Tutorial 06:
Muestreo e intervalos de confianza

Atención:

- Este documento pdf lleva adjuntos algunos de los ficheros de datos necesarios. Y está pensado para trabajar con él directamente en tu ordenador. Al usarlo en la pantalla, si es necesario, puedes aumentar alguna de las figuras para ver los detalles. Antes de imprimirlo, piensa si es necesario. Los árboles y nosotros te lo agradeceremos.
- Fecha: 19 de abril de 2017. Si este fichero tiene más de un año, puede resultar obsoleto. Busca si existe una versión más reciente.

Índice

1. La distribución muestral de la media con R.

La Sección 6.1 del libro contiene el Ejemplo 6.1.1, en el que se discute la distribución muestral de la media para la variable aleatoria

$$X(a, b) = a + b,$$

que representa la suma de puntos obtenidos al lanzar dos dados. En ese ejemplo consideramos muestras aleatorias (con reemplazamiento) de tamaño

$$n = 3.$$

Nuestro primer paso, en ese ejemplo, es el cálculo del número de esas muestras, que resulta ser 46656. En este tutorial vamos a empezar construyendo la lista completa de esas 46656 muestras, para estudiar la distribución de la media muestral a partir de ellas.

Para hacer esto, vamos a utilizar la misma estrategia que se usa en el libro, en el que cada uno de los 36 resultados (equiprobables) distintos que se pueden obtener al lanzar dos dados se representa con un número del 1 al 36. En la Sección 6.2 del Tutorial03 usamos la librería `gtools` de R para obtener todas las tiradas posibles con este código:

```
   library(gtools)
   (dosDados = permutations(6, 2, repeats.allowed=TRUE))
```

```
##       [,1] [,2]
##  [1,]    1    1
##  [2,]    1    2
##  [3,]    1    3
##  [4,]    1    4
##  [5,]    1    5
##  [6,]    1    6
##  [7,]    2    1
##  [8,]    2    2
##  [9,]    2    3
## [10,]    2    4
## [11,]    2    5
## [12,]    2    6
## [13,]    3    1
## [14,]    3    2
## [15,]    3    3
## [16,]    3    4
## [17,]    3    5
## [18,]    3    6
## [19,]    4    1
## [20,]    4    2
## [21,]    4    3
## [22,]    4    4
## [23,]    4    5
## [24,]    4    6
## [25,]    5    1
## [26,]    5    2
## [27,]    5    3
## [28,]    5    4
## [29,]    5    5
## [30,]    5    6
## [31,]    6    1
## [32,]    6    2
## [33,]    6    3
## [34,]    6    4
## [35,]    6    5
## [36,]    6    6
```

El resultado es una matriz de 36 filas y tres columnas, en la que cada fila representa un resultado posible al lanzar los dos dados. Una muestra de tamaño 3 se obtiene al elegir tres filas de esta matriz. Por ejemplo, si elegimos (como en el libro) las filas 2, 15 y 23:

```
muestraFilas = c(2, 15, 23)
```

entonces los resultados correspondientes de los dados son:

```
(muestra = dosDados[muestraFilas, ])
```

```
##      [,1] [,2]
## [1,]    1    2
## [2,]    3    3
## [3,]    4    5
```

Para calcular el valor de X en cada una de las tres tiradas que forman la muestra hacemos:

```
(XenMuestra = rowSums(muestra))
```

```
## [1] 3 6 9
```

Y, finalmente, la media muestral de esa muestra es:

```
(mediaMuestral = mean(XenMuestra))

## [1] 6
```

Hemos calculado la media muestral en una muestra concreta. Pero lo que queremos hacer es repetir este proceso para todas y cada una de las 46656 muestras posibles. Lo primero que vamos a hacer es construir la lista completa de las muestras. Eso significa saber cuáles son las tres filas de la matriz `dosDados` que se han elegido en cada muestra concreta. Podemos obtener la lista de muestras usando de nuevo la librería `gtools`:

```
n = 3
Muestras = permutations(36, n, repeats.allowed=TRUE)
```

El comienzo y final de la matriz `Muestras` son:

```
head(Muestras)

##      [,1] [,2] [,3]
## [1,]    1    1    1
## [2,]    1    1    2
## [3,]    1    1    3
## [4,]    1    1    4
## [5,]    1    1    5
## [6,]    1    1    6

tail(Muestras)

##           [,1] [,2] [,3]
## [46651,]   36   36   31
## [46652,]   36   36   32
## [46653,]   36   36   33
## [46654,]   36   36   34
## [46655,]   36   36   35
## [46656,]   36   36   36
```

Fíjate en que, como hemos dicho, se trata de muestras con reemplazamiento. Por ejemplo, la primera muestra de la lista es aquella en la que hemos elegido las tres veces la primera fila de `dosDados`. Es decir, volviendo a los dados originales, que hemos elegido una muestra de tamaño 3, pero las tres veces hemos elegido la tirada $(1, 1)$ de los dados.

Vamos a comprobar que el número de muestras que hemos construido coincide con el cálculo teórico que aparece en el libro (es decir, 46656):

```
dim(Muestras)

## [1] 46656     3

(numMuestras = dim(Muestras)[1])

## [1] 46656
```

Hemos guardado el número de muestras en `numMuestras` para utilizarlo más adelante. Por otra parte, hay que tener en cuenta que, puesto que se trata de muestras con reemplazamiento, podemos elegir la misma fila varias veces. Por eso hemos incluido la opción `repeats.allowed=TRUE`. La función `dim` nos indica que, como esperábamos, la matriz que contiene la lista de muestras tiene 46656 filas y 3 columnas, que contienen los números de fila de `dosDados` que se usan en cada muestra. Para seguir con el ejemplo del libro, la muestra `muestraFilas = c(2, 15, 23)` que hemos usado antes aparece en la fila número 1823:

```
Muestras[1823, ]

## [1]  2 15 23
```

Recuerda que $2, 15, 23$ corresponde a las tiradas $(1, 2), (3, 3)$ y $(4, 5)$ respectivamente de los dos dados.

Ahora, una vez que tenemos la lista completa de muestras, tenemos que calcular la media muestral para cada una de ellas. El resultado será un vector `mediasMuestrales` de R que contendrá 46656 medias muestrales. Para construir ese vector:

- Usaremos un bucle `for` (ver Tutorial05) para recorrer una a una las filas de `Muestras`.
- Para cada fila (esto es, para cada muestra) calcularemos tres valores de X y, a partir de ellos, la media muestral $\bar{X}$, como hemos hecho en el ejemplo más arriba.
- Y almacenaremos esa media en el vector `mediasMuestrales`.

El codigo correspondiente es este (lee los comentarios):

```
mediasMuestrales = numeric(numMuestras)
for(i in 1:numMuestras){
    ## Identificamos las 3 filas que se han elegido en esta muestra.
    muestraFilas = Muestras[i, ]

    ## Recuperamos los resultados de las 3 tiradas de los dos dados.
    muestra = dosDados[muestraFilas, ]

    ## Aqui calculamos 3 valores de X
    XenMuestra = rowSums(muestra)

    ## A partir de ellos calculamos una media muestral.
    mediaMuestral = mean(XenMuestra)

    ## Y la guardamos en el vector de medias muestrales.
    mediasMuestrales[i] = mediaMuestral
}
```

¡Y ya está! Ya tenemos las 46656 medias muestrales. El comienzo del vector es:

```
head(mediasMuestrales)

## [1] 2.0000 2.3333 2.6667 3.0000 3.3333 3.6667
```

Y para la muestra que hemos usado de ejemplo:

```
mediasMuestrales[1823]

## [1] 6
```

como ya sabíamos.

Ejercicio 1. *Esfuérzate en entender de dónde provienen los primeros valores de la media muestral. Es importante que entiendas el mecanismo de construcción de las muestras, para que el concepto de media muestral quede claro.* □

1.1. Distribución de la media muestral $\bar{X}$ y de la variable original X.

Ahora que tenemos todas las medias muestrales, podemos estudiar cómo se distribuyen. Es decir, tenemos 46656 valores de $\bar{X}$ pero, desde luego, no son todos distintos. Hay muchas muestras

distintas de tamaño 3 que producen el mismo valor de $\bar{X}$. Así que lo que tenemos que hacer es, esencialmente, obtener la tabla de frecuencias de los valores de $\bar{X}$ (Tabla 6.2 del libro, ver pág. 200), y representar gráficamente el resultado (Figura 6.3(b) del libro, pág. 202).

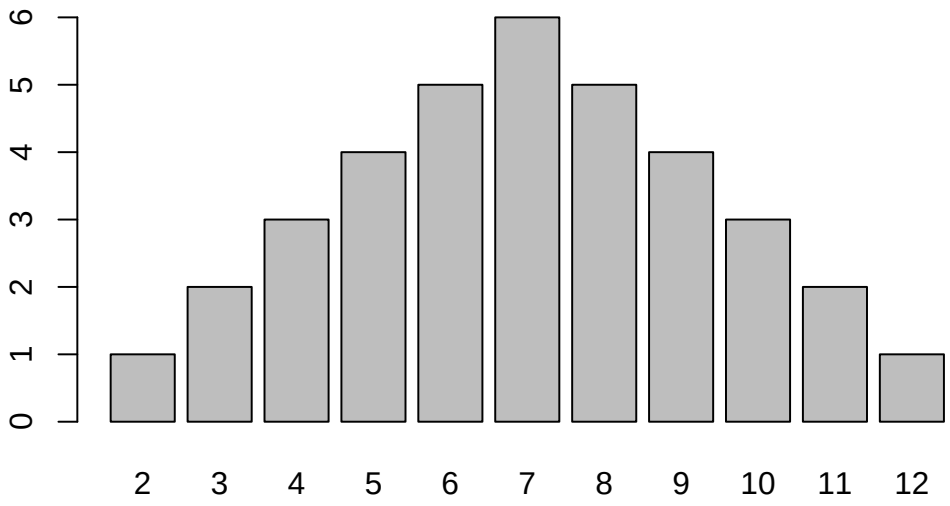
Pero antes vamos a detenernos a estudiar la distribución de la variable original X, para poder compararlas después. La tabla de frecuencias de X es muy fácil de obtener (ya la obtuvimos, por otro método, al principio del Tutorial04). Recuerda que el valor de X se obtiene sumando las filas de `dosDados`:

```
X = rowSums(dosDados)
(tablaX = table(X))
```

```
## X
##  2  3  4  5  6  7  8  9 10 11 12
##  1  2  3  4  5  6  5  4  3  2  1
```

Y su representación gráfica es:

```
barplot(tablaX)
```



La **forma** de la distribución es claramente triangular, sin curvatura alguna. El valor medio de X (media teórica) es $\mu_X = 7$. Podemos confirmar esto, y además calcular la varianza (teórica, o poblacional):

```
(muX = mean(X))
```

```
## [1] 7
```

```
(sigma2_X = sum((X - muX)^2)/length(X))
```

```
## [1] 5.8333
```

Ahora vamos a hacer lo mismo para estudiar la distribución (la forma en que se distribuyen los valores) de $\bar{X}$. La Tabla 6.2 del libro se obtiene con:

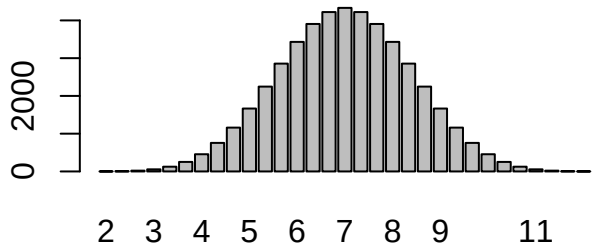
```
(tablaMediaMuestralX = as.matrix(table(mediasMuestrales)))
```

```
##                  [,1]
## 2                   1
## 2.33333333333333    6
## 2.66666666666667   21
## 3                  56
## 3.33333333333333  126
## 3.66666666666667  252
## 4                 456
## 4.33333333333333  756
## 4.66666666666667 1161
## 5                1666
## 5.33333333333333 2247
## 5.66666666666667 2856
## 6                3431
## 6.33333333333333 3906
## 6.66666666666667 4221
## 7                4332
## 7.33333333333333 4221
## 7.66666666666667 3906
## 8                3431
## 8.33333333333333 2856
## 8.66666666666667 2247
## 9                1666
## 9.33333333333333 1161
## 9.66666666666667  756
## 10                456
## 10.3333333333333  252
## 10.6666666666667  126
## 11                 56
## 11.3333333333333   21
## 11.6666666666667    6
## 12                  1
```

Esta es la tabla (o función) de densidad de $\bar{X}$. La hemos convertido en una matriz, porque la presentación por defecto de la tabla que se obtiene de R no es gran cosa (hay librerías que se encargan de esto). Pero en cualquier caso puedes comprobar que los valores son los que aparecen en el libro. Fíjate en que los valores de $\bar{X}$ avanzan de 1/3 en 1/3 (porque la muestra es de tamaño 3, claro).

A partir de aquí, la representación gráfica es inmediata:

```
barplot(table(mediasMuestrales))
```



La forma curvada, aproximadamente normal, de esta distribución se hace ahora evidente.

Ejercicio 2. *En este ejemplo hemos trabajado con muestras realmente pequeñas, en las que $n = 3$. ¿Qué modificaciones habría que hacer en el código para estudiar las muestras de tamaño 4 de la misma variable X? ¿Cuántas muestras de tamaño $n = 20$ hay? ¿Qué crees que sucedería si tratases de ejecutar el código muestras de tamaño $n = 20$ ¡No lo intentes, R se bloqueará! El mejor ordenador en el que hemos probado esto a duras penas podía con las muestras de tamaño $n = 5$.* *Solución en la página 44.* □

Media y desviación típica de $\bar{X}$.

Ahora, una vez que hemos comprobado que la distribución de $\bar{X}$ es aproximadamente normal, vamos a comprobar los resultados sobre su media $\mu_{\bar{X}}$ y su varianza $\sigma^2_{\bar{X}}$. Se trata, en primer lugar, de calcular el valor medio de $\bar{X}$ cuando consideramos todas las 46656 muestras de tamaño 3 posibles. El resultado es el que la figura anterior anunciaba:

```
(mu_barX = mean(mediasMuestrales))

## [1] 7
```

En cuanto a la varianza (¡poblacional!) se tiene:

```
(sigma2_barX = sum((mediasMuestrales - mu_barX)^2) / length(mediasMuestrales) )

## [1] 1.9444
```

Pero lo realmente interesante es el cociente entre esta varianza de $\bar{X}$ y la varianza de X (ambas poblacionales):

```
sigma2_X / sigma2_barX

## [1] 3
```

El resultado es 3. No aproximadamente 3, sino un 3 exacto, igual al tamaño n de las muestras que estamos considerando. Así hemos confirmado la relación

$$\sigma_{\bar{X}} = \frac{\sigma_X}{\sqrt{n}}$$

que aparece en la página 204 del libro.

1.2. Otro ejemplo.

Este resultado nos parece tan importante que vamos a incluir aquí un ejemplo adicional, tal vez incluso más espectacular que el anterior. Lo que vamos a hacer es estudiar la distribución muestral de la media de una variable aleatoria distinta, a la que vamos a llamar W. La variable W toma los valores del 1 al 20, todos con la misma probabilidad. Por lo tanto su tabla (o función) de densidad es:

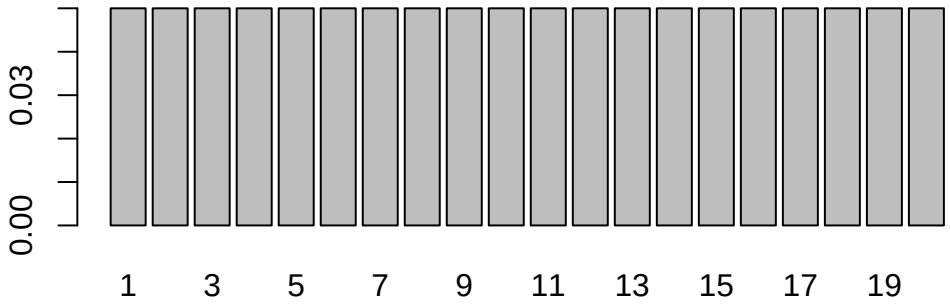
Valor:	1	2	$\cdots$	19	20
Probabilidad:	$\frac{1}{20}$	$\frac{1}{20}$	$\cdots$	$\frac{1}{20}$	$\frac{1}{20}$

Puedes pensar que la variable aleatoria X describe el resultado de un experimento en el que elegimos un número al azar del 1 al 20, de forma que todos los números sean equiprobables. Gráficamente, la forma de la distribución es plana, horizontal a una altura constante (igual a 1/20).

```
W = 1:20
table(W)/20
```

```
## W
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15
## 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05
##   16   17   18   19   20
## 0.05 0.05 0.05 0.05 0.05
```

```
barplot(table(W)/20)
```



Ahora, vamos a considerar muestras *con reemplazamiento* de X de tamaño $n = 4$. Es decir, una muestra consiste en elegir (*¡con reemplazamiento!*) cuatro números del 1 al 20, y queremos pensar en el conjunto de todas las muestras de tamaño 4 posibles. Vamos a proponer al lector que explore este conjunto mediante ejercicios.

Ejercicio 3.

1. *¿Cuántas muestras distintas de tamaño* 4 *existen?*
2. *Constrúyelas todas. Imita lo que hemos hecho en el ejemplo de los dados y usa* `gtools` *para obtener una matriz* `MuestrasW`*, que contenga, en cada fila, una de las muestras.*

Soluciones en la página 45. □

¡No sigas si no has hecho este ejercicio!

Una vez construidas las muestras podemos calcular las medias muestrales de X. En este caso, las cosas son más sencillas, porque las 160000 medias muestrales se obtienen directamente usando la función `rowMeans`, aplicada a la matriz `MuestrasW`:

```
mediasMuestralesW = rowMeans(MuestrasW)
length(mediasMuestralesW)
```

```
## [1] 160000
```

Y ahora podemos hacer una tabla de frecuencia de las medias muestrales,

```
(tablaMediaMuestralW = as.matrix(table(mediasMuestralesW)))
```

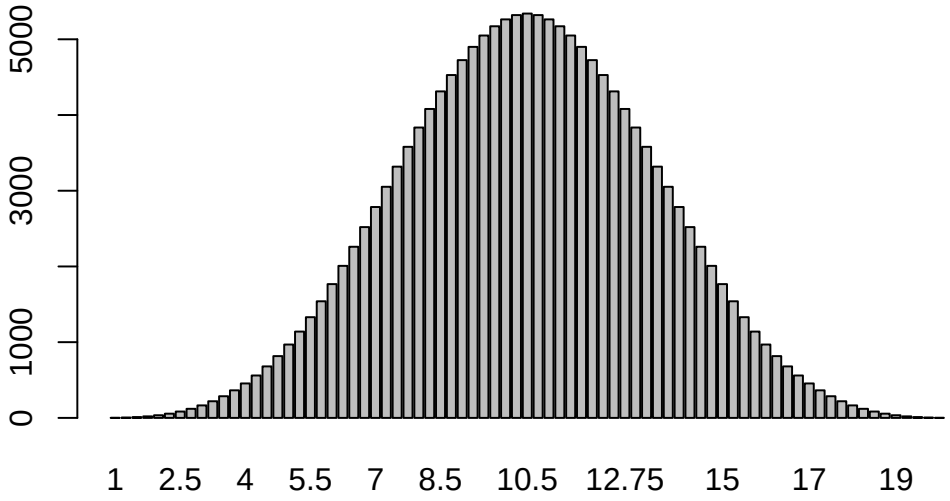
```
##       [,1]
## 1        1
## 1.25     4
## 1.5     10
## 1.75    20
## 2       35
## 2.25    56
```

```
## 2.5     84
## 2.75   120
## 3      165
## 3.25   220
## 3.5    286
## 3.75   364
## 4      455
## 4.25   560
## 4.5    680
## 4.75   816
## 5      969
## 5.25  1140
## 5.5   1330
## 5.75  1540
## 6     1767
## 6.25  2008
## 6.5   2260
## 6.75  2520
## 7     2785
## 7.25  3052
## 7.5   3318
## 7.75  3580
## 8     3835
## 8.25  4080
## 8.5   4312
## 8.75  4528
## 9     4725
## 9.25  4900
## 9.5   5050
## 9.75  5172
## 10    5263
## 10.25 5320
## 10.5  5340
## 10.75 5320
## 11    5263
## 11.25 5172
## 11.5  5050
## 11.75 4900
## 12    4725
## 12.25 4528
## 12.5  4312
## 12.75 4080
## 13    3835
## 13.25 3580
## 13.5  3318
## 13.75 3052
## 14    2785
## 14.25 2520
## 14.5  2260
## 14.75 2008
## 15    1767
## 15.25 1540
## 15.5  1330
## 15.75 1140
## 16     969
## 16.25  816
## 16.5   680
## 16.75  560
## 17     455
## 17.25  364
## 17.5   286
## 17.75  220
## 18     165
## 18.25  120
## 18.5    84
## 18.75   56
```

```
## 19      35
## 19.25   20
## 19.5    10
## 19.75    4
## 20       1
```

y representar gráficamente la forma de la distribución:

```
barplot(table(mediasMuestralesW))
```



Como se ve, obtenemos de nuevo una curva con la forma de campana característica de las curas normales, muy similar a la del ejemplo con dos dados. **Es muy importante** que observes que las dos variables que hemos usado como punto de partida son bastante distintas entre sí, y que el tamaño de muestra tampoco era el mismo. Y, sin embargo, la distribución de la media muestral es, en ambos casos, aproximadamente normal.

Ejercicio 4.

1. *Calcula la media* $\mu_{\bar{W}}$ *y la varianza* $\sigma^2_{\bar{W}}$.
2. *Comprueba que se cumple la relación:*

$$\sigma_{\bar{W}} = \frac{\sigma_W}{\sqrt{n}}$$

Soluciones en la página 45. □

2. Intervalos de confianza para la media con la distribución normal.

En el Tutorial05 hemos aprendido a usar el ordenador para resolver problemas directos e inversos de probabilidad asociados con la normal estándar. Para refrescar esos problemas vamos a empezar con un par de ejercicios.

Ejercicio 5.

1. *Usa R para calcular la probabilidad (recuerda que $Z \sim N(0,1)$)*

$$P(-2 < Z < 1.5),$$

que aparece en el Ejemplo 6.2.1 del libro (pág. 209).

