero se puede concluir que el modelo no lineal de von Bertalanffy (VBGM), ajusta mejor, a pesar que el AIC para el VBGM resulto mayor.

Modelo de Crecimiento para más de un grupo.

No es muy usual más de un grupo en datos aparejados por variable y edad (t). entonces vamos a ver qué sucede entre dos índices de sitio. Como siempre graficaremos los datos para ver los resultados.

```
pppb1<- subset(pppb,sit==1)
pppb1<- as.data.frame(pppb1)
pppb2<- subset(pppb,sit==2)
pppb2<- as.data.frame(pppb2)

plot(pppb1$t, pppb1$vha, pch = 2, type="b",main="crecimiento de Pp en PB", xlab = 'edad',
ylab="Vt/ha", xlim=range(pppb1$t), ylim=range(pppb1$vha), cex.lab=1.5, cex.axis=1.5,
cex.main= 1.5)
lines(pppb2$t,pppb2$vha,pch=16,type="b",xlab="edad(taños)",ylab="V/ha", xlim=range(pppb2$t),
ylim=range(pppb2$vha))
legend (18, 150, c("Sitio 1", "Sitio 2"), pch = c(2, 16))
```



En general son mayores los volúmenes para el sitio 1. La pregunta es si es apropiado mezclar los datos en un solo modelo, o son mejores los modelos por cada sitio. Intentar el modelo VBGM y considerar parámetros específicos por sitio es un poco tedioso sin un paquete en R pero resultaría muy instructivo, de tal manera que trabajaremos combinando los datos y adicionando variables dummy por cada sitio.

```
PP12<-data.frame(rbind(cbind(pppb1, sit1=1, sit2=0), cbind(pppb2, sit1=0, sit2=1)))
str(PP12)
'data.frame':   26 obs. of  5 variables:
 $ t   : int  11 12 13 14 15 16 17 18 19 20 ...
 $ vha : num  111 176 227 263 287 ...
 $ sit : int  1 1 1 1 1 1 1 1 1 1 ...
 $ sit1: num  1 1 1 1 1 1 1 1 1 1 ...
 $ sit2: num  0 0 0 0 0 0 0 0 0 0 ...

head(PP12)
    t   vha sit sit1 sit2
14 11 111.4   1    1    0
15 12 176.1   1    1    0
16 13 227.4   1    1    0
17 14 262.9   1    1    0
18 15 287.3   1    1    0
19 16 311.8   1    1    0

PPconj<-(nls(PP12$vha~PP12$sit1*Linf1*(1-exp(-K1*(PP12$t-t01)))+sit2*Linf2*(1-exp(-
K2*(PP12$t-t02))), start = list (Linf1=a, K1=b, t01=c, Linf2=a, K2=b, t02=c)))
summary(PPconj)
```

```
Formula: PP12$vha ~ PP12$sit1 * Linf1 * (1 - exp(-K1 * (PP12$t - t01))) +
    sit2 * Linf2 * (1 - exp(-K2 * (PP12$t - t02)))

Parameters:
       Estimate Std. Error t value Pr(>|t|)
Linf1 3.564e+02  1.994e+00  178.76   <2e-16 ***
K1    3.208e-01  9.656e-03   33.22   <2e-16 ***
t01   9.841e+00  5.579e-02  176.41   <2e-16 ***
Linf2 3.330e+02  2.173e+00  153.20   <2e-16 ***
K2    2.991e-01  9.527e-03   31.39   <2e-16 ***
t02   9.915e+00  5.781e-02  171.51   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.97 on 20 degrees of freedom

Number of iterations to convergence: 4
Achieved convergence tolerance: 7.996e-06s
```

Este modelo es el más general porque permite a cada uno de sus 3 parámetros variar por sitio. La estructura jerárquica completa del VBGM incluye 7 especificaciones: 3 con un parámetro en común, 3 más con dos parámetros en común, y 1 con los 3 parámetros en común. Acá vamos a considerar 4 de estos 7 modelos para ilustrar el proceso:

#El modelo siguiente tiene 3 parámetros comunes y se dio cuando usamos VBGM

```
Cppg<-(nls(pppb$vha ~ sit1*Linf*(1-exp(-K*(pppb$t-t0))) + sit2*Linf*(1-exp(-K*(pppb$t-t0))),
start = list(Linf=a, K=b, t0=c)))

Cppg
Nonlinear regression model
  model: pppb$vha ~ sit1 * Linf * (1 - exp(-K * (pppb$t - t0))) + sit2 *     Linf * (1 -
exp(-K * (pppb$t - t0)))
   data: parent.frame()
    Linf        K       t0
344.5946   0.3101   9.8755
 residual sum-of-squares: 4525

Number of iterations to convergence: 4
Achieved convergence tolerance: 2.883e-06

summary(Cppg)

Formula: pppb$vha ~ sit1 * Linf * (1 - exp(-K * (pppb$t - t0))) + sit2 *
    Linf * (1 - exp(-K * (pppb$t - t0)))

Parameters:
      Estimate Std. Error t value Pr(>|t|)
Linf 344.59459    6.93655  49.678  < 2e-16 ***
K      0.31006    0.03203   9.679 1.41e-09 ***
t0     9.87549    0.18969  52.062  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
#El modelo siguiente tiene 3 parámetros comunes y se dio cuando usamos VBGM
Cppg<-(nls(pppb$vha ~ sit1*Linf*(1-exp(-K*(pppb$t-t0))) + sit2*Linf*(1-exp(-K*(pppb$t-t0))),
start = list(Linf=a, K=b, t0=c)))
```

#Estos otros modelos tienen un parámetro comun: Linf, K, or t0

```
Cpplinf<-(nls(PP12$vha ~ sit1*Linf*(1-exp(-K1*(PP12$t-t01))) + sit2*Linf*(1-exp(-K2*(PP12$t-
t02))), start = list(Linf=a, K1=b, t01=c, K2=b, t02=c)))#Linf comun
summary(Cpplinf)

Formula: PP12$vha ~ sit1 * Linf * (1 - exp(-K1 * (PP12$t - t01))) + sit2 *
    Linf * (1 - exp(-K2 * (PP12$t - t02)))
```

```
Parameters:
      Estimate Std. Error t value Pr(>|t|)
Linf 348.87538    2.88692  120.85  < 2e-16 ***
K1     0.35209    0.01790   19.67 5.24e-15 ***
t01    9.94911    0.09409  105.74  < 2e-16 ***
K2     0.24616    0.01065   23.12  < 2e-16 ***
t02    9.64774    0.11252   85.74  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.652 on 21 degrees of freedom

Number of iterations to convergence: 7
Achieved convergence tolerance: 4.651e-06

Cppkg<-(nls(PP12$vha ~ sit1*Linf1*(1-exp(-K*(PP12$t-t01))) + sit2*Linf2*(1-exp(-K*(PP12$t-
t02))), start = list(Linf1=a, K=b, t01=c, Linf2=a, t02=c)))#modelos con K comun

summary(Cppkg)

Formula: PP12$vha ~ sit1 * Linf1 * (1 - exp(-K * (PP12$t - t01))) + sit2 *
    Linf2 * (1 - exp(-K * (PP12$t - t02)))

Parameters:
       Estimate Std. Error t value Pr(>|t|)
Linf1 3.582e+02  1.746e+00  205.13   <2e-16 ***
K     3.102e-01  7.024e-03   44.16   <2e-16 ***
t01   9.788e+00  4.839e-02  202.27   <2e-16 ***
Linf2 3.309e+02  1.725e+00  191.81   <2e-16 ***
t02   9.969e+00  4.613e-02  216.11   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.075 on 21 degrees of freedom

Number of iterations to convergence: 5
Achieved convergence tolerance: 1.121e-06

Cppto<-(nls(PP12$vha ~ sit1*Linf1*(1-exp(-K1*(PP12$t-t0))) + sit2*Linf2*(1-exp(-K2*(PP12$t-
t0))), start = list(Linf1=a, K1=b, t0=c, Linf2=a, K2=b)))
> summary(Cppto)

Formula: PP12$vha ~ sit1 * Linf1 * (1 - exp(-K1 * (PP12$t - t0))) + sit2 *
    Linf2 * (1 - exp(-K2 * (PP12$t - t0)))

Parameters:
       Estimate Std. Error t value Pr(>|t|)
Linf1 3.557e+02  1.815e+00  195.99   <2e-16 ***
K1    3.258e-01  7.955e-03   40.96   <2e-16 ***
t0    9.875e+00  4.000e-02  246.86   <2e-16 ***
Linf2 3.338e+02  2.012e+00  165.89   <2e-16 ***
K2    2.936e-01  7.413e-03   39.60   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.958 on 21 degrees of freedom

Number of iterations to convergence: 5
Achieved convergence tolerance: 3.617e-07
```

Modelo más adecuado. Es necesario ver cuál de ellos es más adecuado:

```
AIC(PPconj,Cppg,Cpplinf,Cppkg,Cppto)
        df      AIC
PPconj   7 137.5695
Cppg     4 215.9240
Cpplinf  6 170.2943
Cppkg    6 138.6422
```

```
Cppto     6 136.6290
```

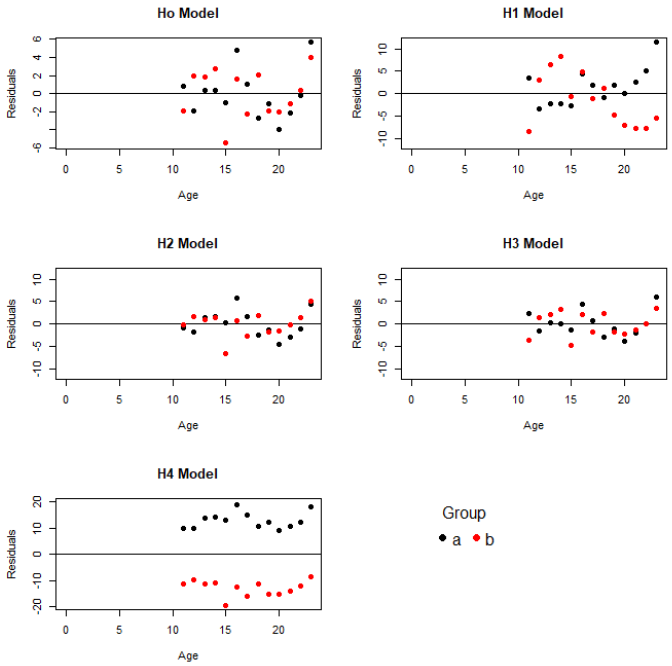
El AIC penaliza los modelos con mayor número de parámetros, y resulta insensato ajustar con muchos parámetros, por ello vale la pena aquel que menor número de ellos tenga.

Como regla general el modelo de más bajo AIC sería el mejor, pero las diferencias en este puntaje < 2 no resultan obligantes. Acá dos modelos Cppkg y Cppto resultaron casi semejantes al conjunto.

Jerarquías anidadas (McBride). Codificar lo anterior resulta complicado y propenso a equivocaciones. Las aproximaciones anteriores debidas a (Kimura) están en R (paquete fishmethods).

Como antes la pregunta es si son mejores modelos separados por sitio o un modelo general. En “fishmethods’ existe el commando, vblrt, (Likelihood Ratio Tests for Comparing Multiple von Bertalanffy Growth Curves), para verificar diferencias entre modelos VB anidados por medio de razones de máxima verosimilitud. (Ojo la variable sitio debe ser no numérica, para lo cual deb crear otra como la mostrada aca is=a para el sitio 1 y b para el 2).

```
PBab<-read.table("clipboard")
attach(PBab)
The following objects are masked from PP12:

    sit, sit1, t, vha

The following objects are masked from pppb:

    sit, t, vha

names(PBab)
[1] "t"    "vha"  "sit"  "sit1"

PPPBLLR<-vblrt(len=pppb$vha, age = pppb$t, group=PBab$sit1, error=2, select=1, plottype=2)
```



El termino error: Asunción de la varianza de los errores,así: 1= varianza constante para todos los lijs; 2= varianza constante para todas las medias a la edad (t) dada; 3=varianza de los lij variando con la edad. Los estadísticos requeridos para cada de error son calculados de las observaciones individuales longitud-edad.

```
PPPBLLR$results
     tests                hypothesis chisq df     p
1 Ho vs H1               Linf1=Linf2 34.72  1 0.000
2 Ho vs H2                     K1=K2  3.07  1 0.080
3 Ho vs H3                   t01=t02  1.06  1 0.303
4 Ho vs H4 Linf1=Linf2,K1=K2,t01=t02 84.35  3 0.000
```

Las razones de varianza se hacen contra una χ_2 para verificar si los modelos que tienen uno o más parámetros en común (H1-H4) son diferentes del modelo conjunto o general (Ho), que permite que los 3 parámetros varíen por índice de sitio.

La fuerte diferencia entre el modelo conjunto con el modelo sin sitios específicos, parámetros (H4) y el modelo conjunto o pleno (Ho) puede interpretarse dasometricamente como que el índice de sitio influye en los volúmenes. Entonces el modelo que mantiene los 3 parámetros en común debe ser rechazado. De acuerdo con los términos de los parámetros individuales, se ve que hay diferencias asintóticas, no para K y to.

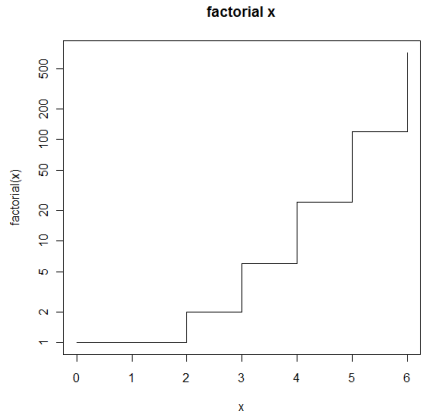
Otras funciones de Probabilidad.

Se muestran las más famosas que deberían ser conocidas por todo IF,*aunque se que se darán repeticiones, se exponen a pesar de ello*.

La función **factorial** se usa en el análisis combinatorio, por ejemplo, para calcular el número de permutaciones de n items. La función en R para ello es factorial(x) y puede graficarse por ejemplo entre 0 y 10 usando la función escala (**step**) acudiendo a la opción (**type="s"**), dentro de la gráfica (**plot**) con una escala logarítmica en el eje y como log="y", por ejemplo para valores entre 0 y 6

x<-0:6

plot(x,factorial(x),type="s",main="factorial x",log="y")



Coeficientes binomiales.

