Estadística y Procesos Estocásticos

P. Saavedra, C.N. Hernández,
J. Artiles, A. Santana

Estadística y Procesos Estocásticos

ISBN:

Departamento de Matemáticas

Universidad de Las Palmas de Gran Canaria

Fecha: 15 de mayo de 2014

Índice general

Capítulo 1

Probabilidad

Una buena parte de los sistemas que se estudian en el campo de las telecomunicaciones contiene elementos de naturaleza aleatoria. Así por ejemplo, las variables que caracterizan el teletráfico toman valores en los que interviene el azar. El objetivo por tanto de esta asignatura es la *matematización de los fenómenos aleatorios*. En este capítulo se introducen los conceptos de fenómeno aleatorio, espacio muestral y álgebra de sucesos. Tras una exposición de los escenarios en los que interviene la probabilidad, se realiza una matematización de ésta a través de sus axiomas y propiedades. La idea de predecir fenómenos de naturaleza aleatoria a partir de otros observados se analiza a través del concepto de probabilidad condicional. El capítulo se completa con los teoremas de la probabilidad total y de Bayes.

1.1. Fenómeno aleatorio y espacio muestral asociado

Una de las acepciones del término *fenómeno* es *manifestación de la naturaleza*. Al fenómeno le subyacen sus propias causas. Cuando a través de las leyes físicas es posible establecer la conexión entre las causas y el fenómeno observable, éste podrá por lo general ser predicho con exactitud. En tal caso decimos que el fenómeno es *determinista*. Cuando todas las posibles causas no determinan el fenómeno o su complejidad hace imposible en la práctica realizar una predicción exacta, podemos

entender que el fenómeno es *aleatorio*. En tal caso, las predicciones del fenómeno habrán de hacerse en términos de probabilidades.

Para la formalización del concepto de probabilidad se requiere definir previamente los conceptos de espacio muestral y álgebra de sucesos. El *espacio muestral asociado a un fenómeno aleatorio es el conjunto de todos los posibles resultados elementales que puede presentar (manifestar) dicho fenómeno*. En los siguientes ejemplos se describen espacios muestrales asociados a diversos fenómenos aleatorios.

Ejemplo 1.1. El conjunto de posibles resultados elementales asociados al experimento consistente en lanzar un dado es el conjunto $\Omega = \{1, 2, 3, 4, 5, 6\}$. *Obtener número par* es un resultado del experimento, pero no es elemental dado que está formado por los sucesos elementales $\{2, 4, 6\}$.

Ejemplo 1.2. Al extraerse aleatoriamente una bola de una urna que contiene 5 bolas blancas y 3 negras, el espacio muestral es el conjunto

$$\Omega = \{b_1, b_2, b_3, b_4, b_5,\ n_1, n_2, n_3\}$$

Aunque las bolas no fuesen físicamente distinguibles, deben distinguirse a efectos teóricos para facilitar el cálculo de probabilidades.

Ejemplo 1.3. El espacio muestral asociado con el lanzamiento dos veces consecutivas de un dado es:

$$\Omega = \{(i, j) :\ i, j = 1, \ldots, 6\}$$

Este es el conjunto de todas las variaciones con repetición de los números enteros comprendidos entre 1 y 6 tomados dos a dos (36 resultados posibles).

Ejemplo 1.4. El tiempo transcurrido entre dos accesos consecutivos a una red de comunicaciones es un elemento frecuente en los problemas de teletráfico. Este tiempo es obviamente aleatorio y razonablemente, el espacio muestral asociado a este fenómeno aleatorio es $\Omega = \mathbb{R}^+$.

Ejemplo 1.5. Considérese el movimiento armónico definido por la ecuación:

$$X(t) = A\cos(\omega t + f)$$

donde la amplitud A y la frecuencia ω son fijas, pero la fase f se elige aleatoriamente en $[-\pi, \pi]$. En tal caso, el espacio muestral asociado puede considerarse como $\Omega = [-\pi, \pi]$.

Ejemplo 1.6. Supóngase ahora la superposición de movimientos armónicos:

$$X(t) = \sum_{j=1}^{K} A_j \cos(\omega_j t + f_j)$$

donde los pares (A_j, ω_j) para $j = 1, \ldots, K$ son fijos, pero las fases f_j se eligen aleatoriamente en en $[-\pi, \pi]$, de forma independiente. En tal caso, el espacio muestral asociado al fenómeno aleatorio de producir la señal $X(t)$ es $[-\pi, \pi]^K$; esto es, el conjunto de todas las secuencias de longitud K de números pertenecientes a $[-\pi, \pi]$.

1.2. Álgebra de sucesos

Un suceso es cualquier resultado (no necesariamente elemental) de un fenómeno aleatorio. Los sucesos son por tanto, subconjuntos del espacio muestral Ω. Así por ejemplo, el suceso *obtener número par,* correspondiente al fenómeno aleatorio descrito en el ejemplo 2.1, es el subconjunto $\{2, 4, 6\}$.

1.2.1. Sucesos especiales

Suceso seguro

Para que un suceso sea seguro debe contener todos los posibles resultados, lo que supone que coincide con Ω.

Suceso imposible

Se corresponde con el conjunto vacío $\emptyset$; esto es, no contiene ninguno de los posibles resultados.

Suceso contrario

El suceso contrario de un suceso A, representado por A^C, está formado por todos los resultados de Ω que no pertenecen a A.

1.2.2. Inclusión de sucesos

Un suceso A está incluido o contenido en otro B ($A \subset B$), si siempre que ocurre A, ocurre también B; esto es: todos los resultados de A lo son también de B.

1.2.3. Operaciones con sucesos

Unión

La unión de dos sucesos A y B es otro suceso que se representa por $A \cup B$ y que consiste en la *ocurrencia de al menos uno de los dos sucesos*. Incluye por tanto a todos los resultados de A y B.

Intersección

La intersección de dos sucesos A y B es otro suceso que se representa por $A \cap B$ y que consiste en la *ocurrencia simultánea de ambos sucesos*. Incluye a los resultados comunes de A y B.

1.2.4. Definición axiomática de álgebra de sucesos

Sea Ω el espacio muestral asociado a un fenómeno aleatorio y $\Im$ una clase de sucesos en Ω. $\Im$ constituye una $\sigma-$*álgebra de sucesos* si se satisfacen las siguiente propiedades:

1. $\emptyset \in \Im$
2. Si $A \in \Im$ entonces $A^C \in \Im$
3. Si $A_n \in \Im$, $\forall n \in \mathbb{N}$, entonces $\bigcup_{n=1}^{\infty} A_n \in \Im$

Se deja como ejercicio probar las siguientes consecuencias de estos axiomas:

1. $\Omega \in \Im$
2. Si $A_n \in \Im$, $\forall n \in \mathbb{N}$, entonces $\bigcap_{n=1}^{\infty} A_n \in \Im$

1.2.5. Incompatibilidad de sucesos

Dos sucesos A y B se dicen incompatibles si no pueden ocurrir simultáneamente; esto es: si $A \cap B = \emptyset$.

1.2.6. Sistema completo de sucesos.

En un espacio muestral Ω, una clase de sucesos $A_1, \ldots, A_n$ forman un sistema completo si:

1. $\Omega = A_1 \cup \ldots \cup A_n$ (ocurre con seguridad alguno de ellos)
2. $A_i \cap A_j = \emptyset$, para $i \neq j$ (incompatibilidad por pares).

Ejemplo 1.7. Para el fenómeno aleatorio definido por el lanzamiento de un dado se tiene:

- Suceso seguro: $\Omega = \{1, 2, 3, 4, 5, 6\}$

- *Obtener número par* = $A = \{2, 4, 6\}$
- Contrario de A: $A^c = \{1, 3, 5\}$
- *Obtener número mayor que tres* = $B = \{4, 5, 6\}$
- $A \cup B = \{2, 4, 5, 6\}$
- $A \cap B = \{4, 6\}$

Diagramas de Venn.

Los sucesos son conjuntos y como tales pueden representarse gráficamente mediante los diagramas de Venn. Estos diagramas permiten resolver de forma simple diversos problemas de probabilidad. La figura 1.1 corresponde a un estudio de simulación de un sistema de teletráfico en el que se ha observado el acceso de 300 usuarios a lo largo de un periodo de 300 minutos (lo que significa una tasa de llegada de uno por minuto). Los tiempos de permanencia de los usuarios en el sistema dependen del tamaño de la línea de espera (cola) y del tiempo que usen en el servicio. Supóngase que se selecciona aleatoriamente a uno de estos usuarios y consideramos los sucesos A_{150} = *acceder al sistema después del instante 150* y W_{10} = *permanecer más de 10 minutos en el sistema.* Ambos sucesos se representan en la figura 1.1.

1.3. Medidas de probabilidad

1.3.1. Criterios de probabilidad

La probabilidad de un suceso es una medida de cuantificación de la verosimilitud de su ocurrencia. Puede usarse en escenarios muy diversos, que van desde aquellos en los que puede definirse sin ambigüedad hasta escenarios en los que expresa la creencia subjetiva de que ocurran ciertos eventos. Consideraremos los siguientes criterios para la asignación de probabilidades a los sucesos de un espacio muestral:

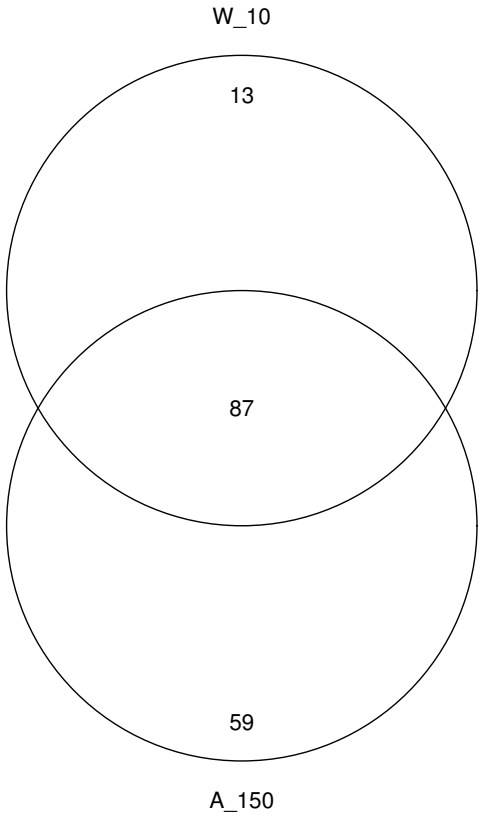


Figura 1.1: A_150 = "Acceso al sistema > 150" ; W_10 = "Permanencia > 10"

Probabilidad exacta.

La probabilidad es exacta cuando puede definirse sin ambigüedad a partir de consideraciones teóricas sobre el fenómeno que genera los sucesos. Así por ejemplo, al lanzar un dado equilibrado todos los posibles resultados deben tener la misma probabilidad, ya que *a priori* están en las mismas condiciones y no hay ninguna causa que favorezca algún resultado sobre los otros. Como hay 6 posibles resultados, podemos afirmar, sin ningún tipo de ambigüedad, que la probabilidad de obtener cada uno de ellos es 1/6.

Probabilidad frecuentista.

Cuando no hay información para asignar una probabilidad exacta, pero es posible observar repetidamente el fenómeno aleatorio en las mismas condiciones, puede asignarse como probabilidad de cada suceso su frecuencia relativa de ocurrencia. Obviamente dos series de observaciones podrían dar lugar a diferentes probabilidades. En cualquier caso, en un número muy elevado de observaciones, la frecuencia relativa de ocurrencias converge a la probabilidad del suceso (ver problema 3.6).

Probabilidad subjetiva.

Hay sucesos para los que no es posible determinar una probabilidad exacta de que ocurran y no se dispone (o no es posible disponer) de réplicas del fenómeno aleatorio. En estos casos, la probabilidad puede asignarse según un criterio subjetivo. Así por ejemplo, podemos formular nuestro grado de creencia en la existencia de vida extraterrestre en una medida de probabilidad.

1.3.2. Definición axiomática de la probabilidad

Sea Ω el espacio muestral asociado a un fenómeno aleatorio y $\Im$ una $\sigma-$álgebra de sucesos en Ω. Una medida de *probabilidad* sobre $\Im$ es cualquier función $\Pr : \Im \to [0,1]$ que satisface los siguientes axiomas:

1. $\Pr(\Omega) = 1$
2. Si $A_n \in \mathfrak{F} : \forall n \in \mathbb{N}$ y $A_m \cap A_n = \emptyset$, para $m \neq n$, entonces:

$$\Pr\left(\bigcup_{n=1}^{\infty} A\right) = \sum_{n=1}^{\infty} \Pr(A_n)$$

La terna de elementos $(\Omega, \mathfrak{F}, \Pr)$ recibe el nombre de espacio de probabilidad. El siguiente teorema da algunas propiedades de la probabilidad que son consecuencia inmediata de la definición axiomática.

Teorema 1.1. *Sea $(\Omega, \Im, P)$ un espacio de probabilidad. Se tiene entonces:*

1. $\Pr(\emptyset) = 0$

2. $\Pr\left(A^C\right) = 1 - \Pr\left(A\right)$

3. *Si* $A \subset B$*, entonces* $\Pr\left(A\right) \leq \Pr\left(B\right)$

4. $\Pr\left(A \cup B\right) = \Pr\left(A\right) + \Pr\left(B\right) - \Pr\left(A \cap B\right)$

Demostración.
1) $\Omega = \Omega \cup \emptyset$, siendo $\Omega \cap \emptyset = \emptyset$. Entonces, de acuerdo con el axioma 2 se tiene que $\Pr\left(\Omega\right) = \Pr\left(\Omega \cup \emptyset\right) = \Pr\left(\Omega\right) + \Pr\left(\emptyset\right)$, y por tanto, que $\Pr\left(\emptyset\right) = 0$.

2) El resultado se sigue de la relación $\Omega = A \cup A^C$ y del segundo axioma de la probabilidad.

3) El resultado es consecuencia inmediata de la relación $B = A \cup \left(A^C \cap B\right)$

4) $A \cup B = A \cup \left(A^C \cap B\right)$, siendo $A \cap \left(A^C \cap B\right) = \emptyset$. De aquí, $\Pr\left(A \cup B\right) = \Pr\left(A\right) + \Pr\left(A^C \cap B\right)$.

Veamos ahora que $\Pr\left(A^C \cap B\right) = \Pr\left(B\right) - \Pr\left(A \cap B\right)$. En efecto, $B = \left(A \cap B\right) \cup \left(A^C \cap B\right)$ lo que supone que $\Pr\left(B\right) = \Pr\left(A \cap B\right) + \Pr\left(A^C \cap B\right)$, y finalmente que $\Pr\left(A^C \cap B\right) = \Pr\left(B\right) - \Pr\left(A \cap B\right)$. □

1.3.3. Regla de Laplace

Considérese un espacio muestral finito Ω en el que todos sus resultados son equiprobables. Entonces, para cualquier suceso A, $\Pr\left(A\right) = \#\left(A\right)/\#\left(\Omega\right)$. (La notación $\#\left(A\right)$ representa el número de resultados en A o favorables al suceso A).

La regla anterior es una consecuencia directa del axioma segundo de la probabilidad. Téngase en cuenta que la probabilidad de cada resultado elemental es $1/\#\left(\Omega\right)$.

Ejemplo 1.8. Considérese que de una urna que contiene k bolas blancas y $n-k$ negras, se extraen dos bolas consecutivamente sin reemplazamiento. Entonces, el espacio muestral Ω está formado por todas las variaciones de las n bolas que hay en la urna tomadas 2 a 2, lo que significa que $\#\left(\Omega\right) = n\left(n-1\right)$. Se tiene además:

- $\Pr(Ambas\,blancas) = k(k-1)/(n(n-1))$
- $\Pr(Primera\,blanca\,y\,segunda\,negra) = k(n-k)/(n(n-1))$
- $\Pr(Una\,blanca\,y\,otra\,negra) = 2k(n-k)/(n(n-1))$

1.4. Probabilidad condicionada

Un suceso A contiene información de otro suceso B, cuando la ocurrencia o no de A modifica la probabilidad de B. Por ejemplo, en el lanzamiento de un dado, el suceso $A =$ *obtener número par* contiene información sobre el suceso $B = \{2\}$. A priori, en ausencia de más información, la probabilidad de B es $\Pr(B) = 1/6$. Ahora bien, si sabemos que A ocurrió, la probabilidad de B es $1/3$.

Considérese por tanto un espacio de probabilidad $(\Omega, \mathcal{F}, P)$ y sea $A \in \mathcal{F}$ tal que $\Pr(A) > 0$. La probabilidad de cualquier suceso $B \in \mathcal{F}$ condicionada por A se define por:

$$\Pr(B \mid A) = \frac{\Pr(A \cap B)}{\Pr(A)}$$

Para cada suceso A fijo, $\Pr(B \mid A)$ es una medida de probabilidad cuyo argumento es B. En efecto:

- $\Pr(\Omega \mid A) = 1$
- Si $B \cap C = \emptyset$, entonces $\Pr(B \cup C \mid A) = \Pr(B \mid A) + \Pr(C \mid A)$

De la definición de probabilidad condicionada se obtiene la siguiente regla multiplicativa:

$$\Pr(A \cap B) = \Pr(A)\Pr(B \mid A)$$

Ejercicio 1.1. Hallar la probabilidad de que al extraer de una baraja española dos cartas consecutivamente, la primera sea copa y la segunda espada.

1.5. Independencia de sucesos

Un suceso B es independiente de A cuando A no contiene información sobre B. Esto significa que $\Pr(B \mid A) = \Pr(B)$.

Es fácil comprobar que si un suceso B es independiente de A, entonces A lo es de B. En efecto, supóngase que B es independiente de A y por tanto que $\Pr(B \mid A) = \Pr(B)$. Entonces:

$$\Pr(A \mid B) = \frac{\Pr(A \cap B)}{\Pr(B)} = \frac{\Pr(A)\Pr(B \mid A)}{\Pr(B)} = \frac{\Pr(A)\Pr(B)}{\Pr(B)} = \Pr(A)$$

lo que supone que A es independiente de B. Se tiene por tanto la siguiente regla multiplicativa: los sucesos A y B son independientes, si y solo si:

$$\Pr(A \cap B) = \Pr(A)\Pr(B)$$

Podemos extender este concepto de la siguiente forma: Un conjunto de sucesos $A_1, \ldots, A_n$, son mutuamente independientes si la probabilidad de que ocurra simultáneamente cualquier subconjunto de ellos es igual al producto de sus probabilidades.

1.6. Teoremas notables de la probabilidad

Los teoremas de la probabilidad total y el de Bayes, que se enuncian a continuación, son de gran utilidad para la resolución de problemas de probabilidad.

Teorema 1.2. ***(de la probabilidad total).***

Sea $A_1, \ldots A_n$ un sistema completo de sucesos. Se tiene entonces que:

$$\Pr(B) = \sum_{i=1}^{n} \Pr(B|A_i)\Pr(A_i)$$

Demostración. Dado que $A_1, \ldots A_n$ un sistema completo, se tiene:

$$B = (A_1 \cap B) \cup \cdots \cup (A_n \cap B), \quad (A_i \cap B) \cap (A_j \cap B) = \varnothing \ : \ i \neq j$$

De aquí se sigue:

$$\Pr(B) = \sum_{i=1}^{n} \Pr(A_i \cap B) = \sum_{i=1}^{n} \Pr(B \mid A_i) \Pr(A_i)$$

□

Teorema 1.3. ***(de Bayes).***

Para un sistema completo de sucesos $A_1, \ldots A_n$ se verifica:

$$\Pr(A_i \mid B) = \frac{\Pr(B \mid A_i) \Pr(A_i)}{\Pr(B)} = \frac{\Pr(B \mid A_i) \Pr(A_i)}{\sum_{j=1}^{n} \Pr(B \mid A_j) \Pr(A_j)}$$

Ejemplo 1.9.

Considérense tres urnas con la siguiente composición de bolas blancas y negras: U_1 contiene tres bolas blancas y dos negras, U_2 cuatro blancas y dos negras y U_3 una blanca y cuatro negras. Se selecciona una urna al azar y seguidamente una bola de la urna elegida.

a. Hallar la probabilidad de que la bola seleccionada sea blanca.

Para $i = 1, 2, 3$, sea el suceso $U_i =$ *seleccionar la urna* U_i. La selección al azar de la urna significa que $P(U_i) = 1/3$.

Sea ahora $B =$ *La bola seleccionada es blanca.* Se tiene entonces:

$$\Pr(B) = \sum_{i=1}^{3} \Pr(B \mid U_i) \Pr(U_i) = \frac{3}{5} \times \frac{1}{3} + \frac{4}{6} \times \frac{1}{3} + \frac{1}{5} \times \frac{1}{3} \approx 0{,}4889$$

b. Probabilidad de que haya sido elegida la segunda urna supuesto que la bola extraída era blanca.

$$\Pr(U_2 \mid B) = \frac{\Pr(B \mid U_2) \Pr(U_2)}{\Pr(B)} \approx 0{,}4545$$

Problemas

1. Sean A, B y C tres sucesos arbitrarios. Encontrar expresiones de los sucesos siguientes:

   a) Solamente ocurre A.

   b) Ocurren tanto A como B pero no C

   c) Los tres eventos ocurren.

   d) Ocurre por lo menos uno.

   e) Ocurren al menos dos.

   f) Ocurre exactamente uno.

   g) Ocurren exactamente dos.

   h) No ocurre ninguno.

2. Dados los sucesos A, B y C, probar las siguientes propiedades distributivas:

   a) $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

   b) $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$.

3. Supóngase que de una baraja española se extraen cuatro cartas al azar sin reemplazamiento. Calcular las probabilidades de los siguientes sucesos:

   a) Obtener cuatro ases.

   b) Que las cuatro cartas sean de la misma figura.

   c) Que las cuatro cartas sean de igual palo.

   d) Obtener al menos una sota.

4. Las letras del alfabeto telegráfico Morse están formadas por una sucesión de rayas y puntos con repeticiones permitidas. ¿Cuántas letras pueden formarse con 10 o menos símbolos?.

5. ¿De cuántas formas diferentes pueden colocarse k bolas en n celdas (en cada celda pueden haber varias bolas).

6. Si la probabilidad de comprar el periódico es $\Pr(D) = 0.3$, la de una revista $\Pr(R) = 0.2$ y la de comprar ambos $\Pr(D \cap R) = 0{,}08$, calcular las probabilidades de los sucesos siguientes:

   a) Comprar un periódico o una revista.

   b) Comprar un periódico y no una revista.

   c) Una revista y no un periódico.

   d) No comprar un periódico o no comprar una revista.

   e) No comprar un periódico y no comprar una revista.

7. De una baraja española se extraen consecutivamente dos cartas al azar sin reemplazamiento. Hallar la probabilidad de que la primera carta sea copa y la segunda una figura. ¿Son éstos sucesos independientes?.

8. Consideremos una urna con 7 bolas blancas y 3 bolas negras, y una segunda urna con 3 bolas blancas y 4 negras. Se extrae al azar una bola de la primera urna y se pasa a la segunda. Se extrae a continuación una bola al azar de la segunda urna. Calcúlese la probabilidad de que esta bola sea blanca.

9. Supóngase que del conjunto de los 10 dígitos se seleccionan k al azar. Hallar las siguientes probabilidades:

   a) De que no aparezca el cero

   b) De que no aparezca ni el cero ni el uno.

   c) De que por lo menos no aparezca uno de los dígitos 0 y 1.

10. Supóngase que del conjunto de los 10 dígitos se seleccionan k al azar. Hallar las sigui

11. Si se colocan aleatoriamente n bolas en n celdas, hallar la probabilidad de que exactamente una quede vacía.

12. En un lote de estacionamiento hay doce lugares en hilera. Un hombre observó que había ocho coches estacionados y que los cuatro lugares vacíos eran adyacentes uno al otro. Dado que hay cuatro lugares vacíos, ¿es sorprendente este arreglo (es decir, indica que no hay aleatoriedad)?.

13. *La prueba de una hipótesis estadística.* A un profesor de Cornell le levantaron infracción doce veces por estacionarse ilegalmente durante las noches. Las doce infracciones fueron hechas en martes o en jueves. Encontrar la probabilidad de este evento. (¿Estaría justificado que alquilara un garaje solamente estos días?).

14. Dadas doce personas, hallar las siguientes probabilidades:

    a) De que sus cumpleaños sean en doce meses diferentes del calendario

    b) De que los cumpleaños de seis personas determinadas sean todos en dos meses del calendario.

15. Una planta armadora recibe microcircuitos procedentes de tres fabricantes A, B y C. El 50 % del total se compra a A, mientras que a B y C se les compra un 25 % a cada uno. El porcentaje de circuitos defectuosos para A, B y C es 5, 10 y 12 % respectivamente. Si los circuitos se almacenan en la planta sin importar quién fue el proveedor:

    a) Determinar la probabilidad de que una unidad armada en la planta contenga un circuito defectuoso.

    b) Si un circuito no está defectuoso, ¿cuál es la probabilidad de que haya sido vendido por el proveedor B?

16. El transmisor de localización de emergencias de los aviones (TLE) es un dispositivo diseñado para transmitir una señal en caso de accidente. La Altigauge Manufacturing Company (A) fabrica el 80 % de los TLE, la Bryant Company (B) el 15 % y la Chartair Company (C) el otro 5 %. Los TLE fabricados por Altigauge tienen una tasa de fallos del 4 %, los de la Bryant el 6 % y los de la Chartair el 9 % (lo que a explica que la Chartair tenga la cuota de mercado más baja).

    a) Hallar la probabildiad de que un TLE seleccionado aleatoriamente de la población general de todos los TLE sea defectuoso.

    b) Supuesto que el TLE sea defectuoso, hallar la probabilidad de que lo haya fabricado A.

17. Se ha recibido una partida de 20 ordenadores, de los cuales dos son defectuosos. La tienda decide seleccionar aleatoriamente dos y aceptar el embarque si ninguno de los dos tiene defectos. ¿Cuál es la probabilidad de aceptación del embarque?

18. De entre 20 tanques de combustible, tres se encuentran defectuosos. Si se seleccionan aleatoriamente cuatro tanques:

    a) ¿Cuál es la probabilidad de que ninguno de los tanques se encuentre defectuoso?

    b) ¿Cuál es la probabilidad de que al menos uno de los tanques tenga defectos?

19. La probabilidad de que cierto componente eléctrico funcione es de 0.9 Un aparato contiene dos de estos componentes. El aparato funciona mientras lo haga uno de los dos componentes.

    a) Sin importar cuál de los dos componentes funcione o no, ¿cuáles son los posibles resultados y sus respectivas probabilidades?.

    b) ¿Cuál es la probabilidad de que el aparato funcione?.

20. Un mecanismo consta de 3 piezas A, B y C. La probabilidad de que falle cada una de ellas es, respectivamente, 0.03, 0.08 y 0.01. Calcular la probabilidad de que el mecanismo deje de funcionar si esto sólo sucede cuando:

    a) Le fallan las tres piezas.

    b) Le fallan al menos dos piezas.

    c) Le falla al menos una pieza.

21. Dados los sucesos A, B y C, probar: $\Pr(A \cup B \cup C) = \Pr(A) + \Pr(B) + \Pr(C) - \Pr(A \cap B) - \Pr(A \cap C) - \Pr(B \cap C) + \Pr(A \cap B \cap C)$.

22. Un avión con tres bombas trata de destruir una línea férrea. La probabilidad de destruir la línea con cualquiera de las bombas es $1/3$. ¿Cuál es la probabilidad de que la línea quede destruida si el avión emplea las tres bombas?.

Capítulo 2

Variables aleatorias

Las variables aleatorias son magnitudes cuyo valor es función del azar. Para predecir sus valores se requiere conocer su distribución de probabilidad. En este capítulo se formalizan estos conceptos y se estudian las características de las distribuciones de probabilidad, así como sus propiedades de mayor interés. Por último se estudian algunas distribuciones de probabilidad particulares de gran importancia práctica.

2.1. Variables aleatorias y sus distribuciones de probabilidad

En un espacio de probabilidad $(\Omega, \Im, P)$, una variable aleatoria X es cualquier función de la forma $X : \Omega \to \mathbb{R}$. Las variables aleatorias son por tanto *funciones del azar*.

Ejemplo 2.1. Sobre el espacio muestral asociado al lanzamiento de un dado dos veces consecutivas (ejemplo 1.3) definimos la función $X(i, j) = i + j$ (suma de los números obtenidos). Por tanto, el valor que presente X en una observación de este fenómeno aleatorio depende del azar.

El problema principal que plantea el manejo de variables aleatorias es la predicción de sus posibles valores. Esta predicción se realiza, en general, a través de su *distribución de probabilidad*, la cual queda determinada la *función de distribución de*

probabilidad acumulativa o simplemente, *función de distribución de probabilidad*, la cual se define por:

$$F(t) = \Pr(X \leq t) \quad : \quad t \in \mathbb{R}$$

En el ejemplo 2.1 resulta muy sencillo obtener la distribución de probabilidad. Así por ejemplo, $F(4)$ es la probabilidad de que ocurra una pareja de resultados (i, j) cuya suma es menor o igual que 4. Por tanto (ver tabla 2.1),

$$F(4) = \Pr(\{(1,1),(1,2),(1,3),(2,1),(2,2),(3,1)\}) = \frac{6}{36}$$

Ejemplo 2.2. El tiempo X transcurrido entre dos accesos consecutivos a una red de comunicaciones es una variable aleatoria. Su distribución de probabilidad a menudo queda determinada por una función acumulativa de la forma:

$$F(t) = 1 - \exp(-(t/\lambda)^{\kappa}) \quad : \quad t \geq 0$$

La figura 2.1 muestra la función acumulativa de esta distribución para $\kappa = 2$ y $\lambda = 10$.