2. *Localiza el valor a para el que se cumple $P(Z > a) = 0.25$, como en el Ejemplo 6.2.2 del libro (pág. 210).*

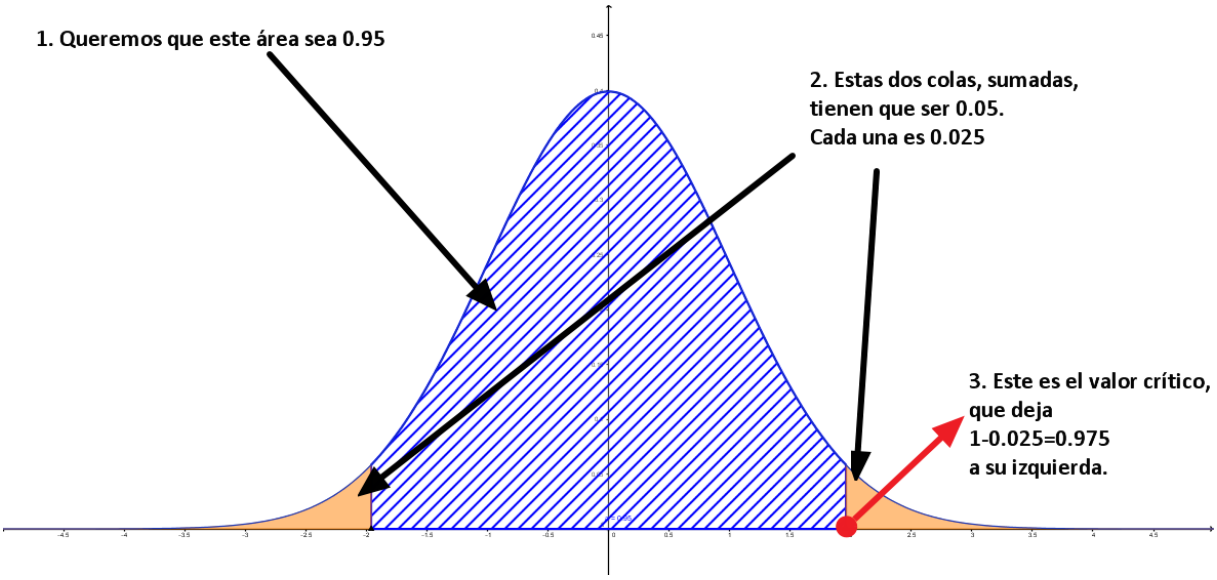
Soluciones en la página 46. □

Volviendo al tema que nos ocupa, en la Sección 6.2 del libro (pág. 206) hemos visto que, concretamente, los problemas inversos (de cálculo de cuantiles) son la clave para obtener los valores críticos necesarios para construir un intervalo de confianza. Es decir, que tenemos que ser capaces de encontrar el valor crítico $z_{\alpha/2}$, que satisface (ver página 213 del libro):

$$F\left(z_{\alpha/2}\right) = P\left(Z \leq z_{\alpha/2}\right) = 1 - \frac{\alpha}{2},$$

Por tanto, $z_{\alpha/2}$ es el valor que deja una probabilidad $\alpha/2$ en la cola derecha de Z, y una probabilidad $1 - \alpha/2$ en la cola izquierda.

Concretando, supongamos que queremos calcular un intervalo de confianza al (nivel de confianza) 95 %. Entonces $\alpha = 1 - 0.95 = 0.05$. O sea, $\alpha/2 = 0.025$, y por tanto $1 - \alpha/2 = 0.975$. La siguiente figura ilustra los pasos que hemos dado:



Esta cuenta, el hecho de que empezamos con 0.95, pero terminamos preguntando por el valor 0.975 es de las que, inicialmente, más dificultades causan. Con la práctica, y dibujando siempre lo que queremos, las cosas irán quedando más y más claras.

Vamos a continuación a ver cómo calcular ese valor $z_{\alpha/2} = z_{0.025}$, y los correspondientes intervalos de confianza, usando R y algunos otros programas. Nuestra recomendación es que te acostumbres a usar R.

2.1. Usando la función `qnorm` de R.

Para calcular el valor $z_{\alpha/2} = z_{0.025}$ en R, basta con ejecutar el comando:

```
    qnorm(0.975)
```

```
## [1] 1.96
```

Pero esta forma de proceder tiene el inconveniente de que nos obliga a hacer nosotros mismos la cuenta desde el nivel de confianza $nc = 0.95$ hasta ese valor $1 - \alpha/2 = 0.975$. Si cambiamos el nivel

de confianza a $nc = 0.90$ tenemos que repetir las cuentas, y corremos el riesgo de equivocarnos en algún paso. Es mejor automatizar, así que haremos que R haga toda la cuenta por nosotros:

```
    nc=0.95
    (alfa=1-nc)

## [1] 0.05

    (alfa2=alfa/2)

## [1] 0.025

    (z_alfa2=qnorm(1-alfa2) )

## [1] 1.96
```

También podrías usar `qnorm` con la opción `lower.tail = FALSE`. Es en el fondo una cuestión de gustos.

Usando este tipo de código, es muy fácil elaborar un fichero con todas las instrucciones necesarias para automatizar casi por completo la tarea de calcular el intervalo de confianza. Hay dos casos posibles, que hemos visto en las Secciones 6.2 y 6.3 del libro:

- Cuando podemos suponer que X es normal y la varianza de la población es conocida, la fórmula del intervalo es (Ecuación 6.10, pág. 216 del libro):
$$\bar{X} - z_{\alpha/2}\frac{\sigma_X}{\sqrt{n}} \leq \mu_X \leq \bar{X} + z_{\alpha/2}\frac{\sigma_X}{\sqrt{n}}.$$
- Si la varianza de la población es desconocida, pero el tamaño de la muestra es grande (en este curso estamos usando $n > 30$), entonces la fórmula del intervalo es (Ecuación 6.14, pág. 223 del libro):
$$\bar{X} - z_{\alpha/2}\frac{s}{\sqrt{n}} \leq \mu_X \leq \bar{X} + z_{\alpha/2}\frac{s}{\sqrt{n}}.$$

Las expresiones de ambos intervalos son muy parecidas, simplemente hay que cambiar σ por s. Así que, **teniendo siempre en cuenta las condiciones teóricas**, podemos usar el mismo código R para ambos casos. el código es muy simple, porque no hay que tomar decisiones: introducimos los datos del problema, y calculamos por orden todos los ingredientes del intervalo.

Pero todavía debemos tener en cuenta otro detalle: cómo recibimos los datos de la muestra. Podemos haber recibido la muestra *en bruto*, como un vector de datos (por ejemplo, en un fichero de tipo `csv`), o podemos tener directamente los valores $\bar{X}$, n, s, etc. Esta última situación es típica de los *problemas de libro de texto*, mientras que la primera es una situación más frecuente en los que llamaremos *problemas del mundo real*, en los que los datos no vienen preparados.

En consecuencia, vamos a presentar dos ficheros *"plantilla"*, uno para cada una de estas situaciones. El código está en los ficheros:

Tut06-IntConf-Media-UsandoZ-Estadisticos.R

y

Tut06-IntConf-Media-UsandoZ-MuestraEnBruto.R

que aparecen en las Tablas 1 y 2, respectivamente. Échales un vistazo antes de seguir leyendo.

Cuando vayas a utilizar estos ficheros, especialmente las primeras veces, lee atentamente las instrucciones que aparecen en los comentarios. En particular,

- Los ficheros **no funcionarán** si no introduces toda la información en las líneas adecuadas de la primera parte del código. Para ayudarte a identificar donde termina la parte del fichero en la que tienes que introducir esa información, hemos incluido un bloque comentado en el código con este aspecto.

```
################################################
# NO CAMBIES NADA DE AQUI PARA ABAJO
################################################
```

- En el segundo de estos ficheros, el que trabaja con datos *en bruto*, existen, a su vez, dos posibilidades:
  - Usar como punto de partida un vector de datos de R.
  - Usar como punto de partida un fichero de datos, por ejemplo de tipo `csv`.

  Para indicarle a R cuál de esas dos posibilidades queremos utilizar **es necesario** comentar (usando `#`) la(s) línea(s) que no vamos a usar, e introducir la información necesaria en las líneas de la opción que sí usamos.
- En cualquier caso, es esencial tener en cuenta las **condiciones teóricas de aplicabilidad** de estos ficheros. Para empezar, se supone que la población es normal. Pero incluso así, si la muestra es pequeña (pongamos $n < 30$) y no conoces σ, entonces **no debes usar estos ficheros**.
- Para usar estos ficheros, no utilices *copiar y pegar* a partir de este pdf. Descarga los ficheros adjuntos en tu ordenador y ábrelos directamente con RStudio.

Para que puedas practicar el funcionamiento de estos ficheros, aquí tienes un ejercicio con unos cuantos apartados, que cubren, esencialmente, todas las situaciones en las que te puedes encontrar.

Ejercicio 6.

1. *Una muestra de $n = 10$ elementos de una población normal con desviación típica conocida $\sigma = 2.3$, tiene media muestral $\bar{X} = 132.5$. Calcula intervalos de confianza al 90 %, 95 % y 99 % para la media μ. ¿Cómo son las anchuras de esos intervalos?*
2. *Una muestra de 10 elementos de una población normal tiene cuasidesviación típica $s = 2.3$, y media muestral $\bar{X} = 132.5$. Calcula un intervalo de confianza al 95 % para la media μ.*
3. *Una muestra de $n = 450$ elementos de una población normal tiene cuasidesviación típica $s = 2.3$, y media muestral $\bar{X} = 132.5$. Calcula un intervalo de confianza al 99 % para la media μ.*
4. *Dado este vector de datos,*

   ```
   datos = c(3.09,3.06,2.79,2.44,2.54,3.52,3.07,2.67,2.99,2.82,2.94,3.57,2.38,3.24,
   3.16,3.45,3.24,2.97,3.39,2.97,2.68,2.91,2.84,3.15,3.15)
   ```

   del que sabemos que procede de una población normal con $\sigma = 0.5$, calcula un intervalo de confianza al 95 % para la media μ de esa población.
5. *El fichero adjunto Tut06-IntConf-Media-UsandoZ-datos.csv contiene una muestra de cierta población. Utiliza esos datos para calcular un intervalos de confianza al 95 % para la media μ de esa población.*
6. *Usa R para comprobar los resultados del Ejemplo 6.2.3 del libro (pág. 216).*

Soluciones en la página 46. □

```
#########################################################
# www.postdata-statistics.com
# POSTDATA. Introduccion a la Estadistica
# Tutorial-06.
#
# Fichero de instrucciones R para calcular
# un intervalo de confianza (1-alfa) para la media de una
# poblacion normal N(mu,sigma), a partir de una
# muestra con n datos.
#
# Este fichero usa los resumenes de una muestra,
# previamente calculados (numero de datos, media muestral, etc.)
#
#################################################################

rm(list=ls()) #limpieza inicial

# Introduce el numero de datos de la muestra,
n =

# Introduce aqui el valor de xbar, la media muestral
xbar =

# LEE ESTAS INSTRUCCIONES ATENTAMENTE:
# Si la muestra tiene mas de 30 elementos
# introduce el valor de s, la cuasidesviacion tipica de la poblacion.
# Si la poblacion es normal, y conoces sigma, la desviacion tipica de
# la poblacion, puedes usarla en lugar de s, aunque sea n<30.
#
# SI LA MUESTRA TIENE < 30 ELEMENTOS Y DESCONOCES SIGMA,
# NO USES ESTE FICHERO!!
# ASEGURATE DE HABER ENTENDIDO ESTAS INSTRUCCIONES

s =    # O sigma, lee las instrucciones.

# y el nivel de confianza deseado.
nc =

################################################
# NO CAMBIES NADA DE AQUI PARA ABAJO
################################################
(alfa = 1 - nc )

# Calculamos el valor critico:

(z_alfa2 = qnorm( 1 - alfa / 2 ) )

#y la semianchura del intervalo
(semianchura=z_alfa2 * s / sqrt(n) )

# El intervalo de confianza (a,b) para mu es este:
(intervalo = xbar + c(-1, 1) * semianchura )
```

Tabla 1: Código R del fichero Tut06-IntConf-Media-UsandoZ-Estadisticos.R

```
#############################################################
# www.postdata-statistics.com
# POSTDATA. Introduccion a la Estadistica
# Tutorial-06.
#
# Fichero de instrucciones R para calcular
# un intervalo de confianza (1-alfa) para la media de una
# poblacion normal N(mu,sigma), a partir de una
# muestra con n datos.
#
# Este fichero usa los datos de la muestra en bruto, en forma
# de vector o en un fichero csv. Lee las instrucciones mas .
# abajo
#############################################################################

rm(list=ls()) #limpieza inicial

#############################################
# EL FICHERO NO FUNCIONARÁ BIEN HASTA
# QUE HAYAS COMPLETADO CORRECTAMENTE
# TODA LA INFORMACIÓN NECESARIA.
#############################################

# Una posibilidad es que tengas la muestra como un vector.
muestra =    # SI NO SE USA, ESCRIBE # AL PRINCIPIO DE ESTA LINEA


# Si lees la muestra de un fichero csv:
# 1. selecciona el directorio de trabajo
# Para eso, escribe el nombre entre las comillas. En RStudio puedes usar el
    tabulador como ayuda.
(setwd(dir=""))  # SI NO SE USA, ESCRIBE # AL PRINCIPIO DE ESTA LINEA

# 2. Ahora introduce entre las comillas el nombre del fichero, y el tipo de
    separador, etc.
muestra = read.table(file=" ", header = , sep=" ",dec=".")[ , 1]  # SI NO SE
     USA, ESCRIBE # AL PRINCIPIO DE ESTA LINEA


# LEE ESTAS INSTRUCCIONES ATENTAMENTE:
# Si la muestra tiene mas de 30 elementos
# calculamos el valor de s, la cuasidesviacion tipica de la poblacion.
# Si la poblacion es normal, y conoces sigma, la desviacion tipica de
# la poblacion, puedes CAMBIAR A MANO s por sigma, aunque sea n<30.
#
# SI LA MUESTRA TIENE < 30 ELEMENTOS Y DESCONOCES SIGMA,
# NO USES ESTE FICHERO!!
# ASEGURATE DE HABER ENTENDIDO ESTAS INSTRUCCIONES

(s = sd(muestra) )   # O sigma, lee las instrucciones.

# y el nivel de confianza deseado.
nc =

#########################################################
#NO CAMBIES NADA DE AQUI PARA ABAJO
#########################################################
# Calculamos la longitud de la muestra

(n = length( muestra ))

# Calculamos la media muestral
(xbar = mean( muestra ) )

# Calculamos alfa
(alfa = 1 - nc )

# Calculamos el valor critico:

(z_alfa2 = qnorm( 1 - alfa / 2 ) )

#y la semianchura del intervalo
(semianchura=z_alfa2 * s / sqrt(n) )

# El intervalo de confianza (a,b) para mu es este:
(intervalo = xbar + c(-1, 1) * semianchura )
```

Tabla 2: Código R del fichero Tut06-IntConf-Media-UsandoZ-MuestraEnBruto.R

2.2. Otros programas.

Vamos a ver brevemente las herramientas que nos ofrecen otros programas pra calcular intevalos de confianza para μ, usando Z.

GeoGebra.

En GeoGebra disponemos, en primer lugar, de la función

```
IntervaloMediaZ[ <Media (muestra)>, <σ>, <Tamaño (muestra)>, <Nivel> ]
```

donde los nombres de los argumentos entre los símbolos < y > indican en que orden debemos introducir los datos del problema. Por ejemplo, para obtener el intervalo de confianza del apartado 3 del Ejercicio 6 usaríamos la función de esta manera (en la *Línea de entrada*):

```
IntervaloMediaZ[132.5, 2.3, 450, 0.99]
```

y GeoGebra contesta con una *lista*, que contiene los dos extremos del intervalo.

Si lo que tenemos es un vector de datos, con un número no demasiado grande de elementos, podemos usar esta misma función de otra manera, siguiendo esta sintaxis:

```
IntervaloMediaZ[ <Lista de datos (muestra)>, <σ>, <Nivel> ]
```

Para ver como funciona, usaremos como ejemplo el apartado 4 del Ejercicio 6. En primer lugar, creamos una lista en GeoGebra con los datos del problema, ejecutando este comando (tendrás que copiar cada línea por separado, para pegarlas formando una única línea de entrada en GeoGebra):

```
datos = {3.09, 3.06, 2.79, 2.44, 2.54, 3.52, 3.07, 2.67, 2.99, 2.82, 2.94, 3.57,
2.38, 3.24, 3.16, 3.45, 3.24, 2.97, 3.39, 2.97, 2.68, 2.91, 2.84, 3.15, 3.15}
```

Una vez hecho esto, calculamos el intervalo mediante:

```
IntervaloMediaZ[datos, 0.5, 0.95]
```

Como puedes ver, es difícil utilizar de forma práctica esta función cuando partimos de los datos *en bruto* de una muestra grande.

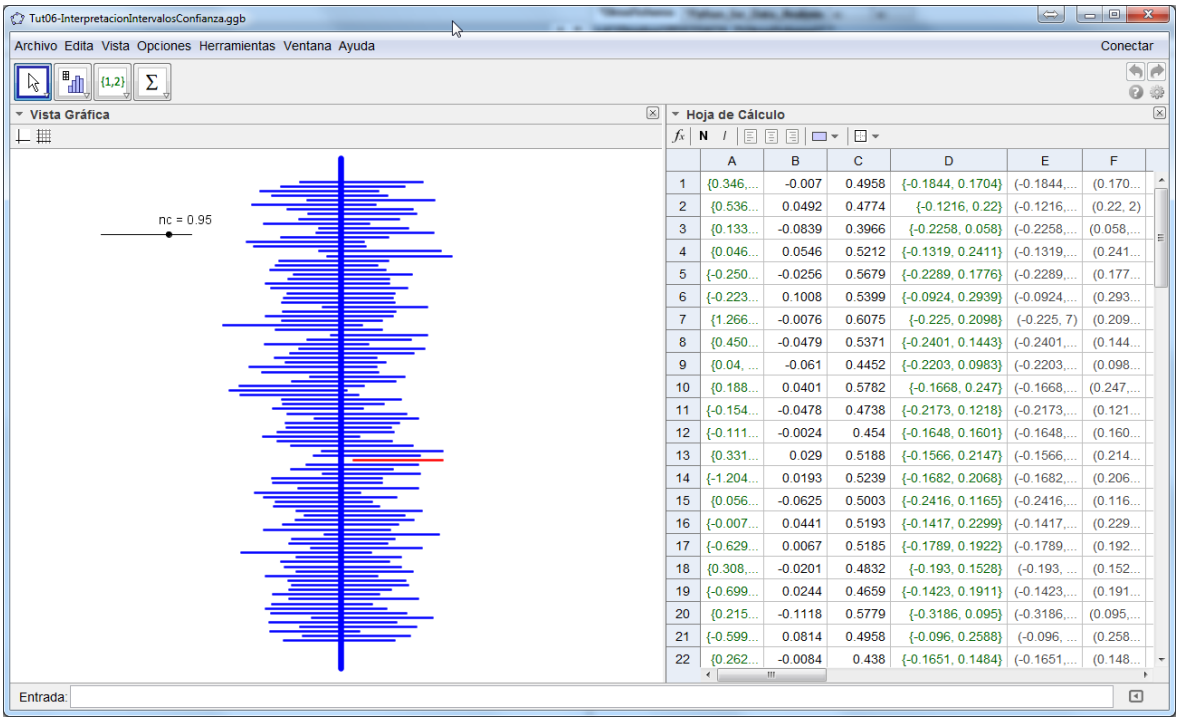
Ejercicio 7. *Usa GeoGebra para hacer el apartado 1 del Ejercicio 6. Solución en la página 49.* □

Interpretación probabilística del intervalo de confianza, usando GeoGebra.

Antes de pasar a otros programas, queremos señalar que la Figura 6.9 del libro (pág. 218), que ilustra la interpretación probabilística de los intervalos de confianza, se ha obtenido con este fichero de GeoGebra:

Tut06-InterpretacionIntervalosConfianza.ggb

Al abrirlo verás una ventana de GeoGebra similar a la de esta figura:



	A	B	C	D	E	F
1	{0.346,...	-0.007	0.4958	{-0.1844, 0.1704}	(-0.1844,...	(0.170...
2	{0.536...	0.0492	0.4774	{-0.1216, 0.22}	(-0.1216,...	(0.22, 2)
3	{0.133...	-0.0839	0.3966	{-0.2258, 0.058}	(-0.2258,...	(0.058,...
4	{0.046...	0.0546	0.5212	{-0.1319, 0.2411}	(-0.1319,...	(0.241...
5	{-0.250...	-0.0256	0.5679	{-0.2289, 0.1776}	(-0.2289,...	(0.177...
6	{-0.223...	0.1008	0.5399	{-0.0924, 0.2939}	(-0.0924,...	(0.293...
7	{1.266...	-0.0076	0.6075	{-0.225, 0.2098}	(-0.225, 7)	(0.209...
8	{0.450...	-0.0479	0.5371	{-0.2401, 0.1443}	(-0.2401,...	(0.144...
9	{0.04, ...	-0.061	0.4452	{-0.2203, 0.0983}	(-0.2203,...	(0.098...
10	{0.188...	0.0401	0.5782	{-0.1668, 0.247}	(-0.1668,...	(0.247,...
11	{-0.154...	-0.0478	0.4738	{-0.2173, 0.1218}	(-0.2173,...	(0.121...
12	{-0.111...	-0.0024	0.454	{-0.1648, 0.1601}	(-0.1648,...	(0.160...
13	{0.331...	0.029	0.5188	{-0.1566, 0.2147}	(-0.1566,...	(0.214...
14	{-1.204...	0.0193	0.5239	{-0.1682, 0.2068}	(-0.1682,...	(0.206...
15	{0.056...	-0.0625	0.5003	{-0.2416, 0.1165}	(-0.2416,...	(0.116...
16	{-0.007...	0.0441	0.5193	{-0.1417, 0.2299}	(-0.1417,...	(0.229...
17	{-0.629...	0.0067	0.5185	{-0.1789, 0.1922}	(-0.1789,...	(0.192...
18	{0.308,...	-0.0201	0.4832	{-0.193, 0.1528}	(-0.193, ...	(0.152...
19	{-0.699...	0.0244	0.4659	{-0.1423, 0.1911}	(-0.1423,...	(0.191...
20	{0.215...	-0.1118	0.5779	{-0.3186, 0.095}	(-0.3186,...	(0.095,...
21	{-0.599...	0.0814	0.4958	{-0.096, 0.2588}	(-0.096, ...	(0.258...
22	{0.262...	-0.0084	0.438	{-0.1651, 0.1484}	(-0.1651,...	(0.148...

La parte izquierda contiene la *Vista Gráfica*, con una colección de 100 intervalos de confianza, correspondientes a 100 muestras distintas extraídas todas de la misma población normal. La parte derecha es la *Vista de hoja de cálculo*. Esta parte de GeoGebra tiene un comportamiento similar al de una hoja de cálculo como Calc. La diferencia fundamental es que las celdas de esta hoja de cálculo pueden guardar objetos geométricos, funciones, etc. Por ejemplo, puedes tener una columna llena de parábolas, por decir algo. En nuestro caso, la columna `A` contiene las 100 muestras, las columnas `B` y `C` contienen, respectivamente, las medias y cuasidesviaciones típicas de esas muestras y la columna `D` contiene los intervalos de confianza. El resto de columnas contienen operaciones auxiliares para dibujar el gráfico, y aquí no nos detendremos en comentarlas.