Muestran el número de éxitos en n items con probabilidad constante en un espacio muestral con experimentos Bernoulli. Por ejemplo, de 8 individuos queremos saber cuántas formas hay de que 3 de ellos fueran varones y 5 mujeres. La respuesta es dada por:

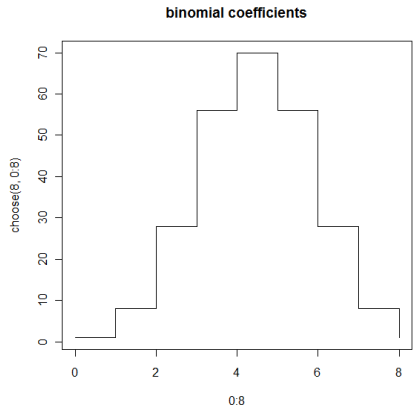
$$\binom{n}{x} = \frac{n!}{x!(n-x)!}; \quad \binom{8}{3} = \frac{8!}{3!(8-3)!} = \frac{8\times 7\times 6}{3\times 2} = 56$$

En R la función para ello es: **choose(n,x)**, en nuestro caso:

```
choose(8,3)
[1] 56
```

El siguiente es el gráfico del número de formas de seleccionar de 0 a 8 varones entre 8 individuos:

plot(0:8,choose(8,0:8),type="s",main="binomial coefficients")



Distribuciones continuas de probabilidad.

R tiene muchas funciones incorporadas de distribuciones de probabilidad para cada una de las cuales es posible obtener 4 funciones:
La función de densidad de probabilidad (cuyo **prefijo** es d);
La función de distribución acumulativa de probabilidades (con **prefijo** p);
Los cuantiles de la distribución (q);
y un generador de números aleatorios de la distribución madre (r). Cada letra puede ser prefijada a los nombres de las funciones de R como se muestra en la Tabla (ejemplos: dbeta, qnorm).

Función R	Distribución	Parámetros
beta	beta	forma1, forma2
binom	binomial	tamaño muestral, probabilidad
cauchy	Cauchy	localización, escala
exp	exponencial	rata (opcional)
chisq	chi-cuadrada	grados de libertad
f	F de Fisher	g.l.1, g.l.2
gamma	gamma	forma
geom	geometrica	probabilidad
hyper	hipergeometrica	*m, n, k*
lnorm	lognormal	media, desviación estándar
logis	logistica	localización, escala
nbinom	binomial negativa	Tamaño, probabilidad
norm	normal	media, desviación estándar
pois	Poisson	media
signrank	Wilcoxon signed rank statistic	tamaño muestral n
t	t de Student	grados de libertad

unif	uniforme	minimo, maximo (opc.)
weibull	Weibull	forma
wilcox	Wilcoxon rank sum	*m*, *n*

Función de distribución acumulativa (*cumulative probability*).

Esta noción es muy importante, se trata de una **curva en forma de S** que muestra para algún valor de la variable ***X=x***, la probabilidad de obtener un valor muestral que es menor o igual a *x*. Por ejemplo con una distribución normal luce así, parte izquierda de la siguiente gráfica:

```
par(mfrow=c(1,2))
curve(pnorm(x),-3,3)
arrows(-1,0,-1,pnorm(-1),col="red")
arrows(-1,pnorm(-1),-3,pnorm(-1),col="green")
```

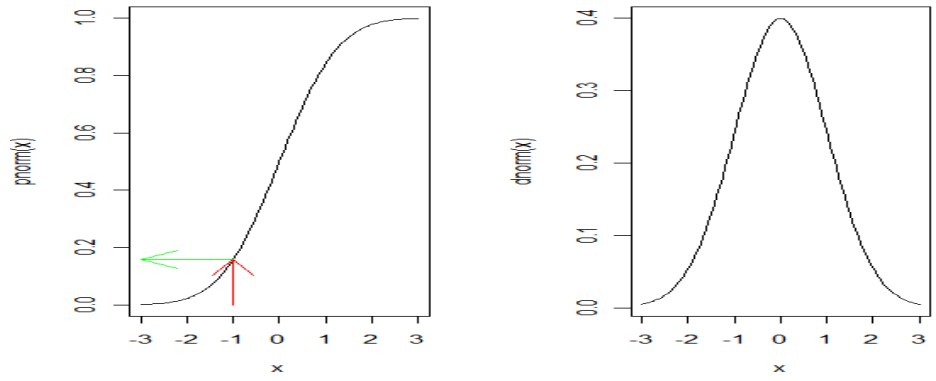
El valor de x(−1) presenta una probabilidad acumulada (flecha roja) y su probabilidad asociada de obtener un x <=−1 en el eje y (flecha verde), parte izquierda de la grafica. El valor en el eje y es 0.1586553:

```
pnorm(-1)
[1] 0.1586553
```

La densidad de probabilidades es la pendiente de la curva anterior (su 'derivada'). Por ejemplo con:

```
curve(dnorm(x),-3,3)
```

Puede verse al mismo tiempo que la pendiente nunca es negativa. La pendiente empieza muy suave hasta *x=−2*, de allí en adelante se incrementa hasta un pico (en *x = 0* para este ejemplo) luego empinadamente en un comportamiento simétrico llega hasta *x=2* y vuelve y se suaviza, parte derecha de la grafica:



Distribución Normal.

Esta distribución es de las más importantes en la teoría de la estadística paramétrica. Considérese la siguiente función exponencial simple:

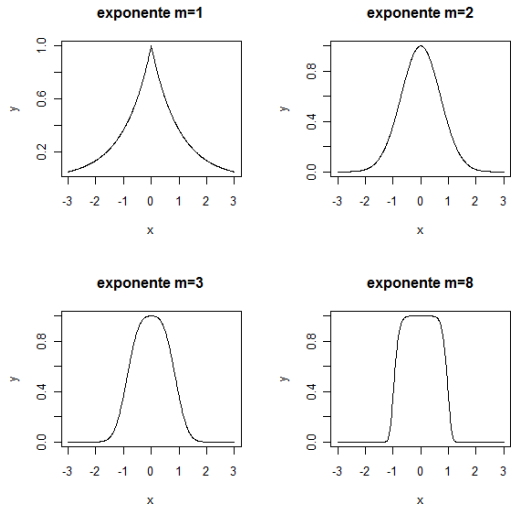
$$y = \exp(-|x|^m)$$

en la medida que el exponente (*m*) se incrementa, la función llega ser similar a una función escalonada. Los paneles siguientes muestran la relación entre las variables y y x para m=1, 2, 3 y 8, respectivamente:

```
par(mfrow=c(2,2))
x<-seq(-3,3,0.01)
```

```
y<-exp(-abs(x))
plot(x,y,type="l" ,ylab="y",main="exponente m=1")
y<-exp(-abs(x)^2)
plot(x,y,type="l",ylab="y",main="exponente m=2")
y<-exp(-abs(x)^3)
plot(x,y,type="l",ylab="y",main="exponente m=3")
y<-exp(-abs(x)^8)
plot(x,y,type="l",ylab="y",main="exponente m=8")
```

El panel superior derecho $y = \exp(-x^2)$, es la base de una función de densidad de probabilidades extremadamente importante y famosa. Una vez que ella ha sido escalada de tal manera que la integral (el área bajo la curva de $-\infty\ a\ \infty$) es la unidad, esta es la distribución normal.



Cuando la ***distribución*** anterior ***tiene media cero y desviación estándar 1***, la distribución normal estandarizada presenta la siguiente ecuación:

$$y = \frac{1}{\sqrt{2\pi}} \exp(-\frac{z^2}{2}) = \frac{1}{\sqrt{2\pi}} e^{(-\frac{z^2}{2})}$$

con la cual se resuelven muchos problemas de probabilidades. Ejemplo, hemos medido las estaturas (*y*) de 100 personas. La estatura media fue de 170 cm y la desviación estándar de 8 cm (panel superior derecho de la siguiente figura). Tres tipos de interrogantes podemos responder de unos datos como estos. Cuál es la probabilidad que un individuo aleatoriamente seleccionado:

- sea mas bajo que una estatura particular dada?
- sea mas alto que una estatura particular dada?
- esté entre dos estaturas dadas?

Ya sabemos que el área bajo la curva es exactamente 1, o 100%, por lo cual cualquier individuo tiene una y entre -∞ e ∞. Cualquiera de las inquietudes anteriores se resuelve en z. Por ejemplo, cuantos individuos tendrán una *y* menor de 160 cm. Debemos convertir esta *y* en valores de *z*, es decir en un número de desviaciones estándar de la media, por medio de:

$$z = \frac{y - \bar{y}}{s}; \quad z = \frac{160 - 170}{8} = -1.25; \text{ entonces } \ p(y \le 160 \text{ cm}) = p(z \le -1.25) \ \infty$$

Esta es un área en la cola izquierda (la integral) de la función de densidad. La función en R para ello es **pnorm(z)**: o sea que debemos proporcionar el cuantil de *z* para la probabilidad buscada:

pnorm(-1.25)

[1] 0.1056498

Entonces, 10.5% de las personas del área de muestreo miden menos de 160 cm de estatura. Así mismo nos podríamos preguntar qué porcentaje de personas miden más de 185 cm. La *z* para esta estatura será 1.875. Ya que conoceríamos la probabilidad acumulada hasta esta *z*, entonces usamos:

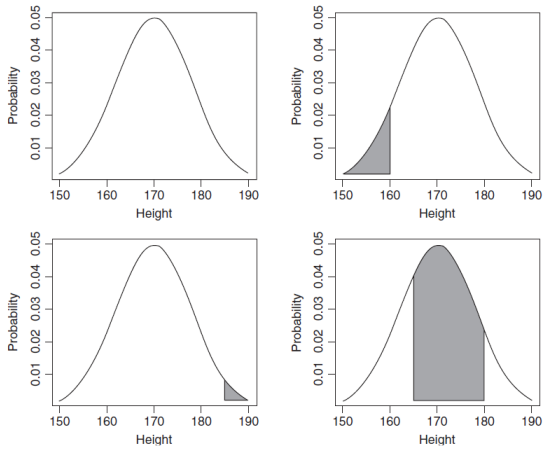
1-pnorm(1.875)

[1] 0.03039636

Solo el 3% alcanza estaturas superiores a 185 cm. Finalmente podríamos preguntarnos por el porcentaje de personas con estaturas entre 165 cm (*z= -0.625*) y 180 cm (*z=1.25*) obteniendo 63%:

pnorm(1.25)-pnorm(-0.625)

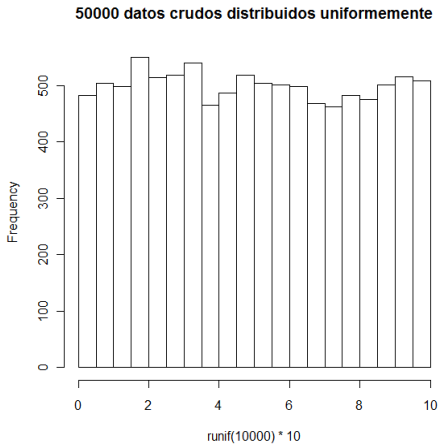
[1] 0.6283647



Teorema del límite central.

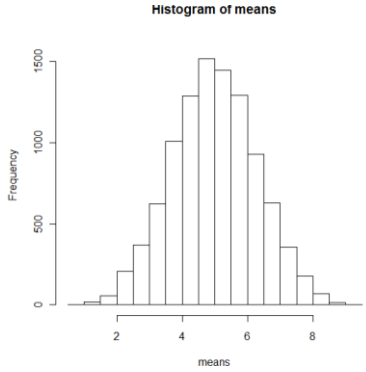
Se generaliza de la siguiente forma: Sí tomamos muestras repetidas de una población con varianza finita y calculamos las medias de cada muestra, estas se distribuirán normalmente. Se pueden mostrar resultados que avalan lo anterior. Tomemos varias muestras de 5 números aleatorios (entre 0 y 10) de una distribución uniforme, y encontrémosle su media. Por ejemplo para una muestra como 2,3,1,2,1 su media sería muy baja y por el contrario una como 9,8,9,6,8 la tendría muy alta. Desde luego el promedio que esperaríamos la mayoría de las veces estaría cercano a 5. Realicemos esto 10000 veces y veamos qué pasa con la distribución de las 10000 muestras. Los datos serán rectangularmente (uniformemente) distribuidos en el intervalo de 0 a 10, por lo cual la distribución de los datos crudos sería bastante aplanada en la cima de ellos. En R lo podemos apreciar con:

```
hist(runif(10000)*10,main="50000 datos crudos distribuidos uniformemente")
```



¿Qué pasa con la distribución de las 10000 medias muestrales basadas en la toma de 5 muestras aleatorias uniformemente distribuidas? Se debe construir un vector con muchos ceros con numeric (x) por ejemplo con 10000, para ir remplazando cada uno con una media de las muestras aleatoriamente seleccionadas, con:

```
means<-numeric(10000)
for (i in 1:10000){
means[i]<-mean(runif(5)*10)
}
hist(means,ylim=c(0,1600))
```



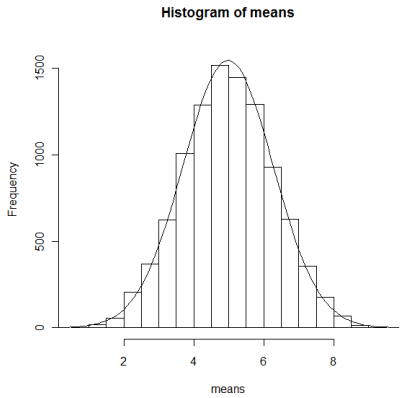
¿Qué tan cercana está la distribución de nuestras medias de una normal? Una forma de saberlo es dibujando una normal con los mismos parámetros en las cimas del histograma. Podemos estimar los dos parámetros (media y varianza) de nuestra muestra de 10000 medias (acuérdese que los valores de estas medias deberían ser bastante diferentes debido a la aleatorización):

```
mean(means)
[1] 4.98049
sd(means)
[1] 1.289689
```

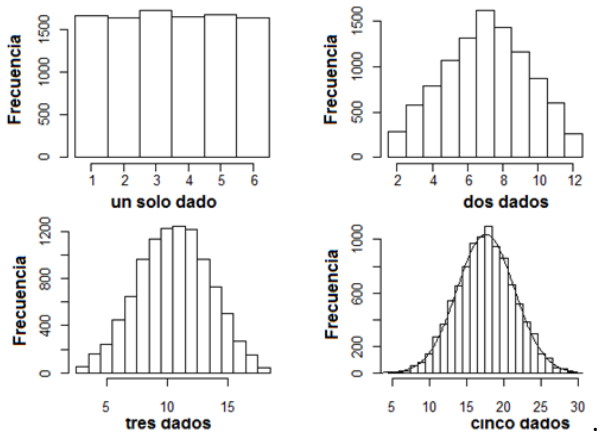
Con los anteriores en la función de R de densidad de probabilidades de la distribución normal (**dnorm**) creamos la curva suavizada por medio de series de valores del eje x; la inspección de los histogramas sugieren que los límites sensibles estarán entre 0 y 10 (los mismos de la distribución uniforme). Una regla generalizada establece que para esta curva son buenos siquiera 100 valores, lo cual logramos con:

```
xv<-seq(0,10,0.1)
yv<-dnorm(xv,mean=4.98049,sd=1.289689)*5000
```

lines(xv,yv)



Se ve un ajuste excelente, o sea que se cumplió en este caso el teorema del límite central. Con otra población no normal por ejemplo la tirada de un solo dado, luego 2,3,5 muchas veces, muestra el cumplimiento del teorema del límite central.