Para una variable aleatoria X con distribución de probabilidad $\mathcal{P}$ usaremos habitualmente la notación $X \cong \mathcal{P}$.

2.2. Clasificación de variables aleatorias

Las variables aleatorias se clasifican según su distribución de probabilidad esté concentrada sobre un conjunto finito o numerable de puntos (discretas) o se distribuya a lo largo de todos los puntos de un intervalo (continuas).

2.2.1. Variables aleatorias discretas.

Una variable aleatoria X se dice discreta cuando el conjunto de valores que puede tomar es finito o numerable. En tal caso, su distribución de probabilidad

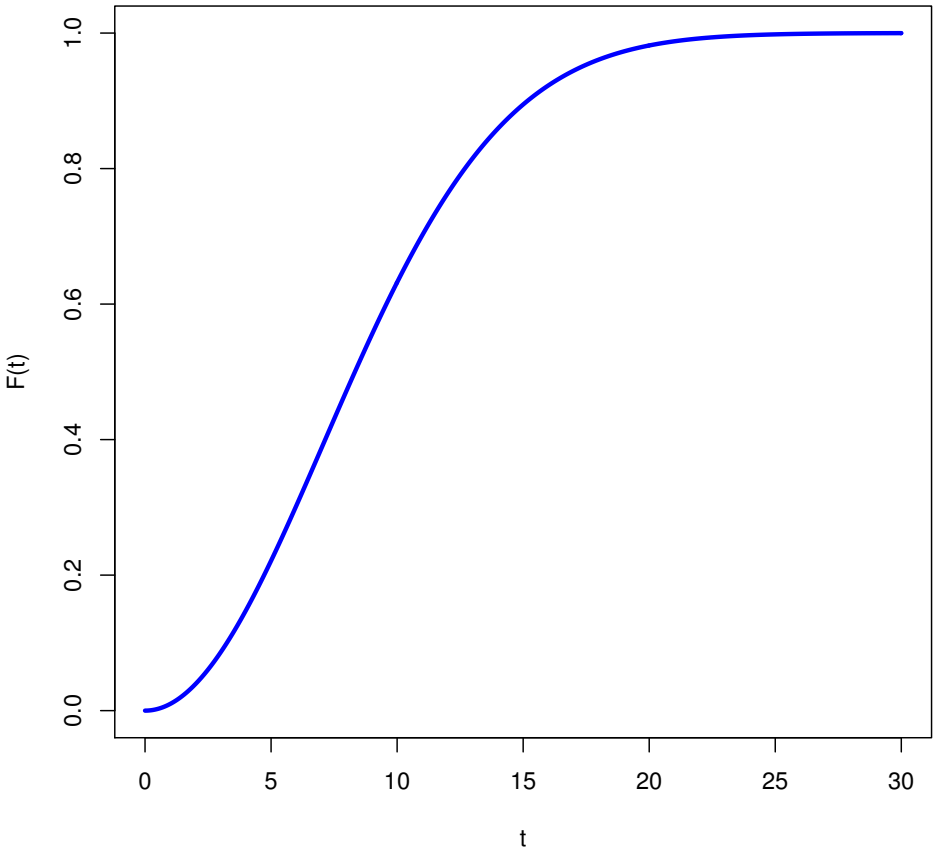


Figura 2.1: Función de distribución de probabilidad para el tiempo entre dos accesos consecutivos a un sistema de tráfico

queda plenamente especificada por la función de probabilidad $\Pr(X = t)$, donde t es cualquier valor que pueda tomar la variable. Obviamente se tiene que: $\sum_t \Pr(X = t) = 1$.

Ejemplo 2.3. La distribución de probabilidad de la variable $X =$ *suma de los números obtenidos en dos lanzamientos consecutivos de un dado* es discreta y su distribución de probabilidad se resume en la siguiente tabla.

X	2	3	4	5	6	7	8	9	10	11	12
$P(X=t)$	$\frac{1}{36}$	$\frac{2}{36}$	$\frac{3}{36}$	$\frac{4}{36}$	$\frac{5}{36}$	$\frac{6}{36}$	$\frac{5}{36}$	$\frac{4}{36}$	$\frac{3}{36}$	$\frac{2}{36}$	$\frac{1}{36}$

Tabla 2.1: Distribución de probabilidad para la suma de los números obtenidos en dos lanzamientos consecutivos de un dado

2.2.2. Variables aleatorias continuas.

Una variable aleatoria X se dice continua cuando lo es su función de distribución de probabilidad $F(t) = \Pr(X \leq t)$.

Hay algunas distribuciones continuas especiales (las distribuciones absolutamente continuas), que se caracterizan porque quedan determinadas a través de las *funciones de densidad de probabilidad*. Una función no negativa $f : \mathbb{R} \to \mathbb{R}^+$ que satisface la condición:

$$\int_{-\infty}^{\infty} f(t) \cdot dt = 1$$

es la función de densidad de probabilidad de una variable aleatoria X, si su función de distribución acumulativa $F(x)$ puede expresarse como:

$$F(x) = \Pr(X \leq x) = \int_{-\infty}^{x} f(t) \cdot dt \ : \ x \in \mathbb{R}$$

En aquellos casos en los que la función $f(x)$ sea continua, el segundo teorema fundamental del cálculo integral establece que $F'(x) = f(x)$. Puede comprobarse fácilmente que para la distribución de probabilidad del ejemplo 2.2, la función de densidad de probabilidad es:

$$f(x) = \kappa x^{\kappa-1} \exp\left(-(x/\lambda)^{\kappa}\right)/\lambda \ : \ x \geq 0$$

De la definición de densidad de probabilidad se deduce que:

$$\Pr(a < X \leq b) = \int_{a}^{b} f(t)\, dt \ : \ a < b$$

La figura 2.2 corresponde a la función de densidad de probabilidad para el tiempo comprendido entre dos llegadas consecutivas a un sistema de teletráfico. El área sombreada corresponde a $\Pr(10 < X \leq 15)$.

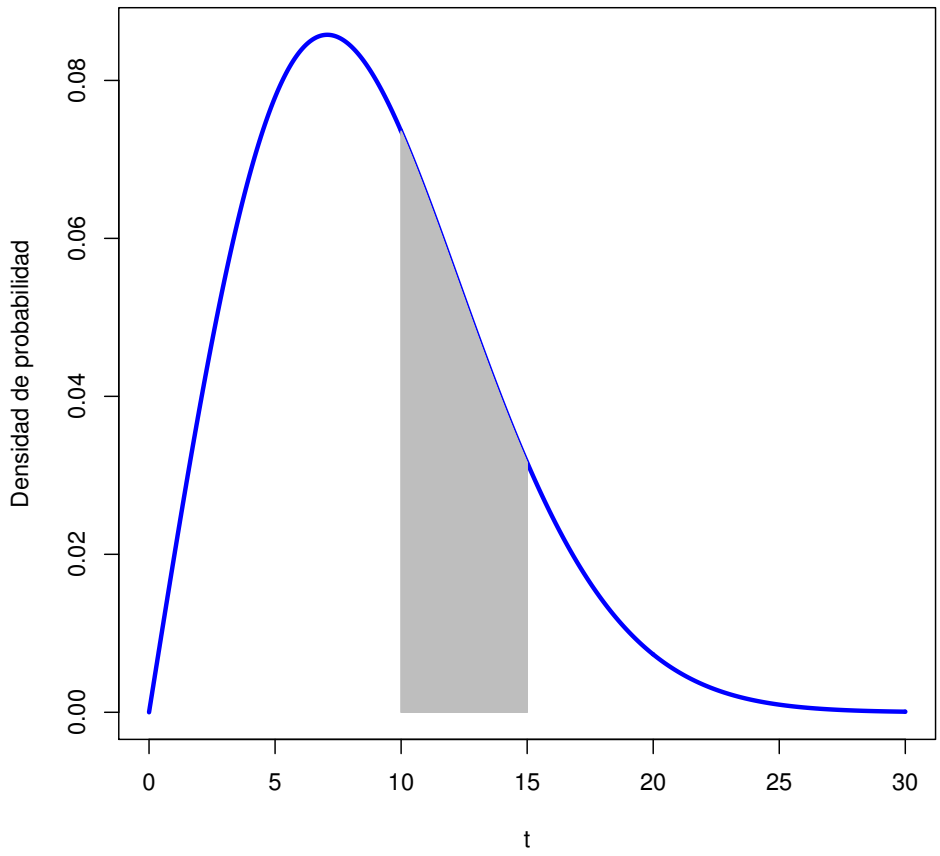


Figura 2.2:

Puede establecerse una analogía entre probabilidad y masa que resulta útil para aclarar intuitivamente algunos significados. En el caso de variables aleatorias discretas, que toman valores en un número finito (o numerable) de puntos, podemos imaginar que cada punto tiene una masa igual a su probabilidad de ocurrir, por lo que la masa total de todos los puntos es 1; de esta forma, la distribución de probabilidad de la variable es equivalente a la forma en que se reparte (o distribuye) esta masa entre el conjunto de puntos. Cuando la variable es continua esa masa total se reparte sobre todo un intervalo; en esta analogía, el valor de la función de

distribución acumulativa a la izquierda del punto x, esto es, $F(x) = P(X \leq x)$, se correspondería con la masa total acumulada sobre el intervalo a la izquierda de x. A su vez, la densidad de probabilidad $f(x)$ se correspondería con la cantidad de masa por unidad de longitud (de acuerdo con las unidades en las que se exprese la variable), en un entorno del valor x.

2.3. Características de las distribuciones de probabilidad

2.3.1. Esperanza matemática

La esperanza matemática de una variable aleatoria tiene como finalidad resumir su distribución de probabilidad en un valor representativo de la totalidad de la distribución. Usando la analogía entre probabilidad y masa, el concepto de esperanza de una variable aleatoria se corresponde con el *centro de gravedad* de su distribución de probabilidad.

Para formalizar este concepto, sea X una variable aleatoria y $g : \mathbb{R} \to \mathbb{R}$. La *esperanza matemática* de la variable aleatoria $g(X)$ se define como:

- Si X es discreta: $E[g(X)] = \sum_t g(t) \cdot \Pr(X = t)$
- Si X es continua y tiene función de densidad $f(t)$, entonces:

$$E[g(X)] = \int_{-\infty}^{\infty} g(t) \cdot f(t) \cdot dt$$

Para la variable aleatoria X cuya distribución de probabilidad se muestra en la tabla 2.1, la esperanza se calcula en la forma que sigue:

$$E[X] = \sum_{t=2}^{12} tP(X = t) = 7$$

Para la distribución del ejemplo 2.2 y siendo $\kappa = 1$, la función de densidad se reduce a: $f(t) = (1/\lambda)\exp(-t/\lambda) \ : \ t \geq 0$, y de aquí:

$$E[X] = \frac{1}{\lambda}\int_0^\infty t\exp(-t/\lambda) = \lambda$$

Teorema 2.1. ***(1ª propiedad de linealidad de la esperanza)***

Sea λ una constante y X una variable aleatoria. Entonces

$$E[\lambda \cdot X] = \lambda \cdot E[X]$$

Demostración. (caso discreto)

$$E[\lambda X] = \sum_t \lambda t \cdot \Pr(\lambda X = \lambda t) = \lambda \sum_t t \cdot \Pr(X = t) = \lambda E[X]$$

□

Se deja al lector la demostración para el caso en el que X sea una variable continua con función de densidad $f(t)$.

2.3.2. Dispersión de variables aleatorias

Para una variable aleatoria X con esperanza μ, la *varianza* se define por:

$$\text{var}(X) = E\left[(X-\mu)^2\right]$$

La figura 2.3 muestra las funciones de densidad de tres distribuciones con la misma esperanza ($\mu = 80$), pero con diferentes varianzas (9, 25 y 64). La menor varianza supone una mayor concentración de la distribución alrededor de la esperanza o centro de gravedad. La siguiente proposición da algunas propiedades de la varianza.

1. $\text{var}(\lambda X) = \lambda^2 \text{var}(X)$
2. $\text{var}(X) = E[X^2] - E[X]^2$

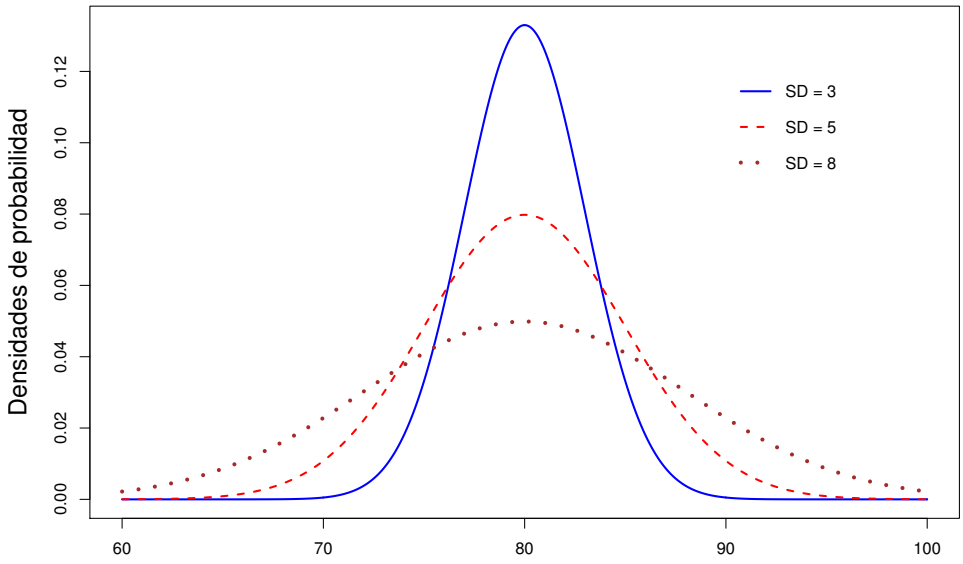


Figura 2.3: Distribuciones de probabilidad con diferentes varianzas

Demostración. La primera propiedad se sigue directamente de la linealidad de la esperanza. En efecto, si $E[X] = \mu$ se tiene que $E[\lambda X] = \lambda\mu$, y de aquí:

$$\text{var}(\lambda X) = E\left[(\lambda X - \lambda\mu)^2\right] = E\left[\lambda^2 (X-\mu)^2\right] = \lambda^2 E\left[(X-\mu)^2\right] = \lambda^2 \text{var}(X)$$

La varianza puede obtenerse alternativamente como:

$$\text{var}(X) = E\left[(X-\mu)^2\right] = E\left[X^2\right] - 2\mu E[X] + \mu^2 = E\left[X^2\right] - \mu^2$$

□

Finalmente, la *desviación típica* se define por: $\text{sd}(X) = \sqrt{\text{var}(X)}$

2.3.3. Momentos

Para cualquier variable aleatoria X, el *momento de orden* k se define por:

$$\mu_k = E\left[X^k\right]$$

Nótese que el momento de orden uno de una variable aleatoria X coincide con su esperanza y que:

$$\text{var}\left(X\right) = \mu_2 - \mu_1^2$$

2.3.4. Cuantiles

Para una variable aleatoria X cuya función de distribución acumulativa es $F\left(t\right)$, el α-ésimo cuantil se define como el valor q_α, tal que $F\left(q_\alpha\right) = \Pr\left(X \leq q_\alpha\right) = \alpha$, siempre y cuando, la ecuación tenga solución.

La mediana de la distribución de probabilidad se define como el cuantil 0.5. Los cuartiles 1 y 3 son los cuantiles 0.25 y 0.75 respectivamente.

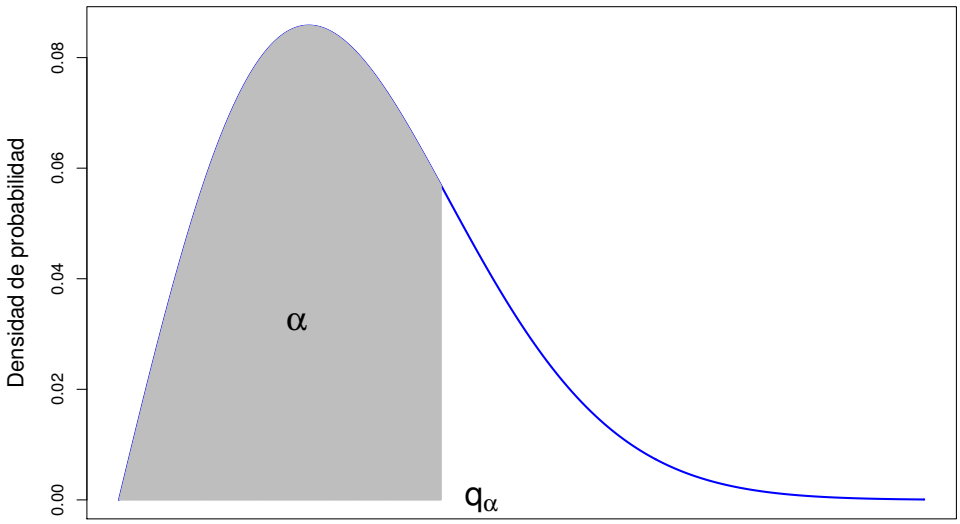


Figura 2.4: El α-ésimo cuantil es q_α

Para la distribución del ejemplo 2.2, la mediana se obtiene fácilmente de la función acumulativa resolviendo la ecuación:

$$F(t) = 1 - \exp\left(-\left(t/\lambda\right)^{\kappa}\right) = 0{,}50$$

La solución de la ecuación es: $t = \lambda\left(-\log\ 0{,}5\right)^{1/\kappa}$.

2.4. Función característica

La función característica de una variable aleatoria X *caracteriza* la distribución de probabilidad en el sentido que existe una correspondencia biunívoca entre la función de distribución y la característica. Tal como se verá en lo sucesivo, es una herramienta que permite simplificar el manejo de las distribuciones de probabilidad.

La *función característica* de cualquier variable aleatoria X se define como:

$$\varphi(u) = E\left[\exp\left(iuX\right)\right]$$

Para $\kappa = 1$, la distribución del ejemplo 2.2 se reduce a:

$$f(t) = \left(1/\lambda\right)\exp\left(-t/\lambda\right)\ :\ t \geq 0$$

La función característica se obtiene entonces:

$$\varphi(u) = \int_0^{\infty} \exp\left(iut\right) f(t)\, dt = \frac{1}{\lambda}\int_0^{\infty} \exp\left(-\left(1/\lambda - iu\right)t\right) dt = \frac{1}{1 - \lambda iu}$$

Los momentos de la distribución pueden entonces obtenerse a través de las sucesivas derivadas de la función característica en la forma:

$$\mu_k = \frac{\varphi^{(k)}(0)}{i^k}$$

Utilizando este resultado, puede probarse fácilmente que el momento de orden uno de la distribución anterior es: $\mu_1 = \mu = \lambda$.

La función característica caracteriza la distribución de probabilidad, en el sentido de que la determina. El siguiente teorema expresa el modo en el que la función característica determina la función de densidad de probabilidad cuando ésta existe.

Teorema 2.2. ***(de inversión de la función característica)***

Sea X una variable aleatoria con densidad de probabilidad $f(t)$ y función característica $\varphi(u)$. Se tiene entonces:

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \exp(-iut)\, \varphi(u)\, du$$

2.5. Distribuciones de probabilidad especiales

2.5.1. Bernoulli.

Considérese un fenómeno aleatorio en el que el suceso A tiene probabilidad p. Asociado a éste se define la variable aleatoria X como 1 ó 0 según ocurra o no A. Nótese que su distribución de probabilidad se puede expresar en la forma:

$$\Pr(X = t) = p^t (1-p)^{1-t} \quad : \; t = 1, 0$$

Esta distribución recibe el nombre de *distribución de Bernoulli de parámetro p*. Su función característica es:

$$\varphi(u) = p \cdot \exp(iu) + 1 - p$$

Es fácil comprobar que $E[X] = p$ y $\operatorname{var}(X) = p(1-p)$.

2.5.2. Binomial

Considérese nuevamente un fenómeno aleatorio en el que un suceso A tiene probabilidad p. Supóngase ahora que el experimento se repite n veces en las mismas condiciones de tal forma que los sucesivos resultados son independientes. Sea la variable aleatoria X definida como *número de veces que ocurre A*. Se tiene entonces que:

$$\Pr(X = t) = \binom{n}{t} p^t (1-p)^{n-t} \quad : \; t = 0, 1, \ldots, n$$

Esta distribución de probabilidad recibe el nombre de *binomia*l de parámetros n y p ($b(n,p)$). Su función característica tiene la forma:

$$\varphi(u) = [p \cdot \exp(iu) + 1 - p]^n$$

De aquí resulta que $E[X] = np$ y $\text{var}(X) = np(1-p)$.

2.5.3. Poisson

Se dice que una variable aleatoria X tiene *distribución de Poisson* de parámetro μ, ($X \cong \mathcal{P}(\mu)$) si su distribución de probabilidad es de la forma:

$$\Pr(X = t) = \exp(-\mu)\frac{\mu^t}{t!} \ : \ t = 0, 1, 2, \ldots$$

La función característica es:

$$\varphi(u) = \exp(-\mu)\sum_{t=0}^{\infty}\frac{(\mu\exp(iu))^t}{t!} = \exp(\mu(\exp(iu) - 1))$$

A partir de $\varphi(u)$ puede comprobarse que $E[X] = \text{var}(X) = \mu$.

Veamos ahora en que forma la distribución de Poisson se relaciona con la binomial.

Teorema 2.3. *La sucesión de distribuciones de probabilidad $b(n; \lambda/n)$ converge a la distribución de Poisson de parámetro λ.*

Demostración. Sea $\varphi_n(u)$ la función característica de la distribución $b(n; \lambda/n)$, para cualquier valor de n y $\varphi(u)$, la de la distribución $\mathcal{P}(\lambda)$. Entonces, para $n \to \infty$:

$$\varphi_n(u) = \left(\frac{\lambda}{n}\exp(iu) + 1 - \frac{\lambda}{n}\right)^n = \left(1 + \frac{\lambda(\exp(iu) - 1)}{n}\right)^n \to \varphi(u)$$

para cualquier valor de u. □

Este resultado permite aproximar la distribución $b(n,p)$ por la $\mathcal{P}(np)$ en aquellos casos en los que el valor de la probabilidad p sea muy pequeño y el de n muy grande.

Ejemplo 2.4. Un sistema de transmisión digital comete un error al transmitir un bit con probabilidad 10^{-4}. Se supone que los sucesivos errores se producen de forma independiente. Supóngase ahora que se envía un mensaje de 10^4 bit y sea $X = \textit{Número de errores en la trasmisión}$. La distribución de probabilidad exacta de X es $b\left(10^4, 10^{-4}\right)$. De acuerdo con el teorema 2.3, esta distribución se puede aproximar por la $\mathcal{P}(1)$, lo cual facilita el manejo de la distribución.

2.5.4. Uniforme

La variable aleatoria X tiene *distribución uniforme* en el intervalo $[a, b]$ ($U[a, b]$) si su función de densidad de probabilidad tiene la forma:

$$f(x) = \frac{1}{b-a} \ : \ a \leq x \leq b$$

La función característica es:

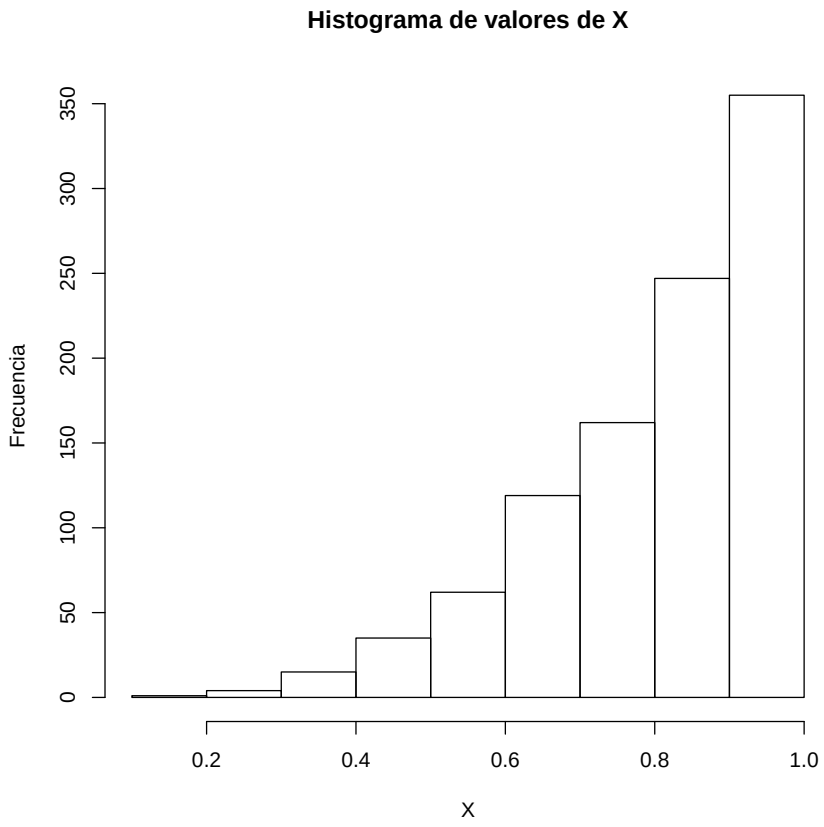
$$\varphi(u) = \frac{\exp(iub) - \exp(iua)}{iu(b-a)}$$

El siguiente resultado es muy útil para la simulación de variables aleatorias.

Teorema 2.4. *Sea $F(t)$ una función de distribución acumulativa continua y estrictamente creciente y $U \cong U[0, 1]$. Entonces, la variable aleatoria $X = F^{-1}(U)$ tiene a $F(t)$ como función de distribución.*

Ejemplo 2.5. Para la simulación de valores de una variable aleatoria con función de distribución acumulativa $F(t) = t^4 \ : \ 0 \leq t \leq 1$ se pueden simular valores $U_1, \ldots, U_n$ de una distribución $U[0, 1]$ y calcular $X_i = F^{-1}(U_i) \ : \ i = 1, \ldots, n$, siendo $F^{-1}(u) = u^{1/4}$. De acuerdo con el teorema anterior, los valores $X_1, \ldots, X_n$ están generados por la distribución de probabilidad cuya función de distribución acumulativa es F. El siguiente código R genera 1000 valores de la distribución F y los representa gráficamente:

```
n = 1000
U = runif(n)
X = U^(1/4)
hist(X, main = "Histograma de valores de X", ylab = "Frecuencia")
```



2.5.5. Normal

Se dice que una variable aleatoria tiene distribución *normal* de esperanza μ y desviación estándar σ ($X \cong N(\mu, \sigma)$) si su distribución de probabilidad puede expresarse por una función de densidad de la forma:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right)$$

La gráfica de esta distribución es la bien conocida campana de Gauss. Las funciones representadas en la figura 2.3 corresponden a distribuciones normales de esperanza 80 y desviaciones estándar 3, 5 y 8. Se deja como ejercicio probar que la función característica es:

$$\varphi(u) = \exp\left(i\mu u - \sigma^2 u^2/2\right)$$

De acuerdo con este resultado, puede verse que $E[X] = \mu$ y $\text{var}(X) = \sigma^2$.

Teorema 2.5. ***(de la tipificación).***

Si una variable aleatoria X tiene distribución de probabilidad $N(\mu, \sigma)$, entonces, $Z = (X - \mu)/\sigma$ tiene distribución de probabilidad $N(0, 1)$ (normal estándar o tipificada).

Demostración. Representaremos por φ_X y φ_Z las funciones características de X y Z respectivamente. Entonces:

$$\varphi_Z(u) = E[\exp(iu(X - \mu)/\sigma)] = \exp(-iu\mu/\sigma) E[i(u/\sigma) X] =$$

$$\exp(-iu\mu/\sigma) \varphi_X(u/\sigma) = \exp\left(-u^2/2\right)$$

El resultado se sigue del hecho de que $\varphi_Z(u) = \exp(-u^2/2)$ es la función característica de la distribución $N(0, 1)$. □

Para una variable aleatoria Z con distribución normal estándar ($Z \cong N(0, 1)$), representaremos:

- La función de distribución acumulativa por $\Phi(t)$, lo que significa que $\Phi(t) = P(Z \leq t)$.
- El cuantil α por z_α; esto es: $\Phi(z_\alpha) = \alpha$

2.5.6. Exponencial

Una variable aleatoria X se dice que tiene *distribución exponencial de parámetro* λ ($X \cong \exp(\lambda)$) si su función de densidad de probabilidad es de la forma:

$$f(x) = \frac{1}{\lambda} \exp(-x/\lambda) \ : \ x \geq 0$$

La función característica es:

$$\varphi(u) = \frac{1}{\lambda} \int_0^\infty \exp(-(1 - \lambda iu) x/\lambda) \, dx = \frac{1}{1 - \lambda iu}$$

De aquí se obtiene que $E[X] = \lambda$ y $\text{var}(X) = \lambda^2$.