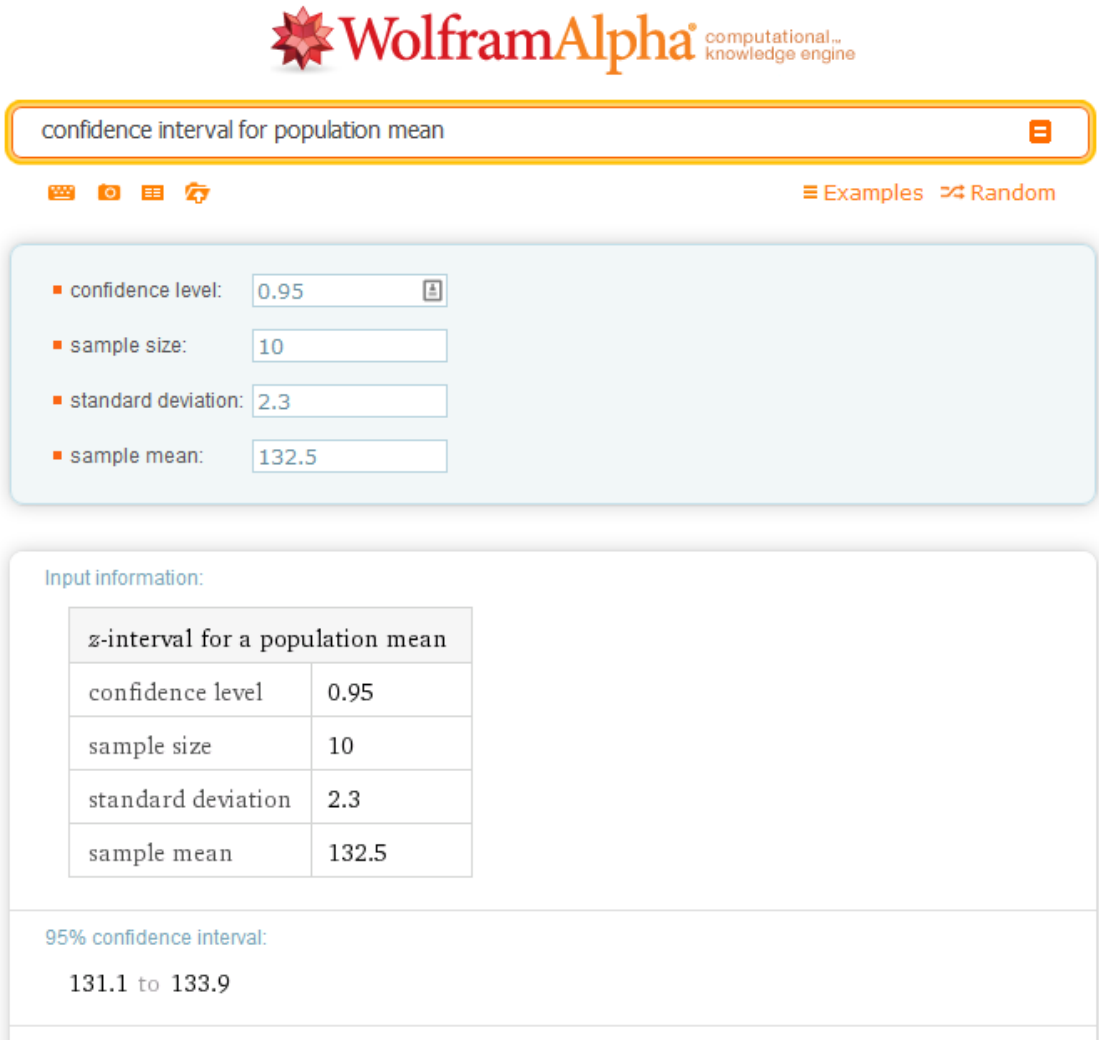
Prueba a pulsar `Ctrl + R` varias veces seguidas. Cada vez que lo haces, GeoGebra genera 100 nuevos intervalos de confianza. Verás, destacados en color rojo, aquellos intervalos de confianza que no contienen a la media real de la población μ, cuyo valor es 0 y cuya posición se indica mediante el segmento vertical. Si el nivel de confianza es del 95 %, es previsible que aparezcan alrededor de 5 intervalos rojos. Tal vez alguno más, o alguno menos. Pero lo que significa ese nivel de confianza es que si tomamos muchas muestras, el 95 % de los intervalos de confianza correspondientes a esas muestras serán intervalos azules, de los que contienen a la media real de la población. Si lo deseas, puedes hacer clic con el botón derecho del ratón sobre cualquiera de los intervalos de la figura, y GeoGebra te indicará el número de fila que corresponde a ese intervalo concreto. Con ese número de fila puedes localizar el intervalo en la columna `D` de la *Hoja de Cálculo*. Si el intervalo es rojo, podrás comprobar que $\mu = 0$ no pertenece al intervalo, y a la recíproca en el caso de intervalos azules.

Wolfram Alpha.

Para calcular intervalos de confianza como estos con Wolfram Alpha prueba a introducir esta frase en el campo de entrada del programa:

```
confidence interval for population mean
```

Al hacerlo Wolfram Alpha abre una página con una serie de campos en los que podemos introducir los valores necesarios para la construcción del intervalo. La siguiente figura muestra esa página, en la que hemos introducido los valores correspondientes al apartado 1 del Ejercicio6. También se muestra el resultado que ha producido Wolfram Alpha.



En la versión gratuita de Wolfram Alpha que estamos usando no hay opciones para calcular un intervalo de confianza a partir de una muestra *en bruto* (esas opciones sí existen en la versión de pago). Alternativamente, también puedes obtener el intervalo de confianza para la media usando un comando como este (es un ejemplo de la propia Web de Wolfram Alpha):

```
z-interval mean 27, standard deviation 9,  sample size 100
```

El resultado $(25.24, 28.76) = 27 \pm 1.76397$, se muestra en una ventana como la que hemos visto arriba.

Calc.

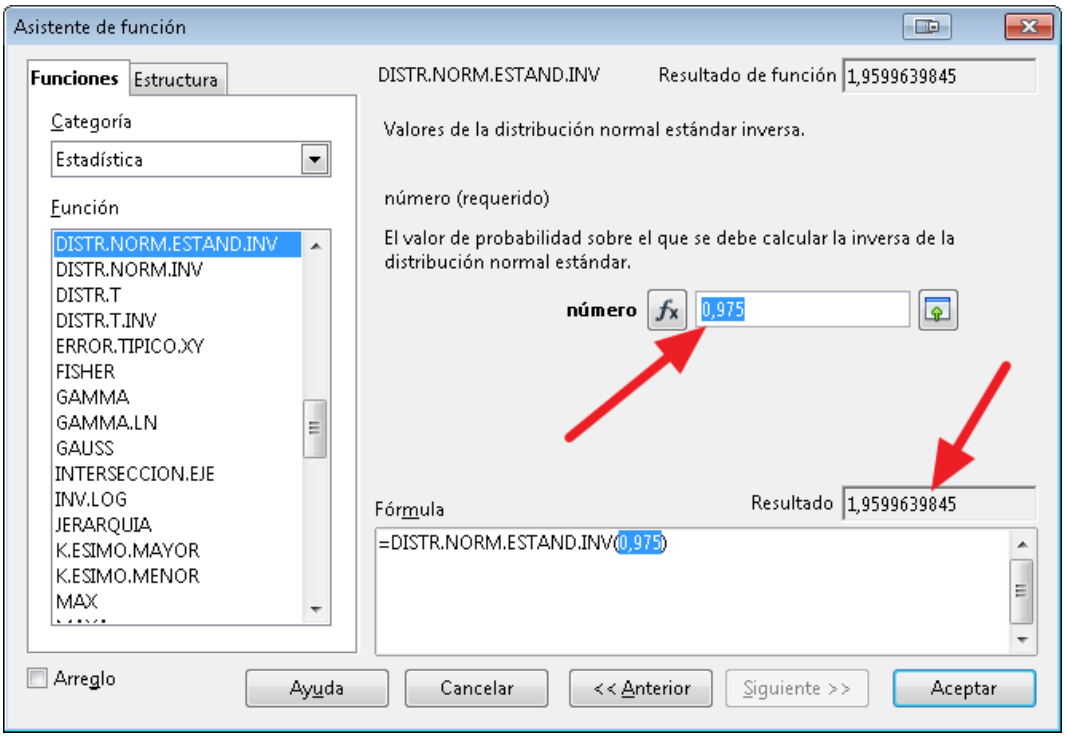
En Calc, para calcular los cuantiles de la distribución normal estándar Z disponemos de la función

```
DISTR.NORM.ESTAND.INV
```

En la siguiente figura puedes ver el cuadro de diálogo correspondiente a esta función, que hemos ilustrado en el caso del cálculo de $z_{\alpha/2}$, para un nivel de confianza $nc = 0.95$. Calc utiliza, en este caso al igual que R, la cola izquierda de Z.

Hemos indicado con las dos flechas rojas el valor de probabilidad (de la cola izquierda) que hemos introducido, y la respuesta que Calc va a producir. A partir de esta función es muy fácil elaborar una hoja de Calc que nos permita automatizar el cálculo del intervalo de confianza a partir de los datos de una muestra, como hemos hecho en el fichero:

Tut06-IntConf-Media-UsandoZ.ods



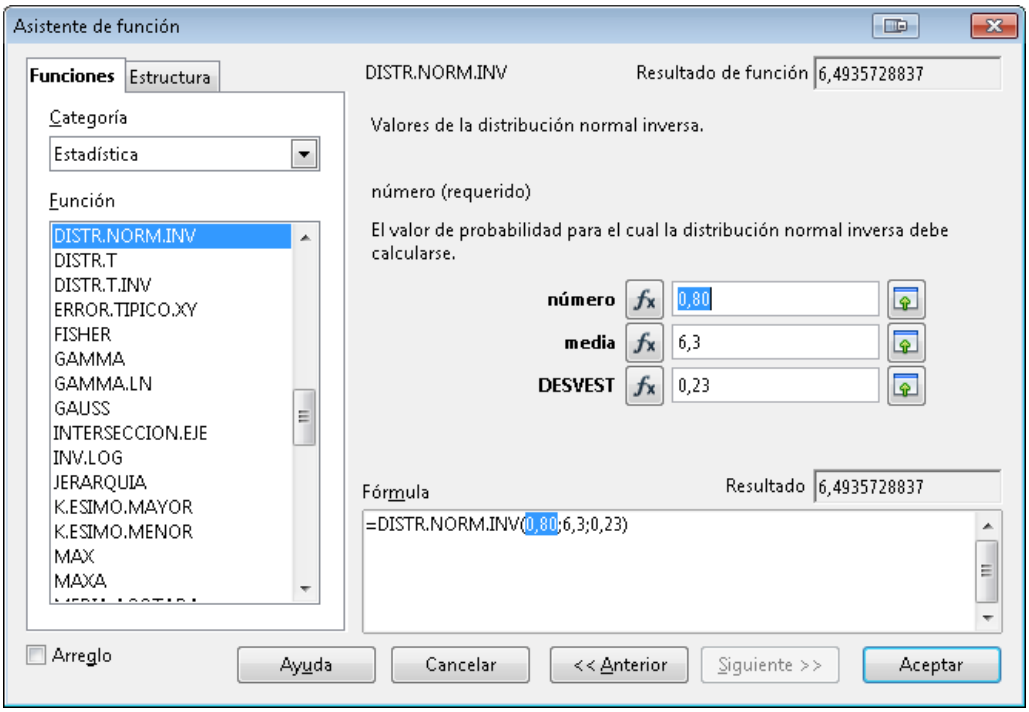
Ejercicio 8. *Usa este fichero para tratar de calcular los intervalos de confianza de los apartados 1 y 3 del Ejercicio 6 (pág. 13). ¿Qué dificultades encuentras? Solución en la página 49.* □

Antes de seguir adelante, vamos a aprovechar nuestro paso por aquí para comentar los recursos que Calc ofrece para trabajar con distribuciones normales, tanto para problemas directos como inversos. Ya hemos comentado la función `DISTR.NORM.ESTAND.INV`. Pero, además de esta, en Calc disponemos de las siguientes funciones:

- `DISTR.NORM.INV`. Sirve para problemas *inversos* en distribuciones normales no necesariamente iguales a Z. En la siguiente figura puedes ver el cuadro de diálogo en el que estamos usando esta función para resolver el problema inverso:

  $$P(X < K) = 0.8,$$

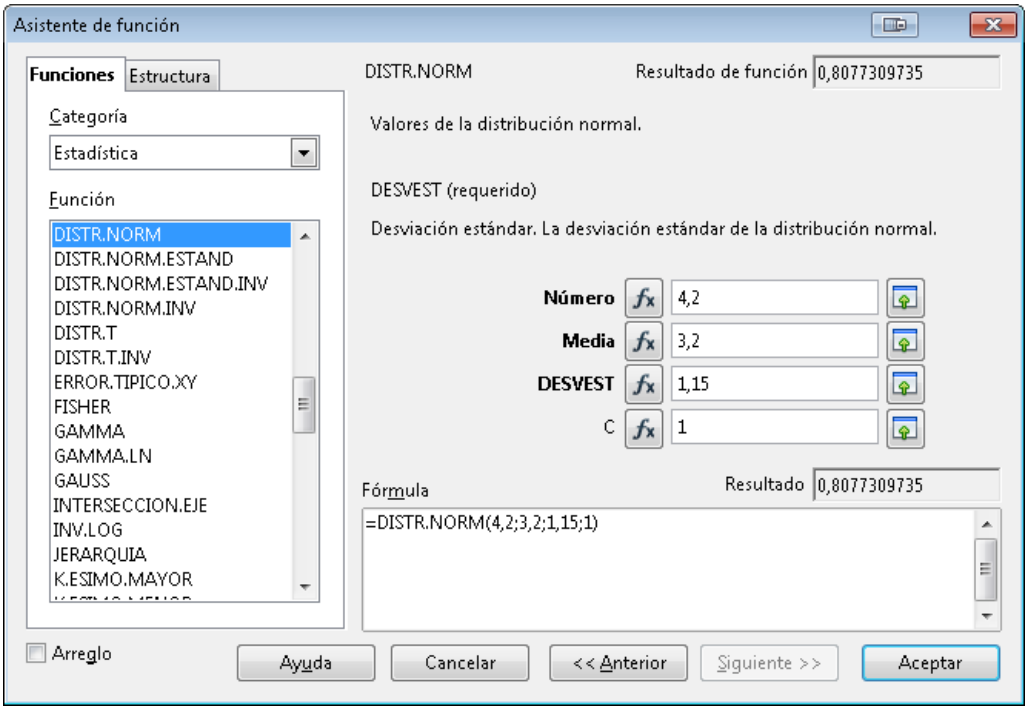
  en una normal X de tipo $N(\mu = 6.3, \sigma = 0.23)$. Usamos `DISTR.NORM.INV(0,80;6,3;0,23)` para obtener 6.4935728837.



- `DISTR.NORM`. Sirve para problemas *directos* en distribuciones normales no necesariamente iguales a Z. La siguiente figura muestra el cuadro de diálogo en el que estamos usando esta función para resolver el problema directo:

$$P(X < 4.2) = ??,$$

en una normal X de tipo $N(\mu = 3.2, \sigma = 1.15)$. Usamos `DISTR.NORM(4,2;3,2;1,15;1)` para obtener $0,8077309735$.



En el caso de esta función debemos introducir un parámetro adicional, que Calc llama `C`, que puede ser $C = 1$ o $C = 0$. El valor $C = 1$ sirve para calcular la función de distribución de la normal (como `pnorm` en R, y también con la cola izquierda). El valor $C = 0$ devuelve valores de la función de densidad de la normal (como `dnorm` en R), y apenas lo vamos a utilizar en este curso.

- `DISTR.NORM.ESTAND` Finalmente, esta función se usa para problemas *directos* en la distribución normal estándar Z. En este caso siempre se calcula la función de distribución (`pnorm`, cola izquierda), y el único argumento es el valor del que sea calcular la probabilidad de su cola izquierda. Por ejemplo, para resolver

$$P(Z < 2.1)$$

usaríamos `DISTR.NORM(2,1)` para obtener $0,9821355794$.

3. Muestras pequeñas. La t de Student.

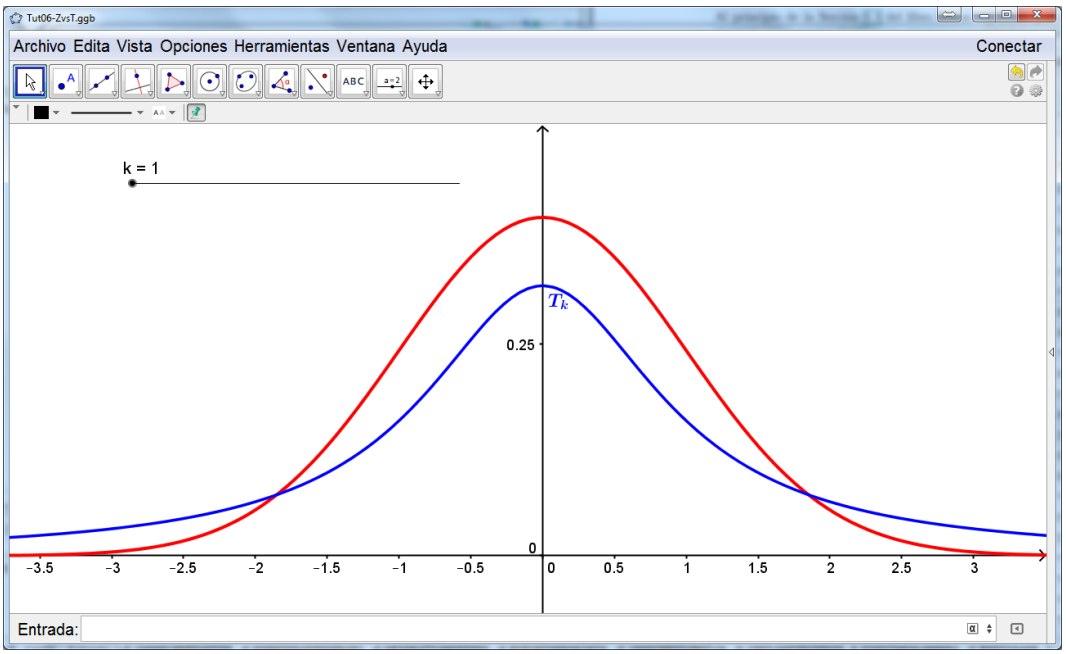
En esta sección vamos a ver cómo utilizar los programas que conocemos para trabajar en problemas en los que interviene la distribución t de Student.

3.1. Relación entre la t de Student y la normal Z, usando GeoGebra.

Al principio de la Sección 6.4 del libro hemos explicado que si T_k es una distribución tipo t de Student, con k grados de libertad, entonces a medida que k aumenta la distribución se parece cada vez más a la normal estándar Z, de manera que a partir de $k = 30$ las gráficas de las densidades de ambas distribuciones son prácticamente idénticas. Para ilustrar esto te hemos preparado un fichero GeoGebra muy sencillo:

Tut06-ZvsT.ggb

Al abrirlo verás algo como esto:



El fichero muestra las dos funciones de densidad, Z en rojo y T_k en azul, y un deslizador que permite ir variando el valor de k, entre 1 y 100, para que puedas ver lo que sucede. Para dibujar las dos curvas hemos usado estos comandos de GeoGebra:

```
DistribuciónT[k, x ]
Normal[0, 1, x ]
```

3.2. Cálculos de probabilidad para la t de Student en R.

Una ventaja de trabajar con R, como ya hemos comentado anteriormente, es la homogeneidad en la sintaxis para todas las distribuciones de probabilidad. Concretamente, el trabajo con la t de Student es prácticamente igual al que hemos visto con la normal estándar Z, salvo por el detalle de los grados de libertad. Si en el caso de Z teníamos las funciones `dnorm`, `pnorm`, `qnorm`, y `rnorm`, ahora tenemos sus análogas `dt`, `pt`, `qt` y `rt`, con los significados evidentes. La primera de ellas, `dt`, sirve para calcular la función de densidad, y apenas la usaremos. La última, `rt`, permite calcular valores aleatorios, distribuidos según la t de Student. La usaremos sobre todo para hacer simulaciones. Las dos restantes, `pt` y `qt` llevarán el peso de nuestro trabajo con esta distribución. Vamos a presentarlas brevemente, antes de ponerlas a trabajar.

Las funciones `pt` y `qt`.

La función `pt` calcula la probabilidad de la cola izquierda (como siempre) de una distribución t de Student. Por ejemplo, si T_{18} es una distribución t de Student con $k = 18$ grados de libertad, el problema directo de probabilidad:

$$P(T_{18} < 2.3)$$

se resuelve mediante este comando en R:

```
pt(2.3, df = 18)

## [1] 0.98319
```

Los grados de libertad se indican, como ves, con el argumento `df` (del inglés *degrees of freedom*). Si deseas usar la cola derecha dispones, como siempre, de la opción `lower.tail=FALSE`. Para calcular en una T_{12} (es decir, una distribución t de Student con 12 grados de libertad) la probabilidad

$$P\left(T_{12} > 3.1\right)$$

podemos usar indistintamente estos dos comandos:

```
1 - pt(3.1, df = 12)

## [1] 0.004595

pt(3.1, df = 12, lower.tail=FALSE)

## [1] 0.004595
```

La función `qt`, por su parte, permite calcular los cuantiles, y por tanto resolver problemas inversos de probabilidad asociados con la t de Student. Como siempre, por defecto, se usa la cola izquierda de la distribución, salvo que usemos `lower.tail=FALSE`. Es decir, que para resolver un problema como el de encontrar el valor K tal que

$$P(T_{24} < K) = 0.87$$

usaríamos este comando de R:

```
qt(0.87, df = 24)

## [1] 1.1537
```

Vamos a practicar el uso de ambas funciones con unos cuantos ejercicios.

Ejercicio 9. *Sea T_{15} una variable aleatoria de tipo t de Student, con $k = 15$ grados de libertad. Calcular los siguientes valores.* **Es muy recomendable** *hacer un dibujo esquemático de la situación en cada uno de los apartados.*

1. $P(T_{15} \leq -1.341)$
2. $P(T_{15} \geq 2.602)$
3. $P(-1.753 \leq T_{15} \leq 1.753)$
4. *Valor de t tal que* $P(T_{15} \leq t) = 0.95$.
5. *Valor de t tal que* $P(T_{15} \geq t) = 0.025$.
6. *Valor de t tal que* $P(T_{15} \leq t) = 0.05$.
7. *Valor de t tal que* $P(T_{15} \geq t) = 0.975$.
8. *Valor de t tal que* $P(-t \leq T_{15} \leq t) = 0.95$.
9. *Valor de t tal que* $P(-t \leq T_{15} \leq t) = 0.93$.