Cada uno de los 6 números se comporta como una distribución uniforme, todos equiprobables, izquierda arriba. Luego dos, luego tres y cinco dados, estos últimos dan media = 17.5937, sd = 3.782198.

Máxima verosimilitud con la distribución normal.

La función de densidad de probabilidades de la normal es:

$$f(y \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}[(y-\mu)/\sigma]^2} \quad -\infty < y < \infty$$

Para una combinación dada de μ y σ^2 esta función dará las probabilidades entre 0 y 1. Se recuerda que la verosimilitud (*likelihood*) es el producto de las densidades de probabilidad, para cada uno de los valores de la variable respuesta *y*. O sea, si tenemos *n* valores de *y* en nuestra muestra o experimento, la función de verosimilitud será:

$$L(\mu,\sigma) = \prod_{i=1}^{n} (\frac{1}{\sqrt{2\pi}\sigma} Exp[-\frac{(y_i-\mu)^2}{2\sigma^2}])$$

donde el único cambio es que *y* ha sido remplazado por y_i y luego se multiplican todas las probabilidades juntas para cada uno de los n puntos. Hay un poco de algebra para simplificar esto.

Nos podemos deshacer del operador producto $(\prod)$ en dos pasos: primero para el término constante que es multiplicado por si mismo n veces, podemos escribirlo como: $\frac{1}{(\sqrt{2\pi}\sigma)^n}$

Segundo, el producto de un conjunto de **antilogs (exp)** puede escribirse como el antilog de una suma de valores de of y_i similar a esto: $\prod \exp(y_i) = \exp(\sum y_i)$. Esto significa que el producto de la parte derecha de la expresión puede escribirse como:

$$Exp[-\frac{\sum_{i=1}^{n}(y_i-\mu)^2}{2\sigma^2}]$$

por lo cual podemos reescribir la verosimilitud de la distribución normal como:

$$L(\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} Exp[-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-\mu)^2]$$

Ambos parámetros $\mu\ y\ \sigma^2$ son desconocidos y el propósito de este ejercicio es usar la modelación estadística para determinar sus valores de máxima verosimilitud de los datos (los n diferentes valores de y). ¿Pero cómo encontrar los valores de $\mu\ y\ \sigma$ que maximicen su verosimilitud? La respuesta la tiene el cálculo: primero encontramos la derivada de la función con respecto a los parámetros, igualamos a cero, y resolvemos. También a causa de la **función exp** en la ecuación, es más fácil la solución con el log de la verosimilitud (*log(L)*):

$$L(\mu,\sigma) = -\frac{n}{2}\log(2\pi) - n\log(\sigma) - \sum_{i=1}^{n}\frac{(y_i-\mu)^2}{2\sigma^2}$$

y maximizar esta en vez de la otra. Obviamente, los valores de los parámetros que maximizan el log(verosimilitud) $l(\mu,\sigma) = \log(L(\mu,\sigma))$ serán los mismos que maximicen la verosimilitud. La sumatoria muestra que es entre i01 a n.

Después del álgebra viene el cálculo. Empecemos con la media μ. La derivada del log(verosimilitud) con respecto a μ es:

$$\frac{dl}{d\mu} = \sum_{i=1}^{n}\frac{(y_i-\mu)}{\sigma^2}$$

Ajustamos la derivada a cero y se resuelve para μ:

$$\sum_{i=1}^{n}\frac{(y_i-\mu)}{\sigma^2} = 0; \quad \text{tal que} \sum_{i}^{n}(y_i-\mu) = 0$$

De la última vemos que el estimado máximo verosímil de μ es la media aritmética

$$\sum_{i}^{n}(y_i-\mu) = \sum_{i}^{n}(y_i) - n\mu) = 0 \therefore \quad \mu = \frac{\sum_{i}^{n}(y_i)}{n}$$

Se procede en seguida a encontrar la derivada del log(verosimilitud) con respecto a σ:

$$\frac{dl}{d\sigma} = -\frac{n}{\sigma} + \sum_{i=1}^{n}\frac{(y_i-\mu)^2}{\sigma^3}$$

Ejecutando un proceso similar al anterior: se llega a:

$$-\frac{n}{\sigma} + \sum_{i=1}^{n}\frac{(y_i-\mu)^2}{\sigma^3} = 0 \ \therefore \sum_{i=1}^{n}(y_i-\mu)^2 = \sigma^3(\frac{n}{\sigma}) = \sigma^2 n \quad \therefore \quad \sigma^2 = \frac{\sum_{i=1}^{n}(y_i-\mu)^2}{n}$$

Entonces le estimado máximo verosímil de la varianza no es más que el promedio aritmético de las desviaciones al cuadrado de los datos con respecto a μ. Como es un estimador sesgado, se evita con (n-1).

R tiene sus funciones de probabilidad incorporadas en el contexto de la distribución normal. La función de densidad (**dnorm**) tiene un valor de z (un cuantil) como su argumento. Argumentos opcionales especifican la media y la desviación estándar. (Entonces por defecto se tiene la normal estandarizada con media 0 y varianza 1). Valores de z por fuera del rango −3.5 a +3.5 son muy improbable:Un ejemplo lo ilustra, grafica superior izquierda:

par(mfrow=c(2,2))
curve(dnorm,-3,3,xlab="z",ylab="Probabilidades de densidad",main="Densidad")

La función de probabilidades (**pnorm**), gráfica superior derecha, también tiene un valor de z (un cuantil) como argumento. Argumentos opcionales especifican la media y la desviación estándar. (Entonces por defecto se tiene la normal estandarizada con media 0 y varianza 1). Esta función muestra la probabilidad acumulada de valores de z menores o iguales al valor especificado y tiene forma de una curva tipo S:

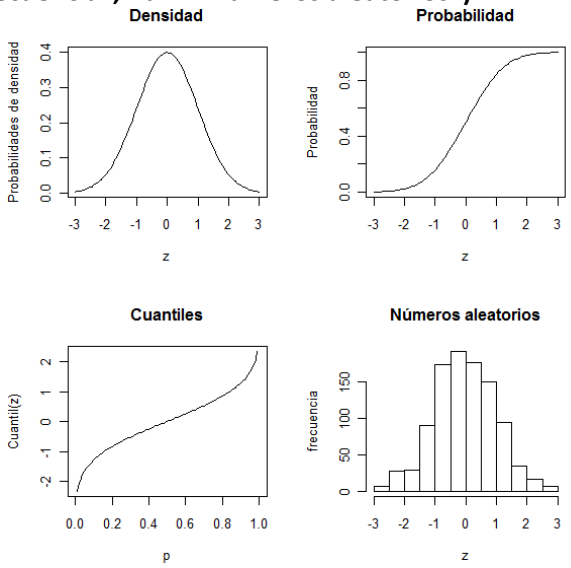
curve(pnorm,-3,3,xlab="z",ylab="Probabilidad",main="Probabilidad")

Los cuantiles de la distribución normal (**qnorm**) tienen una probabilidad acumulada como argumento, grafica inferior izquierda. Ejecutan la opuesta de la función (**pnorm**), al retornar un valor de z cuando se le proporciona como argumento la probabbilidad.

curve(qnorm,0,1,xlab="p",ylab="Cuantil(z)",main="Cuantiles")

El generador de números aleatorios reales dentro de una normal, lo hacemos con la función (**rnorm**) al especificar la media y la desviación estándar, grafica inferior derecha. El primer argumento es el número deseado de números aleatorios a generar. Por ejemplo si deseamos 1000 con media 0 y desviación estandar 1escribimos:

y<-rnorm(1000)
hist(y,xlab="z",ylab="frecuencia",main="Números aleatorios")



Si deseamos generar números aleatorios con una media y desviación estándar aproximadas usamos **rnorm(n,media,desviación estándar)**. Ejemplo generar 100 números aleatorios con media 15 y desviación estándar 6:

vaaleat<-rnorm(100,15,6)
mean(vaaleat)
[1] 15.63647

Si lo que deseamos es una muestra con 100 números aleatorios con media y desviación estándar, exactas, lo debemos hacer en dos pasos. Primero generamos 100 valores aleatorios de z:
zvaaleat<-rnorm(100,0,1)
mean(zvaaleat)
[1] -0.01956842
Entonces debemos compensar lo anterior, pues sucede lo mismo que en el primer caso. Entonces debemos calcular:
zvaaleat <-(zvaaleat -mean(zvaaleat))/sd(zvaaleat)

Este si queda con media cero y varianza 1. Enseguida multiplicamos este vector por su desviación estándar deseada (6 para el ejemplo) y se lo adicionamos a la media deseada (15) y obtenemos lo deseado.
vaaleat<-15 + zvaaleat*6
mean(vaaleat)
[1] 15
sd(vaaleat)
[1] 6

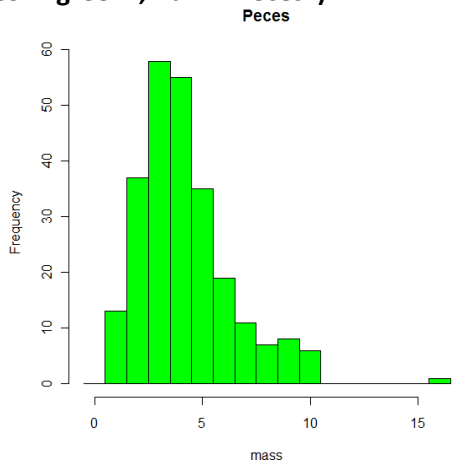
Las cuatro funciones anteriores (d, p, q y r) trabajan en forma similar con cualquier otra función de distribución de probabilidades.

Comparación de datos con la distribución normal.

Existen varias pruebas de normalidad, pero acá se trata de comparar un histograma de datos reales con una curva normal suavizada con igual media y desviación estándar con el fin de buscar evidencias de no-normalidad (como sesgos, curtosis, etc). Veámoslo con un archivo llamado peces (**fishes**):
peces<-read.table("c:\\estadistica\\fishes.txt",header=T)
attach(peces)
names(peces)
[1] "mass"
> mean(mass)
[1] 4.194275
> max(mass)
[1] 15.53216

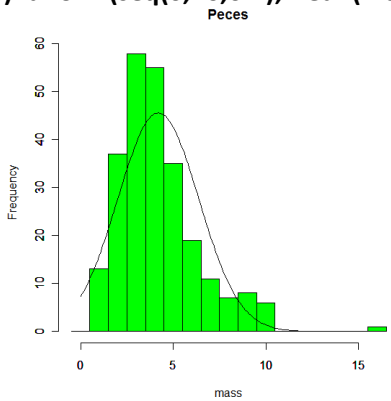
Vamos a construir el histograma de la variable mass, especificando cajuelas enteras (bins) con un gramo de ancho hasta un máximo de 16.5 g con:

hist(mass,breaks=-0.5:16.5,col="green",main="Peces")



Para los propósitos de la demostración generaremos todo lo necesario dentro de la función lines: la secuencia de valores x para graficar (de 0 a 16), y la altura de la función de densidad (el número de peces (length(mass)) por la probabilidad de cada miembro de la secuencia, para una normal (mass promedio, desviación estándar de mass) tal como esto:

lines(seq(0,16,0.1),length(mass)*dnorm(seq(0,16,0.1),mean(mass),sqrt(var(mass))))



La distribución de la masa de los peces no es claramente normal. Hay bastantes más peces de 3 y 4 gr, pero también pocos de 6 o 7 gr, y también algunos peces realmente grandes de más de 8 gr, por lo cual esta distribución tiene probablemente mejores ajuste por otro tipo de eistribuciones continuas por ejemplo la función gamma.

Otras distribuciones usadas en verificación de hipótesis.

Otras distribuciones usadas con este propósito son: La ***chi cuadrada*** (chi-squared), útil en conteos de datos, la de ***Fisher*** (F), en anavas y comparación de 2 varianzas; y la t de Student, con pequeñas muestras para comparación de medias. Todas ellas se basan en un estadístico calculado que pudiera esperarse por azar cuando lo que planteamos en la hipótesis nula se da. Un gran valor de estos estadísticos nos informa de lo sucedido con respecto a la hipótesis alternativa y es, que esta es cierta.

Por ejemplo supóngase una chi cuadrada calculada de 12.5 con 11 grados de libertad. Cuál de las hipótesis respaldaríamos?. Siguiendo las instrucciones dadas en la normal, escribiríamos al 95%:

1-pchisq(12.5,11)

[1] 0.3272558

Con base en lo anterior consideraríamos que 12.5 es un relativo pequeño valor, por lo cual respalda la nula, pues desearíamos que la probabilidad 1-pchisq(12.5,11) hubiera sido menor al 5% antes de rechazarla. Cual debería haber sido la chi cuadrada necesaria para rechazar la hipóteis nula? Para ello escribimos una función con dos argumentos como esta:

qchisq(0.95,11)

[1] 19.67514

Un valor igual o superior a este hubiera sido necesario para rechazar la nula. En idéntica forma procederíamos con una F o una t. Por ejemplo la probabilidad de tener una razón de varianzas de 3 por azar solamente cuando la hipótesis nula es cierta con 10 grados de libertad en el numerador y 14 en el denominador, lo haríamos escribiendo:

1-pf(3.00,10,14)

[1] 0.02997775

Esta probabilidad (0.02997775) está por debajo de 0.05 nos lleva rechazar la nula. El cuantil F límite para nuestro propósito seria 2.602155, obtenido con:

qf(0.95,10,14)

[1] 2.602155

Similarmente sucede con la t, por ejemplo el valor de t en tabla, para una prueba de dos colas dejando a lado y lado un $\alpha = 0.025$ con 15 grados de libertad por ejemplo, se obtiene como:

qt(0.975,15)

[1] 2.131450

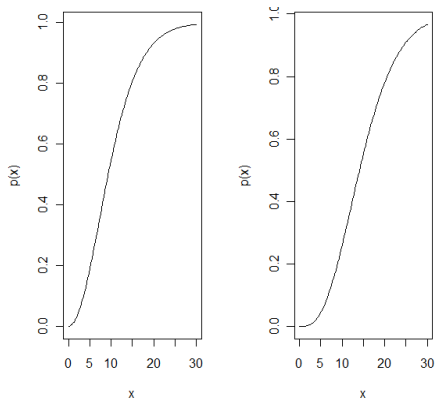
chi-cuadrada.