La distribución exponencial se utiliza en muchos sistemas para modelar la duración de determinados elementos. Hay sin embargo una paradoja curiosa sobre esta distribución que veremos en el siguiente ejercicio.

Ejercicio 2.1. Para una variable aleatoria X con distribución de probabilidad exponencial de parámetro λ, obtener $P(X > s + t \mid X > s)$.

Nótese en primer lugar que $P(X > t) = \exp(-t/\lambda)$. Entonces:

$$\Pr(X > s + t \mid X > s) = \frac{\Pr(\{X > s\} \cap \{X > s + t\})}{\Pr(X > s)} = \frac{\Pr(X > s + t)}{\Pr(X > s)} =$$

$$\frac{\exp(-(s + t)/\lambda)}{\exp(-s/\lambda)} = \exp(-t/\lambda) = \Pr(X > t)$$

Interpretemos qué significa este resultado: si X representa la duración de un cierto objeto, entonces $\Pr(X > s + t \mid X > s)$ representa la probabilidad de que el objeto dure más de un tiempo $s + t$, sabiendo que ya ha durado un tiempo s. El hecho de que esta probabilidad sea igual a $\Pr(X > t)$ significa que si en un cierto instante el objeto tiene una edad s (por tanto que su duración es mayor a s), la probabilidad de que dure al menos un tiempo adicional t es independiente de su edad. Téngase en cuenta que conclusión sería incompatible con el hecho de que en muchos objetos, la duración adicional de vida depende de la edad actual.

2.5.7. Weibull

La *distribución de Weibull* se utiliza muy frecuentemente en problemas de teletráfico. Su función de distribución acumulativa es:

$$F(t) = 1 - \exp(-(t/\lambda)^{\kappa}) \;:\; t \geq 0$$

Los parámetros λ, κ reciben el nombre de parámetros de *escala* y *forma* respectivamente. En la figura 2.5 se muestra las densidades de probabilidad de la

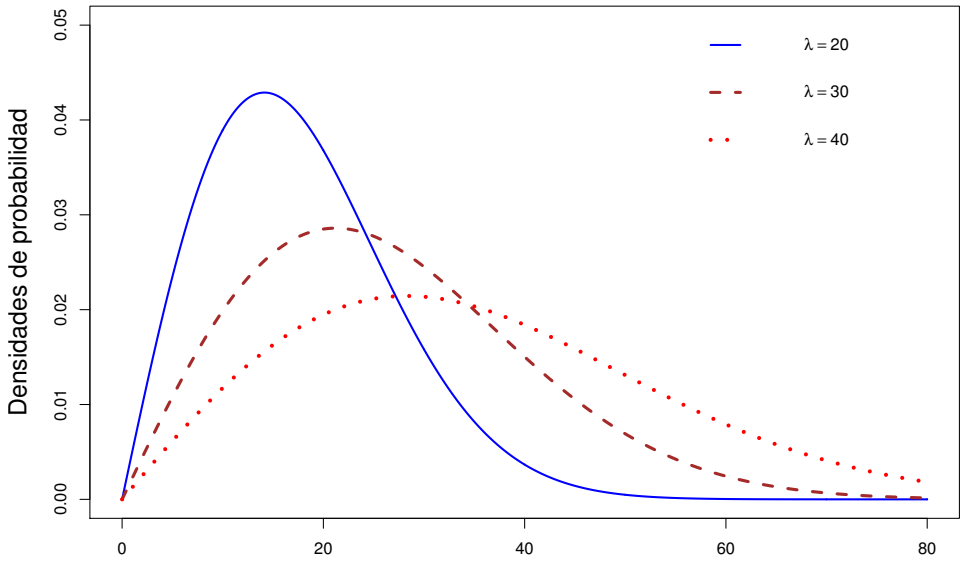


Figura 2.5: Densidades de probabilidad de la distribución de Weibull con el mismo parámetro de forma pero distintos parámetros de escala

distribución de Weibull para un mismo parámetro de forma ($\kappa = 2$) y distintos parámetros de escala ($\lambda = 20, 30, 40$).

Cuando $\kappa = 1$, la distribución de Weibull coincide con la exponencial de parámetro λ.

2.5.8. Gamma

Una variable aleatoria X tiene distribución *gamma* con parámetros α y λ ($X \cong \mathcal{G}(\alpha, \lambda)$) si su función de densidad es de la forma:

$$f(x) = \frac{1}{\lambda^{\alpha}\Gamma(\alpha)} x^{\alpha-1} \exp(-x/\lambda) \ : \ x \geq 0$$

siendo $\Gamma(\alpha) = \int_0^{\infty} x^{\alpha-1} \exp(-x)\, dx$. Nótese que para $\alpha = 1$ la distribución se reduce a la exponencial de parámetro λ.

Ejercicio 2.2. Obtener su función característica de la distribución gamma, y a partir de ella deducir la esperanza y la varianza de la distribución.

Problemas

1. Sea $F(t)$ la función de distribución acumulativa de una variable aleatoria X. Demostrar que la función acumulativa de λX es $F(t/\lambda)$.

2. Por un canal de comunicación con ruido se transmiten datos binarios en bloques de 16 dígitos binarios. La probabilidad de recibir un dígito erróneo es 0.1. Si suponemos que al ocurrir un error en un dígito no influye en lo que le pasa a los demás. Calcular:

   a) Número medio de errores por bloque.

   b) Varianza del número de errores por bloque.

   c) Probabilidad de que el número de errores por bloque sea mayor o igual a 5.

3. Un sistema de transmisión digital (módem) comete un error al transmitir un bit con probabilidad 10^{-4}. Se supone que los errores que se produzcan en las transmisiones son independientes. Un posible comprador hace el siguiente test al módem: envía un mensaje de 10^4 bit y decide comprarlo si el número de errores es a lo sumo de dos. ¿Cuál es la probabilidad de que lo adquiera?.

4. La duración en minutos de una conversación telefónica es una variable aleatoria X cuya función de densidad es $f(t) = K \exp(-t/5)$, para $t > 0$.

   a) Hallar K.

   b) Obtener $\Pr(X > 10)$.

   c) Probar que $\Pr(X > a + b \mid X > a) = \Pr(X > b)$. Interpretar el resultado.

5. La longitud de una determinada componente es una variable aleatoria con distribución normal con media 200 mm y una desviación estándar de 20 mm. Encontrar la probabilidad de que una componente seleccionada al azar tenga una longitud de:

   a) Entre 180 mm y 200 mm.

   b) Menor que 150 mm.

c) Mayor que 225 mm.

d) Entre 190 mm y 210 mm.

6. Una variable aleatoria discreta X se dice que tiene distribución de probabilidad geométrica de parámetro θ si su distribución de probabilidad viene especificada por $\Pr(X = t) = (1 - \theta)\,\theta^{t-1} \; : \; t = 1, 2, \ldots$.

   a) Probar que $\Pr(X = t)$ es una distribución de probabilidad.

   b) Hallar su función característica.

   c) Obtener $E[X]$ y $\text{var}(X)$.

   d) Para $a, b \in \mathbb{Z}$, probar que $\Pr(X > a + b \mid X > a) = \Pr(X > b)$.

Capítulo 3

Distribuciones multivariantes

La distribución de probabilidad de una variable aleatoria, estudiada en el capítulo anterior, es la herramienta natural para realizar predicciones acerca de los posibles valores que puede tomar la variable. Es frecuente sin embargo trabajar con sistemas en los que participan diversas variables que se condicionan unas a las otras. Para modelar tales sistemas se requiere utilizar la distribución simultánea del conjunto de variables que forman el sistema. En este capítulo se introducen las distribuciones multivariantes, a partir de las cuales pueden obtenerse las distribuciones condicionales. Estas son herramientas necesarias para realizar predicciones de unas variables a partir de otras ya observadas.

3.1. Distribución conjunta de dos variables aleatorias

Considérense las variables aleatorias X e Y definidas sobre un mismo espacio de probabilidad $(\Omega, \mathcal{F}, P)$. La función de distribución conjunta del vector (X, Y) se define por:

$$F(s,t) = \Pr(\{X \leq s\} \cap \{Y \leq t\})$$

Por simplicidad usaremos en lo que sigue la notación $F(s,t) = \Pr(X \leq s, Y \leq t)$.

Ejemplo 3.1. La tabla 3.1 muestra la distribución conjunta del vector aleatorio (X, Y). Nótese por ejemplo que $\Pr(X = 0, Y = 2) = 0{,}20$.

	Y		
X	1	2	3
0	0.30	0.20	0.15
1	0.10	0.20	0.05

Tabla 3.1: Distribución conjunta de dos variables aleatorias discretas

Diremos que el vector aleatorio (X, Y) tiene distribución absolutamente continua, si existe una función $f : \mathbb{R}^2 \to \mathbb{R}$, integrable y no negativa, tal que:

$$F(s,t) = \int_{-\infty}^{s} \int_{-\infty}^{t} f(x,y)\, dxdy$$

Obviamente,

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y)\, dxdy = 1$$

La función $f(x,y)$ recibe el nombre de función de densidad de probabilidad. En los casos en los que existe la función de densidad, la distribución de probabilidad recibe el nombre de distribución absolutamente continua. Puede también comprobarse la siguiente relación:

$$\Pr(a < X \leq b, c < Y \leq d) = \int_{a}^{b} \int_{c}^{d} f(x,y)\, dxdy$$

En los casos en los que la función $f(x,y)$ sea continua, la función $F(x,y)$ es derivable, y además:

$$\frac{\partial^2}{\partial x \partial y} F(x,y) = f(x,y)$$

Ejemplo 3.2. Sea $f(x,y) = \lambda xy \ : \ 0 \leq x \leq 1;\ 0 \leq y \leq x$. Determinar λ para que $f(x,y)$ sea función de densidad de probabilidad.

Solución. Nótese en primer lugar que si $\lambda > 0$, $f(x,y) \geq 0$ para $0 \leq x \leq 1;\ 0 \leq y \leq x$. Se tiene que verificar además:

$$1 = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\,dxdy = \int_0^1\int_0^x \lambda xy dxdy = \lambda\int_0^1 x\left\{\int_0^x ydy\right\}dx =$$

$$\frac{\lambda}{2}\int_0^1 x^3 dx = \frac{\lambda}{8} = 1 \implies \lambda = 8$$

Damos ahora un resultado a través del cual puede obtenerse la esperanza de la variable aleatoria $g(X,Y)$, siendo $g:\mathbb{R}^2\to\mathbb{R}$.

Teorema 3.1. *Sea (X,Y) un vector aleatorio con función de densidad de probabilidad $f(x,y)$ y $g:\mathbb{R}^2\to\mathbb{R}$ tal que:*

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |g(x,y)|\, f(x,y)\,dxdy < \infty$$

Entonces:

$$E[g(X,Y)] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\, f(x,y)\,dxdy$$

Ejemplo. Para la función del ejemplo 3.2 calculamos ahora $E[XY]$

$$E[XY] = 8\int_0^1\int_0^x x^2y^2 dxdy = 8\int_0^1 x^2\left\{\int_0^x y^2dy\right\}dx = \frac{8}{3}\int_0^1 x^5dx = \frac{4}{9}$$

Definimos finalmente la función característica para un vector aleatorio (X,Y) como:

$$\varphi(u,v) = E[\exp(i(uX+vY))]$$

3.2. Distribuciones marginales

La distribución conjunta de un vector aleatorio (X,Y) contiene información diversa, en particular la correspondiente a las distribuciones marginales de las variables

Xe Y.

En lo que sigue utilizaremos la notación $F_X(s) = \Pr(X \leq s)$ y $F_Y(t) = \Pr(Y \leq t)$. Nótese que:

$$F(s, \infty) = \Pr(X \leq s, Y \leq \infty) = \Pr(X \leq s) = F_X(s)$$

De forma análoga se tiene que $F(\infty, t) = F_Y(t)$.

Para un vector (X, Y) de variables aleatorias discretas, la distribución de probabilidad marginal de X se obtiene fácilmente por:

$$\Pr(X = s) = \sum_t \Pr(X = s, Y = t)$$

De la misma forma, la distribución marginal de Y es.

$$\Pr(Y = t) = \sum_s \Pr(X = s, Y = t)$$

Para la distribución de probabilidad del vector aleatorio (X, Y), la distribución marginal de X es: $\Pr(X = 0) = 0{,}65$; $\Pr(X = 1) = 0{,}35$.

Cuando la distribución conjunta es continua, las distribuciones marginales se obtienen de forma paralela tal como se indica en la siguiente proposición.

Proposición 3.1. *Sea (X, Y) un vector aleatorio con función de densidad $f(x, y)$. Entonces, las variables aleatorias Xe Y tienen funciones de densidad f_X y f_Y respectivamente, siendo:*

$$f_X(x) = \int_{-\infty}^{\infty} f(x, y)\, dy$$

y

$$f_Y(y) = \int_{-\infty}^{\infty} f(x, y)\, dx$$

Demostración.

$$F_X(s) = F(s, \infty) = \int_{-\infty}^{s} \left\{ \int_{-\infty}^{\infty} f(x, y)\, dy \right\} dx = \int_{-\infty}^{s} f_X(x)\, dx$$

Lo anterior demuestra que f_X es la función de densidad de la variable aleatoria X. De forma análoga se prueba que f_Y es la función de densidad de Y. □

Ejemplo 3.3. Para la distribución de probabilidad del ejemplo 3.1 se tiene:

$$f_X(x) = \int_{-\infty}^{\infty} f(x,y)\,dy = 8\int_0^x xy dy = 4x^3 \;:\; 0 \le x \le 1$$

Análogamente:

$$f_Y(y) = \int_{-\infty}^{\infty} f(x,y)\,dx = 8\int_y^1 xy dx = 4y\left(1-y^2\right) \;:\; 0 \le y \le 1$$

Para la función característica se tiene:

$$\varphi(u,0) = E\left[\exp(iuX)\right] = \varphi_X(u)$$

siendo naturalmente $\varphi_X(u)$ la función característica de la variable aleatoria X.

Estamos ahora en condiciones de dar el segundo teorema de linealidad de la esperanza, a saber: la esperanza de una suma de variables aleatorias es igual a la suma de las esperanzas.

Teorema 3.2. ***(2ª propiedad de linealidad de la esperanza)***

Sea (X,Y) un vector aleatorio definido sobre un espacio de probabilidad $(\Omega, \mathcal{F}, P)$. Entonces, $E[X+Y] = E[X] + E[Y]$

Demostración. (sólo para el caso de distribución absolutamente continua).

Supóngase que la distribución de (X,Y) viene especificada por la densidad $f(x,y)$. entonces:

$$E[X+Y] = \iint (x+y) f(x,y)\,dxdy = \iint x f(x,y)\,dxdy + \iint y f(x,y)\,dxdy =$$

$$\int x\left\{\int f(x,y)\,dy\right\}dx + \int y\left\{\int f(x,y)\,dx\right\}dy =$$

$$\int x f_X(x)\,dx + \int y f_Y(y)\,dy = E[X] + E[Y]$$

□

3.3. Distribuciones condicionales

Si dos variables aleatorias X e Y son discretas, su distribución de probabilidad está determinada por la función $P(X = s, Y = t)$, a lo largo de los valores (s, t) del conjunto donde se concentra la distribución de probabilidad. En este caso, la distribución condicional de la variable Y por la variable X viene dada por la expresión:

$$\Pr(Y = t \mid X = s) = \frac{\Pr(X = s, Y = t)}{\Pr(X = s)}$$

Obsérvese que esta definición es válida si $P(X = s) > 0$. Es fácil probar que para cada valor fijo de s, es una distribución de probabilidad en el sentido de que:

$$\sum_t \Pr(Y = t \mid X = s) = 1$$

Definimos a continuación la probabilidad condicionada $\Pr(Y \leq t \mid X = x)$ para X e Y continuas, en el caso de que la distribución de (X, Y) tenga función de densidad $f(x, y)$ (esto supone que $\Pr(X = s) = 0$ y por tanto, pierde validez la definición anterior). Nótese que para cada valor fijo de t, la probabilidad anterior es una función de x. Tal función es la única solución de la ecuación;

$$F(s, t) = \int_{-\infty}^{s} \Pr(Y \leq t \mid X = x)\, f_X(x)\,dx$$

Puede comprobarse que $\Pr(Y \leq t \mid X = x) = (1/f_X(x)) \int_{-\infty}^{t} f(x, y)\,dy$. De aquí se sigue que la densidad de probabilidad de la variable aleatoria Y condicionada por $X = x$ es:

$$f(y \mid x) = \frac{f(x,y)}{f_X(x)}$$

Nótese que $f(y \mid x)$ es una función de densidad en y para cada valor fijo de x.

Ejemplo. Para la distribución de probabilidad del ejemplo 3.1, la densidad condicional de Y por $X = x$ es:

$$f(y \mid x) = \frac{f(x,y)}{f_X(x)} = \frac{8xy}{4x^3} = \frac{2y}{x^2} \ : \ 0 \leq x \leq 1\,;\, 0 \leq y \leq x$$

Se tiene tiene en general que:

$$E[Y \mid X = x] = \int_{-\infty}^{\infty} y f(y \mid x)\, dy$$

Aplicando esta expresión a la distribución del ejemplo 3.1 queda:

$$E[Y \mid X = x] = \frac{2}{x^2}\int_0^x y^2 dy = \frac{2}{3}x \ : \ 0 \leq x \leq 1$$

El siguiente algoritmo tiene como finalidad simular 10,000 valores de la distribución conjunta del vector aleatorio (X, Y) de este ejemplo:

1. Simular $n = 1000$ valores de la distribución marginal de X. Dado que su densidad marginal es $f_X(x) = 4x^3 \ : \ 0 \leq x \leq 1$, la función acumulativa es $F(x) = x^4 \ : 0 \leq x \leq 1$. Se simulan ahora los valores $U_i \cong U[0,1] \ : \ i = 1, \ldots, 100$. De acuerdo con el teorema 2.4, los valores $X_i = U_i^{1/4} \ : \ i = 1, \ldots, 100$ tienen la distribución especificada por F_X.

2. Para cada uno de los valores X_i, simular 10 valores con distribución dada por la condicional $f(y \mid x)$. Procediendo de forma similar al punto anterior, simulamos valores $V_j \cong U[0,1] \ : \ j = 1, \ldots, 10$. Los valores $Y_{i,j} = X_i\sqrt{V_j}$ siguen la distribución de probabilidad especificada por la densidad condicional $f(y \mid x)$.

3. El conjunto de puntos $\{(X_i, Y_{i,j}) \ : \ i = 1, \ldots 1000\,;\, j = 1, \ldots, 10\}$ obedecerán por tanto a la distribución conjunta especificada por la función de distribución $F(s,t)$.

Se muestra a continuación el algoritmo en el lenguaje R. La salida se resume en la figura 3.1.

```
n = 1000
U = runif(n)
X = U^(1/4)
M = NULL
for (i in 1:n) {
    V = runif(10)
    Y = X[i] * sqrt(V)
    for (j in 1:10) M = rbind(M, c(X[i], Y[j]))
}
```

3.4. Independencia estocástica

Sea un vector aleatorio (X, Y) con función de distribución conjunta $F(s,t)$. Se dice que las variables X e Y son estocásticamente independientes o simplemente, independientes, si se satisface la condición:

$$F(s,t) = F_X(s) F_Y(t) \quad : \quad \forall s,t$$

Si el vector aleatorio (X, Y) tiene tiene función de densidad $f(x,y)$, la independencia es equivalente a que:

$$f(x,y) = f_X(x) f_Y(y)$$

Puede comprobarse fácilmente que las variables aleatorias X e Y cuya distribución conjunta se dio en el ejercicio 3.1 no son independientes.

Proposición 3.2. Si X e Y son independientes, entonces $E[XY] = E[X] E[Y]$

Demostración. (sólo para el caso de distribución absolutamente continua).

$$E[XY] = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} xy f_X(x) f_Y(y) \, dxdy =$$

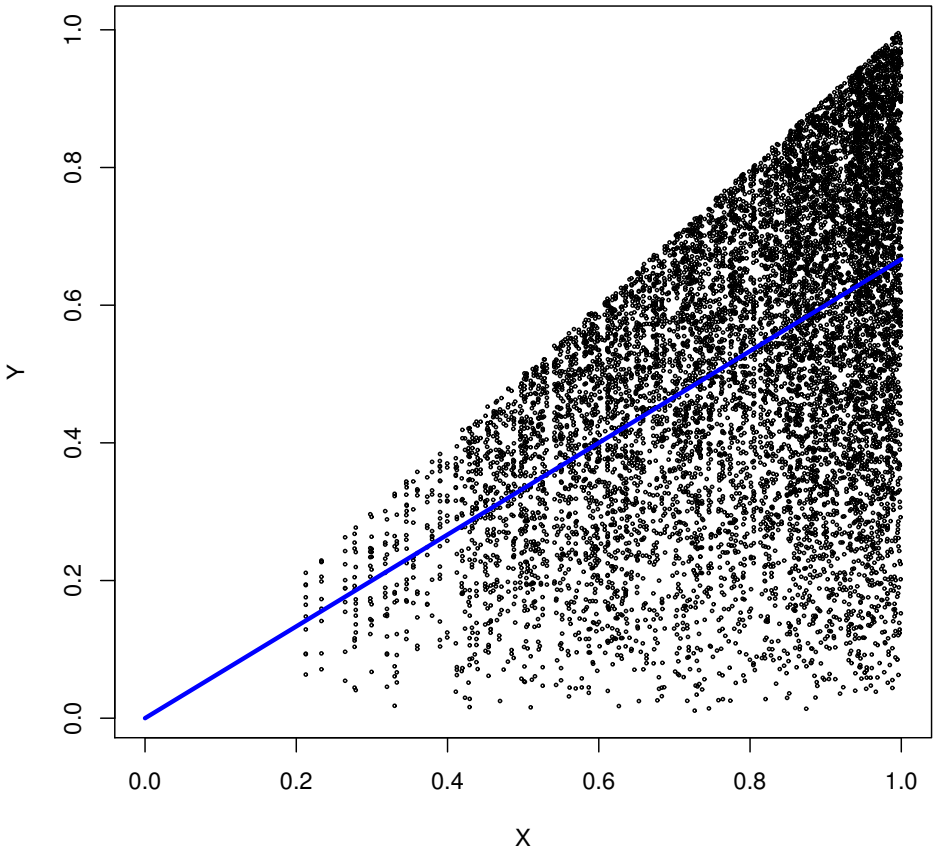


Figura 3.1: Simulación de 10,000 valores de la distribución conjunta del vector aleatorio (X, Y) del ejemplo 3.2.

$$\int_{-\infty}^{\infty} x \left\{ \int_{-\infty}^{\infty} y f_Y(y)\, dy \right\} f_X(x)\, dx = E[X]\, E[Y]$$

□

Proposición 3.3. *Si X e Y son independientes, entonces $var(X+Y) = var(X) + var(Y)$*

Demostración. Sean $\mu_X = E[X]$ y $\mu_Y = E[Y]$. Entonces, $E[X+Y] = \mu_X + \mu_Y$, y además:

$$\text{var}\,(X+Y) = E\left[(X+Y-\mu_X-\mu_Y)^2\right] = \text{var}\,(X)+\text{var}\,(Y)+2E\left[(X-\mu_X)\,(Y-\mu_y)\right]$$

El resultado se sigue de la independencia de X e Y, dado que:

$$E\left[(X-\mu_X)\,(Y-\mu_y)\right] = E\left[(X-\mu_X)\right]E\left[(Y-\mu_y)\right] = 0$$

□

Proposición 3.4. *Si X e Y son independientes, entonces, la función característica conjunta satisface que:*

$$\varphi\,(u,v) = \varphi_X\,(u)\,\varphi_Y\,(v)$$

siendo φ_X y φ_Y las funciones características de X e Y respectivamente.

3.5. Correlación lineal

La asociación lineal entre dos variables puede medirse mediante el coeficiente de correlación lineal de Pearson. Su definición requiere introducir previamente el concepto de *covarianza* entre las dos variables. Esta se define por:

$$\text{cov}\,(X,Y) = E\left[(X-\mu_X)\,(Y-\mu_y)\right]$$

El *coeficiente de correlación lineal de Pearson* se define entonces como:

$$\rho_{Y,Y} = \frac{\text{cov}\,(X,Y)}{\sigma_X\sigma_Y}$$

siendo σ_X^2 y σ_Y^2 las varianzas de X e Y respectivamente.

Proposición 3.5. *Si X e Y son independientes, entonces $\rho_{X,Y} = 0$*

La demostración de este resultado es inmediata.

Cuando $\rho_{X,Y} = 0$, las variables X e Y se dicen *incorreladas.*

De acuerdo con la proposición anterior, la independencia entre dos variables implica la incorrelación. Lo contrario en general no es cierto como se ve en el siguiente ejemplo.

Ejemplo 3.4. Sea (X, Y) un vector aleatorio con distribución uniforme en el círculo de radio unidad centrado en el origen de coordenadas. La función de densidad de probabilidad es entonces $f(x, y) = 1/\pi \ : \ -1 \leq x \leq 1 \, ; \, -\sqrt{1-x^2} \leq y \leq \sqrt{1-x^2}$. Probaremos que estas variables no son independientes pero si incorreladas. En efecto: para $-1 \leq x \leq 1$, la densidad marginal para la variable aleatoria X es:

$$f_X(x) = \int_{-\sqrt{1-x^2}}^{\sqrt{1-x^2}} f(x, y)\, dy = \frac{2}{\pi}\sqrt{1-x^2} \ \ : \ -1 \leq x \leq 1$$

De esta forma, su esperanza es:

$$E[X] = \int_{-\infty}^{\infty} x f_X(x)\, dx = \frac{2}{\pi}\int_{-1}^{1} x\sqrt{1-x^2} \cdot dx = 0$$

El resultado se sigue del hecho de ser $x\sqrt{1-x^2}$ una función impar. En orden a obtener la densidad marginal para la variable aleatoria Y, nótese que la densidad conjunta puede expresarse alternativamente por: $f(x, y) = 1/\pi \ : \ -1 \leq y \leq 1 \, ; \, -\sqrt{1-y^2} \leq x \leq \sqrt{1-y^2}$. Ello obviamente implica, para $-1 \leq y \leq 1$:

$$f_Y(y) = \int_{-\sqrt{1-y^2}}^{\sqrt{1-y^2}} f(x, y)\, dx = \frac{2}{\pi}\sqrt{1-y^2} \ \ : \ -1 \leq y \leq 1$$

De donde se sigue que $E[Y] = 0$. Nótese que:

$$f(x, y) \neq f(x) \cdot f(y)$$

Ello significa que las variables no son independientes. Para ver que son incorreladas obtenemos en primer lugar:

$$E[XY] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} xy f(x, y)\, dxdy = \frac{1}{\pi}\int_{-1}^{1} x \left\{ \int_{-\sqrt{1-x^2}}^{\sqrt{1-x^2}} y dy \right\} dx = 0$$

Se tiene entonces:

$$\operatorname{cov}(X,Y) = E[XY] - E[X]\,E[Y] = 0$$

Se tiene finalmente:

$$\rho_{Y,Y} = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y} = 0$$

lo que significa que las variables son incorreladas.

3.6. Distribuciones multivariantes generales

Los conceptos que se han desarrollado en los epígrafes anteriores para dos variables aleatorias pueden generalizarse fácilmente a un vector de variables aleatorias.

3.6.1. Distribución conjunta de un vector aleatorio

Dado el vector de variables aleatorias $(X_1, \ldots, X_n)$, su distribución de probabilidad conjunta se determina por:

$$F(t_1, \ldots, t_n) = \Pr(X_1 \leq t_1, \ldots, X_n \leq t_n)$$

A partir de la distribución conjunta puede determinarse la distribución marginal $F_i(t_i)$ de cada una de las variables X_i. Así, por ejemplo, a partir de la distribución conjunta $F(t_1, t_2, t_3)$ del vector aleatorio (X_1, X_2, X_3) se obtiene la distribución marginal de X_2 como:

$$F_2(t_2) = F(\infty, t_2, \infty)$$

La distribución de probabilidad de $(X_1, \ldots, X_n)$ se dice absolutamente continua si existe una función $f(x_1, \ldots, x_n) \geq 0$ tal que:

$$F(t_1, \ldots, t_n) = \int_{-\infty}^{t_1} \cdots \int_{-\infty}^{t_n} f(x_1, \ldots, x_n)\, dx_1 \cdots dx_n$$

La función $f(x_1, \ldots, x_n)$ recibe el nombre de función de densidad de probabilidad conjunta.

3.6.2. Independencia multivariante

Las variables aleatorias $X_1, \ldots, X_n$, con distribución conjunta $F(t_1, \ldots, t_n)$ son independientes si se satisface a condición:

$$F(t_1, \ldots, t_n) = F_1(t_1) \cdot \cdots \cdot F_n(t_n) \ : \ \forall t_1, \ldots, t_n$$

En el caso de que exista función de densidad $f(x_1, \ldots, x_n)$, la independencia es equivalente a:

$$f(x_1, \ldots, x_n) = f_1(x_1) \cdots f_n(x_n)$$

donde f_i es la densidad de probabilidad marginal de la variable X_i.