Soluciones en la página 49. □

Los dos últimos apartados de este ejercicio son ejemplos del tipo de cálculo que necesitamos para obtener los valores críticos que se usan en un intervalo de confianza, como los que vamos a construir a continuación.

3.3. La t de Student en GeoGebra, Wolfram Alpha y Calc.

3.3.1. GeoGebra.

La *Calculadora de Probabilidades* de GeoGebra permite trabajar con la distribución t de Student de forma muy parecida a lo que vimos en el caso de la distribución normal. Basta con seleccionar `Student` en el menú desplegable que se sitúa bajo la gráfica.

Aparte de esto, puedes usar directamente algunas funciones en la *Línea de Entrada* de GeoGebra. Por ejemplo

```
DistribuciónT[k, x]
```

produce como resultado la probabilidad de la cola izquierda de x en la distribución T_k:

$$P(T_k \leq x)$$

Es decir, el mismo efecto que si en R utilizaras $pt(x, df = k)$. Recuerda que los resultados se obtienen en el panel de *Vista Algebraica*, a la izquierda (si no es visible, usa el menú *Vista* para hacerlo aparecer).

En la versión actual de GeoGebra, para conseguir la probabilidad de una cola derecha hay que usar el truco de $1 - p$. Es decir,

```
1 - DistribuciónT[k, x]
```

produce, como cabe esperar, el valor

$$P(T_k > x).$$

Para los problemas inversos de probabilidad disponemos de la función

```
DistribuciónTInversa[k, p]
```

cuyo resultado es el valor x tal que:

$$P(T_k \leq x) = p.$$

Es decir el cuantil p de T_k, que en R calcularíamos con `qt(p, df=k)`. De nuevo, si deseas localizar el valor cuya cola derecha representa una probabilidad p, usa un truco como:

```
DistribuciónTInversa[k, 1- p]
```

Ejercicio 10. *Usa estas funciones, o la* Calculadora de Probabilidades *de GeoGebra para repetir el Ejercicio 9.* □

3.3.2. Wolfram Alpha.

La sintaxis de Wolfram Alpha es, como siempre, algo más peculiar. Seguramente existen otras formas de calcular estas cantidades, pero las que incluimos aquí son las más sencillas que conocemos.

Vamos a resolver, en primer lugar un problema directo de probabilidad. Concretamente, dada una distribución T_{21}, vamos a calcular la probabilidad:

$$P(T_{21} > 2.7)$$

Para ello, en Wolfram Alpha ejecutamos el comando:

```
P[X > 2.3] for X student t with 21 dof
```

y obtenemos como resultado aproximado 0.0159011. Si lo que deseas es la probabilidad de un intervalo, como en este ejemplo (procede de la propia página de Wolfram Alpha)

$$P(-1.2 < T_{12} < 2.3)$$

puedes usar una sintaxis como esta:

```
P[-1.2 < X < 2.3] for X student t with 12 dof
```

y obtendrás el valor aproximado 0.853254.

Ejercicio 11. *Usa R o GeoGebra (o, aún mejor, ambos) para comprobar estos resultados. Solución en la página 52.* □

Los problemas inversos se pueden resolver reduciéndolos manualmente a problemas sobre la cola izquierda de la distribución T_k. Por ejemplo, para encontrar el valor x tal que (es una cola derecha)

$$P(T_{12} > x) = 0.07$$

hacemos un pequeño esquema gráfico que nos ayude a entender que esto es lo mismo que buscar el valor x tal que

$$P(T_{12} < x) = 0.93.$$

Y ahora podemos pedirle a Wolfram Alpha que nos diga cuál es ese valor usando esta sintaxis:

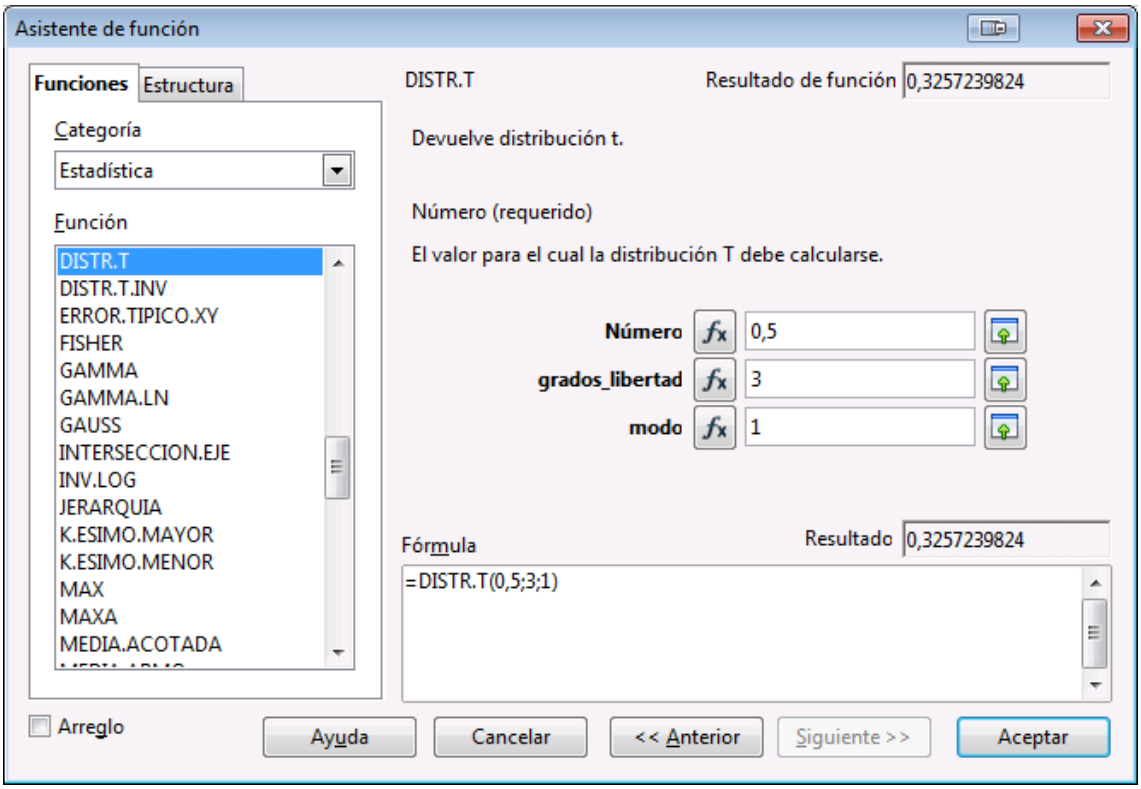
```
93th percentile  for student t with 12 dof
```

El resultado aproximado es 1.5804.

Ejercicio 12. *Usa R o GeoGebra (o, aún mejor, ambos) para comprobar este resultado. Solución en la página 53.* □

3.3.3. Calc.

En Calc, las funciones `DISTR.T` y `DISTR.T.INV` permiten resolver problemas directos e inversos de probabilidad para la distribución t de Student, de modo parecido a lo que hemos visto para la distribución normal. Pero es importante entender las diferencias. Empecemos, en este caso, por los problemas directos. El cuadro de diálogo que aparece, al utilizar la función `DISTR.T` para resolver, usando la distribución T_3 (con $k = 3$ grados de libertad) un problema directo, es este:



La función tiene un parámetro `modo`, que puede tomar los valores 1 o 2. Ese valor indica el número de colas de la distribución que se utilizan. Si `modo` = 2, entonces Calc obtiene el resultado repartiendo la probabilidad entre las dos colas de la distribución t (que es justo lo que se utiliza para construir el intervalo de confianza). Pero si `modo` = 1, entonces Calc sólo utiliza una de las dos colas.

Ejercicio 13.

1. *Observa los datos del ejemplo que aparecen en ese cuadro de diálogo. Es decir,*

```
Número = 0,5
grados_libertad = 3
modo = 1
```

Y el resultado que es aproximadamente 0.3257*. Teniendo en cuenta este ejemplo, ¿qué cola de la distribución t usa Calc cuando* `modo = 1`*? Una representación gráfica puede ayudarte con esta pregunta.*

2. *Para confirmar que lo has entendido, usa R para calcular este mismo valor.*

Solución en la página 53. □

¡No sigas, si no has hecho este ejercicio!

Puesto que la distribución T tiene (como la normal estándar Z) una forma de campana simétrica, centrada en el 0, si tomamos un valor situado a la derecha del cero (como el $0,5$ del ejercicio), su cola derecha tiene que valer menos de $\frac{1}{2}$, y su cola izquierda más de $\frac{1}{2}$. Viendo el resultado de Calc en este ejemplo concluimos que (lamentablemente, desde nuestro punto de vista):

> La función `DISTR.T` de Calc, con `modo = 1`, usa la **cola derecha** de la distribución t de Student.

Es decir que el resultado de un comando como:

```
DISTR.T(a; k; 1)
```

es

$$P(T_k > a)$$

Si se usa $modo = 2$ se obtiene el área de ambas colas. Es decir, se calcula

$$P(T_k < -a) + P(T_k > a),$$

o, lo que es lo mismo, $P(|T_k| > a)$).

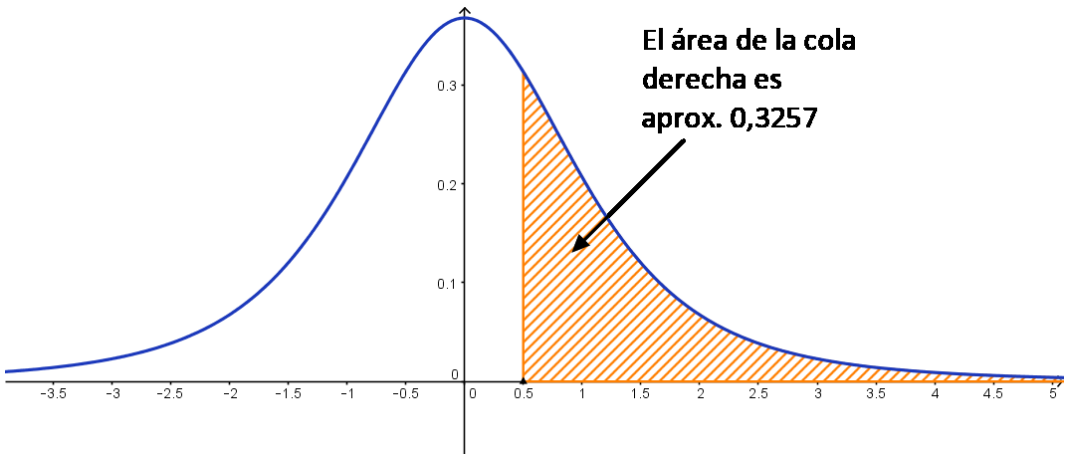
El ejemplo que hemos visto antes, con

```
DISTR.T(0,5; 3; 1) = 0,3257239824
```

corresponde al cálculo de:

$$P\left(T_3 > \frac{1}{2}\right),$$

que se ilustra en esta figura



Si lo que queremos es calcular una cola izquierda, como en $P(T_3 < \frac{1}{2})$, debemos utilizar el truco de $1 - p$:

```
1 - DISTR.T(0,5;3;1) = 0,6742760176
```

Para los problemas inversos de probabilidad y, por lo tanto, para obtener los valores críticos necesarios para los intervalos de confianza, la hoja de cálculo incluye la función `DISTR.T.INV`. Y aquí, de nuevo, hay que ir con cuidado, porque el resultado de

```
DISTR.T.INV(p; k)
```

es el valor x tal que

$$P(T_k < -x) + P(T_k > x) = p,$$

Es decir:

> La función `DISTR.T.INV` siempre usa el área de las dos colas, derecha e izquierda.

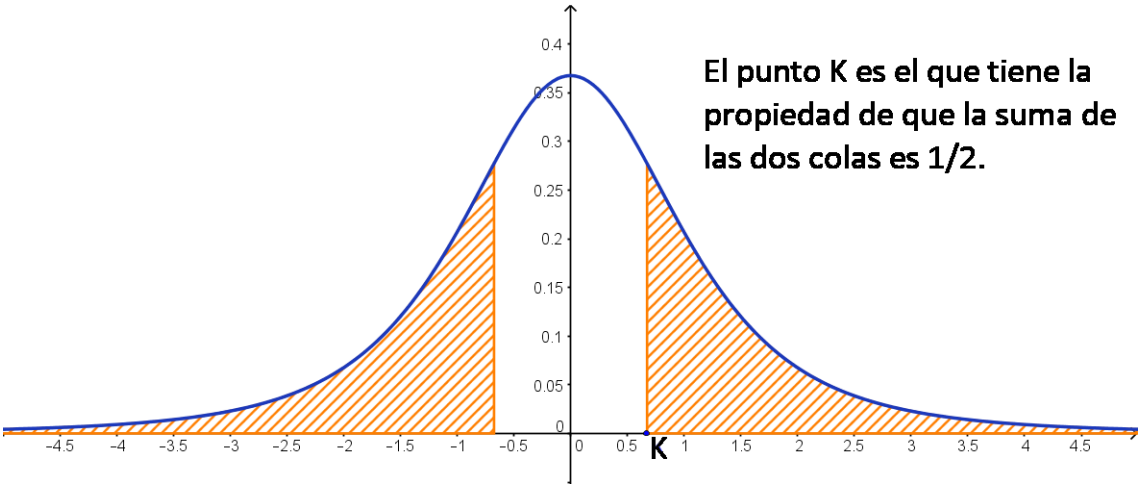
Por ejemplo, para calcular el x tal que

$$\underbrace{P(T_3 < -x)}_{\text{cola izda.}} + \underbrace{P(T_3 > x)}_{\text{cola dcha.}} = \frac{1}{2},$$

usamos

`DISTR.T.INV(0,5; 3) = 0,7648923284`,

El problema se ilustra en esta figura



A pesar de la confusión que, probablemente genera, la ventaja de esto es que, si lo que quiero es encontrar el valor crítico para construir un intervalo de confianza, entonces a partir del nivel de confianza $1-\alpha$, precisamente lo que necesitamos saber es cuál es el valor de K tal que

$$P(X > K) + P(X < -K) = \alpha$$

y eso, directamente, es lo que nos da `DISTR.T.INV`. Por ejemplo, con 3 grados de libertad, el valor crítico para un intervalo de confianza para la media al 95 % (es decir $\alpha = 0.05$), que es $t_{k;\alpha/2} = t_{3;0.025}$, se obtiene con

`DISTR.T.INV(0,05;3)=3,1824463053`,

Como resumen final, nuestra opinión es que la falta de coherencia entre las distintas funciones de Calc (a veces cola izquierda, a veces derecha, a veces las dos...) aconseja ser muy prudentes a la hora de usar el programa. Afortunadamente, creemos haber presentado suficientes alternativas como para hacer esencialmente innecesario el uso de Calc en estos problemas.

4. Intervalos de confianza para la media usando t.

Empecemos recordando que la t de Student se usa para calcular intervalos de confianza cuando se cumplen estas condiciones:

$$\begin{cases} \text{(1) La población original es normal (o, al menos, aproximadamente normal).} \\ \text{(2) Desconocemos la varianza de la población } \sigma_X^2. \\ \text{(3) El tamaño de la muestra es pequeño } (n < 30 \text{ sirve de referencia}). \end{cases}$$

La fórmula del intervalo (Ecuación 6.19 del libro, pág. 229) es:

$$\bar{X} - t_{k;\alpha/2}\frac{s}{\sqrt{n}} \leq \mu_X \leq \bar{X} + t_{k;\alpha/2}\frac{s}{\sqrt{n}}.$$

Es decir, es prácticamente igual a la del caso en que usábamos Z, pero sustituyendo $z_{\alpha/2}$ por $t_{k;\alpha/2}$ (donde $k = n-1$, recuérdalo). En esta sección vamos a ver como usar distintos programas para calcular los intervalos de confianza en esos casos.

4.1. Usando R, primera versión: ficheros plantilla.

Para construir intervalos de confianza usando t en R disponemos de dos ficheros "plantilla", casi idénticos a los que vimos para la normal:

Datos resumidos:	Tut06-IntConf-Media-UsandoT-Estadisticos.R
Datos *en bruto*	Tut06-IntConf-Media-UsandoT-MuestraEnBruto.R

El primero de estos ficheros se usa para los ejercicios *" de libro de texto"*, en los que en lugar de usar como punto de partida los datos de la muestra, disponemos de los valores de $n, \bar{X}$ y s, calculados previamente. El segundo, en cambio, es el adecuado cuando partimos de la muestra completa, en un vector de R, o en un fichero `csv` en la página 30 veremos que esa diferencia no es tan relevante).

El uso de estos ficheros es prácticamente idéntico al que vimos en el caso en el que se usaba Z. La única diferencia reseñable es que aquí *siempre* se va a usar s, la cuasidesviación típica muestral, porque se supone que desconocemos σ (¡de otro modo usaríamos Z!). Lo mejor es, simplemente, practicar haciendo algunos ejercicios.

Ejercicio 14.

1. *El* "ejercicio trampa"*, del apartado 2 del Ejercicio 6 (pág. 13). Una muestra de* 10 *elementos de una población normal, tiene cuasidesviación típica* $s = 2.3$*, y una media muestral* $\bar{X} = 132.5$*. Calcula un intervalo de confianza al* 95 % *para la media* μ*.*

2. *Los siguientes* 15 *valores proceden de una población normal. Calcula un intervalo de confianza al* 95 % *para la media de esa población.*

```
3.14, 3.71, 2.77, 4.08, 4.18, 1.51, 3.65, 3.41, 4.51, 2.27, 3.28,
2.26, 2.80, 2.54, 3.77
```

3. *La t de Student se usa (principalmente) con muestras pequeñas. Por esa razón puede que pienses que aprender a trabajar a partir de ficheros* `csv` *no es, en este caso, tan necesario. Para intentar persuadirte de que puede merecer la pena, aquí tienes un fichero* `csv` *con* 20 *valores procedentes de una distribución normal:*

   Tut06-DatosIntConfConStudent.csv

   Usa R para calcular un intervalo de confianza al 95 % *para estos datos.*

   Yo, desde luego, prefiero leer este fichero usando `read.table` *(y la opción* `dec=","` *para gestionar la coma como separador de decimales). De esa forma, los datos se leen en una sola línea de código en R, en lugar de hacer operaciones de* copia/pega*, reemplazar comas por puntos, etc.*

4. **IMPORTANTE:** *la distribución t de Student es muy útil para trabajar con muestras pequeñas. Pero eso no significa que no pueda usarse con muestras grandes. Para ver lo que sucede, repite el apartado 3 del Ejercicio 6 (en el que* $n = 450$*).*

5. *Comprueba con R la solución del Ejemplo 6.4.1 del libro (pág. 230).*

Solución en la página 53. □

4.2. Usando R, segunda versión: la función `t.test` y similares.

Aparte de las plantillas que hemos descrito en la sección precedente, R incluye una función propia, la función `t.test`, que puede usarse para calcular intervalos de confianza con la t de Student. En realidad, el objetivo principal de esta función es realizar un Contraste de Hipótesis, una técnica que se explica en el Capítulo 7 del libro. Cuando estudiemos ese tema volveremos a visitar esta función `t.test` para conocerla con más detalle (y todavía tendremos ocasión de volver sobre ella más adelante en el curso). Pero, como aperitivo, aquí vamos a ver cómo se puede usar esta función para obtener el intervalo de confianza.

La función `t.test`, como muchas otras funciones de R, está pensada para trabajar directamente sobre un vector de datos. El motivo por el que esto es así es que R se diseñó para un uso profesional,

para tratar los que antes hemos llamado *problemas del mundo real*, para marcar la diferencia con los *problemas de libro de texto*. Recuerda que, en un *problema de libro de texto* partimos de los valores estadísticos ya calculados de la muestra (n, $\bar{X}$, s). Pero en un *problema del mundo real* el punto de partida es un conjunto de datos sin procesar, *en bruto*.

En la Sección previa hemos visto dos ficheros de código R para calcular intervalos de confianza para la media, usando la t de Student. Pues bien, lo que estamos diciendo es que `t.test` se parece al que usa la muestra *en bruto*. Veamos, por ejemplo, cómo usar `t.test` para obtener el intervalo de confianza del apartado 2 del Ejercicio 14.

```
datos=c(3.14,3.71,2.77,4.08,4.18,1.51,3.65,3.41,4.51,2.27,3.28,2.26,2.80,2.54,3.77)
t.test(datos, conf.level=0.95)
```

```
##
##  One Sample t-test
##
## data:  datos
## t = 14.9, df = 14, p-value = 5.7e-10
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  2.7316 3.6524
## sample estimates:
## mean of x
##     3.192
```

De la salida del comando sólo nos interesa, por el momento, la parte en la que aparece el intervalo, que resulta ser $(2.731598, 3.652402)$. Puedes comprobar que este resultado es el mismo que aparece en la solución del Ejercicio 14, con el fichero `Tut06-IntConf-Media-UsandoT-MuestraEnBruto.R`.

Si lo único que queremos es ver cuál es el intervalo, no hay ningún problema en usar la función `t.test` de esta manera. Pero si queremos usar los extremos del intervalo para hacer alguna otra operación con ellos (por ejemplo, ¿cuánto vale la semianchura de ese intervalo?), entonces tenemos que aprender a separar el intervalo del resto de resultados de `t.test` (en el Tutorial05 nos sucedió algo parecido, con el comando para calcular integrales). Afortunadamente, esto es fácil. Basta con guardar el resultado de `t.test` en una variable (la hemos llamado `TtestDatos`):

```
datos=c(3.14,3.71,2.77,4.08,4.18,1.51,3.65,3.41,4.51,2.27,3.28,2.26,2.80,2.54,3.77)
TtestDatos = t.test(datos, conf.level=0.95)
```

y ahora usar la notación `$` para extraer las *piezas* que componen la respuesta de R (en RStudio puedes escribir `TtestDatos$` y pulsar el tabulador para ver la lista de piezas). En particular, el intervalo se extrae usando:

```
TtestDatos$conf.int
```

```
## [1] 2.7316 3.6524
## attr(,"conf.level")
## [1] 0.95
```

(`conf.int` es la abreviatura de *confidence interval*). El resultdo es un vector. No te preocupes por la parte que empieza por `attr`. R a veces añade *atributos* (en inglés, *attributes*) a los vectores, para conservar alguna información relevante sobre ellos. En este caso, el atributo del intervalo es el nivel de confianza que se ha usado para calcularlo. Pero, como decíamos, el resultado es un vector de R que contiene los extremos del intervalo, y con el que podemos trabajar sin problemas. Por ejemplo, para calcular la semianchura hacemos:

```
(intervalo = TtestDatos$conf.int)
```

```
## [1] 2.7316 3.6524
## attr(,"conf.level")
## [1] 0.95
```

```
(extremoInf = intervalo[1])

## [1] 2.7316

(extremoSup = intervalo[2])

## [1] 3.6524

(semianchura = (extremoSup - extremoInf) / 2)

## [1] 0.4604
```

Ejercicio 15. *Usa* `t.test` *para repetir el apartado3 del Ejercicio 14. Solución en la página 54.* □

¿Plantillas o `t.test`, cuál debo usar?