Es un caso especial de la distribución gamma caracterizada por un único parámetro, sus grados de libertad (υ = g.l.). Su media es igual a υ y su varianza a 2υ. Su función de densidad es

$$f(x) = \frac{1}{2^{\upsilon/2}\Gamma(\upsilon/2)} \chi^{\upsilon/2-1} e^{-\chi/2}$$

Es importante porque muchas formas cuadráticas siguen esta distribución, si los datos se comportan normalmente. Por ejemplo, la varianza es una variable escalada a chi cuadrada, la razón de verosimilitud también se distribuye aproximadamente como tal. Cuando se usa la probabilidad acumulativa, aparece un tercer argumento para describir no centralidad. Sí la chi cuadrada no central es una suma de variables normales independientes aleatorias, el parámetro de no centralidad es igual a la suma de las medias al cuadrado de las variables normales. Por ejemplo, se muestra la grafica de probabilidad acumulada para un parámetro no central (ncp) basado en 3 medias normales de (1, 1.5 y 2) y otro con 4 medias y un ncp=10:

```
par(mfrow=c(1,2))
x<-seq(0,30,.25)
plot(x,pchisq(x,3,7.25),type="l",ylab="p(x)",xlab="x")
plot(x,pchisq(x,5,10),type="l",ylab="p(x)",xlab="x")
```



La grafica de la izquierda tiene 3 g.l. y un parámetro de no centralidad de $1^2+1.5^2+2^2=7.25$, mientras la de la derecha tiene 5 grados de libertad incluida la no centralidad y 10 para este valor (nótese una cola más larga a la izquierda para bajas probabilidades).

La chi-cuadrada se usa también para establecer intervalos de confianza para las varianzas muestrales basados en su propiedad $\frac{(n-1)s^2}{\sigma^2} \sim \chi^2_{(n-1)}$. Con esta se pueden establecer intervalos de confianza al 95% de acuerdo con:

$$\frac{(n-1)s^2}{\chi^2_{1-\alpha/2}} \leq \sigma^2 \leq \frac{(n-1)s^2}{\chi^2_{\alpha/2}}$$

Ejemplo sea una varianza muestral s^2 =10.2 con 8 g.l. El intervalo para ella está dado por:
8*10.2/qchisq(.975,8)
[1] 4.653670

8*10.2/qchisq(.025,8)
[1] 37.43582

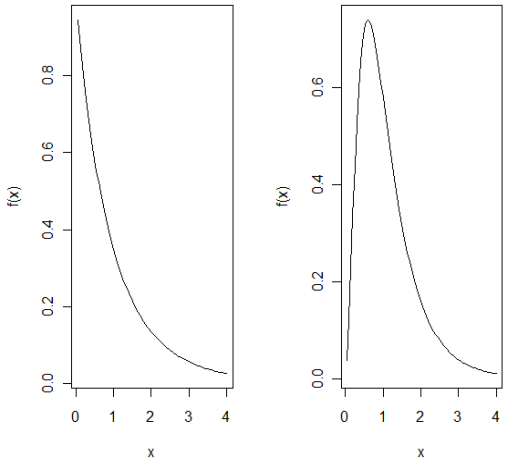
F de Fisher.

La distribución de F es una biparamétrica, definida por la función de densidad:

$$f(x)=\frac{r\Gamma(1/2(r+s))}{s\Gamma(1/2r)\Gamma(1/2s))}\frac{(rx/s)^{(r-1)/2}}{[1+(rx/s)]^{(r+s)/2}}$$

r = g.l. del numerador y **s** = g.l. del denominador. Denominada así por R. A. Fisher, el padre del análisis de varianza, su principal uso. Tiene usualmente 3 argumentos, la probabilidad para una prueba de una cola (usualmente 0.95), los grados de libertad del numerador y los grados de libertad del denominador. Por ejemplo el cuantil F para .95 con 2 y 18 grados de libertad se logra con:
qf(.95,2,18)
[1] 3.554557

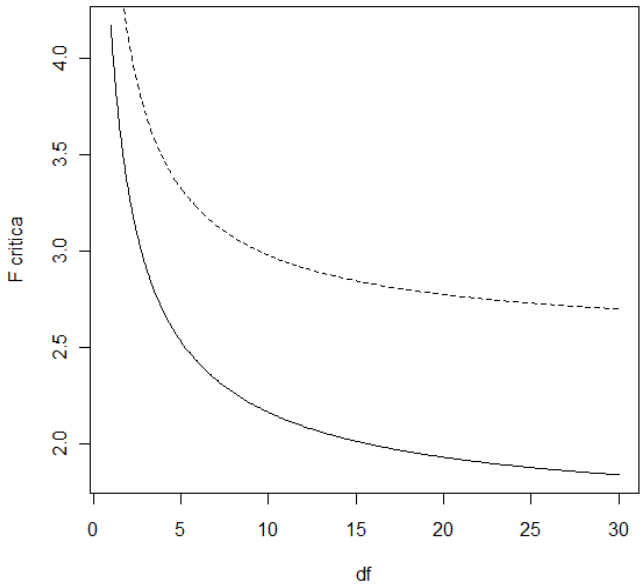
Así luce la F para 2 y 18 g.l. (izquierda) y 6 y 18 g.l. (derecha):
x<-seq(0.05,4,0.05)
plot(x,df(x,2,18),type="l",ylab="f(x)",xlab="x")
plot(x,df(x,6,18),type="l",ylab="f(x)",xlab="x")



Cuando hay un solo grado de libertad para el numerador, la distribución es igual a la t de Student al cuadrado ($F = t^2$). Un valor generalizado de F=4 se convierte en un valor crítico para las pruebas de hipótesis. Para verlo podemos graficar una F crítica contra los g.l. en el numerador:

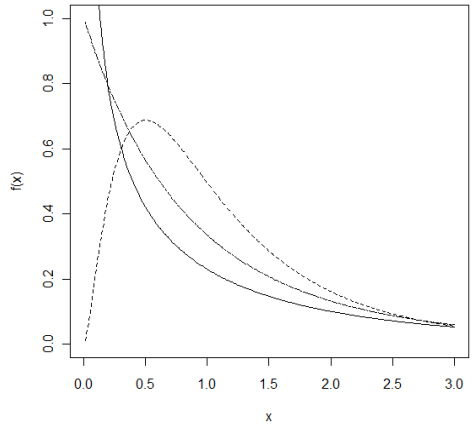
```
df<-seq(1,30,.1)
plot(df,qf(.95,df,30),type="l",ylab="F critica")
lines(df,qf(.95,df,10),lty=2)
```

La regla generalizada (***F critica=4***) muestra que rápidamente llega a ser mucho mayor una vez que los g.l. en el numerador sean mayores de 2. La línea sólida (debajo) muestra los valores críticos de F cuando el denominador tiene 30 g.l. y la superior (punteada) muestra el caso cuando el denominador tiene 10 g.l.



La forma de la función de densidad de la distribución F depende de los grados de libertad del numerador:

```
x<-seq(0.01,3,0.01)
plot(x,df(x,1,10),type="l",ylim=c(0,1),ylab="f(x)")
lines(x,df(x,2,10),lty=6)
lines(x,df(x,5,10),lty=2)
lines(x,df(x,30,10),lty=3)
```



La densidad de probabilidad f(x) declina monotónicamente cuando el numerador tiene 1 o 2 g.l. y alcanza un máximo para 3 o más g.l. (acá se muestran los de 5 y 30), con los g.l. del denominador de 10 para todos los gráficos.

t de Student.

Distribución de ***W.S. Gossett*** (1908) conocida como de ***student*** por no poder publicarla por restricciones de la Guinness Brewing Co para publicar algo con el nombre de sus empleados. Es una función uniparamétrica (r):

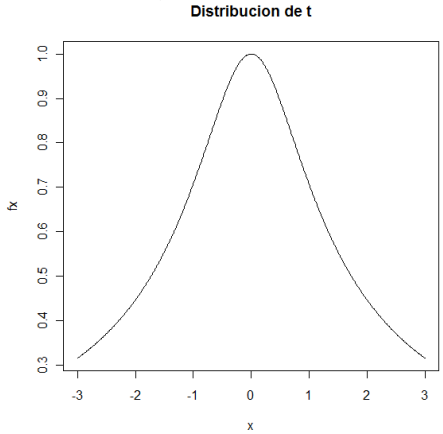
$$f(x) = \frac{\Gamma(1/2(r+1))}{(\pi r)^{1/2}\Gamma(1/2r)}\left(1+\frac{x^2}{r}\right)^{-(r+1)/2}$$

en la cual $-\infty < x < +\infty$. Una vez dejadas de lado las constantes aparece la siguiente mucho mas simple:

$$f(x) = (1+x^2)^{-1/2}$$

La podemos graficar para $-3 < x < +3$ como sigue:

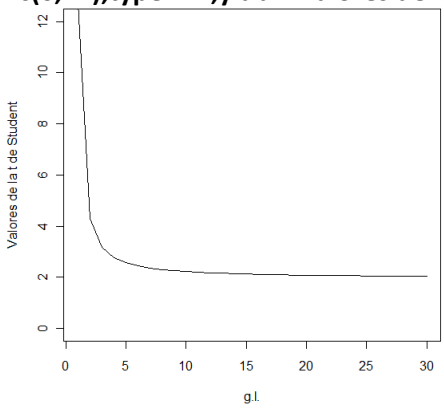
```
x<-seq(-3,3,0.01)
fx<-(1+x^2)^(-0.5)
plot(x,fx,type="l", main="Distribucion de t")
```



Las constantes son necesarias para poder escalar la función de modo que su integral sea 1, o sea el 100% de probabilidades. Sin detallar lo anterior se muestra lo sucedido con cuando *t=2* como *punto usualmente de referencia* al juzgarla. Para ello se graficará *t* vs *tamaño de muestra* para pequeñas

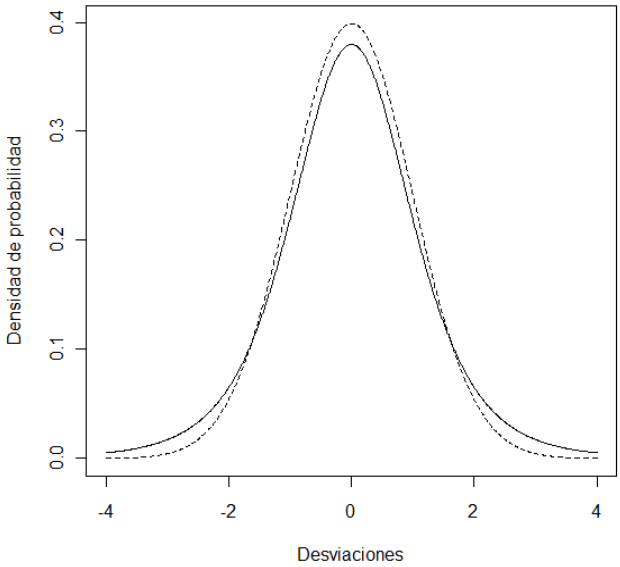
muestras. Para ello usaremos la función de R, **qt** (cuantil de t) y fijamos la probabilidad para un valor de dos colas de 0.975:

plot(1:30,qt(0.975,1:30), ylim=c(0,12),type="l",ylab="Valores de la t de Student",xlab="g.l.")



Como se ve la generalización funciona bien para 5 g.l. en adelante. Pero para muchos propósitos ***t≈2*** realmente da una buena idea para ciertas decisiones. Veamos como luce la t comparada con la normal estandarizada, la cual presentaremos con la línea punteada (lty=2), a la cual sobrepondremos una t con 5 g.l. como línea sólida para ver la diferencia:

xvs<-seq(-4,4,0.01)
plot(xvs,dnorm(xvs),type="l",lty=2,ylab="Densidad de probabilidad",xlab="Desviaciones")
lines(xvs,dt(xvs,df=5))



La diferencia fundamental está en las *colas más robustas de la t*. Lo anterior significa que los valores extremos son más probables con una t que con la normal y que los intervalos de confianza son más amplios. Así, en vez de un intervalo al 95% de ±1.96 con la distribución normal, tendríamos un intervalo de ±2.57 para la t de Student con 95% y 5 g.l. lo cual obtenemos con la función qt(probabilidad, g.l.):

qt(0.975,5)
[1] 2.570582

Distribución gamma.

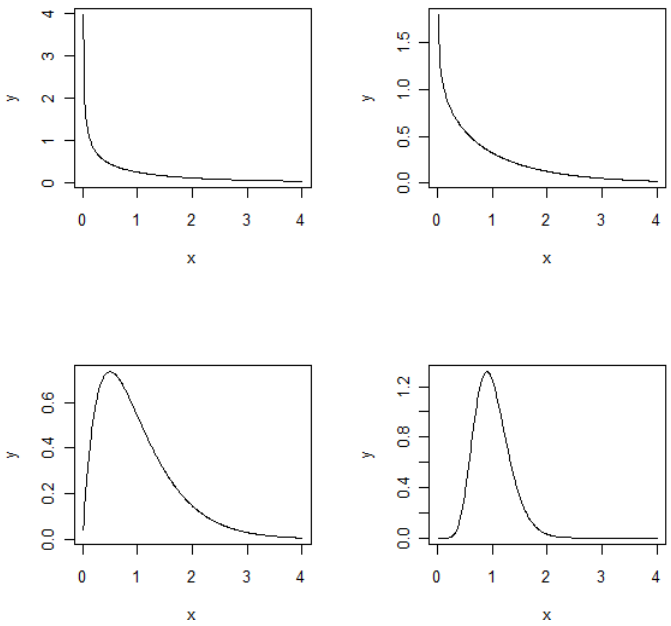
Útil para describir procesos donde los datos son positivamente sesgados (o sea, no normales con una larga cola a la derecha). Es una distribución biparamétrica cuyos parámetros se conocen como: ***forma*** y ***razón*** (*rate*). Su función de densidad es:

$$f(x) = \frac{1}{\beta^{\alpha}\Gamma(\alpha)} x^{\alpha-1} e^{-x/\beta}$$

en la que α es el parámetro de forma y β^{-1} es parámetro razón (media/varianza también conocido como parámetro de escala). Casos especiales de esta distribución son la **distribución exponencial** cuando $\alpha = 1$ y la vista **chi-cuadrada** $\alpha = \upsilon / 2$ *y* $\beta = 2$**.**

Para ver el efecto del parámetro de forma en la función de densidad de probabilidades graficaremos la distribución gamma para diferentes valores de forma y razón en un rango entre 0.1 y4:

```
par(mfrow=c(2,2))
x<-seq(0.01,4,.01)
par(mfrow=c(2,2))
y<-dgamma(x,.5,.5)
plot(x,y,type="l")
y<-dgamma(x,.8,.8)
plot(x,y,type="l")
y<-dgamma(x,2,2)
plot(x,y,type="l")
y<-dgamma(x,10,10)
plot(x,y,type="l")
```



Los gráficos de la parte superior izquierda hasta el del fondo a derecha se muestran para diferentes valores de forma de: 0.5, 0.8, 2 y 10. Nótese que $\alpha < 1$ produce funciones que declinan monotónicamente y $\alpha > 1$produce curvas con protuberancia que pasan a través del origen, cuyo grado de sesgo declina en la medida que se incrementa α .