3.7. Sucesiones de variables aleatorias independientes

Consideramos en esta sección sucesiones de variables aleatorias $\{X_n : n \in \mathbb{N}\}$ independientes, entendiendo por tal que para todo valor de n fijo, las variables $X_1, \ldots, X_n$ son independientes.

Teorema 3.3. *Sea $\{X_n : n \in \mathbb{N}\}$ una sucesión de variables aleatorias independientes e idénticamente distribuidas tales que $E[X_n] = \mu$ y $var(X_n) = \sigma^2$. Sea $\hat{\mu}_n = (1/n)\sum_{i=1}^{n} X_i$ (media aritmética de las n primeras observaciones). Entonces, para $n \to \infty$*

$$E\left[(\hat{\mu}_n - \mu)^2\right] \to 0$$

Demostración. De las propiedades de linealidad de la esperanza se tiene que:

$$E\left[\hat{\mu}_n\right] = E\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right] = \frac{1}{n}\sum_{i=1}^{n} E\left[X_i\right] = \mu$$

De la independencia de las variables X_n se sigue:

$$\text{var}\left(\hat{\mu}_n\right) = \text{var}\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) = \frac{1}{n^2}\sum_{i=1}^{n} \text{var}\left(X_i\right) = \frac{\sigma^2}{n}$$

El resultado se sigue de ser $E\left[\left(\hat{\mu}_n - \mu\right)^2\right] = \text{var}\left(\hat{\mu}_n\right)$ □

Este tipo de convergencia recibe el nombre de *convergencia en media cuadrática* ($\hat{\mu}_n \to \mu$ en media cuadrática)

Un problema frecuente es el del cálculo de la distribución de probabilidad de una suma de variables aleatorias independientes. Cuando las variables $X_1, \ldots, X_n$, además de ser independientes comparten una distribución de probabilidad común (están idénticamente distribuidas) y además $E\left[X_i\right] = \mu$ y $\text{var}\left(X_i\right) = \sigma^2$, es inmediato que:

1. $E\left[\sum_{i=1}^{n} X_i\right] = n\mu$
2. $\text{sd}\left(\sum_{i=1}^{n} X_i\right) = \sigma\sqrt{n}$

El siguiente teorema prueba que la suma de variables tipificada tiene una distribución de probabilidad que converge a la normal estándar.

Teorema 3.4. ***(central del límite)*** *Sea* $\{X_n : n \in \mathbb{N}\}$ *una sucesión de variables aleatorias independientes e idénticamente distribuidas tales que* $E\left[X_n\right] = \mu$ *y* $var\left(X_n\right) = \sigma^2$*. Entonces, para* $n \to \infty$*:*

$$\Pr\left(\frac{\sum_{i=1}^{n} X_i - n\mu}{\sigma\sqrt{n}} \leq t\right) \to \Phi\left(t\right)$$

Problemas

1. Sea un vector aleatorio (X, Y) con distribución de probabilidad conjunta especificada por la función de densidad $f\left(x, y\right) = x + y$ para $0 < x, y < 1$.

a) Hallar las distribuciones marginales de X e Y.

b) Hallar $E[Y \mid X = x]$.

c) Calcular el coeficiente de correlación entre X e Y.

2. Sea el vector aleatorio (X, Y) cuya distribución de probabilidad viene especificada por la función de densidad de probabilidad conjunta:

$$f(x, y) = \frac{K}{x} \ : \ x > 1 \ ; \ 0 \leq y \leq \frac{1}{x}$$

a) Determinar el valor de K.

b) Hallar $E[Y \mid X = x]$ y representarla gráficamente sobre el soporte de la distribución.

3. Sea un vector aleatorio (X, Y) con distribución de probabilidad conjunta especificada por la función de densidad $f(x, y) = \lambda y^2 \exp(-x)$ para $x \geq 0$, $|y| \leq x$.

a) Obtener el valor de λ.

b) Hallar las distribuciones marginales de X e Y.

c) Obtener $E[Y \mid X = x]$.

4. Considérense las variables aleatorias X e Y cuya distribución de probabilidad conjunta viene especificada por la función de densidad: $f(x, y) = 3/4$: $-1 \leq x \leq 1$; $0 \leq y \leq 1 - x^2$.

a) Obtener $E[Y|X = x]$ y representarla gráficamente como función de x.

b) Hallar $\Pr(2Y > X + 1)$.

5. Sea $X \cong b(1, p)$ e $Y \cong b(1, q)$. Demostrar que son independientes si y sólo si son incorreladas.

6. Sean $X_1, \ldots, X_n$ variables aleatorias independientes y con la distribución de probabilidad común $U[0, 1]$. Hallar la distribución de probabilidad de la variable $Z = \text{máx}_i \, X_i$

7. Sean $X_1, \ldots, X_n$ variables aleatorias independientes y con la distribución de probabilidad común exponencial de parámetro λ. Probar que $\sum_{i=1}^{n} X_i \cong \mathcal{G}(n, \lambda)$. (Indicación: obtener la función característica de $\sum_{i=1}^{n} X_i$.

8. Considérese un fenómeno aleatorio en el que un suceso A tiene probabilidad p de ocurrir. Supóngase que el fenómeno se observa n veces en las mismas condiciones siendo los sucesivos resultados independientes unos de otros. Sea X_i la variable aleatoria definida como 1 ó 0 según ocurra o no A en la i-ésima observación. Nótese que $f_n(A) = (1/n) \sum_{i=1}^{n} X_i$ es la frecuencia relativa de las ocurrencias de A.

   a) Probar que $E[f_n(A)] = \Pr(A)$.

   b) $\operatorname{var}(f_n(A)) = \Pr(A)(1 - \Pr(A))/n$.

   c) $E\left[(f_n(A) - \Pr(A))^2\right] \to 0$, para $n \to \infty$. (interpretar el resultado)

Capítulo 4

Inferencia estadística

Los datos manejados en diversos campos de la ciencia proceden habitualmente de la observación de variables aleatorias. Así por ejemplo, los periodos transcurridos entre los accesos consecutivos a un sistema son datos generados por variables aleatorias que pueden compartir una distribución de probabilidad común. Tal distribución modela el *patrón* (causas subyacentes) generador de los datos. A través de ella es posible predecir futuros valores. Se hará necesario, por tanto, estimarla en aquellos casos en los que no sea conocida. Los datos generados por la distribución constituyen uno de los pilares esenciales para su estimación, lo que otorga a los métodos de estimación el carácter de procesos inductivos, que van de la particularidad de los datos a la universalidad de la distribución.

4.1. Muestra aleatoria simple

Para una distribución de probabilidad $\mathcal{P}$, una *muestra aleatoria* es un conjunto de variables aleatorias $X_1, \ldots, X_n$ independientes y con distribución común $\mathcal{P}$.

Como sabemos, la distribución de probabilidad de cualquier variable aleatoria queda determinada por su función de distribución acumulativa $F(t)$. Veremos seguidamente que esta función puede reproducirse a partir de muestras aleatorias aleatorias de tamaño creciente a través de la llamada *distribución empírica.*

Para una muestra aleatoria $X_1, \ldots, X_n$ de una distribución de probabilidad con función acumulativa $F(t)$, la función de *distribución empírica* se define por:

$$\hat{F}_n(t) = \frac{1}{n}\sum_{i=1}^{n} I(X_i \leq t)$$

donde $I(X_i \leq t)$ es uno o cero según ocurra o no el suceso $\{X_i \leq t\}$. Nótese que para cada valor fijo de t, $I(X_i \leq t)$ son variables aleatorias independientes con distribución de probabilidad $b(1, F(t))$.

Teorema 4.1. ***(de Glivenko-Cantelli)***

Para cualquier valor t se verifica, para $n \to \infty$:

$$E\left[\left(\hat{F}_n(t) - F(t)\right)^2\right] \to 0$$

Demostración. Obtenemos en primer lugar la esperanza y varianza de $\hat{F}_n(t)$, para cualquier valor de t.

$$E\left[\hat{F}_n(t)\right] = \frac{1}{n}\sum_{i=1}^{n} E\left[I(X_i \leq t)\right] = F(t)$$

$$\text{var}\left(\hat{F}_n(t)\right) = \frac{1}{n^2}\sum_{i=1}^{n} \text{var}\left(I(X_i \leq t)\right) = \frac{1}{n}F(t)(1 - F(t))$$

Lo anterior significa que:

$$E\left[\left(\hat{F}_n(t) - F(t)\right)^2\right] = \text{var}\left(\hat{F}_n(t)\right) = \frac{1}{n}F(t)(1 - F(t)) \to 0$$

para $n \to \infty$. □

En la figura 4.1 aparecen superpuestas la función de distribución acumulativa de la distribución de Weibull de parámetros $\kappa = 2$ y $\lambda = 10$ y la distribución empírica obtenida de una muestra aleatoria de tamaño $n = 100$. El lector puede i

maginar que según se aumenta el tamaño muestral, la función empírica tenderá a confundirse con la teórica.

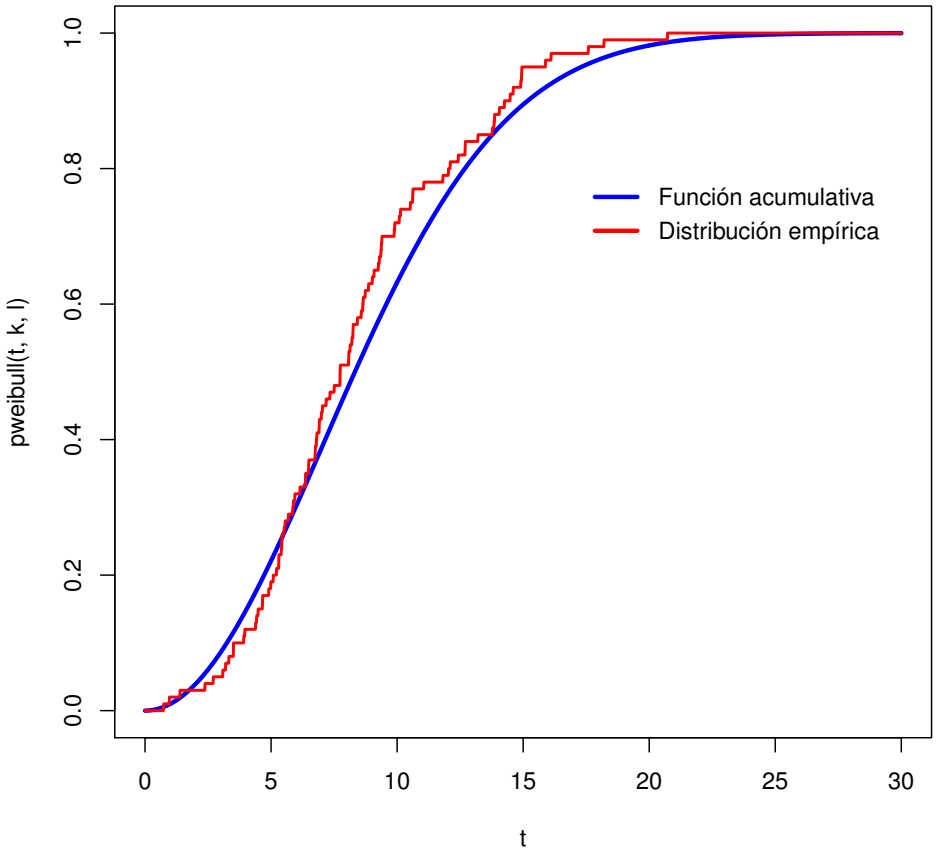


Figura 4.1: Distribución de Weibull: función acumulativa y distribución empírica

4.2. Concepto de estimador puntual

Los problemas de *inferencia estadística paramétrica* son aquellos en los que se pretende estimar una distribución de probabilidad cuando se conoce su forma salvo un número finito de parámetros.

Consideraremos por tanto una distribución de probabilidad F_θ, siendo θ un parámetro desconocido de dimensión finita. Supondremos además que $X_1, \ldots, X_n$ es una muestra aleatoria de la distribución F_θ. Un *estimador* para θ es cualquier función de la muestra que expresamos en la forma que sigue:

$$\hat{\theta}_n = \hat{\theta}_n\left(X_1, \ldots, X_n\right)$$

Nótese que $\hat{\theta}_n$ por ser una función de la muestra es una variable aleatoria. La bondad o fiabilidad del estimador se basará en que todas las posibles estimaciones sean valores muy próximos al verdadero valor del parámetro. Estos aspectos se tratarán en la sección 4.4.

Ejemplo 4.1. Sea $X_1, \ldots, X_n$ una muestra aleatoria de la distribución $U[0, \theta]$. Un estimador natural para el parámetro θ es $\hat{\theta}_n = \text{máx}_{,i}\, X_i$.

4.3. El método de la máxima verosimilitud

Considérese una distribución de probabilidad especificada por su función de densidad f_θ, la cual depende del parámetro desconocido θ. Para su estimación se dispone de una muestra aleatoria $X_1, \ldots, X_n$ de la distribución especificada por f_θ. La idea de la máxima verosimilitud consiste en elegir como estimador de θ el valor que hace más verosímil el conjunto de datos observados.

Dado que las variables $X_1, \ldots, X_n$ son independientes, su distribución conjunta es:

$$f_\theta(x_1, \ldots, x_n) = f_\theta(x_1) \cdots f_\theta(x_n)$$

Para una muestra observada $X_1, \ldots, X_n$, la función de θ, $L(\theta) = f_\theta(X_1, \ldots, X_n)$ recibe el nombre de función de verosimilitud. Supóngase ahora que la función $L(\theta)$ tiene un máximo y sea :

$$\hat{\theta}_n = arg\,\text{máx}\, L(\theta)$$

Entonces, $\hat{\theta_n}$ es por definición el estimador de máxima verosimilitud.

Ejemplo 4.2. Sea $X_1, \ldots, X_n$ una muestra aleatoria de la distribución exponencial de parámetro θ. La función de verosimilitud tiene la forma:

$$L(\theta) = \frac{1}{\theta^n} \exp\left(-\sum_{i=1}^{n} X_i/\theta\right)$$

Maximizar la función $L(\theta)$ es equivalente a maximizar $\ell(\theta) = \log L(\theta)$. Se tiene que:

$$\ell(\theta) = -n\log\theta - \frac{1}{\theta}\sum_{i=1}^{n} X_i$$

Nótese que la maximización de $l(\theta)$ es más sencilla que la de $L(\theta)$. Para obtener el máximo de la función hacemos:

$$\ell'(\theta) = -\frac{n}{\theta} + \frac{1}{\theta^2}\sum_{i=1}^{n} X_i = 0$$

Se obtiene finalmente que $\hat{\theta}_n = (1/n)\sum_{i=1}^{n} X_i$.

Ejercicio 4.1. Probar que el estimador del ejemplo 4.1 es el de máxima verosimilitud.

4.4. Optimalidad de los estimadores

4.4.1. Estimación centrada y error estándar

Un estimador $\hat{\theta}_n$ para un parámetro θ se dice *centrado o insesgado* si $E\left[\hat{\theta}_n\right] = \theta$.

A la diferencia $b\left(\hat{\theta}\right) = E\left[\hat{\theta}_n\right] - \theta$ se le llama *sesgo* del estimador.

A la desviación estándar de un estimador se le llama *error estándar*.

4.4.2. Error cuadrático medio

El error cuadrático medio de un estimador $\hat{\theta}_n$ para un parámetro θ se define por:

$$R\left(\hat{\theta}_n, \theta\right) = E\left[\left(\hat{\theta}_n - \theta\right)^2\right]$$

El siguiente resultado prueba que el error cuadrático medio de cualquier estimador es igual a su varianza más el cuadrado del sesgo.

Proposición 4.1. *Sea $\hat{\theta}_n$ un estimador para un parámetro θ. Entonces,*

$$R\left(\hat{\theta}_n, \theta\right) = var\left(\hat{\theta}_n\right) + b\left(\hat{\theta}_n\right)^2$$

Demostración.

$$R\left(\hat{\theta}_n, \theta\right) = E\left[\left(\hat{\theta}_n - \theta\right)^2\right] = R\left(\hat{\theta}_n, \theta\right) = E\left[\left(\hat{\theta}_n - E\left[\hat{\theta}_n\right] + E\left[\hat{\theta}_n\right] - \theta\right)^2\right] =$$

$$E\left[\left(\hat{\theta}_n - E\left[\hat{\theta}_n\right]\right)^2\right] - 2\left(E\left[\hat{\theta}_n\right] - \theta\right) E\left[\hat{\theta}_n - E\left[\hat{\theta}_n\right]\right] + \left(E\left[\hat{\theta}_n\right] - \theta\right)^2 =$$

$$\text{var}\left(\hat{\theta}_n\right) + b\left(\hat{\theta}_n\right)^2$$

□

4.4.3. Estimación consistente

Un estimador $\hat{\theta}_n$ para un parámetro θ es *consistente* si, para $n \to \infty$

$$R\left(\hat{\theta}_n, \theta\right) \to 0$$

Problemas

1. Probar que el estimador del ejemplo 4.1 es sesgado y consistente.
2. Probar que el estimador del ejemplo 4.2 es consistente.
3. Sea $X_1, \ldots, X_n$ una muestra aleatoria de la distribución $N(\mu, \sigma)$. Hallar el estimador de máxima verosimilitud para el parámetro $\theta = (\mu, \sigma)$.

Capítulo 5

Cadenas de Markov

Muchas veces el fenómeno de interés en nuestro estudio se caracteriza por evolucionar en el tiempo: el patrón de tráfico de datos que se recibe en un router, la señal eléctrica que se registra en un electrocardiograma o la temperatura de funcionamiento de un microcircuito. En estos casos, el fenómeno puede describirse a través de una sucesión de variables aleatorias, que recibe el nombre de *proceso estocástico.* En este capítulo estudiaremos las *cadenas de Markov* que son procesos caracterizados por el hecho de que una vez que se tenga información sobre su presente, el conocimiento de su pasado no mejora las predicciones que podamos hacer sobre su futuro.

5.1. El recorrido aleatorio.

Considérese una partícula que se mueve a lo largo de un eje horizontal ocupando las posiciones correspondientes a los números enteros. Inicialmente la partícula se encuentra en la posición $X_0 = 0$, y en cada etapa $n \in \mathbb{N}$, la partícula sufre un desplazamiento aleatorio Z_n, cuya distribución de probabilidad se especifica en la siguiente tabla:

k	1	-1	0
$\Pr(Z_n = k)$	p	q	$1-p-q$

Supondremos además que los desplazamientos que forman la sucesión $\{Z_n : n \in \mathbb{N}\}$ son independientes. De esta forma, la posición de la partícula en la etapa n puede expresarse como:

$$X_n = X_{n-1} + Z_n$$

La sucesión de variables aleatorias $\{X_n; n \in \mathbb{N}\}$ recibe el nombre de *recorrido aleatorio.* De esta definición pueden obtenerse las probabilidades condicionadas:

$$p_{i,j} = \Pr(X_n = j \mid X_{n-1} = i) = \begin{cases} p & j = i+1 \\ q & j = i-1 \\ 1-p-q & j = i \end{cases}$$

Esta es una característica interesante del recorrido aleatorio $\{X_n; n \in \mathbb{N}\}$: una vez conocida su posición en una etapa $n-1$, no hace falta más información (es irrelevante saber cómo ha llegado a la posición que ocupa en dicha etapa) para hacer una predicción sobre cuál va a ser su posición en la etapa n. Esta característica se conoce como *condición de Markov*, y aquellas sucesiones de variables aleatorias que la cumplen se denominan *cadenas de Markov.*

El recorrido aleatorio es muy fácil de simular en R. Para ello definimos una función que recibe como entrada tres parámetros:

- n: número de etapas a simular.
- $probs = (\Pr(Z_i = -1), \Pr(Z_i = 0), \Pr(Z_i = 1)) = (q, 1-p-q, p)$: la distribución de probabilidad de Z_i.
- X_0 : Posición inicial de la partícula (por defecto será $X_0 = 0$).

Como salida, la función devuelve la sucesión de valores $X_1, X_2, \ldots, X_n$:

```
randomWalk = function(n, probs, X0 = 0) {
    pac = cumsum(probs)
    a = runif(n)
    Z = ifelse(a < pac[1], -1, ifelse(a < pac[2], 0, 1))
```

```
    X0 + cumsum(Z)
}
```

Seguidamente utilizamos esta función para simular 20 etapas de un recorrido aleatorio que empieza en el origen y que tiene por distribución para la magnitud del salto en cada etapa, `probs=(0.2, 0.5, 0.3)`:

```
rw = randomWalk(n = 20, probs = c(0.2, 0.5, 0.3))
rw
```

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

El siguiente código nos permite hacer un gráfico de este recorrido aleatorio, que se muestra en la figura 5.1:

```
plot(rw, xlab = "Etapa", ylab = "Posici<U+00F3>n de la part<U+00ED>cula",
    type = "b", main = "Recorrido Aleatorio", col = "red")
```

5.2. Condición de Markov.

Una sucesión de variables aleatorias de la forma $\{X_n;\, n \in \mathbb{N}\}$ definidas sobre un mismo espacio de probabilidad (Ω, S, P) y con valores en un conjunto E finito o numerable $\{X_n : \Omega \to E\}$ verifica la *condición de Markov* si $\forall n \in N$; $i_0,\, i_1, \cdots,\, i,\, j \in E$:

$$\Pr\left(X_{n+1} = j \,|X_0 = i_0,\, X_1 = i_1,\, \ldots,\, X_n = i\right) = \Pr\left(X_{n+1} = j \,|X_n = i\right)$$

Como hemos visto en el ejemplo del recorrido aleatorio, esta condición significa que la probabilidad de que X_{n+1} tome un valor arbitrario j cuando se conoce *toda la historia anterior* (esto es, todos los valores que han tomado X_0,X_1,...,X_n) es la misma que cuando se conoce *solamente el último valor* X_n. Esta propiedad se suele enunciar diciendo que el pasado $(X,\, X_1,\, \ldots,\, X_{n-1})$ no aporta información para la predicción del futuro (X_{n+1}) una vez que se conoce el presente (X_n). Las sucesiones que verifican la propiedad anterior reciben el nombre de *cadenas de Markov en tiempo discreto* y con espacio de estados E.

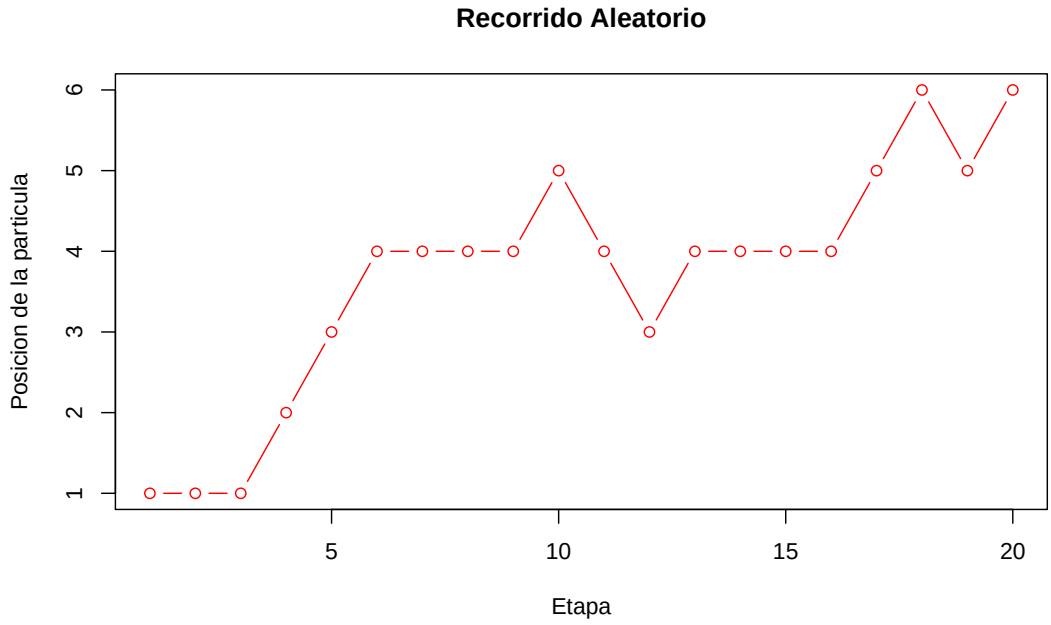


Figura 5.1: Gráfico del recorrido aleatorio de una partícula con probabilidades de desplazamiento: $\Pr(Z_i = -1) = 0{,}2$, $\Pr(Z_i = 0) = 0{,}5$, $\Pr(Z_i = 1) = 0{,}3$.

5.3. Cadenas homogéneas de Markov

En general, la probabilidad $\Pr(X_{n+1} = j \,|X_n = i\,)$ puede depender de la etapa n. En aquellos casos en los que esta probabilidad es independiente de n, la cadena recibe el nombre de *cadena de Markov homogénea*. En tal caso, utilizaremos la notación:

$$p_{i,j} = \Pr(X_{n+1} = j \,|X_n = i\,)$$

El recorrido aleatorio que hemos visto en la sección 5.1 es un ejemplo de cadena homogénea de Markov.

5.3.1. Elementos de una cadena homogénea de Markov

Las probabilidades p_{ij} reciben el nombre de *probabilidades de transición en una etapa* o simplemente, *probabilidades de transición*. Dado que el espacio E es finito

o numerable, las probabilidades de transición pueden disponerse en una matriz de la forma:

$$\mathbf{P} = (p_{i,j})_{i,\, j \in E}$$

La matriz $\mathbf{P}$ recibe el nombre de *matriz de probabilidades de transición* (o simplemente *matriz de transición*). A la distribución de la variable X_0 se le llama *distribución inicial* de la cadena, y suele representarse de la forma:

$$p_i^{(0)} = \Pr\left(X_0 = i\right),\ i \in E$$

Una cadena de Markov homogénea está, pues, determinada por tres elementos, a saber:

1. El espacio de estados E (conjunto finito o numerable)
2. La matriz de probabilidades de transición.
3. La distribución inicial $p^{(0)} = \left(p_i^{(0)}\right)_{i \in E}$

Ejemplo 5.1. Consideremos una linea telefónica cuyo estado se revisa cada cierta fracción de tiempo de longitud fija δ. Denotamos por X_k el estado de la linea en la k-ésima etapa de revisión, siendo posibles solo dos estados $X_k = 0$ (linea libre) o $X_k = 1$ (linea ocupada). Si en cualquier etapa arbitraria la linea está ocupada, la probabilidad de que en la etapa siguiente quede libre es 0.08; asimismo, si la linea está libre, la probabilidad de que en la etapa siguiente pase a estar ocupada es 0.03. Inicialmente la linea está desocupada.

La sucesión de estados por los que pasa esta linea telefónica constituye una cadena de Markov homogénea con espacio de estados $E = \{0, 1\}$. Las probabilidades de transición en una etapa serían:

$$p_{0,0} = \Pr\left(X_{k+1} = 0 \,|X_k = 0\right) = 0{,}97 \ ; \quad p_{0,1} = \Pr\left(X_{k+1} = 1 \,|X_k = 0\right) = 0{,}03$$
$$p_{1,0} = \Pr\left(X_{k+1} = 0 \,|X_k = 1\right) = 0{,}08 \ ; \quad p_{1,1} = \Pr\left(X_{k+1} = 1 \,|X_k = 1\right) = 0{,}92$$

La matriz de probabilidades de transición es entonces:

$$\mathbf{P} = \begin{pmatrix} p_{0,0} & p_{0,1} \\ p_{1,0} & p_{1,1} \end{pmatrix} = \begin{pmatrix} 0{,}97 & 0{,}03 \\ 0{,}08 & 0{,}92 \end{pmatrix}$$

y la distribución inicial de la cadena:

$$p^{(0)} = \left(p_0^{(0)}, p_1^{(0)}\right) = \left(\Pr\left(X_0 = 0\right), \Pr\left(X_0 = 1\right)\right) = (1,\, 0)$$

Simulación: Utilizaremos R para simular el estado de esta linea telefónica a lo largo de n etapas. Para ello construiremos una función que recibe como entrada el número de etapas a simular, el estado inicial de la linea (que por defecto se asume desocupada), y las probabilidades de transición $p_{0,1}$ y $p_{1,1}$:

```
lineaTel = function(n, X0 = 0, probs) {
    actualiza = function(estado) ifelse(estado == 0, rbinom(1, 1, probs[1]),
        rbinom(1, 1, probs[2]))
    estado = numeric(n + 1)
    estado[1] = X0
    for (k in 1:n) estado[k + 1] = actualiza(estado[k])
    estado
}
```

Utilizamos esta función para simular 100 etapas sucesivas del estado de esta linea:

```
estadoLinea = lineaTel(n = 100, X0 = 0, probs = c(0.03, 0.92))
estadoLinea
```

```
##   [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [33] 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [65] 1 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1
##  [97] 1 1 1 1 1
```

La figura 5.2 representa gráficamente el estado de la linea.

```
plot(estadoLinea, type = "n", xlab = "Etapa", ylab = "Estado de la l<U+00ED>nea",
    axes = F, ylim = c(-0.5, 1.5))
axis(1)
```

```
axis(2, at = c(0, 1))
lines(stepfun(2:length(estadoLinea), estadoLinea), do.points = F, col = "red")
box()
```

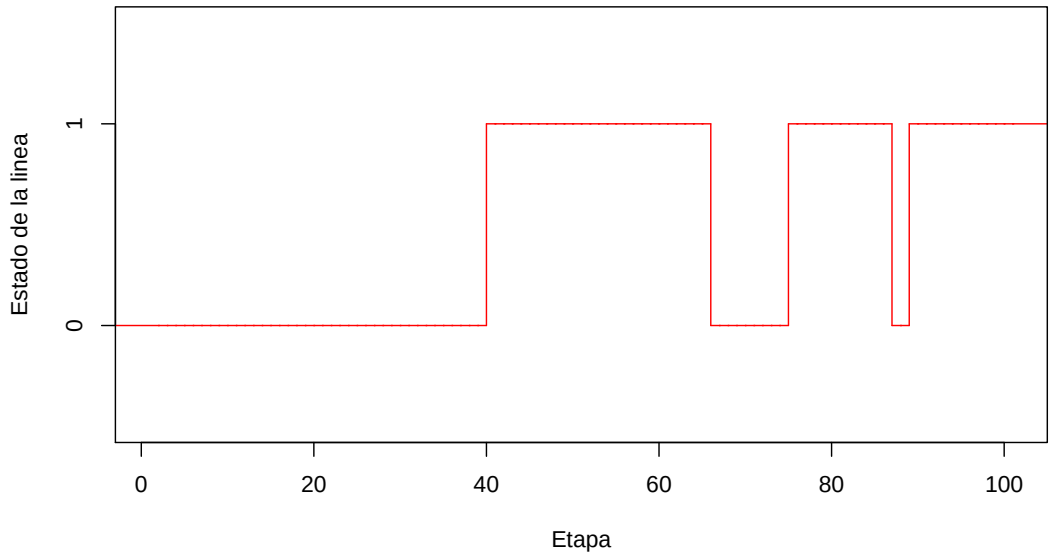


Figura 5.2: Gráfico de la ocupación de la linea telefónica descrita en el ejemplo 5.1.