La función `t.test` es muy ágil, y permite obtener respuestas tecleando muy poco. De hecho, una vez definido el vector `datos`, podemos realizar el cálculo del intervalo de confianza en una sola línea:

```
t.test(datos, conf.level=0.95)$conf.int

## [1] 2.7316 3.6524
## attr(,"conf.level")
## [1] 0.95
```

Los usarios experimentados de R trabajan así, porque les permite obtener los resultados muy rápido. Los usuarios *muy experimentados* de R, en cambio, tienden a ser más cuidadosos, y a trabajar aplicando ese sabio refrán que dice "vísteme despacio, que tengo prisa". Pero la sabiduría sólo llega con la experiencia, y cada uno de nosotros va descubriendo con el tiempo y la práctica cuál es la forma de trabajo que más conviene a cada tarea. Te aconsejamos, en tanto que principiante, que incialmente uses las plantillas que te hemos proporcionado. Tienen la virtud de mostrar paso a paso todos los ingredientes de la construcción del intervalo de confianza. De esa forma, si te equivocas en algo (y, al principio, te equivocarás a menudo), es más fácil que puedas localizar el fallo y corregirlo. Cuando ganes en experiencia y seguridad, empezarás a usar `t.test` más a menudo.

¿Y si no tengo el vector de datos? Cocinando muestras.

La función `t.test`, como decimos, permite trabajar de una forma muy ágil en los casos en los que partimos de *datos en bruto*. ¿Pero qué podemos hacer si el punto de partida no es el vector de datos, sino (como sucede a menudo en los ejercicios de *libro de texto*), la media muestral $\bar{X}$, la cuasidesviación s, etc.?

Afortunadamente hay un remedio, que consiste en pedirle a R que, a partir de los valores n, $\bar{X}$, s, *"cocine"* para nosotros una muestra de una población normal que tenga precisamente esas características. Esto se hace usando la función `mvrnorm` de la librería `MASS`. Por ejemplo, en el apartado 1 del Ejercicio 14 hemos hablado de una muestra de una población normal con estas características:

$$n = 10, \qquad , \bar{X} = 132.5, \qquad s = 2.3.$$

El código necesario para fabricar la muestra que queremos con `mvrnorm`, a partir de esos valores, es este:

```
library(MASS)
n = 10
media_muestral = 132.5
s = 2.3
(muestra = as.vector(mvrnorm(n, mu=media_muestral, Sigma=s^2, empirical=T)))
```

```
##  [1] 130.72 134.65 129.01 135.59 133.77 133.80 131.73 128.96 133.62 133.15
```

Usamos `as.vector` para convertir el resultado en un vector, porque `mvrnorm` devuelve como resultado una matriz. Fíjate en que tenemos que utilizar `Sigma=s^2` porque la función espera la cuasivarianza muestral (la notación `Sigma` es desafortunada...). Además el argumento `empirical=TRUE` sirve para garantizar que la muestra tendrá las características deseadas. Comprobémoslo:

```
length(muestra)

## [1] 10

mean(muestra)

## [1] 132.5

sd(muestra)

## [1] 2.3
```

Como ves, R ha cocinado la muestra a la medida de lo que queríamos. Una vez hecho esto, podemos usar la función `t.test` sobre el vector `muestra`, para obtener el intervalo de confianza, como en el caso anterior.

Hay un detalle que nos parece importante aclarar. El lector puede estar preguntándose: ¿no puedo usar `rnorm` para esto? Por ejemplo, haciendo:

```
n = 10
media_muestral = 132.5
s = 2.3
set.seed(2014)
(muestra2 = rnorm(n, mean=media_muestral, sd=s))

##  [1] 131.20 133.24 132.79 135.61 129.54 133.24 133.11 133.42 133.56 137.45
```

Pero esto no sirve, porque si tomas una muestra *aleatoria* de una población con media μ, la media muestral $\bar{X}$ de esa muestra se *parecerá* a μ, pero no será exactamente μ. ¡Al fin y al cabo por eso hemos tenido que empezar el Capítulo 6 del libro estudiando la distribución de la media muestral! Y lo mismo sucede con s, claro. Para verlo en este ejemplo:

```
length(muestra2)

## [1] 10

mean(muestra2)

## [1] 133.32

sd(muestra2)

## [1] 2.1468
```

Como ves, al usar `rnorm` la media muestral y la cuasidesviación típica de la muestra no coinciden con los valores 132.5 y 2.3, respectivamente, que deseábamos. Eso es lo que hace necesario el uso de `mvrnorm` para *"cocinar"* las muestras.

4.3. Intervalos de confianza para la media usando GeoGebra y Wolfram Alpha.

GeoGebra.

En GeoGebra disponemos de la función `IntervaloMediaT`, que se puede usar de esta forma:

```
IntervaloMediaT[ Media Muestral, Cuasidesviación , Tamaño muestra, Nivel de confianza ]
```

o de esta otra

```
IntervaloMediaT[ Lista de datos, Nivel de confianza ]
```

que se corresponden, aproximadamente, como ves, con los dos ficheros "plantilla" de R que hemos visto antes. El uso de ambas versiones de la función no tiene ninguna complicación. Es aconsejable, en el segundo caso, definir previamente una lista que contenga los valores de la muestra, y usar el nombre de esa lista al ejecutar `IntervaloMediaT`.

Ejercicio 16. *Usa estas funciones para repetir los apartados 1 y 2 del Ejercicio 14.* *Solución en la página 54.* □

Wolfram Alpha.

En Wolfram Alpha, para calcular el intervalo de confianza del primer apartado del Ejercicio 14 puedes usar sintaxis como esta (es un ejemplo de la propia página de Wolfram Alpha)

```
t-interval xbar=132.5, s=2.3, n=10, confidence level=0.95
```

Pero también puedes usar algo parecido introduciendo entre llaves los valores de la muestra:

```
t-interval {3.14, 3.71, 2.77, 4.08, 4.18, 1.51, 3.65, 3.41, 4.51, 2.27, 3.28, 2.26, 2.80, 2.54, 3.77}
```

Lo que sucede es que en este caso no resulta tan fácil indicar el nivel de confianza, y Wolfram Alpha usa 95 % por defecto. Naturalmente, siempre podríamos pedirle a Wolfram Alpha que calcule la media y cuasidesviación típica de esa muestra, y entonces usar el primer método... aunque esa solución no resulte muy cómoda (la versión de pago de Wolfram Alpha no tiene estos inconvenientes, dicho sea de paso).

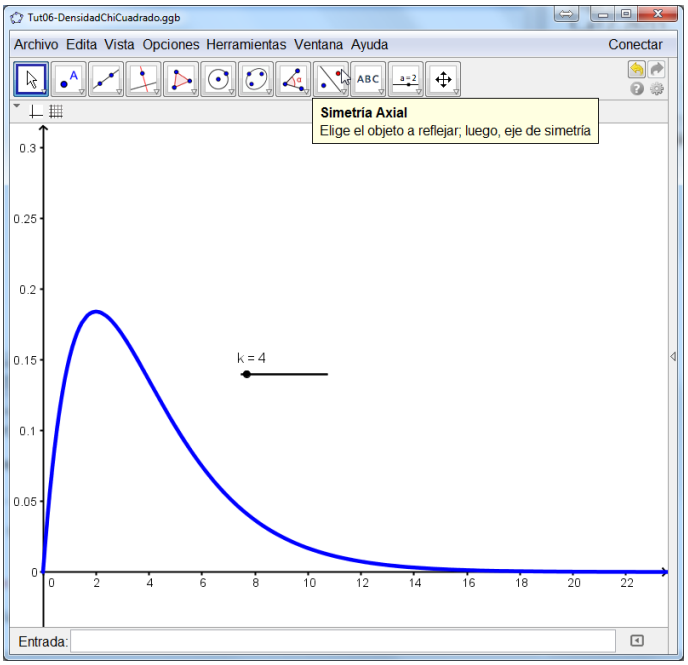
5. La distribución χ^2.

5.1. χ^2 en GeoGebra.

La familia de distribuciones χ^2_k depende de un parámetro k, los grados de libertad, al igual que sucedía con la distribución t de Student. Hemos preparado un fichero GeoGebra para que puedas explorar de forma dinámica cómo son las funciones de densidad de las distribuciones χ^2_k para distintos valores de k:

Tut06-DensidadChiCuadrado.ggb

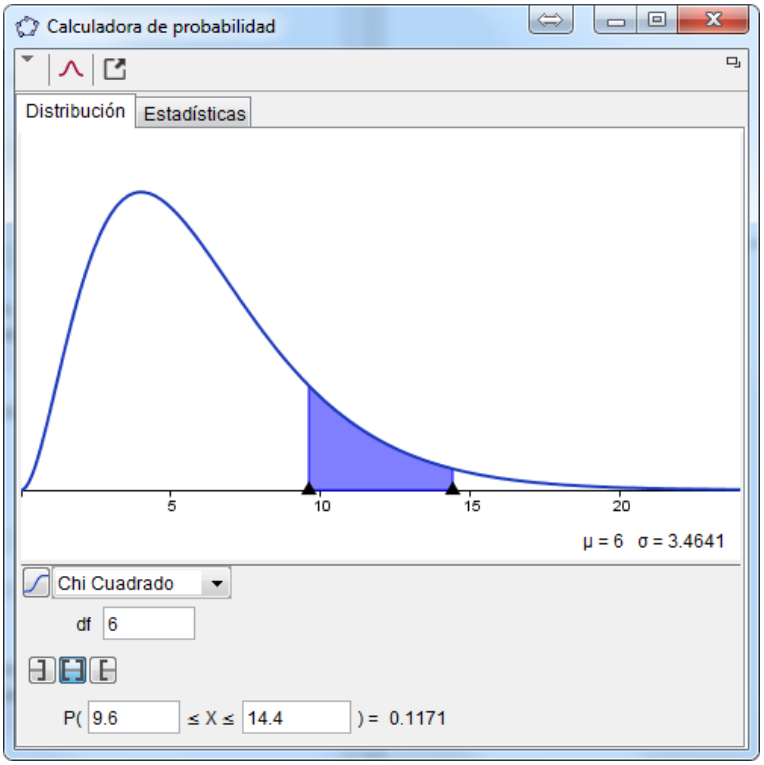
Cuando abras este fichero verás una imagen similar a la de la siguiente figura:



Usa el deslizador para cambiar los valores de k y explorar así esta familia de distribuciones. No dejes de observar el comportamiento cuando $k = 1$, que es excepcional dentro de la familia. Además de esto, observa el aspecto y posición de la función de densidad cuando k crece (tal vez tengas que hacer zoom para verlo). En particular comprobarás que la media de la distribución χ^2_k aumenta a medida que k aumenta. Esta primera exploración debe servir para recordar dos aspectos básicos que **nunca debemos olvidar** al trabajar con χ^2:

- La distribución sólo es distinta de cero en los valores positivos (una variable χ^2 no toma valores negativos; al fin y al cabo, es un cuadrado).
- En consecuencia, la distribución **no es simétrica** en ningún sentido.

Estas características de la distribución condicionan y complican ligeramente el trabajo que tenemos que hacer. La *Calculadora de Probabilidades* de GeoGebra es un buen punto de partida para empezar a trabajar con esta distribución. La siguiente figura muestra la situación que, por defecto, nos encontramos al abrir la *Calculadora* y seleccionar χ^2 en el menú desplegable.



La mejor manera de adentrarnos en este terreno es con una lista de problemas directos e inversos de probabilidad.

Ejercicio 17. *Sea Y una variable aleatoria de tipo χ^2_9 (con $k = 9$ grados de libertad). Usa la Calculadora de Probabilidades de GeoGebra para calcular estos valores:*

1. $P(Y \leq 2.09)$
2. $P(Y \geq 11.4)$
3. $P(14.7 \leq Y \leq 16.9)$
4. *Valor de y tal que* $P(Y \geq y) = 0.05$.
5. *Valor de y tal que* $P(Y \leq y) = 0.01$.
6. *Valores y_1, y_2 tales que* $P(y_1 \leq Y \leq y_2) = 0.90$ *y además* $P(Y \leq y_1) = P(Y \geq y_2)$.

Soluciones en la página 55. □

Aparte de la *Calculadora de Probabilidades*, en GeoGebra disponemos de la función `ChiCuadrado` para efectuar cálculos de probabilidad, usando como en R la cola izquierda de la distribución. Por ejemplo, la respuesta del apartado 1 del Ejercicio 17 se puede obtener con:

```
ChiCuadrado[9, 2.09]
```

Recuerda que las soluciones se obtienen en la *Vista Algebraica* de GeoGebra. Como ves, los grados de libertad se colocan en el primer argumento de la función. Para resolver problemas inversos dispones de la función `ChiCuadradoInversa`. Con esa función el apartado 5 del Ejercicio 17 se resuelve así:

```
ChiCuadradoInversa[9, 0.01]
```

Ejercicio 18. *Usa estas funciones para comprobar las soluciones de los restantes apartados del Ejercicio 17. Soluciones en la página 58.* □

5.2. χ^2 en R.

En R disponemos, como era previsible, de cuatro funciones llamadas `dchisq`, `rchisq`, `pchisq` y `qchisq`. El papel de cada una de estas cuatro funciones, a estas alturas, empieza a ser evidente a partir de sus nombres: `dchisq` es es la función de densidad (y apenas vamos a usarla en este curso), `pchisq` es la función de distribución (probabilidad de la cola izquierda), `qchisq` proporciona los cuantiles (de nuevo, usando la cola izquierda) y, finalmente, `rchisq` sirve para generar valores aleatorios de una variable de tipo χ^2_k. Los grados de libertad (el valor de k) se indican, en todas estas funciones, con el argumento `df`, como sucedía en la t de Student.

Como ejemplo, vamos a usar `pchisq` para resolver un par de problemas directos de probabilidad asociados con χ^2_k. Por ejemplo, para calcular la probabilidad de esta cola derecha

$$P(\chi^2_{14} > 7)$$

usamos uno de estos dos comandos indistintamente:

```
1 - pchisq(7, df=14)

## [1] 0.93471

pchisq(7, df=14, lower.tail=FALSE)

## [1] 0.93471
```

mientras que para encontrar el valor y tal que

$$P(\chi^2_3 > y) = 0.1$$

usaríamos uno de estos comandos:

```
qchisq(1 -0.1, df=3)

## [1] 6.2514

qchisq(0.1, df=3, lower.tail=FALSE)

## [1] 6.2514
```

Para que practiques el uso de estas funciones, haremos un ejercicio.

Ejercicio 19.

1. *Repite con R el Ejercicio 17.*
2. *Usa R para comprobar los resultados del Ejemplo 6.5.1 del libro (pág. 233)*

Soluciones en la página 58. □

5.3. χ^2 en Wolfram Alpha.

El trabajo con χ^2_k en Wolfram Alpha es muy parecido al que hicimos con la t de Student. Por ejemplo, este problema directo de probabilidad

$$P(10 < \chi^2_{12} < 15)$$

se resuelve con este comando:

```
P[10 < Y < 15] for Y chi-square with 12 dof
```

Y el resultado aproximado es 0.3745. Los problemas directos son muy fáciles de resolver, pero los inversos nos darán algo más de trabajo. Para encontrar, por ejemplo, el valor y tal que

$$P(\chi^2_7 > y) = \frac{7}{13}$$

tenemos que comprender primero que esto es lo mismo que buscar el valor de y tal que

$$P(\chi^2_7 \leq y) = 1 - \frac{7}{13} = \frac{6}{13} \approx 0.46154,$$

y ahora usar este comando en Wolfram Alpha:

```
46.154th percentile for chi-square with 7 dof
```

El resultado aproximado es 6.01105.

Ejercicio 20. *Comprueba estos dos ejemplos con R y GeoGebra. Soluciones en la página 60.* □

6. Intervalos de confianza para la varianza con χ^2.

A la hora de construir un intervalo de confianza la asimetría de la distribución χ^2_k marca una diferencia importante con las otras distribuciones que hemos visto antes. En el trabajo con la Z y la t de Student, podíamos aprovechar la simetría para calcular un único valor crítico, con la seguridad de que si, por ejemplo, $z_{\alpha/2}$ deja una probabilidad igual a $\alpha/2$ a su derecha, entonces el simétrico $-z_{\alpha/2}$ deja una probabilidad $\alpha/2$ a su izquierda. Con χ^2 esto no es así, y tendremos que calcular dos valores críticos, $\chi^2_{k;\alpha/2}$, pero también $\chi^2_{k;1-\alpha/2}$. Esos son los valores necesarios

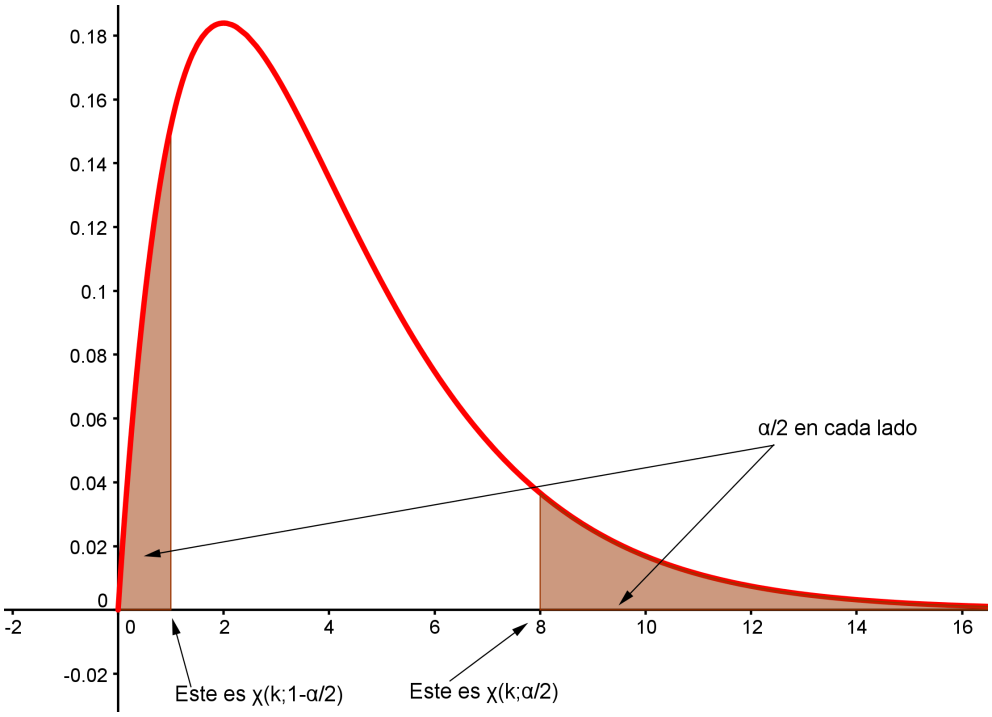
para construir el intervalo de confianza para σ^2 (en poblaciones normales) que hemos visto en la Ecuación 6.24 (pág. 237) del libro:

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

Vamos a ver como construir estos intervalos en varios de los programas que venimos utilizando.

6.1. Intervalo de confianza para σ con R.

Para fijar ideas, supongamos que tenemos una muestra, procedente de una población normal, de tamaño $n = 18$ (de manera que los grados de libertad son $k = n - 1 = 17$), en la que $s = 2.1$, y queremos construir un intervalo de confianza para σ a un nivel de confianza del 99 %. Para seguir la discusión es bueno que tengas presente la Figura 6.15 del libro (pág. 236), que reproducimos aquí por comodidad. ¡Pero atención! Esa figura se corresponde a otro valor de k, así que no esperes que los valores críticos estén donde indica la figura.



Por lo tanto, para comenzar la búsqueda de los valores críticos hacemos en R:

```
n = 17
k = n - 1
s = 2.1

nc = 0.99
(alfa = 1 - nc)

## [1] 0.01

(alfa2 = alfa /2)

## [1] 0.005
```

Para construir el intervalo necesitamos calcular el valor crítico situado más a la derecha, que deja una probabilidad $\frac{\alpha}{2} = 0.005$ en su cola derecha (y $\alpha/2 = 0.995$ en la cola izquierda que usa R), al que hemos llamado:

$$\chi^2_{k;\alpha/2} = \chi^2_{16;0.005}$$

Este valor crítico se obtiene en R con:

```
(chiAlfa2 = qchisq(1 - alfa2, df=k))

## [1] 34.267
```

También necesitamos el otro valor crítico situado más a la izquierda, el que deja una probabilidad $\frac{\alpha}{2} = 0.005$ en su cola izquierda:

$$\chi^2_{k;1-\alpha/2} = \chi^2_{k;0.995}$$

que en R se obtiene con:

```
(chiUnoMenosAlfa2 = qchisq(alfa2, df=k))

## [1] 5.1422
```

Lee varias veces esto si te pierdes. Sabemos que la notación $\chi^2_{k;p}$ es un poco complicada, porque p se refiere a la probabilidad en la cola derecha, y R usa la cola izquierda. Pero es mejor que lo pienses detenidamente, para evitar que este pequeño embrollo notacional te juegue una mala pasada.

Una vez construidos los dos valores críticos, el intervalo es muy fácil de construir. Sólo una última advertencia: el valor crítico situado a la *derecha* se usa en realidad para calcular el extremo *izquierdo* del intervalo, y viceversa. En cualquier caso, si alguna vez te equivocas, los extremos del intervalo se intercambiarían, y eso te serviría de advertencia de que debes revisar tus cuentas para comprobar que no se debe a ningún otro error.

La construcción del intervalo se realiza mediante una operación vectorial de R, en la que obtenemos a la vez los dos extremos:

```
(intervalo = s * sqrt(k / c(chiAlfa2, chiUnoMenosAlfa2)))

## [1] 1.4350 3.7043
```

Así que el intervalo de confianza para σ al 99 % es:

$$1.435 < \sigma < 3.704.$$

Si lo que deseamos es un intervalo de confianza para la varianza σ^2 basta con elevar al cuadrado la desigualdad:

$$2.059 < \sigma^2 < 13.72.$$

Ejercicio 21. *Usa R para comprobar los resultados del Ejemplo 6.5.2 del libro (pág. 238)* *Solución en la página 60.* □

Ficheros "plantilla" de R para intervalos de confianza para σ.