La media de esta distribución es $\alpha\beta$, su varianza $\alpha\beta^2$, el sesgo $2/\sqrt{\alpha}$ y la curtosis $6/\alpha$. Así para la distribución exponencial tendremos una media de β, una varianza de β^2, un sesgo de 2 y una curtosis de 6. Para la chi-cuadrada tenemos una media de υ, una varianza 2υ, un sesgo de $2\sqrt{2/\upsilon}$ y una curtosis $12/\upsilon$. Se observa además que:

$$\beta = \frac{media}{\text{var}\,ianza} \quad y \quad forma = \frac{1}{\beta} media$$

Ahora podemos responder preguntas como: Que valor es el cuantil 95 esperado para una distribución gamma con media 2 y varianza 3? Ello implica entonces que la *razón* será *2/3* y la *forma 4/3* tal que, usando la función cuantil gamma, qgamma (probabilidad, forma, razón) tendríamos para el caso:
qgamma(0.95,2/3,4/3)
[1] 1.732096.

Un uso importante de esta distribución es para datos de medidas continuas no normalmente distribuidas. Por ejemplo, la masa corporal de 200 peces fue graficada como un histograma y superpuesta a este una distribución gamma con la misma media y varianza en una curva suavizada:
fishes<-read.table("c:\\estadistica\\fishes.txt",header=T)
attach(fishes)
names(fishes)
[1] "mass"

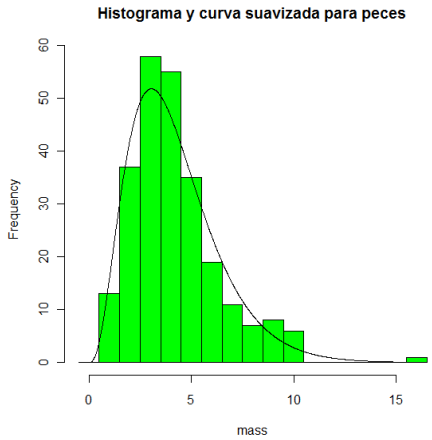
Primero calculamos los dos parámetros para la distribución gamma:
razon<-mean(mass)/var(mass)
forma<-razon*mean(mass)

razon
[1] 0.8775119
> forma
[1] 3.680526

Necesitamos conocer el mayor valor de las masas para hacer las cajuelas (*bin*) del histograma:
max(mass)
[1] 15.53216

Ahora se puede graficar el histograma, usando puntos de ruptura desde 0.5 para tener barras centradas hasta un máximo de 16.5 para acomodar nuestro mayor pez:
par(mfrow=c(1,1))
hist(mass,breaks=-0.5:16.5,col="green",main="")
hist(mass,breaks=-0.5:16.5,col="green",main="Histograma y curva suavizada para peces")

La función de densidad de la distribución gamma se sobrepone utilizando la función **lines** así:
lines(seq(0.01,15,0.01),length(mass)*dgamma(seq(0.01,15,0.01),forma,razon))



Un ajuste mucho mejor que el logrado con una normal.

La distribución exponencial.

Esta es una distribución monoparamétrica, caso especial de la distribución gamma. Es muy usada en análisis de supervivencia. También el generador de números aleatorios de la exponencial es útil en *simulaciones de Monte Carlo* para los tiempos de muerte cuando el riesgo instantáneo de muerte (*hazard*) es constante con la edad. Se especifica el *hazard*, como el recíproco de la edad media de muerte:

rexp(15,0.1)

```
 [1]  6.599385 34.554511  2.676634  9.655766 15.306653  8.139812  1.199781
 [8] 24.279948  5.168529  9.103039 17.026292  6.809087 10.190987  3.827263
[15] 14.582163
```

Estas son 15 tiempos de vida aleatorios con un valor esperado de 1/0.1 = 10 años; con una media de 13.14 años:

```
 mean(rexp(15,0.1))
[1] 13.14967
```

La distribución beta.

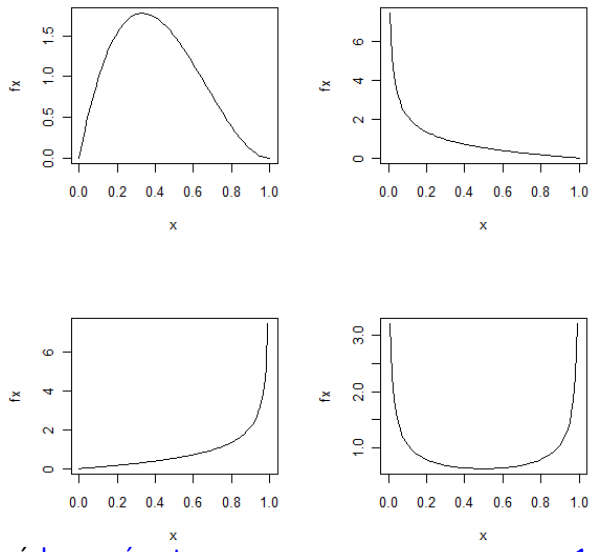
Es una función biparamétrica con dos constantes positivas, ***a*** y ***b***, y acotada ***para valores 0≤x≤1***:

$$f(x)=\frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}x^{a-1}(1-x)^{b-1}$$

En R se genera una familia de funciones de densidad como esta que se muestra para ver la versatilidad de formas encontradas, con:

```
par(mfrow=c(2,2))
x<-seq(0,1,0.01)
fx<-dbeta(x,2,3)
plot(x,fx,type="l")
fx<-dbeta(x,0.5,2)
plot(x,fx,type="l")
fx<-dbeta(x,2,0.5)
plot(x,fx,type="l")
fx<-dbeta(x,0.5,0.5)
```

plot(x,fx,type="l")



Es importante observar sí, los parámetros son mayores o menores que 1. Cuando ambos son >1 se obtiene una curva con un máximo, n formada que llega a ser más sesgada mientras b >a (izquierda arriba).

Sí ***0<a<1*** y ***b>1*** la densidad es negativa (derecha arriba).
Sí ***a>1*** y ***0<b<1*** la densidad es positiva (izquierda abajo).
Sí ***a*** y ***b***, ***ambos***, son fracciones positivas la función alcanza forma de U (derecha abajo).
Sí ***a=b=1***, obtenemos la distribución uniforme en [0,1].

Podemos generar números aleatorios de una beta con la función rbeta(n, forma1, forma2), así ejemplo:

rbeta(20,2,3)
[1] 0.06544228 0.44355812 0.44817866 0.22442084 0.38878443 0.46670716
[7] 0.39072074 0.28108451 0.44519441 0.28409038 0.52163138 0.34043733
[13] 0.56210529 0.49622609 0.58037121 0.33015193 0.31635142 0.64122922
[19] 0.26412883 0.44202023.

Distribución de Cauchy.

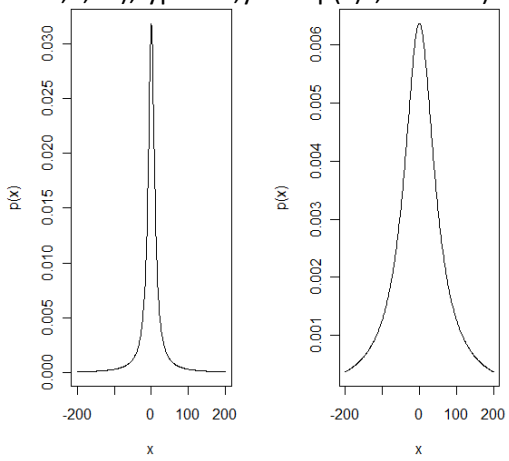
Es una función biparamétrica de largas colas caracterizada por un parámetro de localización ***a*** y un parámetro de escala ***b***. Es valorada en la escala de los reales, simétrica alrededor de *a* (el cual es además ***su mediana***), y como una curiosidad, debido a sus largas colas no tiene esperanza ni momentos. Sin embargo la media harmonica de una variable con densidad positiva desde cero se distribuye típicamente como una Cauchy, y esta también aparece en teorías de movimientos Brownianos (e.g. rutas aleatorias). La forma general de la distribución es:

$$f(x) = \frac{1}{\pi b(1+((x-a)/b^2)}$$

para $-\infty \leq x \leq +\infty$ a. Existe una versión mono paramétrica de Cauchy cuando ***a = 0*** y ***b = 1***, la cual se conoce como distribución estándar de Cauchy y equivale a una t de student con un grado de libertad:

$$f(x) = \frac{1}{\pi(1+x^2)}; \quad -\infty \le x \le +\infty$$

Por ejemplo sea:
par(mfrow=c(1,2))
plot(-200:200,dcauchy(-200:200,0,10),type="l",ylab="p(x)",xlab="x")
plot(-200:200,dcauchy(-200:200,0,50),type="l",ylab="p(x)",xlab="x")



Nótese la larga cola aplanada de la distribución. La parte izquierda con un parámetro de escala =10, la de la derecha con escala = 50 y ambas con parámetro de localización =0.

Distribución lognormal.

Toma solo valores en la recta real positiva del producto cartesiano. Sí se toma el logaritmo de una desviación lognormal, el resultado es una desviación normal, de ahí su nombre. Tiene muchísimas aplicaciones en tamaños de partículas agregadas, flujos de nutrientes, concentraciones de contaminantes del aire y, tiempos de falla. La ***función de riesgo*** (***hazard***) de la lognormal se incrementa para pequeños valores y es decreciente. Una mezcla de objetos u ítems heterogéneos que individualmente presentan riesgos monotónicos pueden generar tal función de riesgo. La función de densidad, distribución acumulativa, cuantiles y generación de números aleatorios con ella empleando la función **dlnorm** se obtiene en R con: dlnorm(x, meanlog=0, sdlog=1).

En ella la media y la desviación estándar son opcionales, pues por defecto presentan meanlog = 0 y sdlog =1. pero se debe notar que estos no son sus parámetros, pues la distribución lognormal tiene:

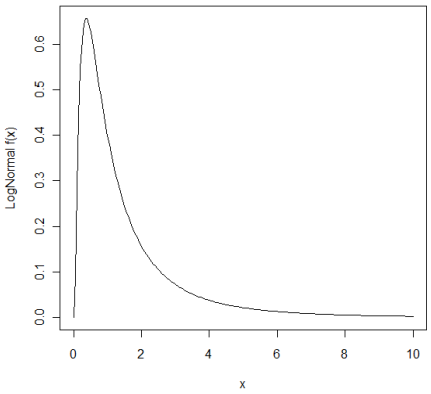
media: $\mu_{\log n} = e^{\mu+\sigma^2/2}$

varianza $\sigma^2_{\log n} = (e^{\sigma^2}-1)e^{2\mu+\sigma^2}$

sesgo: $sesgo_{\log n} = (e^{\sigma^2}+2)\sqrt{e^{\sigma^2}}-1$

y, kurtosis $kurtosis_{\log n} = e^{4\sigma^2}+2e^{3\sigma^2}-6$

Ejemplo: para graficar la lognormal se hace algo como esto:
par(mfrow=c(1,1))
plot(seq(0,10,0.05),dlnorm(seq(0,10,0.05)), type="l",xlab="x",ylab="LogNormal f(x)")



Se puede observar la cola derecha extremadamente larga y un exagerado sesgo positive como características dominantes de ella.

Distribución logística.

La logística es usada en modelos lineales generalizados con errores binomiales, *especialmente en descripción de datos con proporciones*. La función de distribución acumulativa de probabilidades es simétrica con forma de S acotada por debajo y por encima entre 0 y 1. Hay dos formas de escribir esta acumulativa:

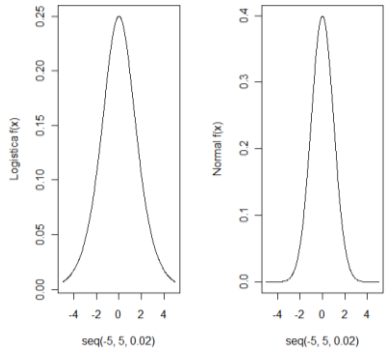
$$(1)\ p(x)=\frac{e^{a+bx}}{1+e^{a+bx}} \quad y \quad (2)\ p(x)=\frac{1}{1+\beta e^{-ax}}$$

La gran ventaja de (1) es la posibilidad de linealizarla con transformación logarítmica como:

$$\log\left(\frac{p}{q}\right)=a+bx$$

p, probabilidad de éxito y q probabilidad de fracaso (1-p). La logística es una distribución unimodal, simétrica y en la recta real con colas más largas que las presentadas por la normal. Es muy usada en modelación de curvas de crecimiento y en bioensayos. Una motivación para su uso al modelar el crecimiento es que f(x) tiene la propiedad que la derivada de f(x) con respecto a x es proporcional a [f(x)-A][B-f(x)] con A<B. La interpretación es que la rata de crecimiento es proporcional a la suma de lo ya crecido x la cantidad de crecimiento aun esperada. Se comparará una logística con una normal equivalente como si fueran z:

```
par(mfrow=c(1,2))
plot(seq(-5,5,0.02),dlogis(seq(-5,5,.02)), type="l",ylab="Logistica f(x)")
plot(seq(-5,5,0.02),dnorm(seq(-5,5,.02)), type="l",ylab="Normal f(x)")
```



Note las colas mucho más gruesas de la logística (aun para probabilidades a ±4 desviaciones estándar y, también las diferencias de escala de los dos ejes y (0.25 para la logística, 0.4 para la normal).

Distribución log-logística.

Esta es una función muy flexible tetraparamétrica, muy usada en procesos de crecimiento y declinación cuya *f(x)* se presenta como:

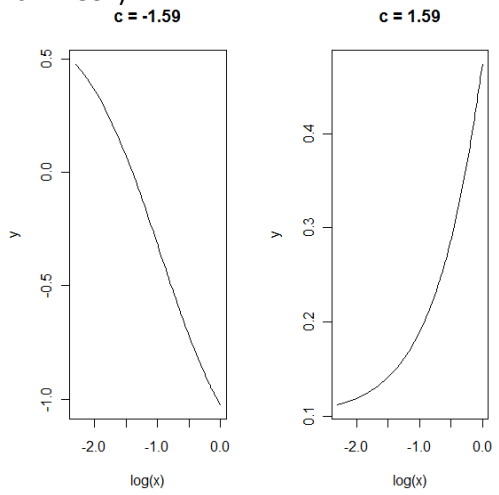
$$f(x) = a + b\left[\frac{\exp(c(\log(x) - d))}{1 + \exp(c(\log(x) - d))}\right]$$

Se mostrarán dos casos: El primero, una sigmoide negativa con ***c=–1.59*** and ***a=–1.4***:

```
x<-seq(0.1,1,0.01)
y<- -1.4+2.1*(exp(-1.59*log(x)-1.53)/(1+exp(-1.59*log(x)-1.53)))
plot(log(x),y,type="l", main="c = -1.59")
```

El Segundo: con ***c=1.59*** y ***a=0.1***:

```
y<-0.1+2.1*(exp(1.59*log(x)-1.53)/(1+exp(1.59*log(x)-1.53)))
plot(log(x),y,type="l",main="c = 1.59")
```



Distribución de Weibull.