5.3.2. Distribuciones marginales

Dada una cadena homogénea de Markov $\{X_n; n \in \mathbb{N}\}$, representaremos la distribución marginal de X_n mediante $p_i^{(n)} = P(X_n = i)$. Si llamamos $\mathbf{p}^{(n)} = \left(p_i^{(n)};\, i \in E\right)$ se tiene que:

$$\mathbf{p}^{(n)} = \mathbf{p}^{(n-1)} \cdot \mathbf{P}$$

En efecto:

$$\Pr(X_n = i) = \sum_{k \in E} \Pr(X_{n-1} = k) \Pr(X_n = i \,|X_{n-1} = k) = \sum_{k \in E} p_k^{(n-1)} p_{k,i}$$

lo cual es equivalente a la expresión matricial $\mathbf{p}^{(n)} = \mathbf{p}^{(n-1)} \cdot \mathbf{P}$

Aplicando esta última expresión reiteradamente obtenemos:

$$\mathbf{p}^{(n)} = \mathbf{p}^{(0)} \cdot \mathbf{P}^n$$

5.3.3. Distribuciones estacionarias.

Sea $\{X_n; n \in \mathbb{N}\}$ una cadena homogénea de Markov con espacio de estados E. Una distribución de probabilidad $\pi = (\pi_i)_{i \in E}$ es una *distribución estacionaria* para $\{X_n; n \in \mathbb{N}\}$ si verifica $\pi \cdot P = \pi$ y $\sum_{i \in E} \pi_i = 1$. No todas las cadenas de Markov poseen alguna distribución estacionaria. Ahora bien, es obvio que en caso de que exista una distribución estacionaria π, si en alguna etapa k la cadena de Markov la alcanza, esto es $p^{(k)} = \pi$, a partir de entonces (para todo $n > k$) ocurrirá que $p^{(n)} = \pi$. En el caso particular de que $p^{(0)} = \pi$, entonces $p^{(n)} = \pi \ \ \forall n$.

Ejemplo. Puede comprobarse que para la cadena de Markov

$$\mathbf{P} = \begin{pmatrix} 0{,}97 & 0{,}03 \\ 0{,}08 & 0{,}92 \end{pmatrix}$$

que describe el estado de ocupación de la linea telefónica del ejemplo 5.1 la distribución $\pi = \left(\frac{8}{11}, \frac{3}{11}\right)$ es una distribución estacionaria.

5.3.4. Probabilidades de transición en n etapas.

Sea $\{X_n;\, n \in \mathbb{N}\}$ una cadena de Markov homogénea con espacios de estados E. La probabilidad de ir del estado i al estado j en n etapas se expresa como:

$$p_{ij}^{(n)} = \Pr(X_{m+n} = j \,|X_m = i)$$

Obviamente al ser la cadena de Markov homogénea, esta probabilidad es independiente de m. Dispondremos estas probabilidades en la matriz:

$$\mathbf{P}^{(n)} = \left(p_{i,j}^{(n)}\right)_{i,j\in E}$$

que recibe el nombre de *matriz de transición en n etapas*. La *ecuación de Chapman-Kolmogorov* establece que esta matriz es igual a la potencia n-ésima de la matriz de transición en una etapa. En efecto:

$$\begin{aligned} p_{i,j}^{(n+1)} &= \Pr\left(X_{n+1} = j \,|X_0 = i\right) = \frac{\Pr\left(X_0 = i, X_{n+1} = j\right)}{\Pr\left(X_0 = i\right)} = \\ &= \frac{1}{\Pr\left(X_0 = i\right)} \sum_{k\in E} \Pr\left(X_0 = i, X_1 = k, X_{n+1} = j\right) = \\ &= \frac{1}{\Pr\left(X_0 = i\right)} \sum_{k\in E} \Pr\left(X_0 = i\right) \Pr\left(X_1 = k \,|X_0 = i\right) \cdot \Pr\left(X_{n+1} = j \,|X_0 = i, X_1 = k\right) = \\ &= \sum_{k\in E} \Pr\left(X_1 = k \,|X_0 = i\right) \Pr\left(X_{n+1} = j \,|X_1 = k\right) = \sum_{k\in E} p_{i,k} p_{k,j}^{(n)} \end{aligned}$$

En forma matricial esta identidad puede expresarse como:

$$\mathbf{P}^{(n+1)} = \mathbf{P} \cdot \mathbf{P}^{(n)}, \quad \forall n$$

Por último, si aplicamos esta igualdad reiteradamente obtenemos:

$$\mathbf{P}^{(n)} = \mathbf{P}^n$$

Esta identidad recibe el nombre de *ecuación de Chapman-Kolmogorov.*

Ejemplo 5.2. Consideremos una cadena de Markov cuyo espacio de estados es $E = \{a, b, c\}$ y su matriz de probabilidades de transición:

$$\mathbf{P} = \begin{pmatrix} 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \\ 0 & 0 & 1 \end{pmatrix}$$

Si el sistema parte del estado a, hallar la distribución de probabilidad de la cadena en la etapa 4. (Distribución de probabilidad de X_4).

Solución: Para obtener la distribución pedida bastará tener en cuenta que la matriz de transición en cuatro etapas es la cuarta potencia de la matriz $\mathbf{P}$. Se tiene entonces:

$$\mathbf{P}^4 = \begin{pmatrix} 1/16 & 0 & 15/16 \\ 0 & 1/16 & 15/16 \\ 0 & 0 & 1 \end{pmatrix}$$

Si el sistema se encuentra inicialmente en el estado a, la distribución inicial de la cadena es:

$$\mathbf{p}^{(0)} = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$$

De donde:

$$\mathbf{p}^4 = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1/16 & 0 & 15/16 \\ 0 & 1/16 & 15/16 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1/16 & 0 & 15/16 \end{pmatrix}$$

El estado c recibe el nombre de *estado absorbente*, pues es obvio que una vez que el sistema haya alcanzado este estado ya no sale más de él. La probabilidad por tanto de que el sistema haya sido absorbido en la etapa cuarta por este estado es 15/16.

Simulación: Construimos a continuación una función que simula n etapas de una cadena de Markov con espacio de estados E, matriz de transición P, y que parte del estado inicial $X0$:

```
markovChain = function(n, X0, P, E = NULL) {
    nE = nrow(P)
    if (is.null(E))
        E = 1:nE
    estado = numeric(n)
    estado[1] = which(E == X0)
```

```
    for (k in 1:n) estado[k + 1] = sample(1:nE, 1, prob = P[estado[k],
        ])
    noquote(E[estado])
}
```

Definimos ahora el espacio de estados E y la matriz $\mathbf{P}$ del ejemplo anterior, y simulamos cuatro etapas partiendo del estado a:

```
P = matrix(c(0, 0.5, 0.5, 0.5, 0, 0.5, 0, 0, 1), nrow = 3, byrow = T)
E = c("a", "b", "c")
markovChain(4, "a", P, E)

## [1] a b a c c
```

Replicamos 1000 veces esta simulación y contamos el número de veces que la cadena se encuentra en cada una de los tres estados posibles en la cuarta etapa:

```
mkSim = replicate(10000, markovChain(4, "a", P, E))
table(mkSim[5, ])

##
##    a    c
##  582 9418
```

De esta forma la proporción de veces que en estas simulaciones se ha alcanzado el estado c es 0.9418 que, como vemos, es un valor muy próximo a $15/16 = 0{,}9375$.

5.3.5. Probabilidades de primer paso.

Con frecuencia resulta de interés el cálculo de la probabilidad de que, partiendo del estado i, la cadena llegue por primera vez al estado j transcurridas n etapas. Esta probabilidad suele denotarse como:

$$f_{i,j}^{(n)} = \Pr\left(X_n = j,\ X_k \neq j,\ k = 1, ..., n-1 / X_0 = i\right),\ n \geq 1$$

Para $n = 0$ se considera:

$$f_{i,j}^{(0)} = 0$$

La probabilidad de que, partiendo del estado i, la cadena llegue alguna vez al estado j es entonces:

$$f_{i,j} = \sum_{n=1}^{\infty} f_{i,j}^{(n)}$$

En particular, $f_{i,i}$ representa la probabilidad de que, partiendo del estado i, la cadena regrese alguna vez a dicho estado. En caso de que $f_{i,i} > 0$, la cantidad $\mu_{i,i} = \sum_{n=1}^{\infty} n f_{i,i}^{(n)}$ recibe el nombre de *tiempo medio de recurrencia* y representa es el número medio de etapas que la cadena tarda en regresar al estado de partida i.

La siguiente relación entre las $p_{i,j}^{(n)}$ y las $f_{i,j}^{(k)}$, $k = 1, 2, ..., n$, es inmediata sin más que aplicar el teorema de la probabilidad total (considerando $p_{j,j}^{(0)} = 1$):

$$p_{i,j}^{(n)} = \sum_{k=1}^{n} f_{i,j}^{(k)} p_{j,j}^{(n-k)} \tag{5.1}$$

(esta igualdad se deduce fácilmente observando que $p_{i,j}^{(n)}$, la probabilidad de ir de i a j en n etapas, puede descomponerse como la suma de las probabilidades de todas las formas de ir por primera vez de i a j en k etapas y regresar a j en $n-k$ etapas adicionales).

A partir de la ecuación (5.1) es fácil deducir que:

$$f_{i,j}^{(n)} = p_{i,j}^{(n)} - \sum_{k=1}^{n-1} f_{i,j}^{(k)} p_{j,j}^{(n-k)} \tag{5.2}$$

Esta ecuación permite calcular los $f_{ij}^{(n)}$ recursivamente. En el caso particular de que la matriz de transición $\mathbf{P}$ sea de dimensión finita, llamando $\mathbf{F}^{(n)}$ a la matriz $\left(f_{i,j}^{(n)}\right)_{i,j\in E}$ la ecuación (5.2) puede escribirse en forma matricial como:

$$\mathbf{F}^{(n)} = \mathbf{P}^n - \sum_{k=1}^{n-1} \mathbf{F}^{(k)} \mathbf{D}^{(n-k)} \tag{5.3}$$

siendo $\mathbf{D}^{(k)}$ una matriz diagonal formada por la diagonal principal de $\mathbf{P}^n$. La siguiente función permite calcular $\mathbf{F}^{(n)}$ en R (nótese que en R no hay, por defecto, ninguna función para calcular la potencia de una matriz. Para ello debemos cargar el paquete *expm,* que contiene al operador %^%; concretamente, utilizando este paquete, la potencia $\mathbf{P}^n$ se calcula como `P %^% n)`).

```
MPP = function(P, n) {
    require(expm)
    Fn = vector("list", n)
    Pn = vector("list", n)
    Dn = vector("list", n)
    In = diag(1, nrow(P))
    for (k in 1:n) {
        Pn[[k]] = P %^% k
        Dn[[k]] = diag(Pn[[k]]) * In
        Fn[[k]] = Pn[[k]]
        if (k > 1)
            for (i in 1:(k - 1)) Fn[[k]] = Fn[[k]] - Fn[[i]] %*% Dn[[k -
                i]]
    }
    return(Fn[[n]])
}
```

Ejemplo 5.3. Con la matriz P del ejemplo 5.2, la matriz de probabilidades de primer paso en 3 etapas puede calcularse mediante:

```
MPP(P, 3)
```

```
##         [,1]   [,2]   [,3]    [,4]    [,5]
## [1,] 0.10175 0.1130 0.1433 0.11550 0.07175
## [2,] 0.09737 0.1139 0.1341 0.11950 0.07562
## [3,] 0.12388 0.1205 0.1251 0.09400 0.07963
## [4,] 0.12500 0.1349 0.1357 0.09350 0.06850
## [5,] 0.12925 0.1419 0.1365 0.09712 0.06113
```

Algunas modificaciones de la función anterior permiten calcular los tiempos medios de primer paso entre los distintos estados:

```
aproxTmPP = function(P, eps = 0.001, nmax = 500) {
    require(expm)
    Fn = vector("list", nmax)
    Pn = vector("list", nmax)
    Dn = vector("list", nmax)
    In = diag(1, nrow(P))
    tmpp0 = matrix(0, nrow = nrow(P), ncol = ncol(P))
    k = 0
    err = 1
    while (err > eps & k < nmax) {
        k = k + 1
        Pn[[k]] = P %^% k
        Dn[[k]] = diag(Pn[[k]]) * In
        Fn[[k]] = Pn[[k]]
        if (k > 1)
            for (i in 1:(k - 1)) Fn[[k]] = Fn[[k]] - Fn[[i]] %*% Dn[[k -
                i]]
        tmpp = tmpp0 + k * Fn[[k]]
        err = max(abs(tmpp - tmpp0))
        tmpp0 = tmpp
    }
    return(round(tmpp, 3))
}
```

Aplicamos esta función a la matriz P anterior:

```
aproxTmPP(P)
```

```
##       [,1]  [,2]  [,3]  [,4]   [,5]
## [1,] 4.829 4.343 4.424 7.067 11.108
## [2,] 4.454 4.242 4.231 7.463 11.754
## [3,] 5.806 4.676 3.964 5.975 11.441
```

```
## [4,] 6.426 5.348 4.146 5.347  9.619
## [5,] 6.893 5.701 4.126 5.133  8.476
```

5.3.6. Clasificación de estados de una cadena de Markov homogénea.

Estados recurrentes: Un estado i es recurrente si $f_{i,i} = 1$

Estados transitorios: el estado i es transitorio si no es recurrente, esto es, si $f_{i,i} < 1$

Estados recurrentes positivos: un estado j se dice recurrente positivo si $\mu_{j,j} < \infty$ esto es, si el tiempo medio que se tarda en regresar a él es finito.

Estados recurrentes nulos: un estado j se dice recurrente nulo si $\mu_{jj} = \infty$ esto es, si se tarda un tiempo medio infinito en regresar a él.

Estados absorbentes: un estado i se dice absorbente si $p_{ii} = 1$, esto es, si una vez que la cadena llega a este estado ya no puede pasar a otro distinto de él.

Estados accesibles: un estado j es accesible desde otro estado i si: $\exists n \geqslant 0$: $p_{i,j}^{(n)} > 0$

Estados comunicantes: dos estados i, j comunican entre sí si cada uno de ellos es accesible desde el otro.

Estados periódicos: un estado recurrente i se dice periódico de periodo d si d es el máximo común divisor de los índices n para los cuáles $p_{i,i}^{(n)} > 0$. Un estado con periodo 1 se dice a-periódico. Puede probarse que si el estado i es a-periódico entonces existe un entero n_0 tal que $p_{i,i}^{(n)} > 0 \ \forall n \geqslant n_0$.

Conjunto cerrado de estados: un conjunto de estados C es cerrado si: $\forall i \in C, \forall j \notin C: \ p_{ij} = 0$

Conjuntos irreducibles de estados: un conjunto de estados C es irreducible si es cerrado y no contiene subconjuntos cerrados. Puede demostrarse que

si C es cerrado, entonces es irreducible si y sólo si todos sus estados comunican entre sí. Los conjuntos irreducibles de estados cumplen las siguientes propiedades:

1. Si C es irreducible, entonces sus estados son todos transitorios o son todos recurrentes.
2. Si C es irreducible y todos sus estados son recurrentes, entonces $f_{i,j} = 1\ \forall i, j \in C$. Más aún, $\mu_{jj} < \infty\ \forall j \in C$, o bien $\mu_{j,j} = \infty\ \forall j \in C$ (esto es, sus estados o son todos recurrentes positivos o son todos recurrentes nulos).
3. Si C es irreducible y todos sus estados son recurrentes, entonces todos los estados de C tienen el mismo periodo.

Ejemplo 5.4. La siguiente es la matriz de transición de una cadena de Markov con 6 estados, numerados de 1 a 6:

$$\begin{array}{c} 1 \to \\ 2 \to \\ 3 \to \\ 4 \to \\ 5 \to \\ 6 \to \end{array} \begin{pmatrix} \frac{1}{6} & 0 & 0 & \frac{1}{2} & \frac{1}{3} & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{5} & 0 & \frac{2}{5} & \frac{1}{5} & 0 & \frac{1}{5} \\ \frac{1}{4} & 0 & 0 & \frac{3}{4} & 0 & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{4} & \frac{1}{4} & 0 & 0 & \frac{1}{2} \end{pmatrix}$$

- El estado 2 es absorbente, pues si el proceso lo alcanza, permanece en él para siempre.
- Los estados 3 y 6 son transitorios, ya que pueden ser visitados unas cuantas veces y no volver a ellos nunca más.
- Los estados 1, 4 y 5 son recurrentes.
- El conjunto $\{1, 4, 5\}$ es un conjunto cerrado de estados; también es irreducible.
- El conjunto $\{2\}$ es también cerrado e irreducible.
- El conjunto $\{3, 6\}$ no es cerrado.
- El conjunto $\{1, 2, 3, 4, 5, 6\}$ es cerrado pero no irreducible.

5.3.7. Distribución de equilibrio de una cadena de Markov.

Un problema de gran interés en las aplicaciones prácticas es el siguiente: transcurrido un número muy grande de etapas, ¿cuál es la probabilidad de que la cadena se encuentre en un estado j determinado?. Esta pregunta sólo tiene sentido cuando el proceso es *ergódico*. Un proceso estocástico es ergódico cuando existe la distribución límite:

$$\pi_j = \lim_{n\to\infty} p_{i,j}^{(n)} \ , \ \forall j \in E$$

esto es, cuando independientemente del estado de partida, transcurrido un número de etapas muy grande existe una probabilidad fija de que la cadena se encuentre en cada uno de sus posibles estados. En tal caso se dice que la cadena está en *equilibrio*. Esta distribución $\pi = (\pi_j)_{j\in E}$ recibe el nombre de *distribución de equilibrio o ergódica* de la cadena.

El siguiente teorema nos indica en qué casos existe la distribución límite de una cadena de Markov.

Teorema: *Si una cadena homogénea de Markov es irreducible y a-periódica entonces existe la distribución límite* $\pi_j = \lim_{n\to\infty} p_{i,j}^{(n)} \ , \ \forall j \in E$. *Además:*

1. *Si todos los estados son transitorios, o todos los estados son recurrentes nulos, entonces* $\pi_j = 0$ *para todo* j*, y no existe distribución estacionaria.*

2. *Si todos los estados son recurrentes positivos entonces* $\pi_j = 1/\mu_{j,j}$ *para todo* j*. En este caso la distribución límite* $\pi = (\pi_j)_{j\in E}$ *es estacionaria y queda unívocamente determinada por las ecuaciones*

$$\pi P = \pi, \quad \pi \cdot \mathbf{1} = 1$$

Problemas:

1. Dos jugadores A y B, que tienen inicialmente a y b euros respectivamente, juegan sucesivas partidas, apostando un euro en cada una. En cada partida, A tiene probabilidad p de ganar y B, probabilidad q (obviamente $p+q=1$).

El juego termina cuando uno de los jugadores se arruina. Sea X_n la cantidad de euros de A al finalizar la n-ésima partida ($X_0 = a$).

a) Probar que la sucesión $\{X_n; n \in \mathbb{N}\}$ es una cadena homogénea de Markov y describir la matriz de transición de probabilidades.

b) Calcular la probabilidad de que A se arruine.

2. Se lanza una moneda repetidamente hasta que se presenten dos caras o tres cruces consecutivas. Si sólo interesa el número de lanzamientos necesarios para que el juego termine, plantear una cadena de Markov que describa la situación después de cada lanzamiento. Determínese la probabilidad de que el juego termine en, a lo sumo, cuatro lanzamientos.

3. Un combate naval entre tres barcos A, B y C se desarrolla de la siguiente manera: en cada unidad de tiempo y simultáneamente cada barco hace un disparo. El barco A tiene probabilidad $1/2$ de alcanzar su blanco, el B tiene probabilidad $1/3$ y el C probabilidad $\frac{1}{4}$. Cada barco elige como blanco al más preciso de sus adversarios presentes. Un solo impacto basta para hundir al barco alcanzado y el combate continúa mientras haya más de un barco en acción. Plantear la cadena de Markov que describa la evolución de la batalla.

4. Cada segundo la tarjeta de red WIFI de un ordenador portátil informa sobre el estado del canal de radio que le permite conectarse a un punto de acceso. El estado del canal puede ser *malo* (0), *regular* (1) o *bueno* (2). Las transiciones entre estados pueden modelarse a través de una cadena de Markov con matriz de transición:

$$P = \begin{pmatrix} 0{,}2 & 0{,}3 & 0{,}5 \\ 0{,}4 & 0{,}4 & 0{,}2 \\ 0{,}2 & 0{,}4 & 0{,}4 \end{pmatrix}$$

a) ¿Cuál es la probabilidad del estado 0 al estado 1 en un segundo?

b) ¿Cuál es la probabilidad de pasar del estado 0 al estado 1 en dos segundos?

c) Calcular la matriz de transición para 3 segundos.

d) Partiendo de que el estado del canal inicialmente es bueno ¿Cuál es la probabilidad de que pasen 3 segundos sin que el estado sea malo?

e) Utilizando el programa R calcula la matriz de transición para 4, 5, 6, 7, 8, 9 y 10 segundos. ¿Tienden a converger estas matrices a alguna matriz fija?

f) Calcula las probabilidades estacionarias en este sistema.

g) Determina si este sistema tiene distribución de equilibrio y en tal caso calcúlala.

5. Considérense N urnas las cuales están inicialmente vacías. Aleatoriamente y en etapas sucesivas se van colocando aleatoriamente bolas en las urnas (una en cada etapa) hasta que todas están llenas. Sea $X_n =$ *número de celdas que permanecen vacías en la etapa* n.

   a) Describir la cadena de Markov asociada a la sucesión de variables aleatorias $X_n : n \in \mathbb{N}$.

   b) Hallar $\Pr\left(X_{n+2} = j \mid X_n = i\right) \ : \quad \forall j$.

6. Dos urnas A y B contienen en total r bolas que supondremos distinguibles. Considérese ahora la secuencia de ensayos consistente en seleccionar aleatoriamente una bola y cambiarla de urna. Sea X_n el número de bolas que hay en la urna A después de n ensayos.

   a) Describir la cadena de Markov asociada a la sucesión de variables aleatorias $X_n : n \in \mathbb{N}$.

   b) Hallar la distribución estacionaria para $r = 3$.

7. Inicialmente, una urna A tiene tres bolas blancas y otra B tres negras. Considérese ahora la secuencia de ensayos consistente en extraer una bola de cada urna e intercambiarlas entre ellas. Sea X_n el número de bolas blancas que hay en la urna A después de n ensayos.

   a) Describir la cadena de Markov asociada a la sucesión de variables aleatorias $\{X_n : n \in \mathbb{N}\}$.

   b) Calcular la distribución estacionaria.

8. Un sistema de control de tráfico dispone de un dispositivo de almacenamiento de ciertas unidades, llamadas *tokens*, que opera del siguiente modo: en el dispositivo cabe un total de c *tokens*; cada vez que llega un cliente al sistema, con probabilidad p añade un *token* al dispositivo, y con probabilidad $1-p$ retira un *token* del mismo (salvo en el caso de que el dispositivo esté vacío en cuyo caso no puede retirar nada, o esté completamente lleno, en cuyo caso no puede añadir nada).

   a) Calcular la distribución de probabilidad del número de tokens en el dispositivo cuando el sistema alcanza el equilibrio.

   b) ¿Cuál es el número medio de tokens en el dispositivo?

Capítulo 6

Procesos de nacimiento y muerte

En este capítulo se estudiarán modelos estocásticos de muy frecuente aplicación en la ingeniería de telecomunicación, en áreas tales como el modelado y gestión de tráfico, o el dimensionamiento de recursos en redes. Los procesos de nacimiento y muerte, que reciben este nombre por su origen en estudios de demografía, constituyen una herramienta muy simple e intuitiva para modelar el teletráfico: un nacimiento representa el acceso de un cliente a un sistema (por ejemplo, un paquete de datos que accede a un router), y una muerte la salida del mismo. Cuando las llegadas y salidas a un sistema se producen siguiendo patrones aleatorios es de esperar que en el sistema se formen lineas de espera o colas. Para diseñar adecuadamente un sistema de comunicación se hace necesario disponer de herramientas que permitan modelar o, en su caso, estimar los tiempos de espera y las longitudes de la cola. Presentaremos algunos modelos simples para ello e indicaremos como puede utilizarse R para la simulación y estimación en modelos más complejos.

6.1. El proceso homogéneo de Poisson

6.1.1. Formulación del proceso homogéneo de Poisson.

Consideremos un sistema que inicialmente está vacío y al que van llegando items de manera sucesiva. El proceso de llegadas de estos items recibe el nombre de

proceso homogéneo de Poisson si, para cada instante t, la probabilidad de que en el intervalo $[t, t+h]$ se produzca:

1. exactamente una llegada es $\lambda h + o(h)$, siendo $o(h)$ un infinitésimo de orden inferior a h.
2. ninguna llegada es $1 - \lambda h + o(h)$.
3. más de una llegada es $o(h)$.

6.1.2. Proceso de recuento

Representaremos por $N(t)$ el número de llegadas producidas hasta el instante t. El proceso $N(t)$ recibe el nombre de *proceso de recuento* asociado al proceso de Poisson. Obsérvese que $N(t)$ es un proceso markoviano, pues la evolución del proceso a partir de cada instante t depende de su valor en dicho instante, pero no de la secuencia de valores que haya ido tomando en el pasado. Una función de interés asociada al proceso homogéneo de Poisson es:

$$P_n(t) = \Pr(N(t) = n) \tag{6.1}$$

Probaremos seguidamente que:

$$P_n(t) = \exp(-\lambda t) \frac{(\lambda t)^n}{n!} \ , \ n = 0, 1, 2, ... \tag{6.2}$$

Para ello, tendremos en cuenta que para un h pequeño la probabilidad de que el instante $t+h$ haya n items coincide con la probabilidad de que en el instante t ya hubiera n items y en el intervalo $[t, t+h]$ no haya llegado ninguno, más la probabilidad de que en t hubiera $n-1$ items y en $[t, t+h]$ haya llegado exactamente uno. Teniendo en cuenta la formulación del proceso de Poisson esta probabilidad puede escribirse formalmente como:

$$P_n(t+h) = P_n(t)(1 - \lambda h) + P_{n-1}(t)\lambda h + o(h) \ ; \ \ n \geqslant 1$$

Para $n = 0$ es evidente que:

$$P_0(t+h) = P_0(t)(1-\lambda h) + o(h)$$

Asimismo, si partimos del supuesto de que el sistema estaba inicialmente vacío:

$$P_0(0) = \Pr(N(0) = 0) = 1$$

De aquí:

$$\frac{P_n(t+h) - P_n(t)}{h} = -\lambda P_n(t) + \lambda P_{n-1}(t) + \frac{o(h)}{h} \; ; \quad n \geqslant 1$$

Tomando límites para h tendiendo a cero en esta última expresión, se obtiene la ecuación diferencial:

$$P_n'(t) + \lambda P_n(t) = \lambda P_{n-1}(t) \; ; \quad n \geqslant 1 \tag{6.3}$$

y para n=0

$$P_0'(t) = -\lambda P_0(t) \tag{6.4}$$

La solución de (6.4) es:

$$P_0(t) = \exp(-\lambda t)$$

por lo que (6.2) es cierta para $n = 0$. Podemos utilizar ahora el método de inducción completa para probar (6.2) para todo n. Supongamos que es cierta para $n-1$. Ello significa que:

$$P_{n-1}(t) = \exp(-\lambda t)\frac{(\lambda t)^{n-1}}{(n-1)!} \tag{6.5}$$

Sustituyendo (6.5) en la ecuación (6.3) se obtiene la ecuación diferencial:

$$P_n'(t) + \lambda P_n(t) = \exp(-\lambda t)\frac{\lambda^n t^{n-1}}{(n-1)!}$$

que puede escribirse como:

$$\exp(\lambda t) P_n'(t) + \lambda \exp(\lambda t) P_n(t) = \frac{\lambda^n t^{n-1}}{(n-1)!}$$

o de manera equivalente:

$$\frac{d}{dt}\left(\exp\left(\lambda t\right) P_n\left(t\right)\right) = \frac{\lambda^n t^{n-1}}{(n-1)!}$$

Esta ecuación puede resolverse fácilmente. Teniendo en cuenta que para $n > 0$ se tiene $P_n\left(0\right) = 0$, se comprueba que la solución de la ecuación es efectivamente (6.2).