Hemos hecho el cálculo anterior paso a paso para que puedas entender cómo se calculan estos intervalos de confianza en R. Pero, como sucede siempre, una vez que se ha entendido esto, lo mejor es tratar de automatizar el proceso de construcción de los intervalos, y para conseguirlo incluimos dos ficheros "plantilla" de código R, que apenas necesitan presentación. El primero de ellos es para *problemas de libro de texto*, y el segundo para trabajar a partir de *datos en bruto*.

Tut06-IntConf-DesvTipica-PoblNormal-Estadisticos.R

Tut06-IntConf-DesvTipica-PoblNormal-MuestraEnBruto.R

Abre los dos ficheros y estudia su código. Verás, entre otras cosas, que se obtienen dos intervalos, uno para σ^2 y otro para σ. Vamos a practicar el uso de estos ficheros con algunos ejercicios.

Ejercicio 22.

1. *Usa estos ficheros para repetir la construcción del intervalo que hemos usado como ejemplo al principio de la Sección 6.1.*

2. *Una muestra de tamaño $n = 87$ de una población normal tiene una cuasidesviación típica igual a $s = 2.81$. Calcula intervalos de confianza al 95 % para la varianza y desviación típica de esa población.*

3. *El fichero adjunto*

   Tut06-datosIntConfDesvTipica.csv

   contiene una muestra con datos procedentes de una distribución normal. Calcula intervalos de confianza al 95 % para la varianza y desviación típica de esa población.

Soluciones en la página 61. □

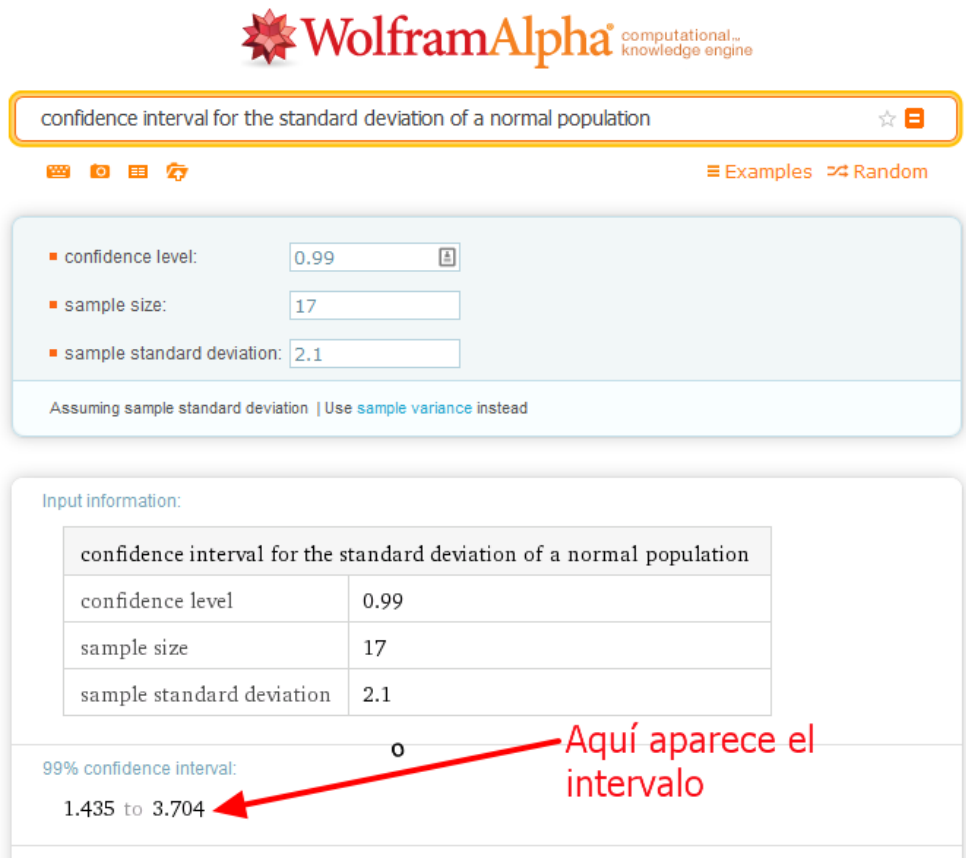
6.2. Intervalo de confianza para σ con GeoGebra y Wolfram Alpha.

6.2.1. Wolfram Alpha.

Para calcular intervalos de tipo *libro de texto* en Wolfram Alpha, puedes teclear esto en la línea de entrada del programa:

```
confidence interval for the standard deviation of a normal population
```

Al hacerlo, aparecerá un cuadro de diálogo en el que puedes introducir los valores necesarios para el cálculo del intervalo. En la siguiente figura se ilustra el cálculo del intervalo de confianza que hemos usado como primer ejemplo para R:



Si lo que tienes es la muestra en bruto, tendrás que calcular los valores necesarios (n y s, en realidad) por algún otro procedimiento, antes de poder usar Wolfram Alpha.

6.2.2. GeoGebra.

GeoGebra no ofrece, en su versión actual (la 4.4 en el momento de escribir esto) ninguna función específica para calcular estos intervalos. Naturalmente, dado que podemos calcular los cuantiles de

χ^2_k, siempre es posible hacer la construcción paso a paso, como hicimos al empezar la discusión con R.

Ejercicio 23. *Repite esa construcción paso a paso usando GeoGebra. Es decir, suponiendo que hemos tomado una muestra procedente de una población normal, de tamaño $n = 18$ y en la que $s = 2.1$, usa las funciones de GeoGebra para construir un intervalo de confianza para σ a un nivel de confianza del 99 %. Solución en la página 62.* □

7. Las funciones de la librería `asbio` de R.

En la Sección 4.2 (pág. 28) hemos visto que la función `t.test` de R se puede usar para obtener intervalos de confianza en el caso de muestras pequeñas. Pero, a diferencia de lo que sucede con la t de Student, R no incluye, en la instalación básica, funciones que permitan calcular intervalos de confianza para la media usando Z, o para σ^2 usando χ^2. Hay, sin embargo, varias librerías que ofrecen ese tipo de fórmulas. Nosotros vamos a utilizar una, bastante completa, llamada `asbio`, creada por Ken Aho. Puedes encontrar información sobre esta librería en el enlace:

http://cran.r-project.org/web/packages/asbio/index.html

en el que encontrarás, entre otras cosas el manual en formato pdf.

Recuerda que antes de usar la librería `asbio` debes instalarla (en la Sección 8 del Tutorial03 se explica como instalar librerías adicionales en R). Después de instalarla, para usarla debes cargarla en la memoria del ordenador utilizando el comando:

```
    library(asbio)

## Loading required package: tcltk
```

La librería `asbio` incluye muchas funciones, pero las tres que más nos interesan en este momento son `ci.mu.z`, `ci.mu.t` y `ci.sigma`. Todas ellas permiten calcular intervalos de confianza (`ci` proviene del inglés *confidence interval*). Concretamente:

- `ci.mu.z` calcula intervalos de confianza para la media usando Z.
- `ci.mu.t` calcula intervalos de confianza para la media usando t de Student. Es decir, lo mismo que antes hemos visto con `t.test`.
- `ci.sigma` calcula intervalos de confianza **para la varianza** σ^2 (¡y no para σ, a pesar del nombre!) usando χ^2.

Y en todos los casos podemos partir de un vector (datos *en bruto*), o de los estadísticos que resumen la muestra (los que hemos llamado *problemas de libro de texto*). Para distinguir en qué caso estamos, basta con utilizar el argumento booleano `summarized`: si es `summarized=TRUE` usamos los valores resumidos (n, $\bar{X}$, s o σ), mientras que si es `summarized=FALSE`, usamos la muestra *en bruto*.

Vamos a ver como usar estas funciones mediante algunos ejemplos, basados en ejercicios que hemos propuesto anteriormente en este tutorial.

Ejemplos con `ci.mu.z`

En el apartado 3 del Ejercicio 6 (pág. 13) teníamos una muestra de 450 elementos de una población normal, con cuasidesviación típica $s = 2.3$, y una media muestral $\bar{X} = 132.5$. Para calcular un intervalo de confianza al 99 % para la media μ, teniendo en cuenta el tamaño de la muestra podemos usar la distribución normal Z. Podemos por tanto utilizar la función `ci.mu.z`, con la opción `summarized=TRUE`, de esta manera:

```
(intAsBio = ci.mu.z(n=450, sigma=2.3, xbar=132.5, conf=0.99, summarized=TRUE))
```

```
##
## 99% z Confidence interval for population mean
## Estimate     0.5%    99.5%
##   132.50   132.22   132.78
```

Compara esto con lo que obtuvimos en el Ejercicio 6. La respuesta de la función `ci.mu.z` es, en realidad, una *lista* de R (puedes usar `class` para comprobarlo). Ya nos hemos encontrado antes con este tipo de situaciones, en las que para acceder a los resultados tenemos que utilizar el operador `$`. Para saber cuales son los elementos de una lista (o de cualquier otro objero de R), disponemos de la función `str`, que en este caso produce:

```
str(intAsBio)

## List of 4
##  $ margin: num 0.279
##  $ ci    : num [1:3] 132 132 133
##  $ head  : chr "99% z Confidence interval for population mean"
##  $ ends  : chr [1:3] "Estimate" "0.5%" "99.5%"
##  - attr(*, "class")= chr "ci"
```

Nos interesa el elemento `ci` de esta lista. Y podemos acceder a ese elemento con este comando

```
intAsBio$ci

## [1] 132.50 132.22 132.78
```

Así se obtiene un vector con tres componentes. La primera es $\bar{X}$, y las dos últimas son los extremos del intervalo de confianza. Podemos usarlos, por tanto

para, por ejemplo, calcular la semianchura del intervalo, de esta manera:

```
(semianchura = (intAsBio$ci[3] - intAsBio$ci[2]) / 2)

## [1] 0.27928
```

Ahora puedes comprobar que la semianchura es el elemento `margin` de la lista `intAsBio` que hemos obtenido como salida de `ci.mu.z`:

```
intAsBio$margin

## [1] 0.27928
```

El resto de los elementos de `intAsBio` son simplemente rótulos.

Vamos ahora a usar a calcular un intervalo de confianza a partir de datos *en bruto*, usando el fichero `Tut06-IntConf-Media-UsandoZ-datos.csv` del apartado 5 del Ejercicio 6. Esto nos va a servir para descubrir una peculiaridad **muy importante** de esta función. Para ello nos aseguramos de que el fichero `csv` está en el directorio de trabajo de R, y usamos de nuevo `ci.mu.z`, pero ahora con `summarized=FALSE`:

```
muestra = read.table(file="../datos/Tut06-IntConf-Media-UsandoZ-datos.csv", sep=" ", dec=".")[ ,
(intAsBio = ci.mu.z(data=muestra, conf=0.95, summarized=FALSE, sigma = sd(muestra)))

##
## 95% z Confidence interval for population mean
## Estimate     2.5%    97.5%
##   7.2313   7.0932   7.3694
```

Como ves, en el caso de datos *en bruto* hemos tenido que pedirle explícitamente a `ci.mu.z` que use la cuasidesviación típica de la muestra, calculada con `sd`. Si no haces esto, comprobarás que el resultado no es correcto.

Ejercicio 24. *Compruébalo. Solución en la página 63.* □

Ejemplos con `ci.mu.t`

Vamos a usar la función `ci.mu.t` para hacer el apartado 2 del Ejercicio 6. Tenemos una muestra de 10 elementos de una población normal, con media muestral $\bar{X} = 132.5$ y cuasidesviación típica $s = 2.3$. Dado que la muestra es pequeña y desconocemos σ debemos usar la t de Student para calcular un intervalo de confianza al 95 % para la media μ. Lo hacemos así:

```
ci.mu.t(n=10, xbar=132.5, sd=2.3, conf=0.95, summarized=TRUE)


##
## 95% t Confidence interval for population mean
## Estimate     2.5%    97.5%
##   132.50   130.85   134.15
```

Puedes comprobar que el resultado coincide con lo que obtuvimos en la solución del Ejercicio 14 (pág. 53) usando `t.test`.

Si partes de una muestra *en bruto*, como la muestra

```
3.14, 3.71, 2.77, 4.08, 4.18, 1.51, 3.65, 3.41, 4.51, 2.27, 3.28,
2.26, 2.80, 2.54, 3.77
```

del apartado 2 del Ejercicio 14, entonces para calcular un intervalo de confianza al 95 % para μ haremos:

```
muestra = c(3.14, 3.71, 2.77, 4.08, 4.18, 1.51, 3.65, 3.41, 4.51, 2.27, 3.28,
    2.26, 2.80, 2.54, 3.77)
(intAsBio = ci.mu.t(data=muestra, conf=0.95, summarized=FALSE))


##
## 95% t Confidence interval for population mean
## Estimate     2.5%    97.5%
##   3.1920   2.7316   3.6524
```

Fíjate en que, en este caso, no ha sido necesario usar la opción `sigma = sd(muestra)` para obtener el resultado correcto, a diferencia de lo que sucede con `ci.mu.z`. Además, si la incluyes, desafortunadamente la función devuelve el error `unused argument`.

Ejemplos con `ci.mu.sigma`

Vamos a empezar usando la función `ci.mu.sigma` para repetir el apartado 2 del Ejercicio 22 (pág. 37 y solución en la página 61). En ese ejercicio teníamos una muestra de tamaño $n = 87$ de una población normal, con cuasidesviación típica igual a $s = 2.81$ y queríamos calcular un intervalo de confianza al 95 % para la varianza de esa población. Lo podemos hacer así:

```
ci.sigma(n=87, S.sq = 2.81^2, conf=0.95, summarized=TRUE)


##
## 95% Confidence interval for population variance
## Estimate     2.5%    97.5%
##   7.8961   5.9807  10.9107
```

Atención: El argumento `S.sq` es la cuasivarianza, no la cuasidesviación típica. Por eso hemos tenido que elevar `2.81` al cuadrado. Y el intervalo que se obtiene es para σ^2, no para σ (a pesar del nombre de la función).

Ejercicio 25. *Usa* `ci.sigma` *para calcular el intervalo de confianza a partir de una muestra* en bruto *(fichero* `csv`*) que aparece en el apartado 3 del Ejercicio 22 (pág. 37). Solución en la página 63.* □

Con esto hemos concluido nuestra primera visita a la librería `asbio`. Volveremos sobre ella más adelante, en futuros tutoriales, y aprenderemos a usar otras funciones de esta librería.

8. Ejercicios adicionales y soluciones.

Ejercicios adicionales.

Ejercicio 26. Simulaciones sobre la interpretación probabilista de los intervalos de confianza.
En este ejercicio vamos a usar R para volver sobre la interpretación probabilística de los intervalos de confianza que vimos, usando GeoGebra, en la Sección 2.2 de este tutorial (pág. 16). Para hacerlo vamos a construir algunas simulaciones.

1. *Empieza por usar* `rnorm` *para generar un número grande (por ejemplo* $N = 1000$*) de muestras de una población norma, cada una de tamaño por ejemplo* $n = 100$*. Puedes usar la normal estándar* $Z = N(0, 1)$ *para esto, el resultado no dependerá de la media o desviación típica de la población. Para cada una de esas muestras, calcula el correspondiente intervalo de confianza* (a, b) *para la media* μ *de la población. En este paso, te puede resultar muy conveniente usar la librería* `asbio`*, como se explica en la Sección 7 (pág. 39). Comprueba, para cada intervalo, si la media real de la población (que es* 0 *si usas* Z*) pertenece al intervalo. Y finalmente, responde a la pregunta: ¿qué porcentaje de los intervalos que has construido contiene a* μ*?*

2. *Repite la anterior simulación con muestras de tamaño* n *mucho menor (por ejemplo de menos de* 10 *elementos), todavía usando* Z *para el cálculo del intervalo.*

3. *Ahora repite la cuenta para las muestras pequeñas usando la* t *de Student. ¿Cambia el porcentaje? Asegúrate de que las muestras sean las mismas, por ejemplo usando* `set.seed` *(o guardándolas en una matriz antes de construir los intervalos con* Z *o* t*).*

4. *Finalmente, en todos los casos anteriores hemos trabajado con poblaciones normales (generadas con* `rnorm`*). Usa ahora* `runif` *para generar muestras pequeñas de una variable aleatoria uniforme (puedes usar el intervalo* $(-1, 1)$ *para que la media sea de nuevo* 0*) y repite el proceso de los anteriores apartados.*

□

Ejercicio 27. Bootstrap. *La técnica que vamos a investigar en este ejercicio se denomina* bootstrap*. En la forma que vamos a ver aquí, esta técnica nos proporciona un método alternativo para construir un intervalo de confianza para la media de una población, a partir de una muestra.*

Originalmente la palabra inglesa bootstrap *denominaba a las correas de cuero que se colocaban en la parte alta de unas botas y que servían para como ayuda para calzarse las botas, tirando de dichas correas. De ahí derivó la expresión popular* pull yourself up by your bootstraps *que podemos tratar de traducir como* elevarte tirando de tus botas *y que representa la idea de conseguir algo mediante el esfuerzo propio, a menudo de forma inesperada. Es ese carácter de conseguir algo de forma inesperada lo que seguramente hizo que se eligiera ese nombre para esta técnica, que podemos describir como una técnica de remuestreo.*

Vamos a utilizar los mismos 15 *valores que aparecían en el apartado 2 del Ejercicio 14. Esos valores eran:*

```
3.14, 3.71, 2.77, 4.08, 4.18, 1.51, 3.65, 3.41, 4.51, 2.27, 3.28,
2.26, 2.80, 2.54, 3.77
```

En aquel ejercicio calculamos un intervalo de confianza para la media usando la fórmula 6.10 (pág. 216) del libro. El resultado era el intervalo:

$$2.732 < \mu < 3.652$$

Vamos a usar el método bootstrap para calcular un intervalo de confianza alternativo para μ. El método consiste simplemente en:

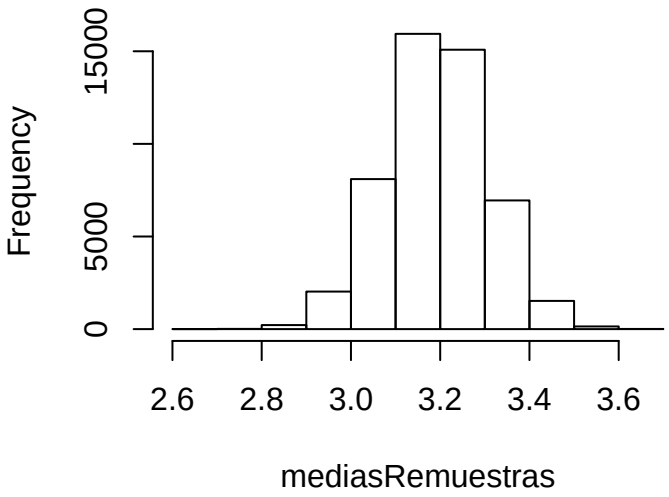
1. *Tomar un número muy grande (miles) de* remuestras*, es decir de muestras aleatorias de la muestra original obtenidas (y esto es muy importante)* **con reemplazamiento**. *El tamaño de las remuestras es un parámetro del método. Se suelen emplear valores como* 20 *o* 30.
2. *Para cada una de esas remuestras calculamos la media, obteniendo un vector con miles de medias de remuestras.*
3. *Si queremos un intervalo de confianza bootstrap al 95 %, nos quedamos con el intervalo definido por los percentiles 0.025 y 0.975 de ese vector de medias de remuestras (la generalización para otros niveles de confianza debería ser evidente).*

En R el código sería este:

```
set.seed(2014)
muestra = c(3.14, 3.71, 2.77, 4.08, 4.18, 1.51, 3.65, 3.41, 4.51, 2.27,
3.28,    2.26, 2.80, 2.54, 3.77)

mean(muestra)

## [1] 3.192

tamannoRemuestra = 50

numRemuestras = 50000

mediasRemuestras = numeric(10000) # este vector almacenara las medias de las remuestras

for(i in 1:numRemuestras){
  # muestras de tamaño tamannoRemuestra y con reemplazamiento
  remuestra = sample(muestra, size = tamannoRemuestra, replace = TRUE)
  mediasRemuestras[i] = mean(remuestra)
}
# El intervalo bootstrap para nc=0.95 es:
quantile(mediasRemuestras, probs = c(0.025, 0.975))

##   2.5%  97.5%
## 2.9698 3.4148
```

Para entender el método puede ayudar la visualización de la distribución de las medias de las remuestras:

```
hist(mediasRemuestras, breaks = 10, main="")
```



Como puede verse, las medias de las remuestras se concentran alrededor de la media de la muestra original, con una distribución que sin ser normal, recuerda bastante a la distribución de la media muestral que aparece en el Teorema Central del Límite.

Ahora es tu turno:

1. *Modifica el código en R para calcular intervalos bootstrap al 90 %, al 99 % o, más en general, para cualquier nivel de confianza dado.*
2. *Modifica el valor de* `tamannoRemuestra`*, haciéndolo más grande. Prueba con tamaños* $30, 40, 50$*. ¿Qué sucede con la anchura del intervalo bootstrap?*
3. *Usa* `rbinom` *para fabricar una muestra de* 100 *valores de una variable binomial* $X \sim B(35, 1/7)$*. A continuación construye un intervalo de confianza para la media* μ_X *usando el método bootstrap. Fíjate en que en este caso conocemos la media de la población. Construye también un intervalo de confianza usando* Z *como hemos visto en el libro. Compara los dos intervalos. ¿Cuál es más ancho? Ya sabemos que el intervalo que usa* Z *está centrado en* $\bar{X}$*. El intervalo bootstrap, en cambio, no es simétrico con respecto a* $\bar{X}$*. ¿Hacia que lado está sesgada la distribución de valores de las medias de remuestras en este caso? Compara la respuesta con el sesgo de la propia variable binomial.*

□

Soluciones de algunos ejercicios.

• Ejercicio 2, pág. 6

La única modificación necesaria es la línea de código

```
n = 4
```

Todo lo demás funciona exactamente igual. Es importante que te esfuerces en escribir código lo más reutilizable posible. En cuanto al número de muestras de tamaño $n = 20$, son

$$36^{20} = 1336749453884373406783884597657 6,$$

del orden de 10^{31}. Otro ejemplo de la *Explosión Combinatoria*, de la que ya hemos hablado.

- **Ejercicio 3, pág. 8**

1. El número de 4-muestras (con reemplazamiento) es simplemente

$$20^4 = 160000.$$

   Si lo quieres pensar usando Combinatoria, son las variaciones con repetición de 20 elementos, tomados de 4 en 4.