El origen de esta distribución fue para análisis de conexiones o encadenamientos débiles. Si hay r eslabones en una cadena y las fuerzas de cada uno de ellos (Z_i) son independientemente distribuidos en $(0, \infty)$, entonces la distribución de los eslabones más débiles V = min(Z_j) se aproxima a una distribución de Weibull en la medida que se incrementa el número de eslabones.

Esta función es un modelo biparamétrico que tiene la distribución exponencial como un caso especial de ella. Es muy utilizada en estudios demográficos, distribuciones diamétricas en el campo forestal y análisis de supervivencia y la que permite que las tasas de mortalidad se incrementen o decrezcan con la edad. Así, hay tres tipos de curvas de supervivencia que pudieran analizarse con ella. Las funciones de densidad, supervivencia y de riesgo con $\lambda = \mu^{-\alpha}$ son:

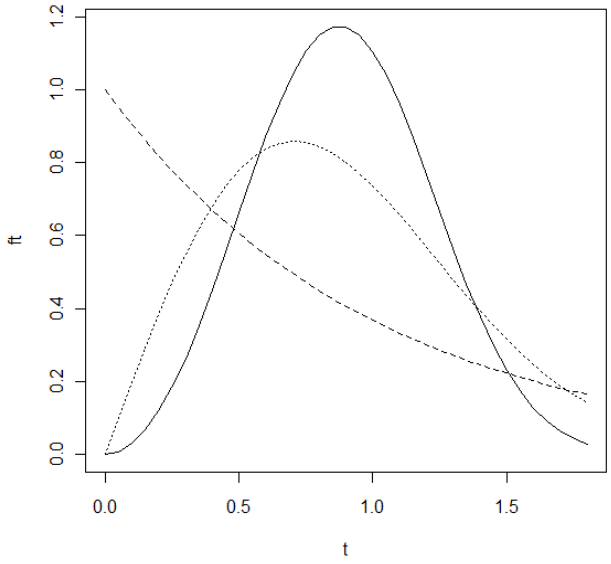
$$(1)\ (f(t)=\alpha\lambda t^{\alpha-1}e^{-\lambda t^{\alpha}};\quad (2)\ S(t)=e^{-\lambda t^{\alpha}};\quad (3)\ h(t)=\frac{f(t)}{S(t)}=\alpha\lambda t^{\alpha-1}$$

La media de una Weibull es $\Gamma(1+\alpha^{-1})\mu$, la varianza $\mu^2\left(\Gamma(1+2/\alpha)-(\Gamma(1+1/\alpha))^2\right)$ y, el parámetro α describe la forma de la función de riesgo. Para $\alpha=1$ (distribución exponencial) el riesgo es constante, mientras que para $\alpha>1$ el riesgo se incrementa con la edad y para $\alpha<1$el riesgo disminuye con ella.

Debido a que las funciones de Weibull, lognormal y log-logística tienen todas sesgos positivos, es difícil discriminar entre ellas con pequeñas muestras. Esto resulta relevante si tenemos en cuenta que cada una tiene diferentes formas de funciones del riesgo lo cual hace difícil además discriminar entre diferentes asunciones acerca de la edad específica para las tasas de muerte. En estudios de supervivencia la parsimonia requiere que ajustemos primero la exponencial antes que la Weibull, a menos que el parámetro de forma α resulte significativamente diferente de 1.

Se presenta una familia de 3 distribuciones Weibull con α = 1, 2, 3 (línea: punteada, a trazos y solida, respectivamente). Se nota que para grandes valores de α la distribución llega a hacerse simétrica, mientras que para α ≤1 la distribución tiene su moda en t = 0.

```
par(mfrow=c(1,1))
a<-3
l<-1
t<-seq(0,1.8,.05)
ft<-a*l*t^(a-1)*exp(-l*t^a)
plot(t,ft,type="l")
a<-1
ft<-a*l*t^(a-1)*exp(-l*t^a)
lines(t,ft,type="l",lty=2)
a<-2
ft<-a*l*t^(a-1)*exp(-l*t^a)
lines(t,ft,type="l",lty=3)
```



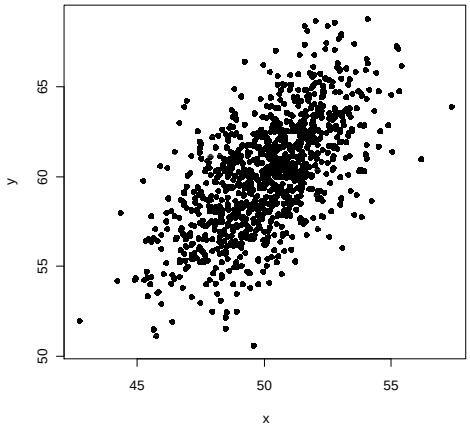
Distribución normal Multivariada.

Si se desea generar 2 (o más) vectores o números aleatorios normalmente distribuidos *correlacionados de alguna manera* uno a otro en un grado especificado debemos acudir a la función **mvrnorm** de la librería MASS:
library(MASS)

Supongamos que deseamos 2 vectores de 1000 números aleatorios cada uno. El primer vector con media 50 y el segundo con media 60. La diferencia con la función ya usada: rnorm es que necesitamos especificar tanto su covarianza como las desviaciones estándar de cada variable por separado. Esto es posible con una matriz definida positiva simétrica, la matriz de covarianzas. Por ejemplo:
xy<-mvrnorm(1000,mu=c(50,60),matrix(c(4,3.7,3.7,9),2))

Podemos chequear la cercanía de las varianzas a nuestros valores especificados:
var(xy)
var(xy)
[,1] [,2]
[1,] 4.059469 3.913759
[2,] 3.913759 9.846204

Bastante cercanas, la *covarianza debía ser 3.70* y nuestros *datos simulados la tienen de 3.91*. Podemos además extraer los dos vectores separados ***x*** y ***y*** y, graficarlos para ver la correlación:
x<-xy[,1]
y<-xy[,2]
plot(x,y,pch=16,ylab="y",xlab="x")



Es también importante ver las dos varianzas de x y y con más detalle:
var(x)
[1] 4.059469
var(y)
[1] 9.846204

Sí ambas muestras fueran independientes entonces la varianza de la suma de las variables sería la suma de las dos varianzas. Es este el caso de los datos anteriores?
var(x+y)

[1] 21.73319
var(x)+var(y)
[1] 13.90567

No lo es. La varianza de la suma (21.73) es mucho mayor que la suma de las varianzas (13.91). Esto sucede porque x y y son positivamente correlacionados; grandes valores de x tienden a asociarse con grandes valores de y y viceversa. Siendo esto así, podríamos esperar que la varianza de la diferencia entre x y y sea menor que la suma de las dos varianzas:
var(x-y)
[1] 6.078155
Podemos concluir que *la varianza de la suma de dos variables es únicamente igual a la varianza de la de la diferencia de las dos variables cuando ambas son independientes*. Qué pasa acerca de la covarianza de x y y? Encontramos esta aplicando la función **var** a la matriz xy (ya mostrada). Las diferencias encontradas se deben simplemente a la selección aleatoria de los puntos. La covarianza está relacionada a las varianzas separadas mediante el coeficiente de correlación, como sigue:

$$\mathrm{cov}(x, y) = \rho\sqrt{s_x^2 s_y^2}$$

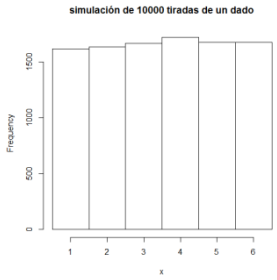
Para nuestro ejemplo, lo chequeamos como sigue, donde el valor muestral de ρ es cor(x,y):
cor(x,y)*sqrt(var(x)*var(y))
[1] 3.913759

Con
cor(x,y)
[1] 0.6190491.

Distribución uniforme.

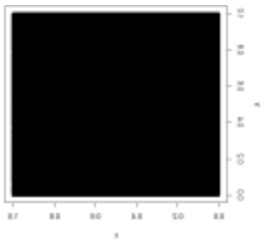
Esta es la que usualmente usan las calculadoras para *emular números aleatorios*. La idea es generar números entre 0 y 1 entre los cuales cualquier número real posible tenga la misma probabilidad de ser obtenido. Pero al pensar en ello algo falso ocurre. Los computadores producen números por medio de recetas. Sí se está siguiendo una receta en que el resultado es predecible, como podría ser aleatorio? Para muchos estadísticos cualquiera que use métodos aritméticos para producir números aleatorios comete una falta. Para verlo la pregunta es, ¿como exactamente una computadora genera el número aleatorio? La respuesta gira alrededor del encriptamiento (luego de un numero suficientemente grande de salidas sería posible predecir el resto de ellas). Acá estamos únicamente interesados en lo bien que los generadores de pseudo números aleatorios trabajan como el caso de R. Por ejemplo se presenta la salida de la función R runif simulando la tirada de un dado de 6 lados 10000 veces: el histograma debería ser plano:
x<-ceiling(runif(10000)*6)
table(x)
x
    1   2   3   4   5   6
1619 1636 1667 1724 1679 1675

hist(x,breaks=0.5:6.5,main="simulación de 10000 tiradas de un dado")



muy cercano al lo esperado, por lo cual se considera el generador de números aleatorios de R eficiente. Se puede intentar mapificar 1000000 de puntos para buscar claros con:

```
x<-runif(1000000)
y<-runif(1000000)
plot(x,y,pch=16)
```



Se produce este mapa absolutamente negro sin claros en el, por lo cual no hay pares de números que no fueran generados en esta escala de resolución (pch=16). Para una mas sesuda apreciación de ello podríamos contar la frecuencia de frecuencia de combinaciones de números: con 36 celdas, la frecuencia esperada será 1000000/36=27777.78: números por cada una. Usaremos la función **cut** para producir las 36 cajuelas del histograma:

table(cut(x,6),cut(y,6))

	(-0.001,0.166]	(0.166,0.333]	(0.333,0.5]	(0.5,0.667]
(-0.000999,0.166]	27556	27859	28035	27910
(0.166,0.333]	27767	27992	27692	27823
(0.333,0.5]	28011	27988	27599	27821
(0.5,0.667]	27939	27801	27915	27938
(0.667,0.834]	27461	28057	28171	27779
(0.834,1]	27419	27682	27590	27826

	(0.667,0.834]	(0.834,1]
(-0.000999,0.166]	27462	27392
(0.166,0.333]	27790	27761
(0.333,0.5]	27882	27736
(0.5,0.667]	27932	27709
(0.667,0.834]	27813	27602
(0.834,1]	27628	27662

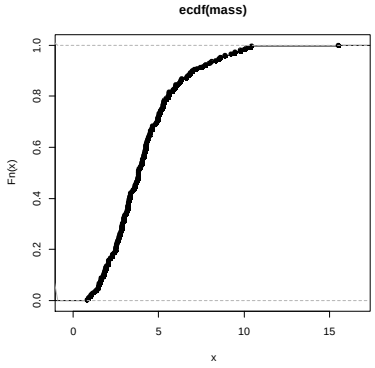
donde se ve lo cercano de lo obtenido con lo esperado.

Graficación de funciones de distribución empíricas acumulativas.

R usa la función **ecdf** para ello. Por ejemplo, con el archivo de peces se obtiene:

peces<-read.table("c:\\estadistica\\fishes.txt",header=T)
attach(peces)
names(peces)
[1] "mass"

plot(ecdf(mass))



El pronunciado sesgo positivo de los datos es evidente por el hecho que el lado izquierdo de la distribución es mucho más inclinado que el derecho.

Distribuciones discretas de probabilidad

Distribución Bernoulli.

Es la que subyace para pruebas de respuesta binaria (uno de dos valores: 1 con probabilidad ***p*** (**éxito**) y 0 con probabilidad ***1–p=q*** (**falla o fracaso**). Su función de densidad se da por

$$p^x(1-p)^{1-x}$$

con E(X)=p y var(X) = pq.

Distribución binomial.

Describe la *probabilidad de éxito en n ensayos independientes* de eventos Bernoulli *con probabilidad constante de intento a intento*. Su función de densidad es una monoparamétrica (p):

$$b(x,n,p)=\binom{n}{x}p^x(1-p)^{n-x}$$

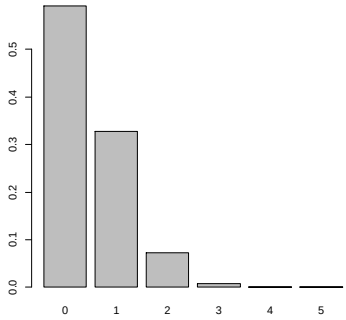
Con ella puede por ejemplo calcular lo improbable de ganar el baloto, aproximadamente 1/14000000 si usted es invitado a seleccionar 6 números entre 1 y 49. Se puede usar la función factorial incorporada en R:

factorial(49)/(factorial(6)*factorial(49-6))
[1] 13983816

También se puede calcular el número de combinaciones posibles con la función combinatoria ***choose(n,x)*** incorporada a R:
choose(49,6)
[1] 13983816

La media de la binomial es np y su varianza npq. Como 1−p < 1 es obvio que la varianza resulta menor que la media excepto para x=0. Es fácil también visualizar la distribución para valores particulares de n y p.

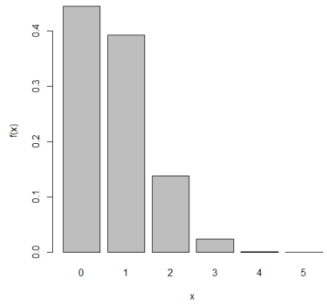
```
p<-0.1
n<-5
x<-0:n
px<-choose(n,x)*p^x*(1-p)^(n-x)
barplot(px,names=as.character(x))
```



Las 4 funciones de distribución incorporadas a R son las de densidad, acumulativa, probabilidad, de cuantiles y generación de números aleatorios) similares a esto:

La función de densidad muestra la probabilidad para valores específicos de conteos de *x* (ejemplo el número de peces parasitados en una muestra de tamaño n con probabilidad constante de éxito = prob. Por ejemplo si se capturaran 5 peces cuando la población está parasitada en un 15% obtenemos un grafico de densidad de probabilidades contra el número de peces parasitados, así:

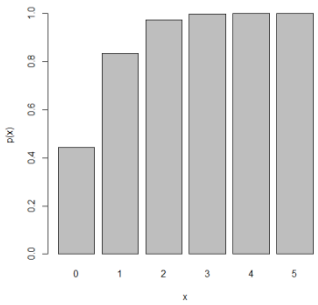
```
barplot(dbinom(0:5,5,0.15),names=as.character(0:5),xlab="x",ylab="f(x)")
```



El número de peces parasitados más probables es 0. Note que podemos generar la secuencia de valores x para graficarlos (0:5 en este caso) dentro de la función de densidad. La distribución acumulativa de probabilidades muestra que la suma de probabilidades, hasta e incluida p(x), grafica la probabilidad acumulada contra el numero de éxitos para una muestra de tamaño n y probabilidad = prob. Nuestros peces lucen así:

barplot(pbinom (0:5,5,0.15), names=as.character(0:5),xlab="x",ylab="p(x)")

Esta muestra que la probabilidad de tener 2 o menos de 2 peces parasitados en la muestra de 5, es muy cercana a 1.