6.1.3. Distribución conjunta de los instantes de llegada

Sean $T_1, T_2, ..., T_n$ los n primeros instantes de llegada correspondientes a un proceso homogéneo de Poisson. Las variables aleatorias definidas por $X_i = T_i - T_{i-1}$ representan los periodos transcurridos entre dos llegadas consecutivas (asumimos $T_0 \equiv 0$). Dadas las características del proceso de Poisson, es obvio que las variables aleatorias $X_1, \ \cdots \ , X_n$ son independientes y siguen la misma distribución de probabilidad. Esta distribución es precisamente la exponencial. Para comprobarlo consideremos $\bar{F}\left(t\right) = \Pr\left(X_1 > t\right)$. Dado que:

$$\Pr\left(X_1 > t + h\right) = \Pr\left(X_1 > t\right)\left(1 - \lambda h\right) + o\left(h\right)$$

se tiene entonces la ecuación diferencial $\bar{F}'\left(t\right) = -\lambda \cdot \bar{F}\left(t\right)$. Con la condición inicial $\bar{F}\left(0\right) = 1$, la solución de esta ecuación es $\bar{F}\left(t\right) = \exp\left(-\lambda t\right)$ que corresponde precisamente a la distribución exponencial.

Si tenemos en cuenta que $T_i = \sum_{j=1}^{i} X_j \ , \ i = 1, ..., n$, se sigue que la distribución de probabilidad conjunta de $T_1, T_2, ..., T_n$ viene dada por la función de densidad:

$$f\left(t_1, \ \cdots \ , t_n\right) = \lambda^n \cdot \exp\left(-\lambda \cdot t_n\right) \ , \quad 0 < t_1 < t_2 < \ \cdots \ < t_n \tag{6.6}$$

6.1.4. Simulación del proceso de Poisson

Construimos a continuación una función en R para simular un proceso de Poisson con tasa de llegadas λ. Esta función recibe como parámetros el número n de llegadas a simular, el instante inicial t_0 en que empieza a observarse el sistema, la

ocupación N_0 en el instante inicial y el valor λ de la tasa de llegadas. Para realizar la simulación utilizaremos que los tiempos entre llegadas siguen una distribución exponencial de parámetro λ. La función nos devuelve los sucesivos instantes de llegada así como el número de clientes en el sistema en esos instantes.

```
procPoisson = function(n, N0 = 0, t0 = 0, lambda) {
    t_entre_llegadas = rexp(n, lambda)
    tllegada = c(t0, t0 + cumsum(t_entre_llegadas))
    N = N0:(N0 + n)
    return(data.frame(t = tllegada, N = N))
}
```

Simulamos la llegada de 10 clientes:

```
proceso = procPoisson(n = 10, N0 = 0, t0 = 0, lambda = 2)
proceso
```

```
##         t  N
## 1  0.0000  0
## 2  0.4762  1
## 3  1.7276  2
## 4  2.0208  3
## 5  2.1832  4
## 6  2.7017  5
## 7  3.8080  6
## 8  4.4785  7
## 9  4.5343  8
## 10 4.6728  9
## 11 4.9845 10
```

La siguiente función genera una gráfica de esta simulación, que se muestra en la figura 6.1.

```
with(proceso, plot(stepfun(t[-1], N), xlab = "Tiempo", ylab = "Numero de llegadas
    main = "Proceso de Poisson"))
```

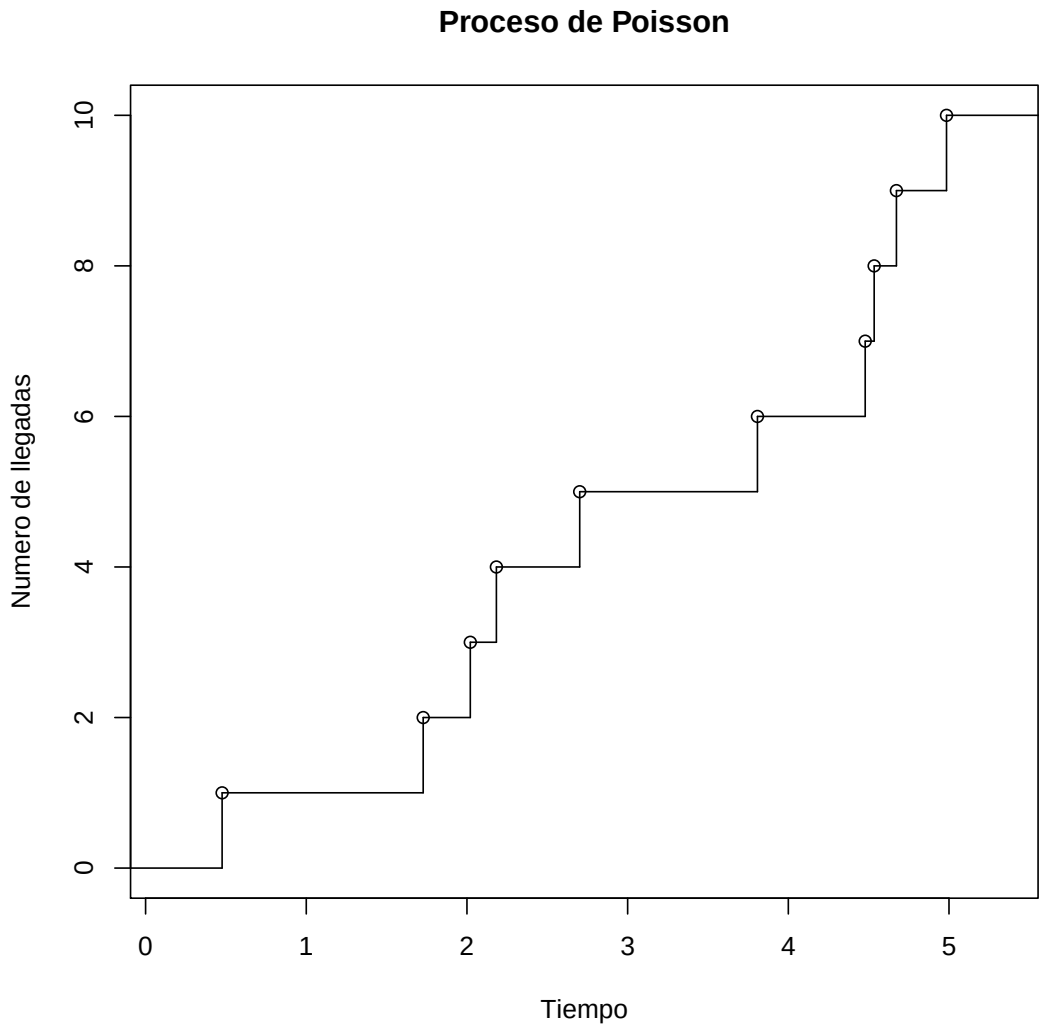


Figura 6.1: Simulación de un proceso de Poisson de tasa $\lambda = 2$.

6.2. Proceso puro de nacimiento

En el proceso de Poisson, la probabilidad de una llegada en un intervalo $[t, t+h]$ es independiente de $N(t)$. Sin embargo, en muchas aplicaciones prácticas, la tasa de

llegadas de clientes al sistema puede estar condicionada por el número de clientes que se encuentran ya en el mismo. Por ejemplo, si los clientes de un supermercado ven que una caja está muy ocupada, se dirigen a otras cajas, con lo que la tasa de llegadas a la primera caja disminuye.

Los sistemas en los que sólo se producen llegadas con tasa dependiente de la ocupación del sistema en cada instante pueden modelarse mediante *procesos puros de nacimiento.* Estos procesos representan las sucesivas llegadas a un sistema que inicialmente contiene N_0 items; si en un instante arbitrario t el número de clientes en el sistema es n, la probabilidad de que en el intervalo $[t, t+h]$ se produzca:

1. exactamente una llegada es $\lambda_n h + o(h)$, siendo $\lim_{t\to 0} o(h)/h = 0$.
2. ninguna llegada es $1 - \lambda_n h + o(h)$, siendo como antes, $o(h)$ un infinitésimo de orden inferior a h.
3. más de una llegada es $o(h)$.

Si $N(t)$ representa la ocupación del sistema en el instante t, y definimos

$$P_n(t) = \Pr(N(t) = n) \tag{6.7}$$

tenemos:

$$P_n(t+h) = P_n(t)(1 - \lambda_n h) + P_{n-1}(t)\lambda_{n-1}h + o(h) \;\; ; \;\; n > i \tag{6.8}$$

siendo,

$$P_i(0) = 1$$

(la población tiene inicialmente i elementos). Para $n = i$, se tiene además:

$$P_i(t+h) = P_i(t)(1 - \lambda_i h) + o(h) \tag{6.9}$$

(cómo sólo hay llegadas y el sistema parte de una ocupación $N(0) = i$ nunca podrá haber menos de i clientes en el sistema). Las ecuaciones (6.8) y (6.9) dan lugar a la ecuación diferencial:

$$P_n'(t) + \lambda_n P_n(t) = \lambda_{n-1} P_{n-1}(t) \; ; \quad n > i \tag{6.10}$$

junto a $P_i(t) = e^{-\lambda_i t}$ y $P_n(0) = 0 \; ; \; n > i$.

La solución de la ecuación diferencial (6.10) depende de la forma particular de la sucesión λ_n. Para el caso $\lambda_n = n\lambda$, con $n \geq 1$, el proceso de nacimiento puro recibe el nombre de *proceso de Yule*, y la expresión de la distribución marginal es:

$$P_n(t) = \begin{pmatrix} n-1 \\ n-i \end{pmatrix} \cdot e^{-i\lambda t} \left(1 - e^{-\lambda t}\right)^{n-i}$$

Simulación: *El proceso de Yule.*

Para simular en R un proceso puro de nacimiento podemos utilizar un código muy similar al que ya hemos empleado para el proceso de Poisson. La única modificación a realizar es tener en cuenta que si T_k es el instante de llegada del cliente k y en ese momento la ocupación del sistema es n, el tiempo hasta la llegada del cliente $k+1$ seguirá una distribución exponencial de parámetro λ_n. La siguiente función simularía el proceso de Yule, con $\lambda_n = n\lambda$.

```
procYule = function(n, N0 = 1, t0 = 0, lambda) {
    t_entre_llegadas = numeric(n)
    for (k in 1:n) t_entre_llegadas[k] = rexp(1, (N0 + k - 1) * lambda)
    tllegada = c(t0, t0 + cumsum(t_entre_llegadas))
    N = N0:(N0 + n)
    return(data.frame(t = tllegada, N = N))
}
```

El gráfico 6.2 muestra una simulación de 10 llegadas de este proceso, con tasa $\lambda_n = 2n$.

6.3. Procesos de nacimiento y muerte

Consideraremos ahora procesos en los que además de las llegadas (*nacimientos*), son también posibles las salidas (*muertes*). Supondremos que tanto la tasa de llegadas como la tasa de salidas en cada instante t son dependientes de la ocupación

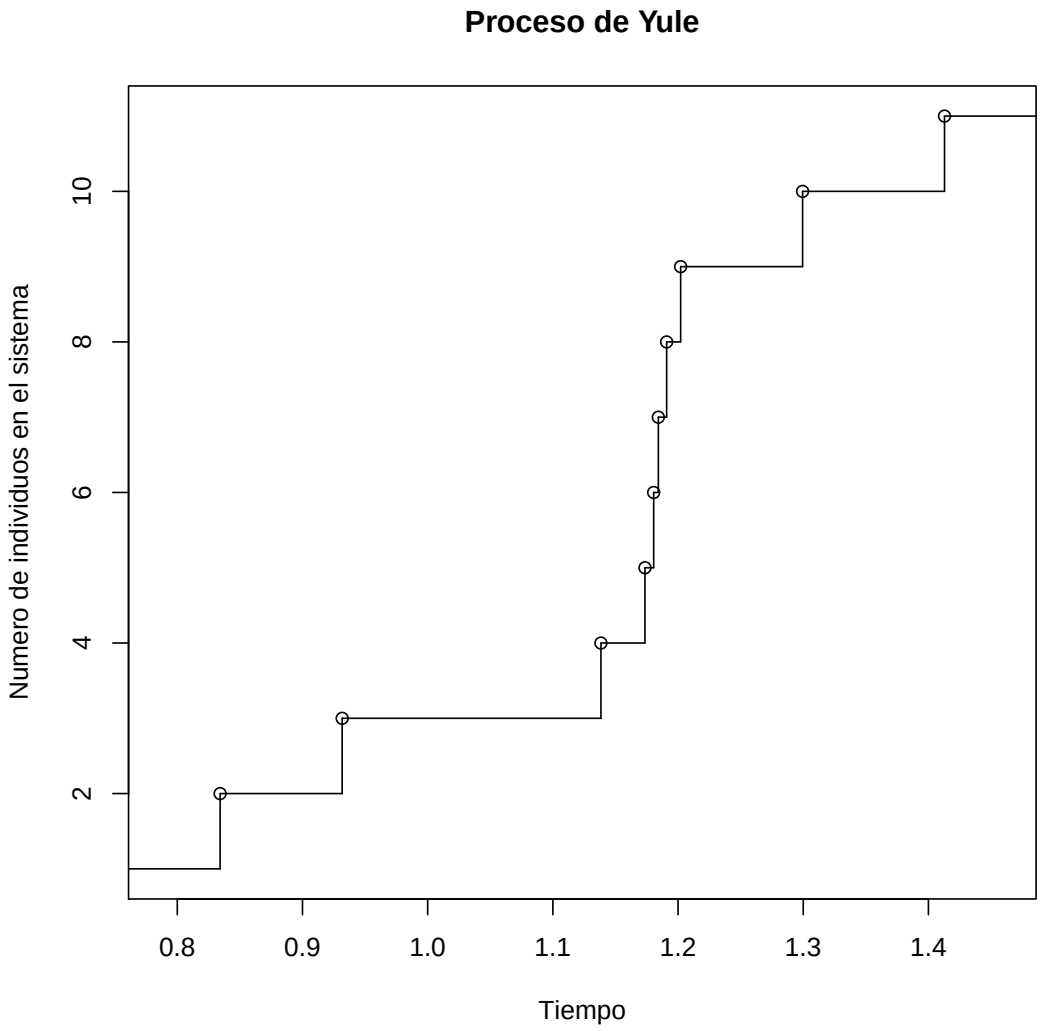


Figura 6.2: Trayectoria de un proceso de Yule con $\lambda_n = 2n$

$N(t)$ del sistema en ese instante. De manera más concreta, consideraremos sistemas en los que, para cada instante t, condicionado a que la ocupación del sistema es $N(t) = n$, la probabilidad de que en el intervalo $[t, t+h]$ se produzca:

1. Exactamente una llegada es $\lambda_n h + o(h)$, siendo $\lim_{h\to 0} o(h)/h = 0$.
2. Ninguna llegada es $1 - \lambda_n h + o(h)$.
3. Más de una llegada es $o(h)$.
4. Exactamente una salida es $\mu_n h + o(h)$.

5. Ninguna salida es $1 - \mu_n h + o(h)$.

6. Más de una salida es $o(h)$.

Supondremos además que las llegadas (nacimientos) y las salidas (muertes) se producen de modo independiente. Si nuevamente representamos el tamaño poblacional por $N(t)$, y la distribución marginal del proceso

$$P_n(t) = \Pr(N(t) = n)$$

se tiene, para $n \geq 1$:

$$\begin{aligned} P_n(t+h) = {} & P_n(t)(1-\lambda_n h)(1-\mu_n h) + P_{n-1}(t)\lambda_{n-1}h(1-\mu_n h) + \\ & + P_{n+1}(t)(1-\lambda_{n+1}h)\mu_{n+1}h + o(h) \end{aligned}$$

y para n=0,

$$P_0(t+h) = P_0(t)(1-\lambda_0 h) + P_1(t)(1-\lambda_1 h)\mu_1 h + o(h)$$

Estas ecuaciones dan lugar a las siguientes ecuaciones diferenciales para $n \geq 1$:

$$P_n'(t) = -(\lambda_n + \mu_n) P_n(t) + \lambda_{n-1} P_{n-1}(t) + \mu_{n+1} P_{n+1}(t) \tag{6.11}$$

junto a:

$$P_0'(t) = -\lambda_0 P_0(t) + \mu_1 P_1(t) \tag{6.12}$$

Supongamos ahora que el sistema alcanza a lo largo del tiempo una situación de equilibrio (sistema ergódico). Ello significa que, para cualquier valor de n, $\lim_{t\to\infty} P_n(t) = p_n$, lo que a su vez implica que $\lim_{t\to\infty} P_n'(t) = 0$. Si tomamos límites para t tendiendo a infinito en las ecuaciones (6.11) y (6.12), obtenemos las siguientes ecuaciones en diferencias finitas:

$$0 = -(\lambda_n + \mu_n) p_n + \lambda_{n-1} p_{n-1} + \mu_{n+1} p_{n+1} \quad ; \quad n \geqslant 1 \tag{6.13}$$

junto a:

$$0 = -\lambda_0 p_0 + \mu_1 p_1 \tag{6.14}$$

De (6.13) y (6.14) es fácil deducir que:

$$p_n = \frac{\lambda_0 \lambda_1 \ldots \lambda_{n-1}}{\mu_1 \mu_2 \ldots \mu_n} \cdot p_0 \tag{6.15}$$

Dado que la sucesión $\{p_n;\ n = 0, 1, ...\}$ es una distribución de probabilidad, verificará que $\sum_{n=0}^{\infty} p_n = 1$, de donde se deduce que:

$$p_0 = \frac{1}{1 + \sum_{n=1}^{\infty} \prod_{i=0}^{n-1} \frac{\lambda_i}{\mu_{i+1}}} \tag{6.16}$$

6.3.1. Simulación de un proceso de nacimiento y muerte

A continuación simularemos un proceso de nacimiento y muerte con tasa de llegada constante $\lambda_n = \lambda$ y tasa de salida también constante $\mu_n = \mu$. Para ello construimos la función `procNM()` que deberá devolvernos un total de n instantes de llegada y salida, acompañados de la ocupación del sistema en cada uno de ellos. La construcción de esta función es simple teniendo en cuenta lo siguiente: en cualquier instante arbitrario t (y por tanto en los instantes de llegada o salida) el tiempo hasta la próxima llegada o hasta la próxima salida depende sólo de la ocupación del sistema en ese instante, y no del tiempo transcurrido desde las últimas llegada o salida. En particular, en un instante t en que la ocupación del sistema sea n el tiempo hasta la próxima llegada será exponencial de parámetro λ_n y el tiempo hasta la próxima salida será también exponencial de parámetro μ_n. Por tanto, en cada instante de llegada o de salida, bastará con simular sendas exponenciales de parámetros λ_n y μ_n respectivamente. El siguiente evento en el sistema será el que primero ocurra de estos dos. Actualizamos la ocupación del sistema en consecuencia y repetimos. Hay que tener precaución si el sistema queda vacío ya que en tal caso ya no será posible una salida, sino tan sólo una llegada. Con todas estas consideraciones, el siguiente código genera nuestro proceso de nacimiento y muerte:

```
procNM = function(n, N0 = 0, t0 = 0, lambda = 0.1, mu = 0.2) {
    ocurr = function(N, mu, lambda) {
        tnac = rexp(1, lambda)
        if (N == 0)
            ocur = c(tnac, 1) else {
            tmuer = rexp(1, mu)
            if (tnac > tmuer)
                ocur = c(tmuer, -1) else ocur = c(tnac, 1)
        }
        return(ocur)
    }
    estado = data.frame(matrix(nrow = n + 1, ncol = 2))
    names(estado) = c("t", "N")
    estado[1, ] = c(t0, N0)
    for (k in 2:(n + 1)) estado[k, ] = estado[k - 1, ] + ocurr(estado$N[k -
        1], mu, lambda)
    return(estado)
}
```

El siguiente código simula 10 eventos en un proceso nacimiento/muerte, los muestra y los representa gráficamente. El resultado se muestra en la figura 6.3.

```
proceso = procNM(10, lambda = 2, mu = 1)
proceso
```

```
##         t N
## 1  0.0000 0
## 2  0.6865 1
## 3  0.7787 2
## 4  1.6965 3
## 5  2.5970 4
## 6  2.8925 3
## 7  3.1945 4
## 8  3.5669 3
## 9  3.9910 2
```

```
## 10 4.0516 1
## 11 4.1137 2

with(proceso, plot(stepfun(t[-1], N), xlab = "Tiempo", ylab = "Numero de individu
    main = "Proceso de Nacimiento y muerte"))
```

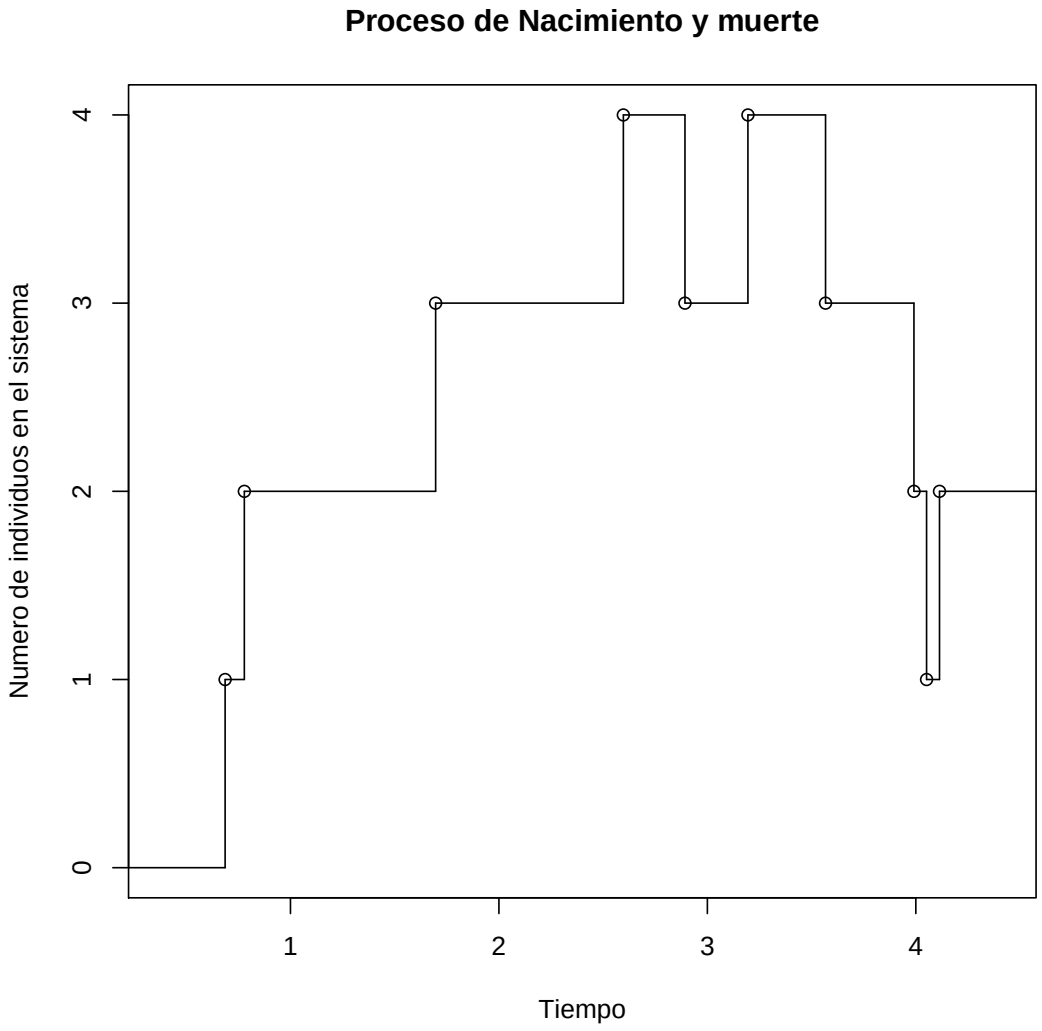


Figura 6.3: Trayectoria de un proceso de nacimiento-muerte con tasas de llegada y salida constantes.

El código anterior es puede modificarse fácilmente para simular procesos de nacimiento y muerte con tasas λ_n y μ_n no constantes.

6.4. Sistemas de colas

Los sistemas de colas constituyen un ejemplo de una clase más amplia de sistemas dinámicos conocidos como *sistemas de flujo*. Un sistema de flujo se caracteriza por la presencia de objetos que fluyen, se mueven o son transferidos a través de uno o más canales de capacidad finita. Las actuales redes de telecomunicación son sistemas de flujo: los objetos que fluyen son en este caso paquetes de datos, voz o video (en redes digitales), o llamadas telefónicas (en redes analógicas); los canales en estas redes son los cables o sistemas de radio que permiten el transporte de los objetos. Por capacidad finita de los canales entendemos el hecho de que un canal solo permite el transporte de objetos a una velocidad o tasa finita.

Un *sistema de colas* elemental es un sistema compuesto por una o más unidades, llamadas *procesadores* o *servidores*, encargados de realizar las tareas que demandan otras unidades, llamadas *clientes*, que van accediendo al sistema a lo largo del tiempo. Cuando la llegada de clientes supera la capacidad de procesamiento del sistema, los clientes permanecen en cola hasta que pueden acceder al servicio. La política o disciplina de la cola establece el orden en que los clientes son atendidos: puede consistir en que el primer cliente en llegar sea el primero en ser atendido, pero pueden adoptarse estrategias alternativas muy diversas. En el contexto particular de los sistemas de conmutación de paquetes, los clientes son los paquetes o mensajes generados por alguna fuente (datos, voz, video, etc.); los servidores son los canales a través de los cuales deben ser enviados estos paquetes o mensajes. Las colas se forman en los buffers disponibles en los conmutadores o multiplexores, donde los paquetes deben esperar a que se libere el canal correspondiente para poder ser transmitidos.

Los procesos de nacimiento-muerte permiten modelar aquellos sistemas de colas en los que las tasas de llegada y servicio de los clientes dependen únicamente de la ocupación del sistema en cada momento. Si cuando en el sistema hay n clientes la tasa de llegadas es λ_n y la tasa de servicio es μ_n, las ecuaciones (6.16) y (6.15) nos proporcionan las probabilidades de que haya n clientes en el sistema ($n \geq 0$) cuando éste se encuentra en equilibrio. Es importante entender el concepto de *"sistema en equilibrio"* en el contexto de los sistemas de colas: que las probabilidades p_n no dependan de t no quiere decir que el sistema no cambie de estado; de hecho los clientes seguirán entrando y saliendo del sistema, y la cola seguirá llenándose

y vaciándose en un proceso cíclico que puede considerarse estable. El valor de p_n representa la probabilidad de encontrar n clientes en el sistema en cualquier futuro instante arbitrario, una vez que ha transcurrido tiempo suficiente para que el sistema se estabilice.

El auge de los sistemas informáticos y de telecomunicación ha impulsado el desarrollo de un amplio conjunto de métodos analíticos y numéricos para resolver cuestiones relacionadas con sistemas de colas: análisis y diseño de protocolos de comunicación, control y gestión de recursos en redes de telecomunicaciones, dimensionado de redes de comunicación o modelado del tráfico que circula por las redes (voz, video y datos). En el caso de sistemas simples es posible desarrollar modelos matemáticos del comportamiento de la cola que permiten resolver los problemas que se plantean en las aplicaciones citadas. Sin embargo en el caso de sistemas complejos, la evaluación de su comportamiento debe llevarse a cabo a través de la simulación. Presentaremos algunos modelos de colas, acompañados de algoritmos en R que permiten simularlos, así como de otras funciones que llevan a cabo la estimación de sus parámetros característicos. Estos algoritmos pueden reutilizarse como componentes básicos para la simulación y evaluación de modelos de colas más complejos.

6.4.1. Elementos de un sistema de colas.

En general en los sistemas de colas, tanto los tiempos entre llegadas de clientes como las duraciones de los servicios son aleatorios; como consecuencia son también aleatorios los tiempos de espera y la longitud de la cola. Para el análisis de los sistemas de colas utilizaremos la siguiente notación:

- C_n: representa el n-ésimo cliente que llega al sistema.
- τ_n : instante de llegada de C_n.
- T_n : es tiempo transcurrido entre las llegadas de C_{n-1} y C_n, esto es, $T_n = \tau_n - \tau_{n-1}$.
- X_n: Tiempo de duración del servicio para el cliente C_n.
- W_n : Tiempo en línea de espera (cola) del cliente C_n.