2. Para construir las muestras, ver el tamaño del espacio muestral y el aspecto de las primeras y últimas muestras, hacemos:

```
n = 4
MuestrasW = permutations(20, n, repeats.allowed=T)
dim(MuestrasW)

## [1] 160000      4

head(MuestrasW)

##      [,1] [,2] [,3] [,4]
## [1,]    1    1    1    1
## [2,]    1    1    1    2
## [3,]    1    1    1    3
## [4,]    1    1    1    4
## [5,]    1    1    1    5
## [6,]    1    1    1    6

tail(MuestrasW)

##           [,1] [,2] [,3] [,4]
## [159995,]   20   20   20   15
## [159996,]   20   20   20   16
## [159997,]   20   20   20   17
## [159998,]   20   20   20   18
## [159999,]   20   20   20   19
## [160000,]   20   20   20   20
```

- **Ejercicio 4, pág. 10**

1. Para calcular $\mu_{\bar{W}}$ y $\sigma^2_{\bar{W}}$ hacemos:

```
(mu_barW = mean(mediasMuestralesW))

## [1] 10.5

(sigma2_barW = sum((mediasMuestralesW - mu_barW)^2) / length(mediasMuestralesW))

## [1] 8.3125
```

2. Primero calculamos la media y varianza de W hacemos (para entender el vector `probabilidades`, recuerda que los valores son equiprobables):

```
probabilidades = 1 / length(W)
(muW = sum(W * probabilidades))

## [1] 10.5

(sigma2_W = sum((W - muW)^2 * probabilidades ))

## [1] 33.25
```

Ahora se tiene:

```
sigma2_W / sigma2_barW

## [1] 4
```

- **Ejercicio 5, pág. 10**

1. La solución es:

```
pnorm(1.5) - pnorm(-2)

## [1] 0.91044
```

2. La solución es:

```
qnorm(1 - 0.25)

## [1] 0.67449
```

- **Ejercicio 6, pág. 13**

1. Las condiciones se cumplen porque, aunque la muestra es pequeña, se supone que conocemos el valor poblacional de σ. Vamos a mostrar como calcular el intervalo al 90 %. Usa el fichero `Tut06-IntConf-Media-UsandoZ-Estadisticos.R` e introduce estos valores en las líneas del código que se indican:

   a) `n = 10` en la línea 19 del código.

   b) `xbar = 132.5` en la línea 22.

   c) `s = 2.3` en la línea 34. Este valor corresponde al valor poblacional de σ, aunque usemos `s` para representarlo en el código.

   d) `nc = 0.95` en la línea 37.

   El resultado que produce el código es (sólo mostramos las líneas finales):

```
> ################################################
> # NO CAMBIES NADA DE AQUI PARA ABAJO
> ################################################
> (alfa = 1 - nc )
[1] 0.1
>
> # Calculamos el valor critico:
>
> (z_alfa2 = qnorm( 1 - alfa / 2 ) )
[1] 1.645
```

```
>
> #y la semianchura del intervalo
> (semianchura=z_alfa2 * s / sqrt(n) )
[1] 1.196
>
> # El intervalo de confianza (a,b) para mu es este:
> (intervalo = xbar + c(-1, 1) * semianchura )
[1] 131.3 133.7
```

Así que el intervalo de confianza al 90 % es $\mu = 132.5 \pm 1.196$ o, de otro modo:

$$131.3 < \mu < 133.7$$

Te dejamos que compruebes que el intervalo al 95 % es:

$$131.1 < \mu < 133.9, \text{ es decir } \mu = 132.5 \pm 1.426,$$

mientras que al 99 % es

$$130.6 < \mu < 134.4, \text{ es decir } \mu = 132.5 \pm 1.873,$$

Fíjate en que, con los mismos datos, a medida que queremos un nivel de confianza más alto, el intervalo tiene que ser más ancho, para intentar atrapar el valor de μ.

2. Este es un *"ejercicio trampa"*. Las condiciones no se cumplen, porque la muestra es demasiado pequeña y desconocemos la desviación típica poblacional. ¡No debe hacerse usando Z! Es necesario usar en su lugar la t de Student, como se explica en la Sección 6.4 del libro (pág. 224).

3. A diferencia del anterior, este ejercicio se puede hacer con Z, porque la muestra es suficientemente grande. Usando el fichero `Tut06-IntConf-Media-UsandoZ-Estadisticos.R` se obtiene sin dificultad este intervalo:

$$132.2 < \mu < 132.8, \text{ es decir } \mu = 132.5 \pm 0.2793,$$

Compáralo con el primer apartado. Fíjate en que los datos son iguales salvo por el tamaño de la muestra. Y al disponer de una muestra mucho más grande, la anchura del intervalo se ha reducido mucho (la precisión ha aumentado), de manera que este intervalo al 99 % resulta más estrecho que el intervalo al 90 % que obtuvimos cuando la muestra era pequeña. La lección que hay que aprender aquí es que, como nos dice la intuición, a mayor tamaño de muestra, más precisión en los resultados.

4. Para hacer este ejercicio debes usar el fichero `Tut06-IntConf-Media-UsandoZ-MuestraEnBruto.R`, copiar el vector de datos, introducir el nivel de confianza y comentar las líneas que no se usarán. Además, puesto que conocemos σ, es **muy importante** que introduzcas ese valor a mano en la variable `s`. En caso contrario, R calculará y usará la cuasidesviación típica, con lo que el resultado no sería correcto (la muestra es demasiado pequeña). Las líneas afectadas del principio del fichero quedarán así:

```
# Una posibilidad es que tengas la muestra como un vector.
muestra =  c(3.09,3.06,2.79,2.44,2.54,3.52,3.07,2.67,2.99,2.82,2.94,3.57,
    2.38,3.24,3.16,3.45,3.24,2.97,3.39,2.97,2.68,2.91,2.84,3.15,3.15)
             # SI NO SE USA, ESCRIBE # AL PRINCIPIO DE ESTA LINEA


# Si lees la muestra de un fichero csv:
# 1. selecciona el directorio de trabajo
# Para eso, escribe el nombre entre las comillas.
# En RStudio puedes usar el tabulador como ayuda.
#    (setwd(dir=""))  # SI NO SE USA, ESCRIBE # AL PRINCIPIO DE ESTA LINEA

# 2. Ahora introduce entre las comillas el nombre del fichero,
# y el tipo de separador, etc.
# SI NO SE USA, ESCRIBE # AL PRINCIPIO DE LA SIGUIENTE LINEA
# muestra = read.table(file=" ", header = , sep=" ", dec=".")
```

```
# LEE ESTAS INSTRUCCIONES ATENTAMENTE:
# Si la muestra tiene mas de 30 elementos
# calculamos el valor de s, la cuasidesviacion tipica de la poblacion.
# Si la poblacion es normal, y conoces sigma, la desviacion tipica de
# la poblacion, puedes CAMBIAR A MANO s por sigma, aunque sea n<30.
#
# SI LA MUESTRA TIENE < 30 ELEMENTOS Y DESCONOCES SIGMA,
# NO USES ESTE FICHERO!!
# ASEGURATE DE HABER ENTENDIDO ESTAS INSTRUCCIONES

(s = 0.5 )    # O sigma, lee las instrucciones.

# y el nivel de confianza deseado.
nc = 0.95
```

Fíjate en que el vector de datos ocupa dos filas de código, pero eso no afecta al correcto funcionamiento del fichero. El resultado, tras ejecutar el código del fichero, es el intervalo:

$$2.805 < \mu < 3.197, \text{ es decir } \mu = 3.001 \pm 0.196.$$

Observa que el valor $\bar{X} \approx 3.001$ en este caso no es un dato. Hay que calcularlo, pero también forma parte de la salida del programa.

5. En este apartado también debes usar el fichero `Tut06-IntConf-Media-UsandoZ-MuestraEnBruto.R`, pero previamente tienes que guardar el fichero de datos en alguna carpeta de tu ordenador. Luego introduce en la línea correspondiente del fichero la ubicación de esa carpeta (en mi caso, verás que he usado como carpeta mi *Escritorio*, en una máquina con Windows 7). Además, introduce el nombre del fichero de datos, y el nivel de confianza deseado. Las líneas afectadas del principio del fichero quedarán así:

```
# Una posibilidad es que tengas la muestra como un vector.
# SI NO SE USA, ESCRIBE # AL PRINCIPIO DE LA SIGUIENTE LINEA
# muestra =

# Si lees la muestra de un fichero csv:
# 1. selecciona el directorio de trabajo

# 2. Ahora introduce entre las comillas el nombre del fichero, y el tipo de separador, etc.
# SI NO SE USA, ESCRIBE # AL PRINCIPIO DE LA SIGUIENTE LINEA.
muestra = read.table(file="../datos/Tut06-IntConf-Media-UsandoZ-datos.csv", header=FALSE,
                     sep=" ", dec=".")[ ,1]


# LEE ESTAS INSTRUCCIONES ATENTAMENTE:
# Si la muestra tiene mas de 30 elementos
# calculamos el valor de s, la cuasidesviacion tipica de la poblacion.
# Si la poblacion es normal, y conoces sigma, la desviacion tipica de
# la poblacion, puedes CAMBIAR A MANO s por sigma, aunque sea n<30.
#
# SI LA MUESTRA TIENE < 30 ELEMENTOS Y DESCONOCES SIGMA,
# NO USES ESTE FICHERO!!
# ASEGURATE DE HABER ENTENDIDO ESTAS INSTRUCCIONES

(s = sd(muestra) )    # O sigma, lee las instrucciones.

# y el nivel de confianza deseado.
nc = 0.95
```

El resultado, tras ejecutar el código del fichero, es el intervalo:

$$7.093 < \mu < 7.369, \text{ es decir } \mu = 7.231 \pm 0.1381.$$

De nuevo, el valor $\bar{X} \approx 7.231$ se obtiene como parte de la salida del programa.

- **Ejercicio 7, pág. 16**

Ejecuta este comando:

```
IntervaloMediaZ[132.5, 2.3, 10, 0.99]
```

- **Ejercicio 8, pág. 19**

El fichero adjunto

Tut06-IntConf-Media-UsandoZ-SolucionEjercicio.ods

muestra como se obtiene la solución.

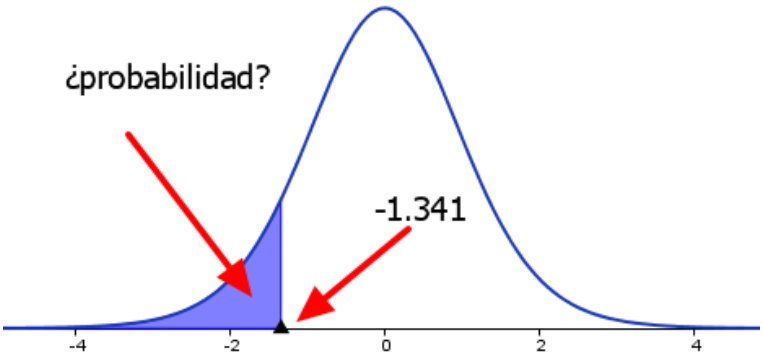
- **Ejercicio 9, pág. 22**

Vamos a guardar en k los grados de libertad para todos los apartados.

```
k = 15
```

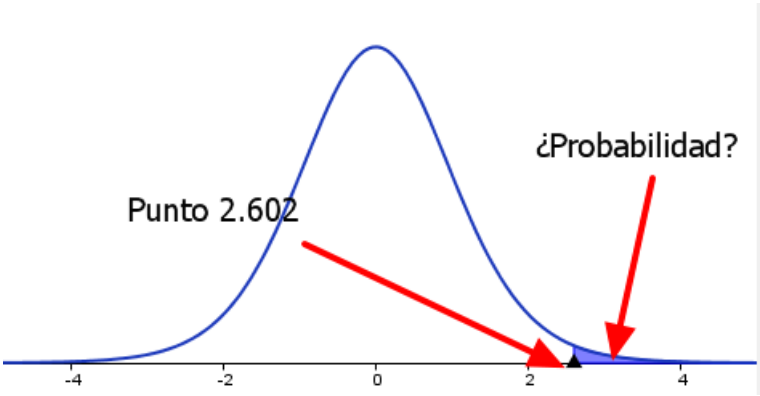
Y ahora vamos con cada uno de los apartados. Por si te has despistado, recuerda que en las distribuciones continuas, a diferencia de lo que sucede por ejemplo en la binomial, no hay ninguna diferencia entre $<$ y $\leq$, o entre $>$ y $\geq$. Los tres primeros ejercicios piden una **P**robabilidad, y por eso usamos `pt`. Los siguientes piden un cuantil, **Q**uantile en inglés, y por eso usamos `qt`.

1.
```
pt(-1.341, df=k)

## [1] 0.099937
```



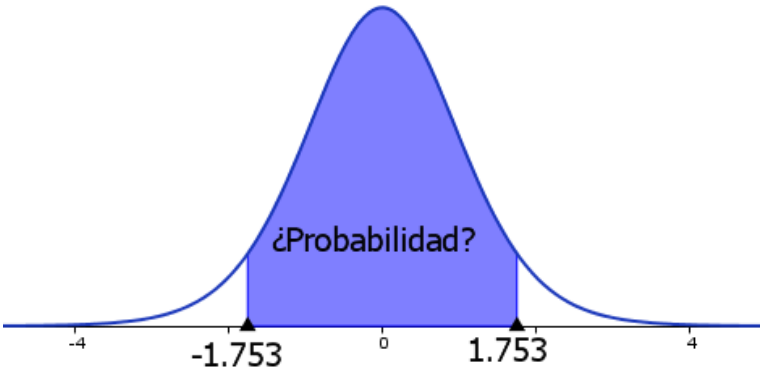
2. Se puede hacer de dos formas:

```
1 - pt(2.602, df=k)

## [1] 0.01001

pt(2.602, df=k, lower.tail=FALSE)

## [1] 0.01001
```



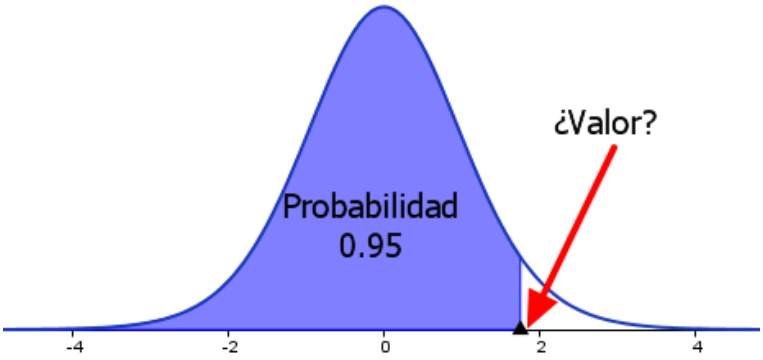
3.
```
pt(1.753, df=k) - pt(-1.753, df=k)
```

```
## [1] 0.89999
```



4.
```
qt(0.95, df=k)
```

```
## [1] 1.7531
```



¿Ves la relación con el apartado anterior? En el apartado anterior las dos colas suman 0.1. Así que. ¿cuál es el área de cada una de las colas? ¿Y cuál es el área de la cola derecha en la figura de este apartado?

5. De dos formas:

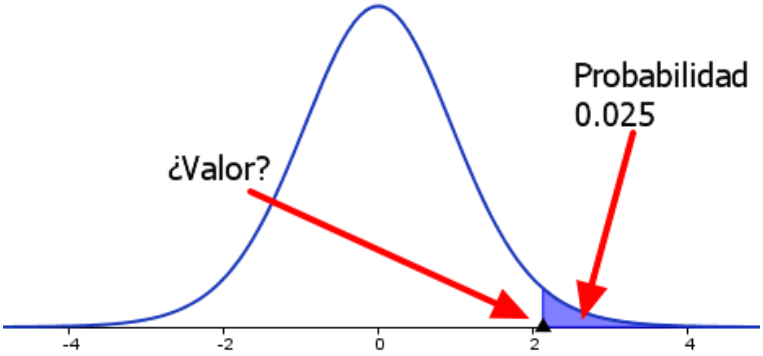
```
qt(0.025, df=k, lower.tail=FALSE)
```

```
## [1] 2.1314
```

```
qt(1 - 0.025, df=k)
```

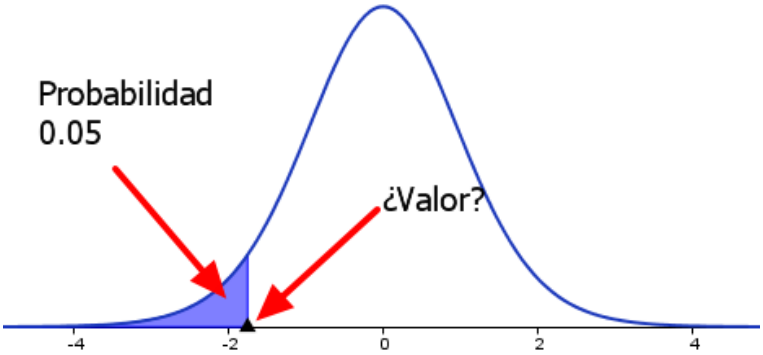
```
## [1] 2.1314
```

En el segundo método fíjate en que siempre que usamos el truco de *restar de* 1, lo que restamos tiene que ser una probabilidad p, de forma que hacemos $1 - p$. Cuando usamos `qt` la probabilidad está *dentro* de la función, y por eso expresiones como `qt(1-p, ...)` tienen sentido. En cambio, una expresión como `1 - qt(...)` es, casi siempre, el resultado de un error de interpretación.



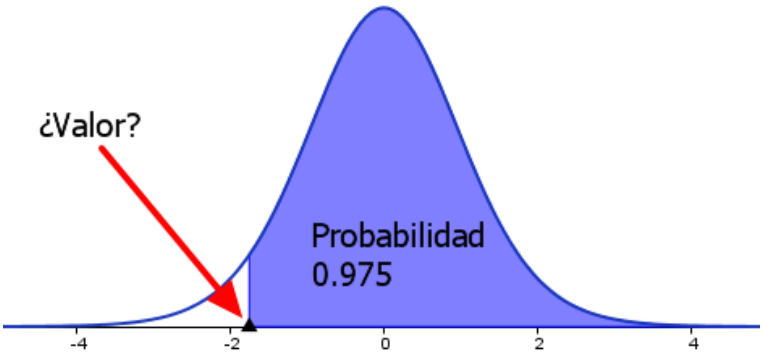
6.
```r
qt(0.05, df=k)

## [1] -1.7531
```

El resultado es el opuesto del que obtuvimos en el apartado 4, y la figura debería dejar claro por qué es así. ¿Cuánto vale la cola derecha en la figura del apartado 4?



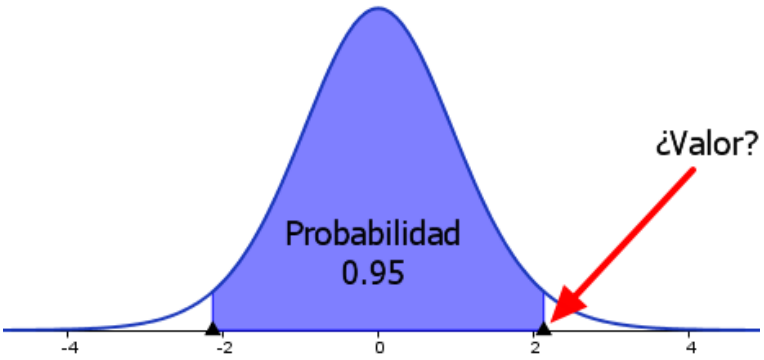
7. De dos formas:

```r
qt(0.975, df=k, lower.tail=FALSE)

## [1] -2.1314

qt(1- 0.975, df=k)

## [1] -2.1314
```



La relación con el apartados 5 debe estar clara a estas alturas.

8. Este apartado es el más directamente se relaciona con la construcción de los intervalos de confianza. Y en este caso es mejor empezar por la figura.



Como muestra la figura, el área conjunta de las dos colas debe ser igual a $1 - 0.95 = 0.05$. Así que el área de la cola derecha no sombreada es 0.025. Y eso demuestra que el valor que buscamos es, por tanto, el que mismo que hemos calculado en el anterior apartado.

9. Dejamos la figura de este apartado al lector, es muy parecida a la del anterior, cambiando sólo los valores numéricos. En cualquier caso, la suma conjunta de las dos colas es, en este apartado, $1 - 0.93 = 0.07$. Por lo tanto la cola derecha vale 0.035. Y entonces el valor se obtiene con

```
qt(0.035, df=k, lower.tail=FALSE)

## [1] 1.9509

qt(1- 0.035, df=k)

## [1] 1.9509
```

• **Ejercicio 11, pág. 23**

En R, el primer valor se obtiene de una de estas dos maneras:

```
pt(2.3, df=21, lower.tail=FALSE)

## [1] 0.015901

1 - pt(2.3, df=21)

## [1] 0.015901
```

Y el segundo mediante:

```
pt(2.3, df=12) - pt(-1.2, df=12)

## [1] 0.85325
```

En GeoGebra, para el primer valor puedes usar la *Calculadora de Probabilidades*, o más directamente el comando:

```
1 - DistribuciónT[21, 2.3]
```

Para el segundo valor usaríamos:

```
DistribuciónT[12, 2.3] - DistribuciónT[12, -1.2]
```

- **Ejercicio 12, pág. 24**

En R, de cualquiera de estas maneras:

```
qt(0.93, df=12)

## [1] 1.5804

qt(0.07, df=12, lower.tail=FALSE)

## [1] 1.5804
```

En GeoGebra con:

```
DistribuciónTInversa[12, 0.93]
```

- **Ejercicio 13, pág. 24**

1. Como se explica en la página que sigue al enunciado de este ejercicio, la función `DISTR.T` de Calc, con `modo = 1`, usa la cola derecha de la distribución t de Student.
2. En R puedes calcular ese valor con:

   ```
   pt(0.5, df=3, lower.tail=FALSE)

   ## [1] 0.32572
   ```

- **Ejercicio 14, pág. 28**

1. El intervalo es $\mu = 132.5 \pm 1.645$ o, de otro modo:

   $$130.9 < \mu < 134.1$$

2. El intervalo es $\mu = 3.192 \pm 0.4604$ o, de otro modo:

   $$2.732 < \mu < 3.652$$

   El valor $\bar{X} = 3.192$ se obtiene como parte de la salida del programa.

3. Después de elegir el directorio de trabajo (el que contiene el fichero), en el paso 2 hemos usado la función `read.table` así:

   ```
   muestra = read.table(file="../datos/Tut06-DatosIntConfConStudent.csv",
   header=FALSE, sep=" ",dec=",")[,1]
   ```

   El intervalo es $\mu = 3.5 \pm 0.5616$ o, de otro modo:

   $$2.939 < \mu < 4.062$$

   De nuevo, el valor $\bar{X} = 3.5$ se obtiene como parte de la salida del programa.