La función *quantiles* solicita 'con una *probabilidad especificada p* (¿a menudo 0.025 y 0.975 para pruebas de dos colas al 95%), cual es el numero esperado de peces a obtener en una muestra de tamaño n y una probabilidad = prob?'. Para nuestro ejemplo los valores de dos colas, más bajo y alto al 95% de peces parasitados esperados serán:

qbinom(.025,5,0.15)
[1] 0
qbinom(.975,5,0.15)
[1] 3

Lo anterior significa que una certidumbre de 95% podríamos capturar entre 0 y 3 peces parasitados de 5 si repitiéramos el ejercicio de muestreo y además que resulta muy improbable lograr más de 3 capturas con la probabilidad dada.

Estos cálculos resultan importantes cuando estamos interesados en determinar si o no, el tamaño de muestra escogido (n=5 en este caso) es capaz de hacer bien nuestro trabajo. Supóngase que lo fundamental para nuestro muestreo sería averiguar si el parasito está presente o no en el lago. Basta encontrar un pez parasitado para responder que sí. Pero que tan probable resulta no obtener ninguno parasitado y concluir erróneamente que el parasito no está presente en el lago. Con esta pequeña muestra de n=5 and p=0.15 tenemos una probabilidad de no encontrar el parásito de 0.85 por cada pez capturado y una probabilidad de 0.85^5 =0.4437 conjunta de no encontrarlo. Esto obviamente es insatisfactorio, por lo cual debemos reflexionar en el tamaño de la muestra. Cuál será la muestra más pequeña, *n*, que haga la probabilidad de no encontrar el parásito menor o igual a 0.05? Para ello necesitamos resolver:

$$0.05 = 0.85^n$$

para lo cual tomamos logaritmos y despejamos n:

$$\log(0.05) = n\log(0.85) \therefore \quad n = \frac{\log(0.05)}{\log(0.85)}$$

log(0.05)/log(0.85)
[1] 18.43313

Lo cual significa que debemos repetir nuestro proceso hasta encontrar más de 18 peces sin parásitos antes de rechazar la hipótesis que el parasitismo está presente a una tasa de 15%. Desde luego a una tasa más baja se requiere un tamaño de muestra mayor.

Los **números aleatorios** serán generados de una binomial dándole una función similar a esta rbinom(#,n,p) que tenga como primer argumento el número deseado de números aleatorios, el segundo el tamaño muestral n y el tercer argumento la probabilidad de éxito, ejemplo:
rbinom(10,5,0.15)

[1] 2 0 0 1 0 0 0 2 1 0

Acá repetimos la muestra de 5 peces 10 veces. Obtendríamos 1 pescado parasitado de los 5 muestreados en dos ocasiones, 2, dos ocasiones, y 0 en las otras siete. Nunca capturaríamos 2 o más parasitados en esta muestra de tamaño 5.

Distribución geométrica.

Suponga, unas series de ensayos independientes Bernoulli con probabilidad p; corridas 1, 2, 3, veces… Sea W el tiempo de espera hasta que el primer suceso ocurra, tal que:

$$p(w>x)=(1-p)^x.$$

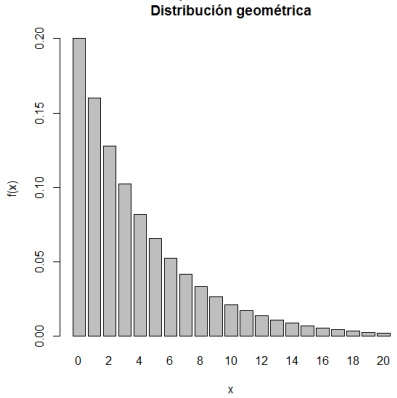
lo cual significa que

$$p(w=x)=p(w>x-1)-p(w>x)$$

Su función de densidad será

$$f(x)=p(1-p)^{x-1}$$

```
fx<-dgeom(0:20,0.2)
barplot(fx,names=as.character(0:20),xlab="x",ylab="f(x)",main="Distribución geométrica")
```



Los parámetros de esta distribución son: media y varianza respectivamente:

$$\mu=\frac{1-p}{p};\quad \sigma^2=\frac{1-p}{p^2}$$

Como observa tiene una larga cola a derecha. Acá se presenta una tabla de números aleatorios generados con ella, en la cual la moda es 4 y ha genrado valores tan grandes como 34 y 35:

```
table(rgeom(100,0.1)
0  1  2  3  4  5  6  7  8  9 10 12 13 14 15 16 17 19 20 21 22 23 27 28 29 33
9  8 10  8 11  4  2  6  5  8  1  4  2  3  1  3  2  2  1  1  2  1  1  1  1  1
34 35
 1  1
```

Distribución hipergeométrica.

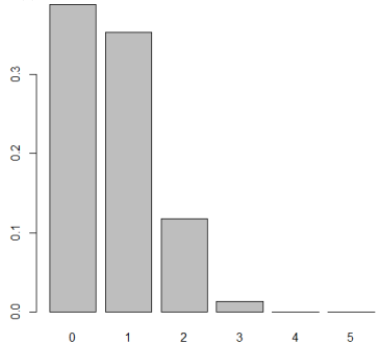
Esta es especialmente conocida con el nombre de ‘Bolas en urnas’ por ser estos casos típicos ejemplos para ella. Su función de densidad es

$$f(x) = \frac{\binom{b}{x}\binom{N-b}{n-x}}{\binom{N}{n}}$$

Ejemplo sea N un conjunto de bolas de diversos colores de las cuales b son azules y el resto de ellas de otros colores: r = N−b. Nos preguntaríamos al sacar *n*, bolas **sin reemplazamiento**, la probabilidad de obtener x bolas azules en las n extraídas. La función incorporada en R para ellos será:
dhyper(q, b,r,n),
rhyper(m, b,r,n).

En la las cuales:
• q es un vector de valores de la variable aleatoria x representando el numero de bolas azules de las n.
• b número de bolas azules en la urna (número real no negativo entero.
• r número de del resto de bolas no azules en la urna r= N – b (enteros no negativos)
• n muestra extraída de la urna.
• p un vector de probabilidades entre 0 y 1.
• m # de números aleatorios generados distribuidos hipergeometricamente.
Ejemplo sea una urna con N =20 bolas de las cuales 6 son azules y 14 de otros colores. Tomaremos una muestra de n=5 , por lo cual x podría ser 0, 1, 2, 3, 4 o 5 de ellas azules, como la proporción de azules es apenas de 6/20, altas frecuencias serán muy improbables. Para evaluar nuestro ejemplo procedemos como sigue:

```
ph<-numeric(6)
for(i in 0:5) ph[i]<-dhyper(i,6,14,5)
barplot(ph,names=as.character(0:5))
```



Resulta muy improbable lograr mas de dos bolas azules de las 5. Un simulacro de conjuntos de Monte Carlo con ensayos de tamaño 5, muestran en 20 realizaciones de muestreo ejemplo:

```
rhyper(20,6,14,5)
[1] 1 1 1 1 2 1 2 2 1 2 3 2 1 1 1 2 3 1 0 2
```

Podrá apreciar que la distribución binomial es un caso limite de la hipergeometrica, lo cual surge cuándo N, b y r se aproximan al infinito de modo que b/N se aproxima a p, y r/N a 1 – p. esto por cuanto con grandes números el muestreo con o sin reemplazamiento se comportan muy similarmente. Por lo anterior una distribución binomial asume muestreos con reemplazamiento en una población finita, o muestreos sin reemplazamiento en una infinita.

Distribución multinomial.

Semejante a la binomial pero con más de dos estados discretos para la variable multinomial, por ejemplo ***t*** posibles resultados en un ensayo experimental cada uno con su probabilidad p_i. Sí tomamos ***n*** ensayos independientes en que $n=n_1+n_2+...+n_t$, y nos preguntamos la probabilidad de obtener el vector de N_i ocurrencias del iésimo suceso:

$$p(N_i = n_i) = \frac{n!}{n_1!n_2!n_3!\cdots n_t!} p_1^{n_1} p_2^{n_2} p_3^{n_3} \cdots p_t^{n_t}$$

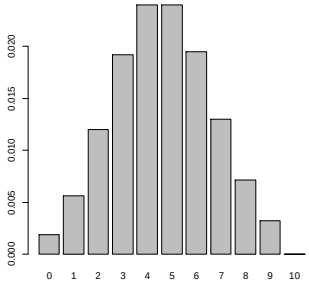
para i entre 1 a t. Por ejemplo, sean tres resultados como, bueno, regular malo para calificar un evento, el primero con el doble de probabilidad de los otros dos (p_1=0.5, p_2=0.25 y p_3=0.25). Si hacemos 4 ensayos con n_1=6, n_2=5, n_3=7 y n_4=6, en total n=24. Evaluamos la fórmula para i=1, 2 and 3 (por ser tres resultados). Arrancaríamos por escribir una función llamada **multi** para llevar a cabo los cálculos del número de éxitos a, b y c, para los tres sucesos dadas sus probabilidades 0.5, 0.25 y 0.25 así:

```
multi<-function(a,b,c) {
factorial(a+b+c)/(factorial(a)*factorial(b)*factorial(c))*.5^a*.25^b*.25^c}
```

a

Ahora pongamos esta función en un bucle (loop) para obtener la probabilidad de tener los patrones de éxitos requeridos, **psuc** para los 3 sucesos. Ilustraremos justamente un caso, en el cual el tercer resultado sea fijado para lograr 4 éxitos, con base en el ejemplo anterior en que n =24. Lo anterior significa que el primero y segundo casos variarían paso a paso entre 19 y 1 y 9 y 11 éxitos respectivamente ya que n=24 (24-4), me deja 20 para los dos primeros sucesos, pero además los dos primeros no deberían sobrepasar 11 (6+5), por lo cual haremos un vector de 0 con 11 espacios.

```
psuc<-numeric(11)
for (i in 0:10) psuc[i]<-multi(19-i,1+i,4)
barplot(psuc,names=as.character(0:10))
```



El resultado mas notable acá es que podríamos tener 19−4=15 o 19−5=14 éxitos de tipo 1 en un ensayo de tamaño 24 con las probabilidades anotadas: 0.5, 0.25 y 0.25 cuando el numero de éxitos del tercer caso se deja en 4 de 24. Estas funciones y resultados las podrá modificar para otras probabilidades y números de éxitos.

Distribución de Poisson.

Muy importante sobre todo en conteos de datos. Sabemos muchas veces que algo ha ocurrido (ej: coces de caballos, estallidos de bombas, etc.), pero no sabemos cuántas veces no ocurren. La distribución de Poisson es una uniparamétrica con la interesante propiedad de que coinciden media y varianza. La función de densidad de Poisson muestra la probabilidad de obtener un conteo de x cuando el conteo promedio por unidad es λ, parámetro de la distribución:

$$p(x) = \frac{e^{-\lambda}\lambda^{x}}{x!}$$

La probabilidad de obtener un conteo de 0 se obtiene con x=0

$$p(0) = \frac{e^{-\lambda}\lambda^{0}}{0!} = e^{-\lambda}$$

o sea, el antilogaritmo de la media por (-1), así podremos obtener, dado p(0), que p(1) es justamente

$$p(1) = \frac{e^{-\lambda}\lambda^{1}}{1!} = e^{-\lambda}\lambda = p(0)\lambda$$

cualquier probabilidad subsiguiente es realmente obtenido multiplicando la probabilidad precedente por la media y dividiendo por el conteo, así:

$$p(x) = p(x-1)\frac{\lambda}{x}$$

En R, las funciones de densidad, distribución acumulativa, cuantiles y genrador de números aleatorios son obtenidos con:
dpois(x, lambda)
ppois(q, lambda)
qpois(p, lambda)
rpois(n, λ)

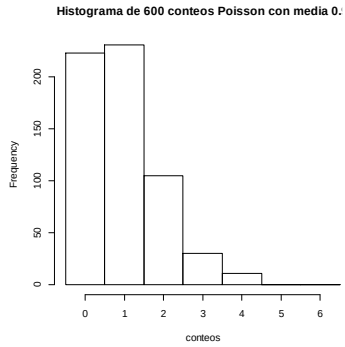
En que λ es la media de conteos por muestra. Esta distribución desempeña un papel central en 3 aspectos bastante separados del área estadística:
• En la descripción de patrones espaciales de puntos.
• Como la distribución de frecuencias de conteos de eventos raros pero independientes.
• Como distribuciones del error en modelo GLM para conteos de datos.

Sí deseamos 600 conteos simulados de una distribución de Poisson con media λ=0.90 células por portaobjetos, entonces escribimos:

conteos<-rpois(600,0.9)
y para ver la tabla de frecuencias usamos la función **table** de cada conteo generado así:

table(conteos)
```
  0   1   2   3   4
223 231 105  30  11
```

O, el histograma con la función **hist** para verlo:
hist(conteos,breaks = - 0.5:6.5,main="Histograma de 600 conteos Poisson con media 0.9")



Nótese el uso de puntos de quiebra para el vector en incrementos enteros desde −0.5 para crear las cajuelas (bin) del histograma de barras.

Distribución binomial negativa.

Esta distribución discreta, biparamétrica es útil en la descripción de conteos cuando la varianza resulta mucho mayor que la media. Los dos parámetros de ella son: la media ***μ*** y el parámetro de agregación (clumping) ***k***, dados por

$$k = \frac{\mu}{\sigma^2 - \mu}$$

Mientras mas pequeño el valor de k, mayor el grado de agregación de los datos. La función de densidad es:

$$p(x) = (1+\frac{\mu}{k})^{-k} \frac{(k+x-1)!}{x!(k-1)!}\left(\frac{\mu}{\mu+k}\right)^x$$

El término cero de esta distribución hacienda x=0 y simplificando da:

$$p(0) = (1+\frac{\mu}{k})^{-k}$$

Los términos sucesivos en la distribución pueden calcularse iterativamente mediante:

$$p(x) = p(x-1)\left(\frac{k+x-1}{x}\right)\left(\frac{\mu}{\mu+k}\right)$$

Un estimado inicial del valor de k puede obtenerse como:

$$k \approx \frac{\overline{x}^2}{s^2 - \overline{x}}$$

Cómo k no puede ser negativo, la distribución binomial no debería ser ajustada a datos en que la varianza resultara menor que la media. El estimado máximo verosímil de k se puede hallar numéricamente iterando progresivamente para cada vez mas finos valores ajustados de k hasta que las ramas derecha e izquierda de la siguiente ecuación sean iguales:

$$n\ln\left(1+\frac{\mu}{k}\right) = \sum_{x=0}^{\max}\left(\frac{A(x)}{k+x}\right)$$

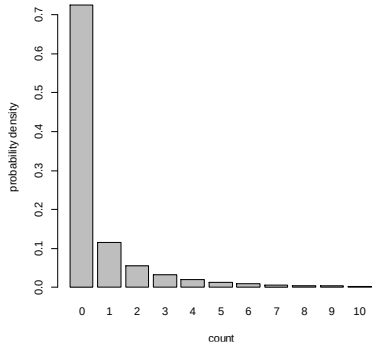
en la cual el A(x) contiene la frecuencia total de valores mayores que x. Entonces se podría escribir una función para obtener la densidad de probabilidad similar a esta:

negbin<-function(x,u,k) (1+u/k)^(-k)*(u/(u+k))^x*gamma(k+x)/(factorial(x)*gamma(k))

Luego usar la función para producir un gráfico de barras de las densidades de probabilidad para un rango de valores de x (dígase 0 a 10, para una distribución con una media y parámetro de agregación especificados, por ejemplo como μ = 0.8, k=0.2) similar a esto:

xf<-sapply(0:10, function(i) negbin(i,0.8,0.2))

barplot(xf,names=as.character(0:10),xlab="count",ylab="probability density")



Forma alternativa para la binomial negativa.