- S_n : Tiempo en el sistema $(W_n + X_n)$ del cliente C_n.
- $N(t)$: Número de clientes en el sistema en el instante t.
- $Q(t)$: Tamaño de la cola en el instante t.

Si suponemos que las variables T_n tienen la misma distribución de probabilidad (tiempos entre llegadas idénticamente distribuidos), representaremos la función de distribución común por:

$$\Lambda(t) = \Pr(T_n \leqslant t)$$

Si los tiempos de servicio X_n tienen también la misma distribución de probabilidad (tiempos de servicio idénticamente distribuidos), representamos la función de distribución común por:

$$B(x) = \Pr(X_n \leqslant x)$$

6.4.2. Notación de Kendall

La notación de Kendall permite establecer una clasificación de los distintos modelos de colas en los que cada cliente debe recibir un solo servicio. Esta notación consiste en un un código de la forma $L/S/m/C/M$, donde:

- L especifica la distribución de probabilidad de los tiempos entre llegadas de clientes a la cola.
- S especifica la distribución de los tiempos de servicio.
- m es el número de servidores.
- C es la capacidad disponible en el sistema para acumular los clientes en cola (si no se especifica valor de K se entiende que la capacidad del sistema es infinita.
- M es el tamaño de la población de la que proceden los clientes. No se especifica si el tamaño es infinito.

Los códigos más habituales para estas distribuciones de los tiempos entre llegadas y de servicio son: M (exponencial), G (general), y D (determinista). Ejemplos de sistemas descritos por la notación de Kendall son:

M/M/1: Llegadas según un proceso de Poisson homogéneo (y por tanto tiempos entre llegadas con distribución exponencial), tiempos de servicio exponenciales y un solo servidor.

M/M/1/K: idéntico al anterior, pero con un buffer de capacidad finita K.

M/G/1: Llegadas según un proceso de Poisson homogéneo, tiempos de servicio con distribución general, un único servidor.

6.4.3. Intensidad de tráfico y ergodicidad

El cociente $\rho_k = \frac{\lambda_k}{\mu_k}$ se conoce con el nombre de *intensidad de tráfico,* y mide la relación entre el número de llegadas y el número de salidas por unidad de tiempo. Puede probarse que la condición necesaria y suficiente para que un proceso de nacimiento-muerte sea ergódico (alcance el equilibrio) es que exista un valor k_0 tal que para todo $k \geq k_0$:

$$\rho_k = \frac{\lambda_k}{\mu_k} < 1$$

Esta condición tiene un significado muy intuitivo: indica que el sistema sólo alcanza el equilibrio si a partir de cierto número de clientes en el sistema (k_0), la velocidad a que llegan los clientes es inferior a la velocidad con que son atendidos. Es fácil darse cuenta de que si ocurriese lo contrario, la cola crecería indefinidamente (todos los estados del sistema serían transitorios) y no se alcanzaría el equilibrio.

6.4.4. Fórmula de Little

Consideremos un sistema de colas en equilibrio, y sean $E[T]$ el tiempo esperado entre llegadas, $E[S]$ el tiempo esperado que permanece en el sistema un cliente arbitrario y $E[N]$ el número esperado de clientes en el sistema en un instante arbitrario. La *fórmula de Little* establece que:

$$E[N] = \frac{E[S]}{E[T]} \tag{6.17}$$

Si λ es el número esperado de llegadas por unidad de tiempo se tiene que $\lambda = 1/E[T]$ y la fórmula de Little puede expresarse también como:

$$E[N] = \lambda E[S]$$

Una demostración intuitiva de este resultado parte de las observaciones siguientes:

- Cuando un cliente arbitrario accede al sistema en equilibrio, el número esperado de clientes que encuentra es $E[N]$.
- Cuando se marcha del sistema, habrá permanecido en él durante un tiempo esperado $E[S]$.
- Como el tiempo medio entre llegadas es $E[T]$, durante el tiempo $E[S]$ habrá llegado un número esperado de clientes $E[S]/E[T]$.
- Si la disciplina de la cola es FIFO (*First In First Out,* los clientes son atendidos en su orden estricto de llegada), cuando el cliente se vaya, los $E[S]/E[T]$ que han llegado durante su servicio son los que quedan en el sistema.
- Como en un sistema de colas en equilibrio, el número esperado de clientes a la llegada de un cliente arbitrario debe coincidir con el número esperado a su salida, se sigue que $E[N] = E[S]/E[T]$.

Puede probarse que la fórmula de Little es válida bajo procesos de llegada y servicio muy generales, e incluso cuando la disciplina de la cola no es FIFO.

6.5. El sistema *M/M/1*

El sistema $M/M/1$ es el modelo de cola más simple. Consiste en un único canal de servicio o servidor en el que las llegadas y salidas se producen de acuerdo con

un proceso de nacimiento y muerte con tasas de llegada y salida respectivas:

$$\lambda_n = \lambda \quad n = 0, 1, 2, \cdots$$
$$\mu_n = \mu \quad n = 1, 2, 3, \cdots$$

Para un sistema de este tipo, los tiempos entre dos llegadas consecutivas, $\{T_n; n \in N\}$, son variables aleatorias independientes y con distribución de probabilidad exponencial de parámetro λ. Asimismo, los sucesivos tiempos de servicio $\{X_n; n \in N\}$ son también variables aleatorias independientes y con distribución de probabilidad exponencial de parámetro μ. De (6.15) y (6.16) se deduce que la distribución del tamaño de la cola en el equilibrio (sistema ergódico) viene dada por la expresión:

$$p_n = \left(1 - \frac{\lambda}{\mu}\right) \cdot \left(\frac{\lambda}{\mu}\right)^n \quad ; \quad n = 0, 1, 2, ... \tag{6.18}$$

que corresponde a la distribución geométrica de parámetro $\frac{\lambda}{\mu}$. Como se ha apuntado en 6.4.3, para que exista solución de equilibrio debe ocurrir que $\lambda < \mu$. Ello garantiza además que la ecuación (6.18) representa una distribución de probabilidad correctamente definida.

Si llamamos N a la variable aleatoria que representa el número de clientes en el sistema en el equilibrio, dado que su distribución de probabilidad es la distribución geométrica de parámetro $\rho = {}^{\lambda}/_{\mu}$, su esperanza vendrá dada por:

$$E[N] = \frac{\lambda}{\mu - \lambda} = \frac{\rho}{1 - \rho} \tag{6.19}$$

y su varianza:

$$\text{var}(N) = \frac{\rho}{(1 - \rho)^2} = \tag{6.20}$$

Si $E[S]$ es el tiempo esperado que cada cliente permanece en el sistema y tenemos en cuenta que el número medio de clientes que acceden al sistema por unidad de tiempo es λ (ya que los tiempos entre dos llegadas consecutivas tienen distribución de probabilidad exponencial de parámetro λ), aplicando la fórmula de Little se tiene que, en los sistemas $M/M/1$:

$$E[S] = \frac{E[N]}{\lambda} = \frac{1/\mu}{1-\rho} \tag{6.21}$$

El tiempo medio de espera en cola puede calcularse como $E[W] = E[S] - E[X] = E[S] - \frac{1}{\mu}$. Sustituyendo el valor obtenido en (6.21) se tiene:

$$E[W] = \frac{\rho/\mu}{1-\rho} \tag{6.22}$$

Obsérvese que tanto $E[N]$ como $E[S]$ y $E[W]$ son inversamente proporcionales a $1-\rho$. Por tanto el número medio de clientes en el sistema y los tiempos medios el cola y en el sistema, crecen ilimitadamente a medida de la intensidad de tráfico ρ se aproxima a 1.

6.5.1. Simulación de la cola *M/M/1*

En el capítulo anterior ya presentamos una simulación del proceso de nacimiento-muerte con tasas de llegada y salida constantes, proceso que coincide con una cola M/M/1 como acabamos de ver. En aquella simulación sólo se obtenía el número de sujetos en el sistema. A continuación desarrollamos una nueva función que, además, para cada cliente guarda su instante de llegada, su tiempo de servicio y su instante de salida. La función recibe como parámetros el número `n` de clientes a simular, y las tasas λ y μ. La salida es una lista formada por dos *data.frames:* uno que contiene los tiempos citados y otro que contiene el número de clientes en el sistema en cada instante de llegada y salida

```
colaMM1 = function(n, lambda, mu) {
    llegada = cumsum(rexp(n, lambda))
    servicio = rexp(n, mu)
    salida = numeric(n)
    salida[1] = llegada[1] + servicio[1]
    for (k in 2:n) salida[k] = max(llegada[k], salida[k - 1]) + servicio[k]
    tiempos = data.frame(llegada = llegada, servicio = servicio, salida = salida)
    inout = rep(c(1, -1), n)
    tes = c(rbind(llegada, salida))
```

```
    ot = order(tes)
    sistema = data.frame(t = c(0, tes[ot]), n = c(0, cumsum(inout[ot])))
    return(list(tiempos = tiempos, sistema = sistema))
}
```

La siguiente función estima algunos parámetros de interés a partir de la simulación realizada por la función anterior:

```
describeCola = function(cola) {
    tmel = mean(diff(cola$tiempos$llegada))  # tiempo medio entre llegadas
    tmserv = mean(cola$tiempos$servicio)  # tiempo medio de servicio
    tmsist = with(cola$tiempos, mean(salida - llegada))  # tiempo medio en el sis
    tmcola = tmsist - tmserv  # tiempo medio en cola
    ns = nrow(cola$sistema)
    numCola = with(cola$sistema, ifelse(n <= 1, 0, n - 1))
    numMedSistema = with(cola$sistema, sum(diff(t) * n[-ns])/t[ns])
    numMedCola = with(cola$sistema, sum(diff(t) * numCola[-ns])/t[ns])
    resumen = data.frame(cbind(tmel, tmserv, tmsist, tmcola, numMedSistema,
        numMedCola))
    return(resumen)
}
```

A continuación simulamos una cola $M/M/1$, le aplicamos la función `describeCola()` y hacemos una gráfica de la evolución del número de clientes en el sistema, que se muestra en la figura 6.4 :

```
cola = colaMM1(50, 2, 3)
describeCola(cola)

##     tmel tmserv tmsist tmcola numMedSistema numMedCola
## 1 0.4135  0.334 0.6663 0.3323         1.472     0.7343

with(cola$sistema, plot(stepfun(t[-1], n), xlab = "Tiempo", ylab = "Numero en el
    main = "Cola M/M/1"))
```

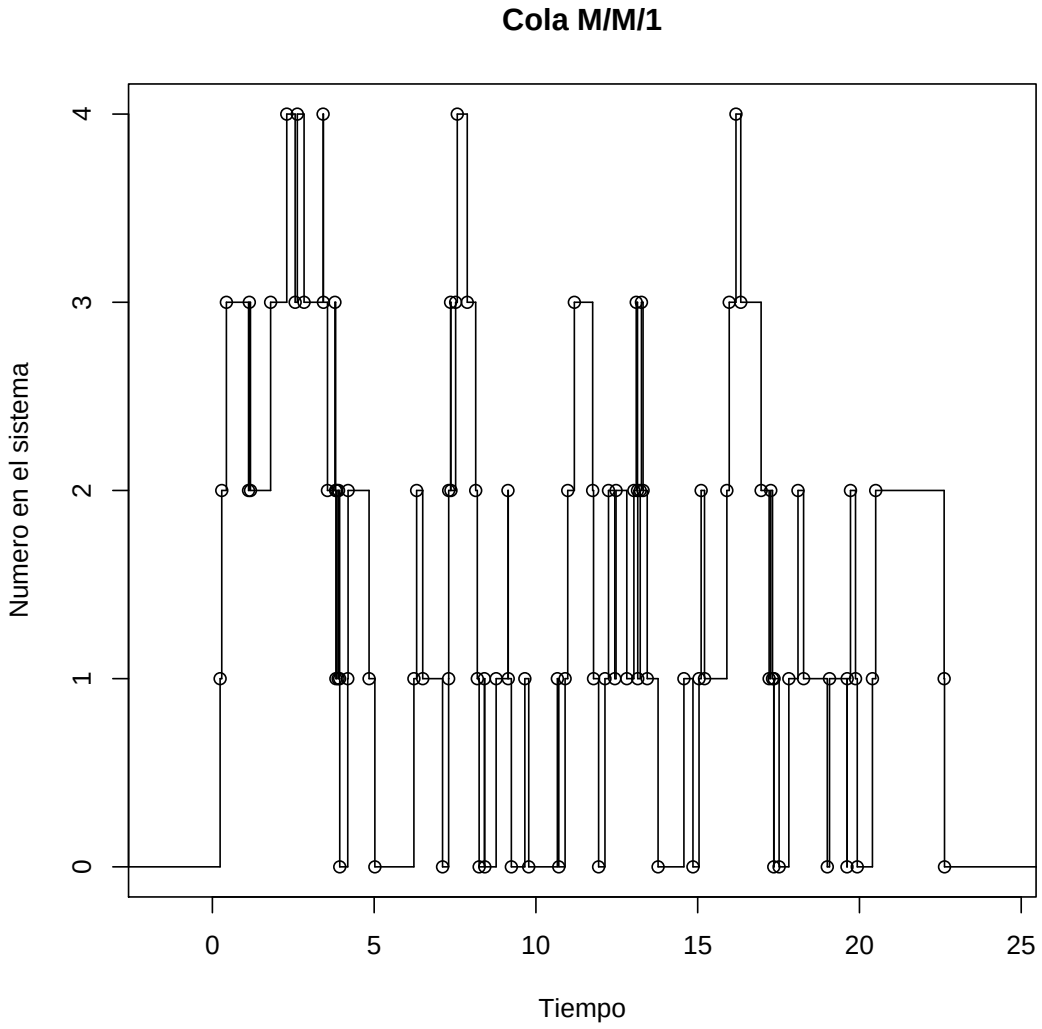


Figura 6.4: Simulación de una cola M/M/1

6.6. Sistemas con capacidad finita: la cola $M/M/1/C$

El modelo $M/M/1$ admite múltiples generalizaciones, siendo una de las más inmediatas que la capacidad de la cola sea un valor finito C. Ello significa que el sistema puede albergar como máximo C clientes (incluido el que está recibiendo servicio) por lo que los que lleguen mientras el sistema está lleno son rechazados. Esta situación puede ser modelada fácilmente mediante un proceso de nacimiento

y muerte si definimos:

$$\begin{aligned} \lambda_k &= \begin{cases} \lambda & k < C \\ 0 & k \geq C \end{cases} \\ \mu_k &= \mu, \quad k = 1, 2, 3, ..., C \end{aligned}$$

Este sistema siempre alcanza el equilibrio, pues para $k \geqslant C$ se tiene $\lambda_k/\mu_k = 0$. Aplicando las fórmulas (6.15) y (6.16) con estos coeficientes se obtiene sin dificultad:

$$p_k = \begin{cases} \frac{1-(\lambda/\mu)}{1-(\lambda/\mu)^{C+1}} \left(\frac{\lambda}{\mu}\right)^k & 0 \leq k \leq C \\ 0 & k > C \end{cases}$$

En este sistema la probabilidad de que un cliente sea rechazado coincide con la probabilidad de encontrar el sistema lleno a su llegada, esto es, $P\,(Rechazo) = p_C$.

6.6.1. Simulación de la cola M/M/1/C

La siguiente función permite simular en R una cola $M/M/1/C$. Es una función notablemente más compleja que la que hemos usado ya que ahora es preciso revisar para cada llegada si el sistema dispone de capacidad suficiente para admitirla o bien si debe ser rechazada.

```
colaMM1C = function(n, lambda, mu, C) {
    tllegada = cumsum(rexp(n, lambda))
    tservicio = rexp(n, mu)
    tsalida = numeric(n)
    nllegadas = 0
    nsistema = 0
    sistema = NULL
    evolsistema = NULL
    rechazados = NULL
    while (nllegadas < n) {
        nllegadas = nllegadas + 1
        # Se actualiza el sistema, eliminando del mismo todos los que salen
```

```
        # antes de esta llegada
        if (nsistema > 0) {
            salen = which(tsalida[sistema] < tllegada[nllegadas])
            nsalen = length(salen)
            if (nsalen > 0) {
                evolsistema = rbind(evolsistema, cbind(tsalida[sistema[salen]],
                  -sistema[salen], (nsistema - 1):(nsistema - nsalen)))
                sistema = sistema[-salen]
                nsistema = nsistema - nsalen
            }
        }
        # Ahora el nuevo sujeto se a<U+00F1>ade o es rechazado si en el sistema
        # hay C clientes
        if (nsistema < C) {
            tacser = ifelse(nsistema == 0, tllegada[nllegadas], tsalida[sistema[n
            tsalida[nllegadas] = tacser + tservicio[nllegadas]
            sistema = c(sistema, nllegadas)
            nsistema = nsistema + 1
            evolsistema = rbind(evolsistema, c(tllegada[nllegadas], nllegadas,
                nsistema))
        } else rechazados = c(rechazados, nllegadas)
    }
    tservicio[rechazados] = 0
    tiempos = data.frame(llegada = tllegada, servicio = tservicio, salida = tsali
    evolsistema = data.frame(evolsistema)
    names(evolsistema) = c("t", "cliente", "nsistema")
    list(tiempos = tiempos, cola = evolsistema, rechazados = rechazados)
}
```

6.7. Colas con más de un servidor: cola $M/M/m$

La cola $M/M/m$ es la generalización del modelo M/M/1 al caso de m servidores. Supondremos que el cliente que ocupa la primera posición de la cola es atendido

por el primer servidor que queda libre. Para modelar esta cola mediante un proceso de nacimiento-muerte basta con hacer las siguientes consideraciones:

- La tasa de llegadas es $\lambda_k = \lambda \ \forall k$
- La tasa de servicio μ de cada servidor es también constante, de forma que la tasa de servicio global del sistema es

$$\mu_k = \begin{cases} k\mu & 0 \leq k \leq m \\ m\mu & k > m \end{cases} = \text{mín}\left\{k\mu, m\mu\right\}$$

La intensidad de tráfico en este sistema es entonces $\rho = \lambda/m\mu$.

Aplicando las fórmulas (6.15) y (6.16) se sigue sin dificultad que la probabilidad en el equilibrio de que haya k clientes en el sistema es:

$$p_k = \begin{cases} p_0 \frac{(\rho m)^k}{k!} & k \leq m \\ p_0 \frac{\rho^k m^m}{m!} & k \geq m \end{cases} \qquad p_{0=} = \left[\sum_{k=0}^{m-1} \frac{(m\rho)^k}{k!} + \frac{(m\rho)^m}{m!\,(1-\rho)}\right]^{-1}$$

Una cantidad que suele resultar de interés es la probabilidad de que un nuevo cliente tenga que hacer cola; en este caso, dicha probabilidad es:

$$Q = \sum_{k=m}^{\infty} p_k = \frac{\frac{(m\rho)^m}{m!}\left(\frac{1}{1-\rho}\right)}{\left[\sum_{k=0}^{m-1} \frac{(m\rho)^k}{k!} + \frac{(m\rho)^m}{m!(1-\rho)}\right]}$$

Esta expresión se conoce como *fórmula C de Erlang*, que la empleó a principios del siglo pasado para el estudio de sistemas telefónicos.

6.8. Sistemas de colas no markovianos.

Los modelos de colas elementales vistos hasta ahora son puramente markovianos: suponen que el proceso de llegadas de clientes al sistema es de Poisson, lo que significa que el tiempo entre llegadas sucesivas de clientes sigue una distribución exponencial, y que la duración del servicio es también exponencial. Estos modelos resultan extremadamente simples pues la falta de memoria de la distribución

exponencial da lugar a que, una vez conocido el número de clientes presente en el sistema en un instante determinado, cualquier otra información sobre la historia del sistema es irrelevante para el conocimiento (probabilista) de su futuro. Ahora bien, ello también da lugar a que las características de estos sistemas sean muchas veces inverosímiles en la práctica. La situación más habitual es que el conocimiento de cuánto tiempo lleva un cliente ocupando el servicio proporcione información sobre el tiempo que aún le resta para terminar. Asimismo, el tiempo transcurrido desde la última llegada de un cliente al sistema puede informar sobre el tiempo que falta hasta la próxima llegada. El tratamiento analítico de estos modelos es bastante más complicado y requiere el uso de métodos matemáticos que exceden el nivel de este curso. No obstante, su simulación puede llevarse a cabo mediante procesos muy similares a los vistos en este capítulo. En el apéndice se ilustra la programación en R mediante la simulación de un modelo de colas con llegadas y salidas no markovianos que permite estimar en dicho modelo parámetros de interés relativos a la ocupación del sistema y los tiempos de espera.

Problemas

1. Una centralita recibe llamadas según un proceso de Poisson de tasa promedio 10 llamadas por hora.

   a) Calcula la probabilidad de que en 10 minutos lleguen 3 ó más llamadas.

   b) ¿Cuál es la probabilidad de que el tiempo entre dos llamadas sucesivas sea inferior a 6 minutos?

2. Consideremos un sistema de cola markoviana en la cual

$$\lambda_n = \begin{cases} \lambda & 0 \leq n \leq K \\ 2\lambda & n > K \end{cases}$$
$$\mu_n = \mu \quad n = 1, 2, \cdots$$

   a) Encontrar la distribución de probabilidad del tamaño del sistema en el equilibrio.

   b) ¿Qué relación debe existir entre los parámetros del problema en relación a que el sistema sea estable y de esta forma pueda alcanzarse la solución

de equilibrio?

3. Sea el sistema de cola markoviana en la cual

$$\lambda_n = \alpha^n \lambda \qquad n \geq 0,\ 0 \leq \alpha \leq 1$$
$$\mu_n = \mu \qquad n \geq 1$$

Encontrar la distribución de probabilidad del tamaño del sistema en el equilibrio.

4. Un conmutador recibe paquetes de acuerdo con un proceso de Poisson de parámetro λ paquetes/segundo. Cada paquete tarda exactamente c segundos en ser procesado (lectura de su cabecera+reenvío). El conmutador dispone de un buffer de capacidad infinita para almacenar los paquetes en espera. Si en $t = 0$ el conmutador está libre calcular:

   a) La probabilidad de que el segundo paquete tenga que esperar.

   b) El tiempo medio de espera del segundo paquete.

5. Se dispone de dos sistemas de colas, el primero un M/M/1 con tasa de llegadas λ_1 y tasa de servicio μ_1 y el segundo un M/M/2 con idéntica tasa de llegadas λ_2 y tasa de servicio μ_2 por cada servidor.

   a) ¿Cuánto debería valer μ_1 para que el tiempo medio en cola de los clientes del primer sistema sea igual al tiempo medio en cola de los clientes del segundo?

   b) ¿y para que la distribución de probabilidad del tiempo en cola de los del primer sistema sea igual a los del segundo?

   c) Conociendo el coste relativo de incrementar la velocidad del servidor frente a incrementar el número de servidores, ¿cuándo resultaría más ventajoso incrementar la velocidad de servicio?

6. Se dispone de un sistema de colas con buffer finito de capacidad K. Las llegadas se producen según un proceso de Poisson y el tiempo de servicio es exponencial. Las tasas de llegada y de servicio vienen dadas respectivamente, por:

$$\lambda_n = (K - n)\,\lambda, \quad \mu_n = n\mu, \quad n = 0, 1, \ldots, K$$

a) ¿En qué condiciones alcanza el equilibrio este sistema?

b) Hallar la distribución de probabilidad del número de clientes en el sistema en equilibrio.

c) Hallar el número medio de clientes en cola y el número medio de clientes en el sistema.

d) Determinar el tiempo medio de espera en cola y el tiempo medio de espera en el sistema.

7. Calcular la probabilidad de pérdida en un sistema M/M/c/c.

8. Un sistema de conmutación de paquetes recibe paquetes a razón de 10 por segundo. Cada paquete tarda en ser transmitido 0.05 segundos con probabilidad 3/4 y 0.1 segundos con probabilidad 1/4. Suponiendo que se dispone de un buffer ilimitado para acumular los paquetes hasta su transmisión, determinar el número medio de paquetes en cola y la probabilidad de que haya más de un paquete en el sistema (suponer llegadas y servicios markovianos)

9. Una conexión entre centralitas dispone de 5 líneas. Esta conexión es utilizada por un promedio de 50 llamadas cada hora, siendo 3 minutos la duración media de las llamadas.

   a) ¿Cuál es la intensidad de tráfico soportada por la conexión?

   b) ¿Cuál es la probabilidad de que una línea esté ocupada?

   c) ¿Cuál es el número medio de líneas ocupadas?

10. Una sucursal bancaria con cuatro teléfonos dispone de dos líneas telefónicas privadas para conectarse con la oficina central. Cada teléfono origina un promedio de 2 llamadas por hora, con una duración media de 3 minutos. Se supone que las llamadas se producen completamente al azar según una distribución de Poisson y que la duración de las mismas sigue una distribución exponencial. (a) Si el sistema no admite llamadas en espera, calcular la probabilidad de bloqueo (esto es, que una llamada no se pueda realizar por encontrarse las dos líneas ocupadas) (b) El tiempo medio de espera cuando

se habilita un buffer que permite la espera de dos llamadas (a la vez que otras dos están activas).

11. Supongamos que a un conmutador en una red de conmutación de paquetes le llegan, por término medio, 6 paquetes por segundo, de acuerdo con una distribución de Poisson. Los paquetes tienen una longitud fija de 1200 bits. Estos paquetes son transmitidos a través de dos líneas de 4800 bits/seg. (a) Determinar el tamaño del buffer para que la proporción de paquetes rechazados sea inferior a 10-5. (b) Con este tamaño de buffer determinar el tiempo medio de espera de los paquetes en el conmutador.

12. Considérense dos sistemas de colas M/M/1 con las idénticas tasas de llegadas λ y servicio μ. Si estos dos sistemas operan lado a lado (con distintas colas), demostrar que el número total de clientes en los dos sistemas tomados conjuntamente tiene una distribución de probabilidad dada por:

$$P\left(N=n\right)=\left(n+1\right)\left(1-\rho\right)^{2}\rho^{n},\ n\geq 0$$

13. Considérense:

    (1) c colas *M/M/1* cada una de ellas con tasa de llegadas λ y tasa de servicio μ.

    (2) Una cola *M/M/c* con tasa de llegadas $c\lambda$ y tasa de servicio μ.

    Demostrar que el tiempo medio de espera en el sistema (2) es inferior al tiempo medio de espera en el sistema (1) para cualquier valor de $c > 1$ (Nótese de paso que ello justifica que sea mejor el sistema de cola única en bancos y administración).

Capítulo 7

Procesos Estacionarios

Las señales, tanto si son voltajes como ondas de corriente, son funciones del tiempo y pueden ser deterministas o aleatorias. Las señales deterministas pueden describirse como funciones en el sentido matemático habitual, siendo el tiempo la variable independiente. En contraste con las señales deterministas, una señal aleatoria tiene elementos de incertidumbre y por ello no es posible determinar exactamente su valor en un instante dado de tiempo. En esta lección estudiamos los *procesos estocásticos estacionarios* a través de los cuales modelizaremos las señales aleatorias. Éstas son *funciones aleatorias* del tiempo que representaremos por $X(t)$, donde $t \in T$. El conjunto $T \subset \mathbb{R}$ representa el tiempo, y a menudo $T = [a, b]$ (señal observada sobre el continuo $[a, b]$) ó $T = \mathbb{Z}$ (señal digitalizada).

7.1. Estacionariedad en sentido amplio

Un proceso estocástico, $\{X(t) : t \in T\}$, con espacio de tiempos $T \subset \mathbb{R}$ se dice *estacionario en un sentido amplio* si satisface las siguientes condiciones:

1. $E[X(t)] = \mu \ , \forall t \in T$
2. $\mathrm{cov}(X(t), X(t-\tau)) = R(\tau)$ (función independiente de t)

La función $R(\tau)$ recibe el nombre de función de autocovarianza y satisface las siguientes propiedades:

1. $R(0) = \text{var}(X(t)) = \sigma_X^2$
2. $|R(\tau)| \leqslant R(0) \quad \forall \tau.$
3. $R(-\tau) = R(\tau), \ \forall \tau$

Sin pérdida de generalidad, asumiremos que $E[X(t)] = 0$, lo que significa que la función de autocovarianza se reduce a:

$$R(\tau) = E[X(t) X(t-\tau)]$$

En lo que sigue consideraremos procesos estacionarios en los que $T = \mathbb{Z}$ (procesos en tiempo discreto). Tales procesos en la práctica modelizan señales digitalizadas.

7.2. Procesos estacionarios especiales

En esta sección se describen algunos procesos estacionarios en tiempo discreto ($T = \mathbb{Z}$) que son de uso frecuente.

7.2.1. Ruido blanco.

Se dice que un proceso estacionario en tiempo discreto $\{\varepsilon(t) \ , : \ t \in \mathbb{Z}\}$ es un ruido blanco, si su función de autocovarianza es:

$$R(\tau) = \begin{cases} \sigma_\varepsilon^2 & si \quad \tau = 0 \\ 0 & si \quad \tau \neq 0 \end{cases} \tag{7.1}$$

En definitiva, una sucesión de variables aleatorias incorreladas, de varianza constante es un *ruido blanco*. El ruido se dice *gaussiano* si además se verifica que $\varepsilon(t) \cong N(0, \sigma_\varepsilon) \ \forall t \in \mathbb{Z}$.