4. Al usar la t de Student con $n = 450$ (y por tanto 449 grados de libertad) se obtiene un intervalo de confianza con semianchura `0.2805`, frente a la semianchura `0.2793` que se obtuvo en el Ejercicio usando Z. La diferencia entre los dos intervalos es del orden de milésimas, así que son prácticamente iguales. Este ejercicio indica que si usamos la t de Student en lugar de Z cuando las muestras son grandes, los resultados serán muy parecidos.

5. Mostramos aquí el código correspondiente al intervalo al 95 %. Es simplemente, el fichero plantilla `Tut06-IntConf-Media-UsandoT-Estadisticos.R`, del que hemos eliminado los comentarios introductorios y las instrucciones. Para hallar el intervalo al 99 % sólo hay que cambiar una línea del código.

```
# Introduce el numero de datos de la muestra,
n = 10

# Introduce aqui el valor de xbar, la media muestral
xbar = 2.35

# Cuasidesviacion tipica muestral
s = 0.61

# y el nivel de confianza deseado.
nc = 0.95

################################################
#NO CAMBIES NADA DE AQUI PARA ABAJO
################################################
(alfa = 1 - nc )

## [1] 0.05

# Calculamos el valor critico:

(t_alfa2 = qt( 1 - alfa / 2 , df=n-1 ) )

## [1] 2.2622

#y la semianchura del intervalo
(semianchura=t_alfa2 * s / sqrt(n) )

## [1] 0.43637

# Y el intervalo de confianza (a,b) para mu es este:
xbar + c(-1, 1) * semianchura

## [1] 1.9136 2.7864
```

- **Ejercicio 15, pág. 30**

Asegúrate, usando por ejemplo `setwd`, de que el directorio de trabajo (en mi caso, es el *Escritorio*) es el que contiene el fichero de datos `Tut06-DatosIntConfConStudent.csv`. Entonces leemos los datos en un vector `muestra`, como hicimos en el Ejercicio 14, y aplicamos `t.test`:

```
muestra = read.table(file="../datos/Tut06-DatosIntConfConStudent.csv", sep=" ", dec=",")[,1]
T.testMuestra = t.test(muestra, conf.level=0.95)
(intervalo = T.testMuestra$conf.int)

## [1] 2.9385 4.0618
## attr(,"conf.level")
## [1] 0.95
```

La respuesta es la misma que en el apartado 3 del Ejercicio 14.

- **Ejercicio 16, pág. 32**

1. Para el primer apartado ejecutamos:

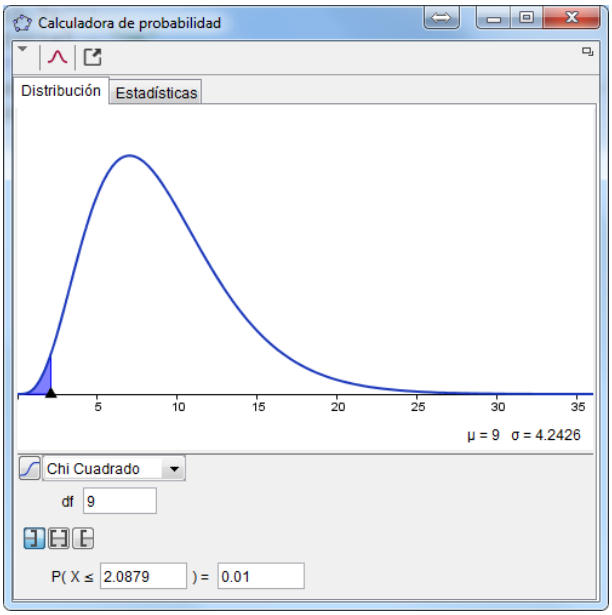
```
IntervaloMediaT[ 132.5, 2.3 , 10, 0.95 ]
```

2. Para el segundo, ejecutamos, consecutivamente:

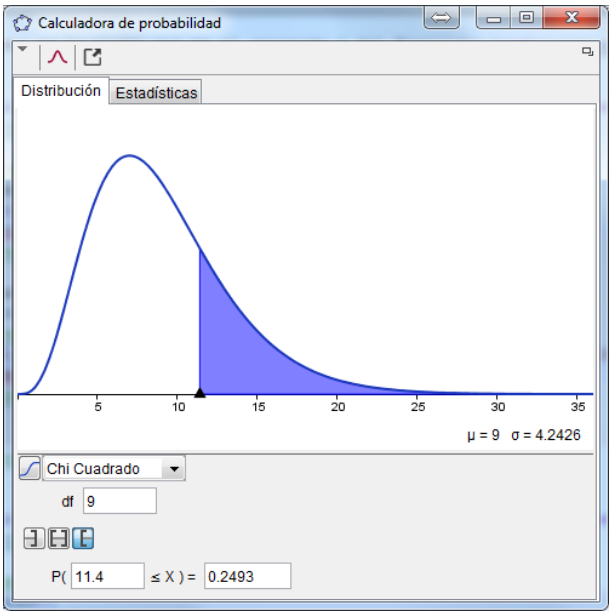
```
datos = {3.14, 3.71, 2.77, 4.08, 4.18, 1.51, 3.65, 3.41, 4.51, 2.27, 3.28,2.26, 2.80, 2.54, 3.77}
IntervaloMediaT[datos, 0.95 ]
```

- **Ejercicio 17, pág. 34**

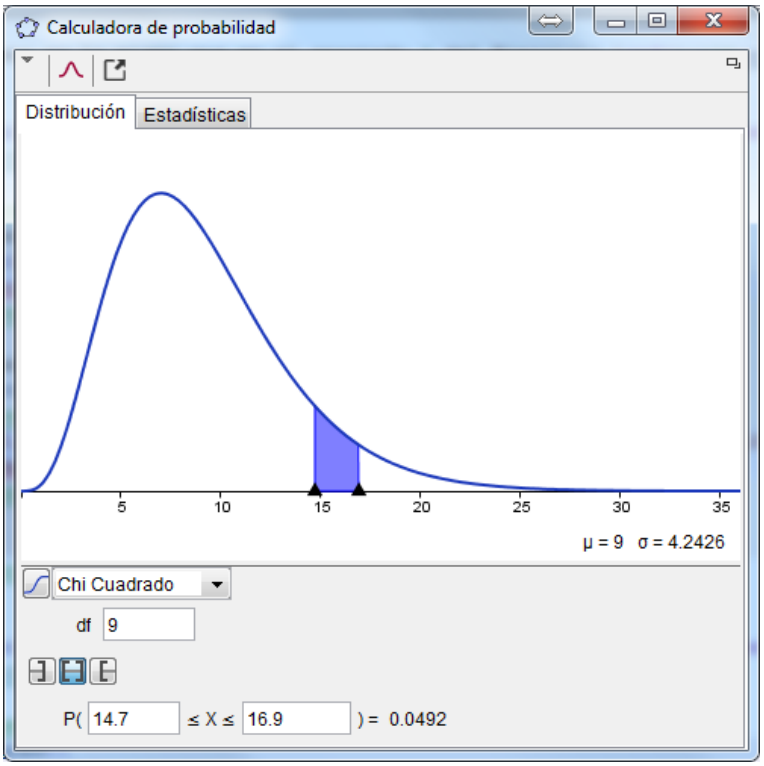
1. La probabilidad es aproximadamente 0.01.



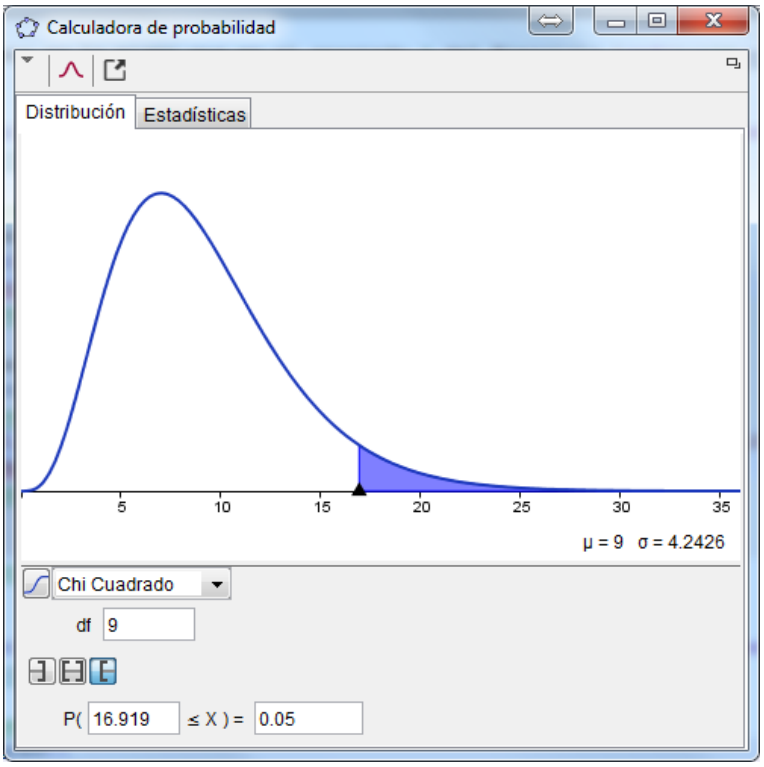
2. La probabilidad es aproximadamente 0.2493.



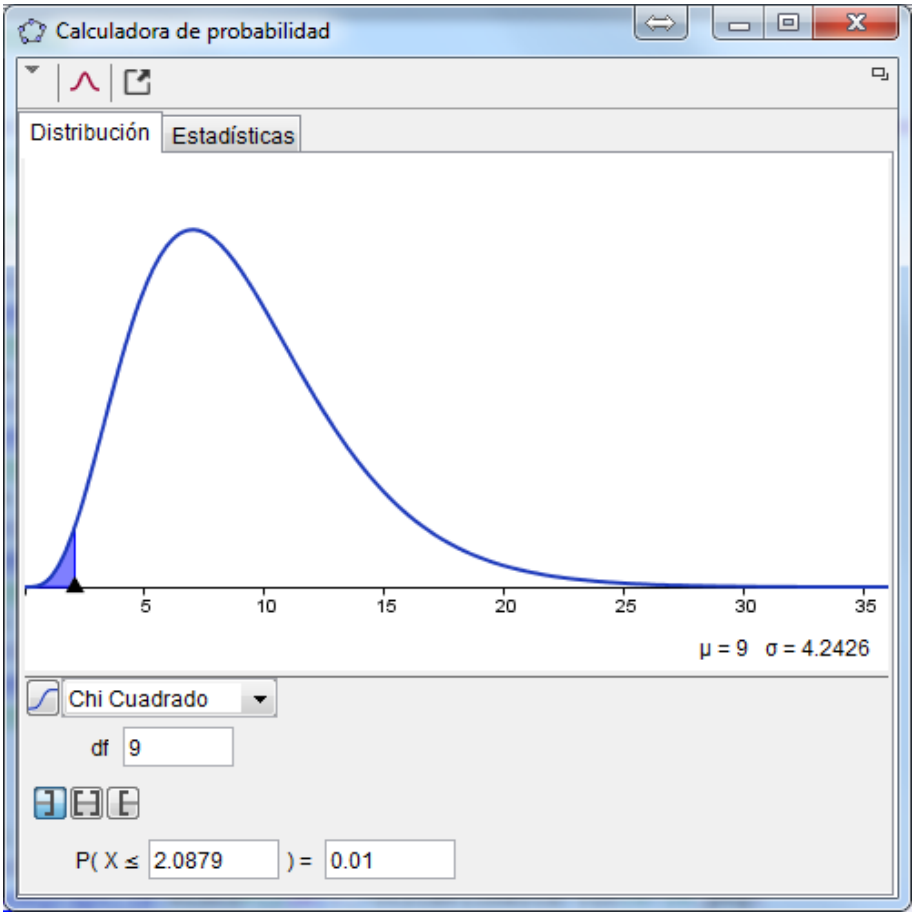
3. La probabilidad es aproximadamente 0.0492.



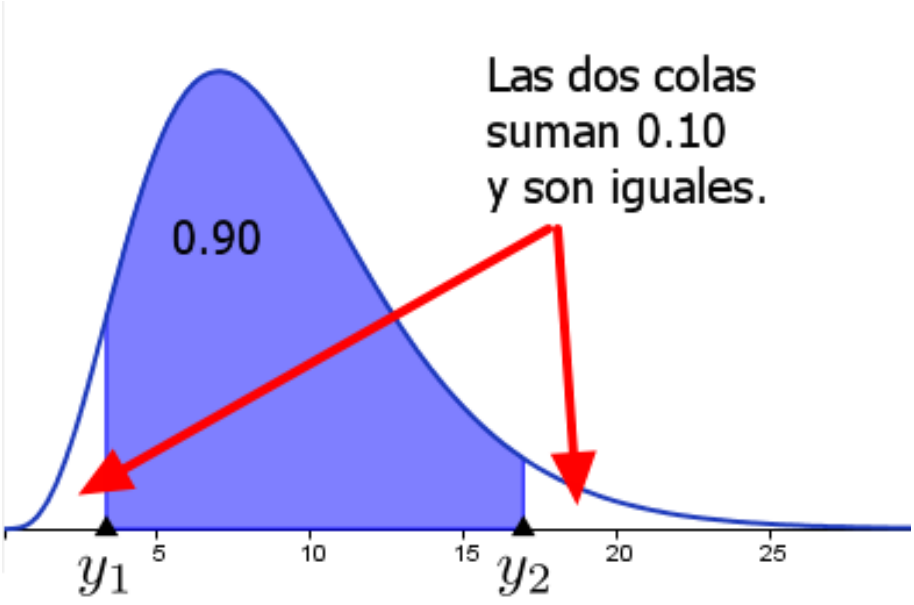
4. El valor es aproximadamente 16.919.



5. El valor es aproximadamente 2.0879



6. El último apartado es el más difícil, a causa de la falta de simetría. Como ilustra la siguiente figura, nuestro objetivo es conseguir que la zona central sombreada, que corresponde a la probabilidad $P(y_1 < Y < y_2)$, tenga un área igual a 0.90. Eso significa que las dos colas suman 0.10. Pero como además, son iguales (es esencial saber esto, porque no hay simetría), eso significa que cada una de ellas vale 0.05.



En particular eso implica que:

$$P(Y < y_2) = 0.90 + 0.05 = 0.95$$

y que:

$$P(Y < y_1) = 0.05$$

Ahora es fácil proceder como en el apartado 1 para obtener $y_1 = 3.3251$ e $y_2 = 16.919$. ¿Ves la relación entre y_2 y el apartado 4 de este ejercicio? ¿Y ves también que no hay ningún tipo de simetría entre y_1 e y_2?

- **Ejercicio 18, pág. 34**

Usa el menú *Opciones* de GeoGebra para ver más cifras decimales si es necesario.

Apartado 2:

```
1- ChiCuadrado[9, 11.4]
```

Apartado 3:

```
ChiCuadrado[9, 16.9] - ChiCuadrado[9, 14.7]
```

Apartado 4: (Atención a la posición del $1 - p$).

```
ChiCuadradoInversa[9, 1- 0.05]
```

Apartado 6: El valor y_1 se obtiene con:

```
ChiCuadradoInversa[9, 0.05]
```

mientras que el valor y_2 se obtiene con:

```
ChiCuadradoInversa[9, 0.95]
```

- **Ejercicio 19, pág. 35**

1. Código del apartado 1:

```
# Fijamos los grados de libertad:

k = 9

# Apartado 1

pchisq(2.09, df=k)

## [1] 0.010037

# Apartado 2

1 - pchisq(11.4, df=k)

## [1] 0.24928

pchisq(11.4, df=k, lower.tail=FALSE)

## [1] 0.24928
```

```r
# Apartado 3

pchisq(16.9, df=k) - pchisq(14.7, df=k)

## [1] 0.049208

# Apartado 4

qchisq(1 - 0.05, df=k)

## [1] 16.919

qchisq(0.05, df=k, lower.tail=FALSE)

## [1] 16.919

# Apartado 5

qchisq(0.01, df=k)

## [1] 2.0879

# Apartado 6

# y1 se obtiene con

qchisq(0.05, df=k)

## [1] 3.3251

# mientras que y2 se obtiene con

qchisq(0.95, df=k)

## [1] 16.919
```

2. Código del apartado 2: El valor a que cumple

$$P(\chi^2_4 < a) = 0.05$$

es:

```r
qchisq(0.05, df=4)

## [1] 0.71072
```

y el valor b que cumple

$$P(\chi^2_4 > b) = 0.05$$

se calcula de una de estas dos formas:

```r
qchisq(1 - 0.05, df=4)

## [1] 9.4877

qchisq(0.05, df=4, lower.tail=FALSE)

## [1] 9.4877
```

• **Ejercicio 20, pág. 35**

En R, para el primer ejemplo hacemos:

```
pchisq(15, df=12) - pchisq(10, df=12)

## [1] 0.37452
```

y para el segundo, uno de estos comandos:

```
qchisq(7/13, df=7, lower.tail=FALSE)

## [1] 6.011

qchisq(1 - (7/13), df=7)

## [1] 6.011
```

En GeoGebra, para el primero ejecuta en la *Línea de Entrada*:

```
ChiCuadrado[12, 15] - ChiCuadrado[12, 10]
```

y para el segundo:

```
ChiCuadradoInversa[7, 1 - (7/13)]
```

• **Ejercicio 21, pág. 37**

El código es este:

```
n = 30
k = n - 1
s = 2.6

nc = 0.95
(alfa = 1 - nc)

## [1] 0.05

(alfa2 = alfa /2)

## [1] 0.025

(chiAlfa2 = qchisq(1 - alfa2, df=k))

## [1] 45.722

(chiUnoMenosAlfa2 = qchisq(alfa2, df=k))

## [1] 16.047

(intervalo = s * sqrt(k / c(chiAlfa2, chiUnoMenosAlfa2)))

## [1] 2.0707 3.4952
```

• **Ejercicio 22, pág. 37**

Para el segundo apartado, el código del fichero con los datos de ese apartado es este:

```
# Introducimos el valor de la desviacion tipica muestral,
s = 2.81

# el tamaño de la muestra,
n = 87

# y el nivel de confianza deseado.
nc = 0.95

################################################
#NO CAMBIES NADA DE AQUI PARA ABAJO
################################################
# Calculamos alfa
(alfa = 1 - nc)

## [1] 0.05

(alfa2 = alfa / 2)

## [1] 0.025

# y los grados de libertad:

(k= n - 1)

## [1] 86

# Calculamos los valores criticos necesarios:

(chiAlfa2 = qchisq(1 - (alfa/2), df=k))

## [1] 113.54

(chiUnoMenosAlfa2 = qchisq(alfa/2, df=k))

## [1] 62.239

#Para la varianza, el intervalo de confianza sera
# extremo inferior
(intervaloVar = s^2 * k / c(chiAlfa2, chiUnoMenosAlfa2))

## [1]  5.9807 10.9107

# Y para la desviacion tipica el intervalo de confianza es este:
(intervaloS = s * sqrt(k / c(chiAlfa2, chiUnoMenosAlfa2)))

## [1] 2.4455 3.3031
```

Para el tercer apartado, fijamos el directorio de trabajo (en mi caso el *Escritorio* de una máquina Windows 7) para que sea el que contiene el fichero de datos, y usamos este código:

```
# Una posibilidad es que tengas la muestra como un vector.
#muestra =   # SI NO SE USA, ESCRIBE # AL PRINCIPIO DE LA LINEA
```

```r
# Si lees la muestra de un fichero csv:
# 1. selecciona el directorio de trabajo

# 2. Ahora introduce entre las comillas el nombre del fichero, y el tipo de separador, etc.

# SI NO SE USA, ESCRIBE # AL PRINCIPIO DE LA SIGUIENTE LINEA
muestra = read.table(file="../datos/Tut06-datosIntConfDesvTipica.csv", dec=".")[ ,1]


# Introducimos el nivel de confianza deseado.
nc=0.95

################################################
#NO CAMBIES NADA DE AQUI PARA ABAJO
################################################

# alfa
(alfa = 1 - nc )

## [1] 0.05

# Numero de datos de la muestra, grados de libertad
(n = length(muestra))

## [1] 800

k = n - 1

# Cuasidesviacion tipica muestral
(s = sd(muestra))

## [1] 7.1999

# Calculamos los valores criticos necesarios:

(chiAlfa2 = qchisq(1 - (alfa/2), df=k))

## [1] 879.23

(chiUnoMenosAlfa2 = qchisq(alfa/2, df=k))

## [1] 722.56

#Para la varianza, el intervalo de confianza sera
# extremo inferior
(intervaloVar = s^2 * k / c(chiAlfa2, chiUnoMenosAlfa2))

## [1] 47.108 57.322

# Y para la desviacion tipica el intervalo de confianza es este:
(intervaloS = s * sqrt(k / c(chiAlfa2, chiUnoMenosAlfa2)))

## [1] 6.8635 7.5711
```

- **Ejercicio 23, pág. 39**

Ejecuta, uno tras otro, estos comandos en la *Línea de Entrada* de GeoGebra. Verás que la estructura

de la construcción es prácticamente la misma que hemos usado en R, salvo detalles sintácticos de cada programa. El objeto entre llaves en el último comando es una *lista* de GeoGebra, y como ves se puede multiplicar listas por números (una especie de *"operación vectorial"*, como las de R, pero *a la GeoGebra*).

```
n = 17
k = n - 1
s = 2.1
nc = 0.99
alfa = 1 - nc
alfa2 = alfa / 2
chiAlfa2 = ChiCuadradoInversa[k, 1 - alfa2]
chiUnoMenosAlfa2 = ChiCuadradoInversa[k, alfa2]
intervalo = s * {sqrt(k / chiAlfa2),   sqrt(k / chiUnoMenosAlfa2) }
```

• **Ejercicio 24, pág. 41**
Ejecutamos el código sin fijar el valor de `sigma` (recuerda fijar el directorio de trabajo), y el resultado es erróneo.

```
muestra = read.table(file="../datos/Tut06-IntConf-Media-UsandoZ-datos.csv", sep=" ", dec=".")[,1
(intAsBio = ci.mu.z(data=muestra, conf=0.95, summarized=FALSE))

##
## 95% z Confidence interval for population mean
## Estimate     2.5%    97.5%
##   7.2313   7.1436   7.3190
```

• **Ejercicio 25, pág. 42**

```
muestra = read.table(file="../datos/Tut06-datosIntConfDesvTipica.csv", sep=" ", dec=".")[,1]
(intAsBio = ci.sigma(data=muestra, conf=0.95, summarized=FALSE))

##
## 95% Confidence interval for population variance
## Estimate     2.5%    97.5%
##   51.838   47.108   57.322
```

El intervalo que hemos obtenido es para la varianza σ^2. Para obtener un intervalo para σ hacemos:

```
sqrt(intAsBio$ci[2:3])

## [1] 6.8635 7.5711
```

Puedes compararlo con la solución del Ejercicio 22, que está en la página 61.

Fin del Tutorial06. ¡Gracias por la atención!