Hay otra forma de verla, con la variable respuesta como el tiempo de espera Wr para el éxito r-ésimo:

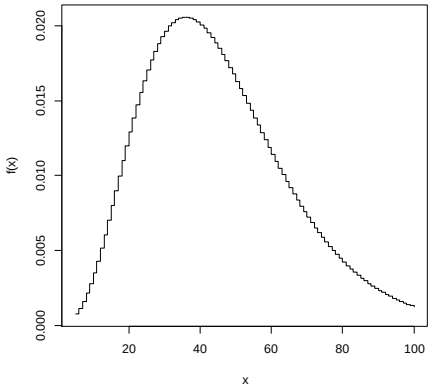
$$f(x) = \binom{x-1}{r-1} p^r (1-p)^{x-r}$$

Es importante comprender que x inicia en r e incrementa desde acá (obviamente, el r-ésimo éxito no puede ocurrir antes del r-ésimo intento). La función **dnbinom** representa el número de fallas x (ejemplo, sellos al tirar una moneda) antes de que el tamaño de éxitos suceda (o caras en la moneda) sean obtenidos cuando la probabiidad de éxito (o de cara) es prob:

dnbinom(x, size, prob)

Supongamos que estamos interesados en la distribución de tiempos de espera hasta que se presente el 5 éxito en un proceso binomial negativo con p=0.1. Iniciamos la secuencia de valores de x hasta 5:

plot(5:100,dnbinom(5:100,5,0.1),type="s",xlab="x",ylab="f(x)")



esta muestra que el tiempo de espera mas probable para el 5 éxito, con probabilidad de éxito 1/10, esta cercano a 31 intentos. Note que la distribución binomial negative es bastante sesgada a la derecha.

El generador de datos binomiales negativos usa la función de R:

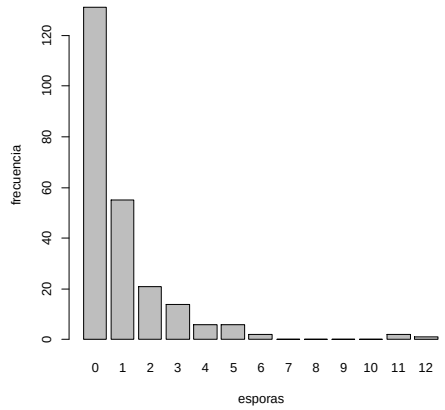
rnbinom(n, size, prob)

Con n, numero requerido de datos aleatorios, el segundo parametro (***size***), es ajustado a 1 cuando la distribución llega ser la geometrica. El parámetro final (***prob***), es la probabilidad de éxito por ensayo, p. Ejemplo vamos a generar 100 conteos BN con una media de 0.6:
contbn<-rnbinom(100,1,0.6)

Usamos luego la función ***table*** par aver las distribuciones de frecuencia de los diferentes conteos:
table(contbn)
contbn
 0 1 2 3 4 5
62 23 9 4 1 1
Es importante revisar que la media sea 0.6 o cercana a ella por medio de la función mean(contbn):
mean(contbn)
[1] 0.62

La varianza deberá resultar sustancialmente mayor que la media:
var(contbn)
[1] 0.9854545
y, esto da un estimado de k de:
0.62^2/(0.9854545-0.62)
[1] 1.051841

Los siguientes datos muestran el número de esporas contadas en 238 portaobjetos de cristal esmerilado. Estamos interesados en saber si estos datos quedan bien descritos por la distribución binomial negativa. Si lo fueran, podríamos encontrar el estimado máximo verosímil del parámetro de agregación k.
x<-0:12
freq<-c(131,55,21,14,6,6,2,0,0,0,0,2,1)
barplot(freq,names=as.character(x),ylab="frecuencia",xlab="esporas")



Empezamos por calcular las relaciones media varianza de los conteos. Pero, no podemos usar la media y varianza directamente ya que nuestros datos son frecuencias de conteos mas que conteos por si mismos. Esto es fácil de rectificar: usaremos la function rep para crear un vector de conteos y en el cual cada conteo es repetido el numero relevante de veces (freq). Ahora podemos usar directamente media y varianza:
y<-rep(x,freq)

mean(y)
y<-rep(x,freq)
> mean(y)
[1] 1.004202
var(y)
[1] 3.075932

Lo anterior muestra que se trata de datos altamente agregados (la relación varianza media es burdamente 3, y si fueran distribución Poisson sería de 1). Nuestro estimado aproximado de k será:
mean(y)^2/(var(y)-mean(y))
[1] 0.4867531

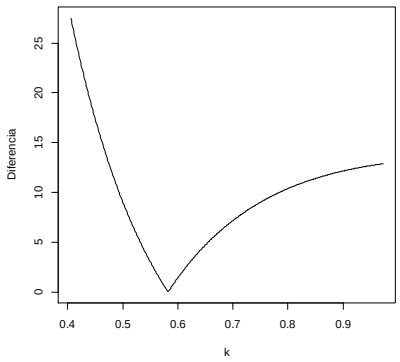
Aca se propone una funcion que toma un vector de frecuencias de conteos de x (entre 0 y length(x)−1) y calcula los estimados máximo verósimiles del parámetro de agregación , k:

```
kfit <-function(x)
{
lhs<-numeric()
rhs<-numeric()
y <-0:(length(x) - 1)
j<-0:(length(x)-2)
m <-sum(x * y)/(sum(x))
s2 <-(sum(x * y^2) - sum(x * y)^2/sum(x))/(sum(x)- 1)
k1 <-m^2/(s2 - m)
a<-numeric(length(x)-1)
for(i in 1:(length(x) - 1)) a[i] <-sum(x [- c(1:i)])
i<-0
for (k in seq(k1/1.2,2*k1,0.001)) {
i<-i+1
lhs[i] <-sum(x) * log(1 + m/k)
rhs[i] <-sum(a/(k + j))
}
k<-seq(k1/1.2,2*k1,0.001)
plot(k, abs(lhs-rhs),xlab="k",ylab="Diferencia",type="l")
d<-min(abs(lhs-rhs))
sdd<-which(abs(lhs-rhs)==d)
k[sdd]
}
```

Ahora la usaremos con nuestros datos de conteos de esporas:

```
kfit(freq)
```

[1] 0.5826276



La mínima diferencia es cercana a 0 y ocurre alrededor de k=0.55. La salida muestra que el estimado máximo verosímil de k es 0.582 (con 3 decimales, simulado; las ultimas 4 cifras decimales (6276) son irrelevantes y no deberíamos ajustar una función más perfecta).

¿Cómo esta distribución binomial negativa, con una media de 1.0042 y una k de 0.583, describe nuestros datos de conteos? Para ello calculamos las frecuencias esperadas multiplicando la respectiva probabilidad por la muestra total (238portaobjetos en este caso).

nb<-238*(1+1.0042/0.582)^(-0.582)*factorial(.582+(0:12)-1)/
(factorial(0:12)*factorial(0.582-1))*(1.0042/(1.0042+0.582))^(0:12)

Existen tres pasos para ello:

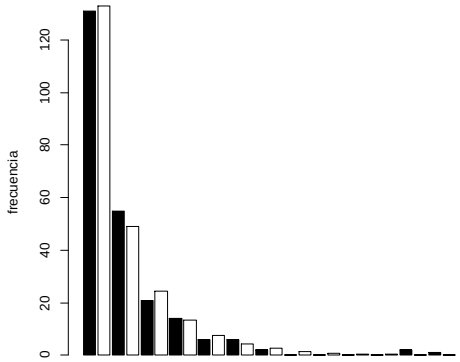
- Concatenar las frecuencias observadas y esperadas en una secuencia alternante.
- Crear una lista de etiquetas para denominar las barras (alternando blancos y conteos).
- Producir una leyenda para describir los diferentes colores de las barras.

La lista concatenada de frecuencias (llamada ***both***) se construye similar a esto, disponiendo los 13 conteos observados (***freq***) en las barras numeradas impares y los 13 conteos esperados (***nb***) en las pares (note el uso de la función modulo %% para ello):

both<-numeric(26)
both[1:26 %% 2 != 0]<-freq
both[1:26 %% 2 == 0]<-nb

En seguida se puede dibujar el grafico de barras combinado:

barplot(both,col=rep(c(1,0),13),ylab="frecuencia")



Debido a que dos barras adyacentes se refieren al mismo conteo (frecuencias observadas y esperadas), no deseamos usar los nombres de los argumentos incorporados a la función barplot para etiquetar las barras (si deseáramos una etiqueta para cada barra, 26 en total). En lugar de lo

anterior, deseamos escribir el conteo justamente para cada par de barras localizado entre el observado y el esperado, usando **as.character(0:12)**. Esto es trabajo para la función **mtext** la cual escribe texto en los márgenes de la gráfica. Necesitamos especificar el margen en el cual deseamos escribir. El margen en la base en side = 1. El único truco llamativo de esto es el desarrollo de las coordenadas x para las 13 etiquetas a lo largo del eje. Para ver como ha sido escalado el eje x por la función **barplot**, llevamos la función **barplot** (como antes) a un vector name, e inspeccionamos su contenido:

xscale<-barplot(both,col=rep(c(1,0),13),ylab="frecuencia")
as.vector(xscale)

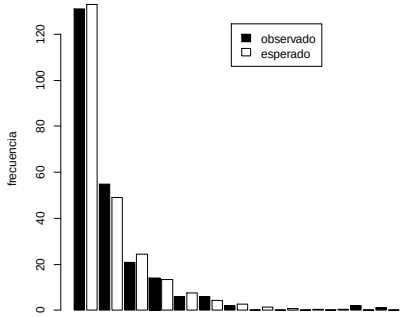
Acá usted podrá ver que el lado izquierdo de la primera barra esta en x=0.7 y el 26 iésimo en x =30.7. Un pequeño experimento nos mostrará que si deseamos colocar la primera etiqueta (label) en x =1.4 y luego a intervalos de 2.4 (2 barras son separadas por, por ejemplo, 11.5 – 9.1=2.4). Nosotros especificamos la secuencia **seq(1.4,30.2,2.4)** como el argumento de **at** dentro de **mtext**:

mtext(as.character(0:12),side=1,at=seq(1.4,30.2,2.4))

El argumento usado por defecto acá para **mtext** es que escriba las etiquetas en la línea número 0 alejadas del lado en cuestión: sí usted desea cambiar esto, adicione el argumento **line=1** a **mtext**. La función leyenda (**legend**) crea una leyenda para mostrar cuales barras corresponden a las frecuencias observadas (negro en este caso) y cuales representan las esperadas frecuencias binomiales negativas (barras abiertas). Luego, de justamente ***click*** cuando el cursor este en la posición donde usted decida ubicar la caja de leyendas en la esquina superior izquierda:

legend(locator(1),c("observado","esperado"),fill=c("negro","blanco"))

El ajuste es bastante cercano por lo cual pensamos que es razonablemente confiable que los conteos dados sean descritos por una distribución binomial negativa. La cola de la distribución observada es menudo mas aplanada que la cola esperada por lo cual le aplicamos una prueba de bondad de ajuste usando la chi cuadrada de Pearson, tomando la precaución de que los casos deben ser aquellos esperados en las frecuencias nb mayores o iguales a 5:



Esta se basa entonces en 5 comparaciones legítimas, que contamos con:

sum(nb>5)
[1] 5

Acá aparecen 5−p−1=2 g.l. ya que estimamos p=2 parametros para los datos al estimar la distribución esperada (la media y k) y se pierde un grado de libertad por contingencia (el número total de conteos debe adicinarse hasta 238). Nuestro valor calculado de la chi cuadrada =1.63 resulta mucho menor que el tabulado:

qchisq(0.95,2)
[1] 5.991465

Por lo cual no rechazamos la Ho y nuestros datos no resultan significativamente diferentes de una Distribución binomial negativa con ***media =1.0042 y k=0.582***.

La suma estadística de rangos de Wilcoxon.

Esta función calcula la distribución (también conocida como de **Mann–Whitney**), y retorna los valores para la probabilidad exacta de los valores discretos de: **dwilcox(q, m, n).** Acá, ***q*** es un vector de cuantiles, ***m*** es el número de observaciones en una muestra ***x*** (un entero positivo no mayor a 50), y ***n*** es el número de observaciones en otra muestra ***y*** (también un entero positivo no mayor a 50). La prueba de rangos de Wilcoxon es la suma de los rangos de of x **en la muestra combinada c(x,y)**. La suma estadística de rangos de Wilcoxon toma valores para W entre los límites:

$$\frac{m(m+1)}{2} \leq W \leq \frac{m(m+2m+1)}{2}$$

usada como prueba no no-parametrica de localización transferida entre las poblaciones padres de ***x*** y ***y***. Ejemplo sean números de esporas en dos medios diferentes para ver ese efecto. R tiene una prueba incorporada **wilcox.test**(x, y, alternative = "g")

Prueba de dos colas.
Numero de esporas en un medio 1
Numero de esporas en un medio 2.
La hipótesis alternativa de interés es si es mayor el número de esporas en el medio 1 que en el medio 2
esporas1<-c(131,55,21,14,6,6,2,0,0,0,0,2,1)
esporas2<-c(21,14,6,6,2,0, 133,554,0,1,2,1)

wilcox.test(esporas1, esporas2, alternative = "g") # greater

Wilcoxon rank sum test with continuity correction

data: esporas1 and esporas2
W = 69, p-value = 0.6995
alternative hypothesis: true location shift is greater than 0

Mensajes de aviso perdidos
In wilcox.test.default(esporas1, esporas2, alternative = "g") :
cannot compute exact p-value with ties