Ejemplo. Para un ruido blanco $\{\varepsilon(t) \ , : \ t \in \mathbb{Z}\}$, probar que, para $s < t$ se satisface que la variable aleatoria $\sum_{k=1}^{s} \beta_k \cdot \varepsilon(k)$ es incorrelada con $\varepsilon(t)$. En efecto,

$$E\left[\left(\sum_{k=1}^{s}\beta_k\cdot\varepsilon\left(k\right)\right)\cdot\varepsilon\left(t\right)\right]=\sum_{k=1}^{s}\beta_k E\left[\varepsilon\left(k\right)\cdot\varepsilon\left(t\right)\right]=0$$

dado que $E\left[\varepsilon\left(k\right)\cdot\varepsilon\left(t\right)\right]=0$ al ser $\varepsilon\left(k\right)$ incorrelada con $\varepsilon\left(t\right)$ para $k<t$.

7.2.2. Proceso autorregresivo de orden 1 ($AR(1)$)

Un proceso estacionario en tiempo discreto $\{X\left(t\right):t\in\mathbb{Z}\}$ se dice *autorregresivo de orden 1 (AR(1))* si satisface la ecuación:

$$X\left(t\right)=\alpha X\left(t-1\right)+\varepsilon\left(t\right) \tag{7.2}$$

siendo $\mid\alpha\mid<1$ y $\{\varepsilon\left(t\right):t\in\mathbb{Z}\}$ un ruido blanco, tal que $\text{var}\left(\varepsilon_t\right)=\sigma_\varepsilon^2$. De esta definición se sigue que:

$$X\left(t\right)=\alpha X\left(t-1\right)+\varepsilon\left(t\right)=$$

$$\alpha\left(\alpha X\left(t-2\right)+\varepsilon\left(t-1\right)\right)+\varepsilon\left(t\right)=$$

$$\alpha^2X\left(t-2\right)+\alpha\cdot\varepsilon\left(t-1\right)+\varepsilon\left(t\right)=$$

$$\alpha^3X\left(t-3\right)+\alpha^2X\left(t-2\right)+\alpha\cdot\varepsilon\left(t-1\right)+\varepsilon\left(t\right)$$

De lo anterior surge la siguiente representación lineal alternativa de los procesos $AR\left(1\right)$:

$$X\left(t\right)=\sum_{k=0}^{\infty}\alpha^k\cdot\varepsilon\left(t-k\right)$$

Nótese que $X(t-\tau)$ es incorrelada con $\varepsilon(t)$. Esto supone que:

$$\text{var}\left(X\left(t\right)\right)=\text{var}\left(\alpha X\left(t-1\right)+\varepsilon\left(t\right)\right)=\alpha^2\text{var}\left(X\left(t-1\right)\right)+\sigma_\varepsilon^2$$

Lo anterior significa que:

$$\sigma_X^2=\alpha^2\sigma_X^2+\sigma_\varepsilon^2$$

Se tiene finalmente que:

$$\sigma_X^2 = \frac{\sigma_\varepsilon^2}{1-\alpha^2}$$

Calculemos ahora la función de autocovarianza del proceso. Si $\tau > 0$, se tiene:

$$R(\tau) = E[X(t)\,X(t-\tau)] = E[(\alpha X(t-1) + \varepsilon(t))\,X(t-\tau)] = \alpha R(\tau-1)$$

puesto que $E[\varepsilon(t)\,X(t-\tau)] = 0$. Aplicando repetidamente este resultado se sigue que, para $\tau > 0$, $R(\tau) = \alpha^\tau \sigma_X^2$. Para $\tau < 0$ y dado que $R(-\tau) = R(\tau)$, ocurre: $R(\tau) = R(-\tau) = \alpha^{-\tau}\sigma_X^2$.

De todo lo anterior se deduce finalmente:

$$R(\tau) = \alpha^{|\tau|}\sigma_X^2 = \frac{\sigma_\varepsilon^2 \alpha^{|\tau|}}{1-\alpha^2} \quad : \quad \forall \tau \in \mathbb{Z}$$

El siguiente código R simula una trayectoria de un proceso $AR(1)$, siendo el parámetro $\alpha = 0.8$ y el ruido blanco gausiano.

```
procesoAR1 = function(a, se, n) {
    sx = se/sqrt(1 - a^2)
    X = numeric(n)  # Se inicializa un vector (vacio) de longitud n
    e = rnorm(n, 0, se)  # Ruido blanco
    X[1] = rnorm(1, 0, sx)  # Primera observacion
    for (t in 2:n) X[t] = a * X[t - 1] + e[t]
    return(X)
}
X = procesoAR1(a = 0.8, se = 1, n = 500)
```

La trayectoria simulada se muestra en la figura 2.1.

7.2.3. Proceso de medias móviles de orden q ($MA(q)$).

Un proceso estacionario en tiempo discreto $\{X(t) : t \in \mathbb{Z}\}$ se dice que es un proceso de medias móviles *de orden q* (MA por sus siglas en inglés *moving average*) si satisface la ecuación:

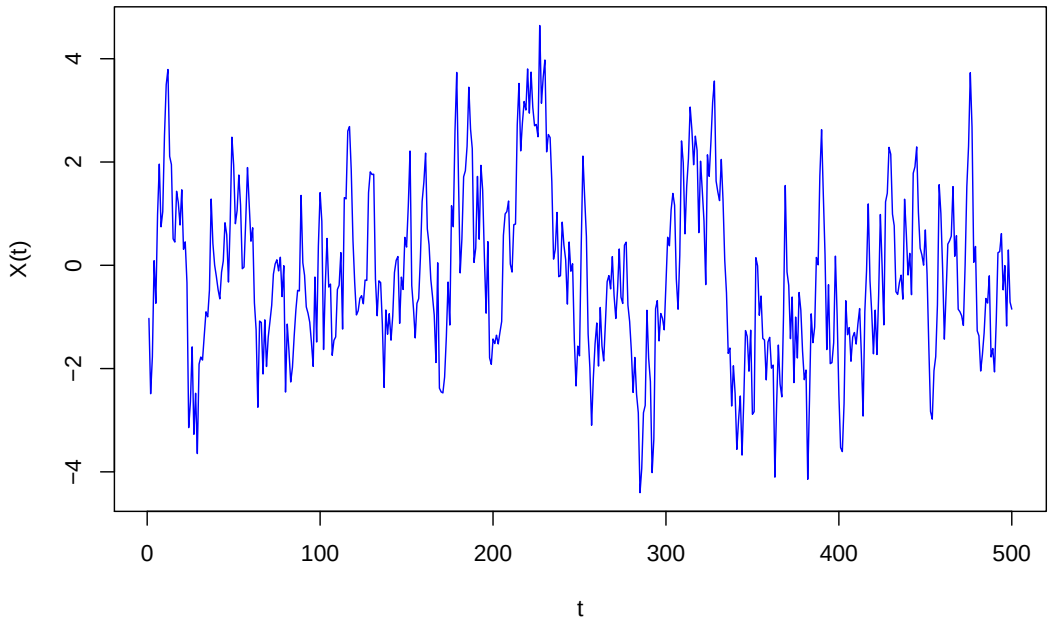


Figura 7.1: Trayectoria de un proceso AR(1) con $\alpha = 0{,}8$

$$X(t) = b_0\varepsilon(t) + b_1\varepsilon(t-1) + \ \ldots \ + b_q\varepsilon(t-q) \tag{7.3}$$

siendo $\{\varepsilon(t) \ : t \in \mathbb{Z}\}$ un ruido blanco.

Mediante el siguiente código R se simula una trayectoria de un proceso $MA(4)$. Ésta se representa en la figura 2.2 conjuntamente con el ruido blanco que lo define. Obsérvese que el proceso $MA(4)$ es un *alisamiento* del proceso de ruido blanco.

```
se=1
n=50
e=rnorm(n,0,se)    # Ruido blanco
b0=0.25; b1=0.2; b2=0.2; b3=0.2; b4=0.15
X=numeric(n)
for(t in 5:n)
  X[t]=b0*e[t]+b1*e[t-1]+b2*e[t-2]+b3*e[t-3]+b4*e[t-4]
```

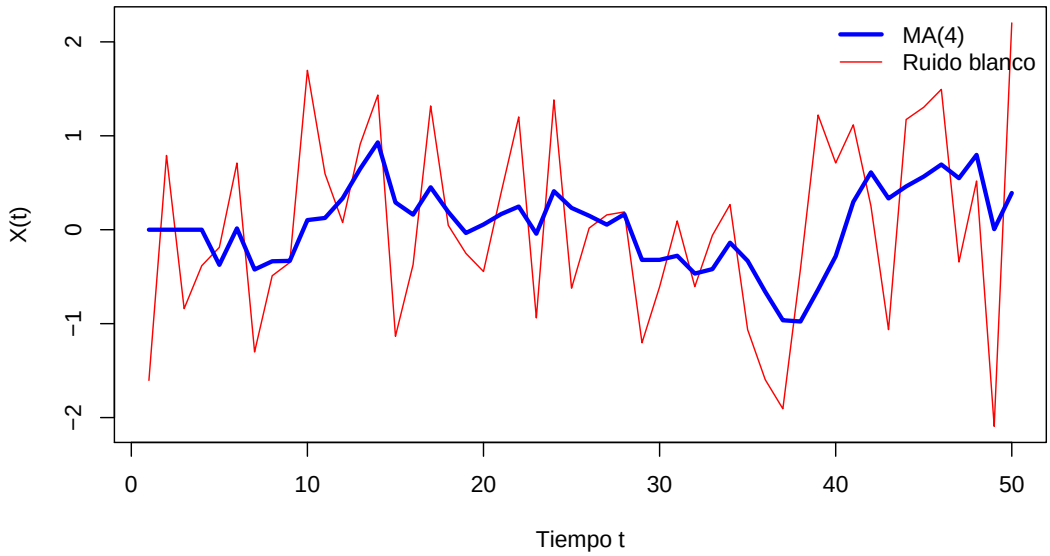


Figura 7.2: Proceso $MA(4)$ y el ruido blanco que lo genera. Nótese que el MA es un alisamiento del ruido

7.2.4. Procesos armónicos

Un proceso estacionario $\{X(t) : t \in \mathbb{R}\}$ es armónico si tiene la forma:

$$X(t) = \sum_{i=1}^{K} A_i \cos(\omega_i t + \phi_i) \tag{7.4}$$

siendo $A_1, ..., A_K, \omega_1, ..., \omega_K$ constantes especificadas y $\phi_1, ..., \phi_K$ variables aleatorias independientes e idénticamente distribuidas con distribución de probabilidad uniforme en el intervalo $[0, 2\pi]$ ($\phi_i \cong U[0, 2\pi]$). En definitiva, un proceso armónico es una *superposición finita de movimientos armónicos* de distintas amplitudes, frecuencias y fases.

Veamos ahora que $E[X(t)] = 0$. Para ello, nótese en primer lugar que si $\phi \cong U[0, 2\pi]$, entonces:

$$E[\cos(\omega t + \phi)] = \frac{1}{2\pi} \int_0^{2\pi} \cos(\omega t + u)\, du = 0$$

De esta forma se tiene:

$$E\left[X\left(t\right)\right] = \sum_{i=1}^{K} A_i E\left[\cos\left(\omega_i t + \phi_i\right)\right] = 0$$

Puede probarse también que:

$$R\left(\tau\right) = \frac{1}{2}\sum_{i=1}^{K} A_i^2 \cos\omega_i \tau$$

De aquí,

$$R\left(0\right) = \sigma_X^2 = \mathrm{var}\left(X_t\right) = \frac{1}{2}\sum_{i=1}^{K} A_i^2$$

Esta representación indica que la contribución del armónico $A_i \cos\omega_i t$ a la varianza del proceso σ_X^2 es obviamente $A_i^2/2$. Este resultado es la base del análisis espectral que se introduce en la siguiente sección.

En la figura 2.3 se representan conjuntamente tres movimientos armónicos y la suma correspondiente en trazo grueso. El movimiento armónico X_1 tiene amplitud $A_1 = 10$ y frecuencia $\omega_1 = 0{,}1$, para X_2, $A_2 = 6$ y $\omega_2 = 0{,}3$ y para X_3, $A_3 = 3$ y $\omega_3 = 0{,}6$. Obsérvese que el armónico X_1 tiene la mayor amplitud y la menor frecuencia, y por tanto la mayor contribución a la varianza del proceso armónico.

7.3. Análisis espectral

Los procesos armónicos son sumas (*superposiciones finitas*) de movimientos armónicos de diferentes amplitudes, frecuencias y fases, siendo su varianza la semisuma de los cuadrados de las amplitudes de los armónicos. De esta forma, *cada armónico* $(A_i \cos\left(\omega_i t + \phi_i\right))$ *contribuye a la varianza del proceso en la mitad del cuadrado de su amplitud* $(A_i^2/2)$.

En general, y bajo determinadas condiciones, los procesos estacionarios son superposiciones (no necesariamente finitas o numerables) de movimientos armóni-

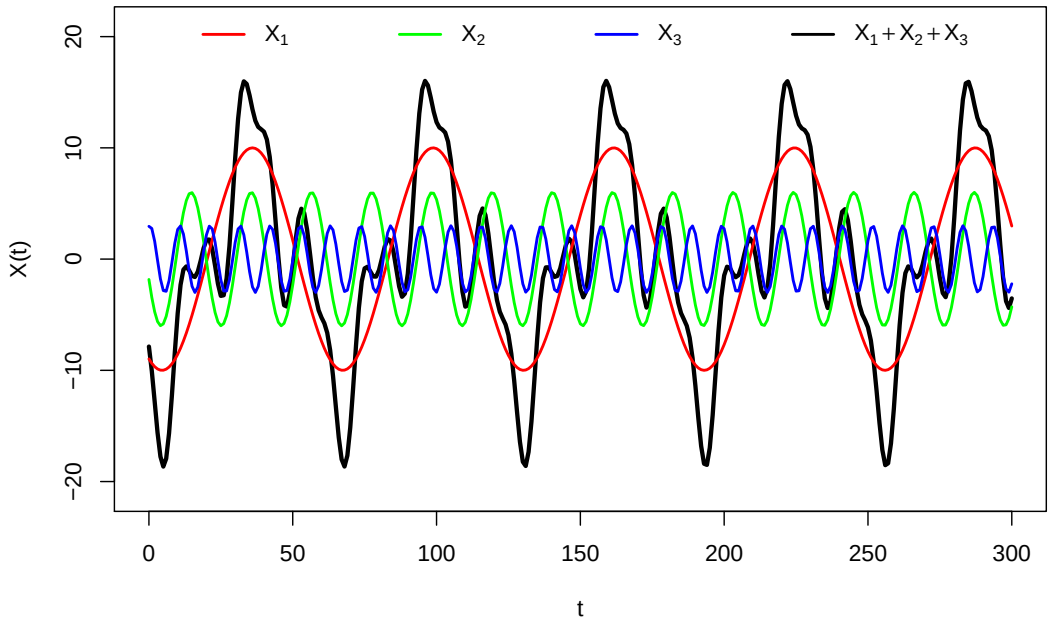


Figura 7.3: Superposición de tres armónicos

cos. El teorema de *representación espectral* prueba que un proceso estacionario $\{X(t) : t \in \mathbb{Z}\}$, bajo determinadas condiciones satisface:

$$X(t) = \int_{-\pi}^{\pi} A(\omega) \cos \omega t \cdot dZ(\omega)$$

donde $Z(\omega)$ es un proceso estocástico de una cierta clase. La integral respecto del proceso $Z(\omega)$ se entiende en el sentido de *Stieltjes-Lebesgue* (el lector puede obviar estos detalles). Nótese que cada función $A(\omega)$ se corresponde con un armónico de frecuencia ω, siendo $\omega \in [-\pi, \pi]$. Como ejemplo podemos citar la *luz blanca,* que es una superposición *no numerable de armónicos* que barren todas las frecuencias del intervalo $[-\pi, \pi]$, siendo $A(\omega)$ una función constante. Nótese también que los procesos armónicos constituyen un caso particular de la representación espectral (las integrales se reducen a sumas finitas).

El *espectro* o *distribución espectral* es una medida similar a la de masa o probabilidad, que asigna a cada conjunto de frecuencias la contribución de los correspon-

dientes armónicos a la varianza del proceso. Para el proceso armónico definido en 7.2.4, la distribución espectral asigna a la frecuencia ω_i la cantidad $A_i^2/2$.

Consideremos ahora un proceso estacionario $\{X(t) \ : \ t \in \mathbb{Z}\}$ tal que: $\sum_\tau |R(\tau)| < \infty$. En tal caso, la distribución espectral es absolutamente continua (tiene densidad espectral), la cual es precisamente $f(\omega) = A(\omega)^2$. Ésta puede obtenerse a partir de la función de autocorrelación en la forma:

$$f(\omega) = \frac{1}{2\pi} \sum_{\tau=-\infty}^{\infty} R(\tau) \cdot \exp(-i\omega\tau) \ ; \quad -\pi \leqslant \omega \leqslant \pi \tag{7.5}$$

Veamos en primer lugar que la serie es convergente. En efecto:

$$| f(\omega) | \leq \frac{1}{2\pi} \sum_{\tau=-\infty}^{\infty} | R(\tau) \cdot \exp(-i\omega\tau) | = \frac{1}{2\pi} \sum_{\tau=-\infty}^{\infty} | R(\tau) | < \infty \tag{7.6}$$

Veamos ahora que la función tiene su recorrido en $\mathbb{R}$ ($f(\omega) \in \mathbb{R}$). Para ello téngase en cuenta que $R(-\tau) = R(\tau)$. Entonces:

$$f(\omega) = \frac{1}{2\pi} \left(R(0) + \sum_{\tau=1}^{\infty} (R(\tau) \cdot \exp(-i\omega\tau) + R(-\tau) \cdot \exp(i\omega\tau)) \right) =$$

$$\frac{1}{2\pi} \left(R(0) + 2 \sum_{\tau=1}^{\infty} R(\tau) \cdot \cos \omega\tau \right)$$

Finalmente se tiene que, para cualquier $\tau \in \mathbb{Z}$ y $\omega \in [-\pi, \pi]$, $\cos \omega\tau \geq 0$, lo que implica que $f(\omega) \geq 0$.

A partir de la función de densidad espectral del proceso puede calcularse la función de autocovarianza mediante la siguiente fórmula de inversión:

$$R(\tau) = \int_{-\pi}^{\pi} \exp(i\omega\tau) f(\omega) \, d\omega \ , \quad \tau \in Z \tag{7.7}$$

Obsérvese ahora que:

$$R(0) = \sigma_X^2 = \int_{-\pi}^{\pi} f(\omega)\, d\omega$$

Estos resultados nos permiten hacer la siguiente consideración: La contribución de las frecuencias de un intervalo $[\omega_1, \omega_2]$ (*banda de frecuencias*) a la varianza del proceso (*potencia de la señal*) puede obtenerse por:

$$\int_{\omega_1}^{\omega_2} f(\omega)\, d\omega$$

Dado que el proceso (en tiempo discreto) es sólo superposición de frecuencias de $[-\pi, \pi]$, la contribución total ($\int_{-\pi}^{\pi} f(\omega)\, d\omega$) es la varianza del proceso σ_X^2, tal como confirma el resultado último.

7.4. Espectros especiales

7.4.1. Ruido blanco.

Para un proceso ruido blanco en tiempo discreto, la función de densidad espectral es:

$$f(\omega) = \frac{1}{2\pi} \sum_{\tau} R(\tau) \cdot \exp(-i\omega\tau) = \frac{\sigma_X^2}{2\pi} \; ; \quad -\pi \leqslant \omega \leqslant \pi \tag{7.8}$$

Obsérvese que esta función es constante sobre el intervalo $[-\pi, \pi]$, lo que significa que es una distribución uniforme. El proceso de ruido blanco se caracteriza pues por el hecho de que todos los armónicos que determinan el proceso tienen una contribución uniforme a la varianza de dicho proceso. Esto es exactamente lo que ocurre con la luz blanca, de ahí la denominación de este proceso.

7.4.2. Proceso autorregresivo de orden 1 ($AR(1)$).

Sea el proceso estacionario autorregresivo $X(t) = aX(t-1) + \varepsilon(t)$. Dado que su función de autocovarianza es $R(\tau) = a^{|\tau|} \cdot \sigma_X^2 \; ; \; \tau \in Z$, su función de densidad espectral es:

$$f(\omega) = \frac{\sigma_X^2}{2\pi} \sum_{\tau=-\infty}^{\infty} \exp(-i\omega\tau)\, a^{|\tau|} =$$

$$\frac{\sigma_X^2}{2\pi}\left(\sum_{\tau=0}^{\infty} (a\exp(-i\omega))^{\tau} + \sum_{\tau=1}^{\infty} (a\exp(i\omega))^{\tau}\right) = \frac{\sigma_X^2(1-a^2)}{2\pi(1-2a\cos\omega+a^2)} \qquad (7.9)$$

La figura 7.4 muestra la función de densidad espectral del proceso $AR(1)$, una de cuyas trayectorias se muestra en la figura 7.1. Obsérvese como la mayor contribución al espectro procede de las frecuencias más próximas a cero.

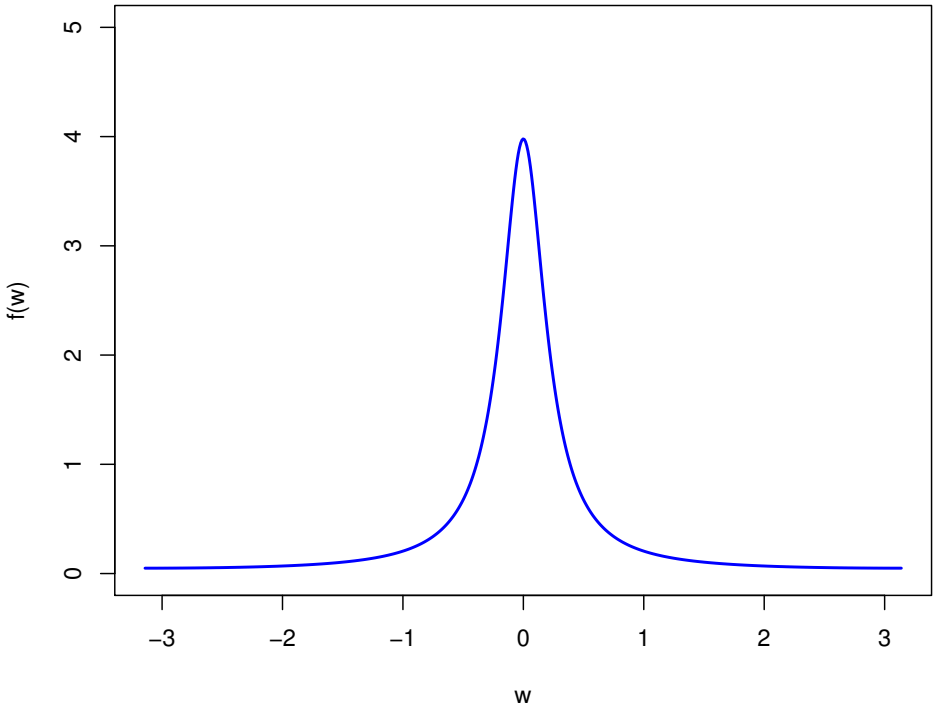


Figura 7.4: Densidad espectral de un proceso AR(1)

7.4.3. Proceso de medias móviles de orden q ($MA(q)$).

Para el proceso $X(t) = b_0\varepsilon(t) + b_1\varepsilon(t-1) + \ \ldots \ + b_q\varepsilon(t-q)$, la función de densidad espectral es:

$$f(\omega) = \frac{\sigma_\varepsilon^2}{2\pi} \cdot \left| \sum_{j=0}^{q} b_j \exp(-i\omega j) \right|^2 \; ; \quad -\pi \leqslant \omega \leqslant \pi \tag{7.10}$$

7.5. Transformaciones lineales y filtros

Una de las principales razones del amplio uso del análisis espectral radica en que el espectro de un proceso explica de una forma sencilla los efectos de las transformaciones lineales del proceso. La idea física de filtro se corresponde con un dispositivo o sistema que recibe una señal $\{X(t)\,;\, t \in T\}$, la transforma y produce una salida $\{Y(t)\,; t \in T\}$.

$$X(t) \rightarrow \boxed{\text{SISTEMA}} \rightarrow Y(t)$$

Supongamos que ambas señales se recogen en tiempo discreto (digitalizadas) y por tanto consideraremos $T = \mathbb{Z}$. Un filtro lineal tiene entonces la forma:

$$Y(t) = \sum_{u=-\infty}^{\infty} g_u \cdot X(t-u) \tag{7.11}$$

donde $\sum\limits_{u=-\infty}^{\infty} |g_u| < \infty$. El filtro se dice que es físicamente realizable si $g_u = 0,\ u < 0$. Obsérvese que la señal de salida en un instante t es una superposición de la señal de entrada en t y todos los instantes anteriores, ponderada por la sucesión de coeficientes $\{g_u; u \geqslant 0\}$.

Si el proceso $\{X\{t\} : t \in T\}$, que se corresponde con la señal de entrada es estacionario con función de autocovarianza $R_X(\tau)$, el proceso $\{Y(t)\,; t \in T\}$ que se corresponde con la señal de salida es también estacionario, y su función de autocovarianza es:

$$R_Y(\tau) = \sum_{u=-\infty}^{\infty} \sum_{v=-\infty}^{\infty} g_u g_v R_X(\tau + u - v) \tag{7.12}$$

La demostración de este resultado se deja al lector.

Es fácil probar que $\sum_\tau |R_Y(\tau)| < \infty$, lo que supone que el proceso $\{Y(t) : t \in T\}$ tiene densidad espectral. Mediante cálculos directos puede además obtenerse que:

$$f_Y(\omega) = f_X(\omega) \cdot |\Gamma(\omega)|^2 \tag{7.13}$$

siendo:

$$\Gamma(\omega) = \sum_u g_u \cdot \exp(-i\omega u) \tag{7.14}$$

Esta última función recibe el nombre de *función de transferencia*.

Estudiamos como aplicación los llamados *filtros de paso de banda*. Para ello, consideremos una señal modelizada a través de un proceso estacionario $\{X(t); t \in \mathbf{Z}\}$ cuya densidad espectral es $f_X(\omega)$. Esta señal se va a filtrar de tal forma que la densidad espectral $f_Y(\omega)$, correspondiente al proceso de salida $\{Y(t); t \in T\}$ concentre toda su varianza sobre los armónicos pertenecientes al intervalo $[\omega_1, \omega_2]$. Para tal fin, consideramos la siguiente función de transferencia:

$$\Gamma(\omega) = \begin{cases} 1 & : \ \omega_1 \leq |\omega| \leq \omega_2 \\ 0 & : \ en\ otro\ caso \end{cases}$$

Obtenemos entonces los coeficientes g_u invirtiendo (7.14) y se tiene entonces:

$$g_u = \frac{1}{2\pi} \int_{-\infty}^{\infty} \Gamma(\omega) \exp(i\omega u)\, d\omega = \frac{1}{2\pi} \int_{\omega_1}^{\omega_2} \exp(i\omega u)\, d\omega + \frac{1}{2\pi} \int_{-\omega_2}^{-\omega_1} \exp(i\omega u)\, d\omega = \tag{7.15}$$

$$\frac{\exp(i\omega_2 u) - \exp(i\omega_1 u)}{2\pi i u} + \frac{\exp(-i\omega_1 u) - \exp(-i\omega_2 u)}{2\pi i u} = \frac{\sin \omega_2 u - \sin \omega_1 u}{\pi u}$$

Finalmente, a partir de esta expresión se obtiene la señal filtrada $\{Y(t); t \in \mathbb{Z}\}$ utilizando (7.11). El siguiente código R muestra la simulación de un proceso $X(t)$ autoregresivo de orden 1, y la aplicación de un filtro de paso de banda a este proceso para obtener un nuevo proceso Y que concentra su varianza en los armónicos del intervalo $[\pi/12; \pi/6]$. Nótese que los valores de u van de 0 a ∞; por tanto, en

principio, para generar la salida Y necesitaríamos generar infinitos valores de la entrada $X(t)$. Obviamente ello no es posible computacionalmente; no obstante es fácil comprobar que $g_0 = (\omega_2 - \omega_1)/\pi$ y que a medida que u crece el valor de g_u se va aproximando a 0 (véase figura 7.5), lo que significa que a partir de cierto valor en adelante los productos $g_u X(t-u)$ toman valores tan pequeños que pueden desecharse. En esta simulación para obtener $Y(t)$ utilizaremos valores de u desde 0 hasta 10000 (lo que implica que necesitamos simular 10000 valores previos de X). La figura 7.6 muestra la serie original $X(t)$, así como la serie filtrada $Y(t)$ observadas a partir de $t = 10000$.

```
longitudProceso=5000
longitudFiltro=1000
X=procesoAR1(a=0.8, se=1, n=longitudProceso)
u=1:longitudFiltro
w2=pi/6
w1=pi/12
g0=(w2-w1)/pi
g1=(sin(w2*u)-sin(w1*u))/(pi*u)
g=c(g0,g1)
Y=filter(X,g,side=2)
```

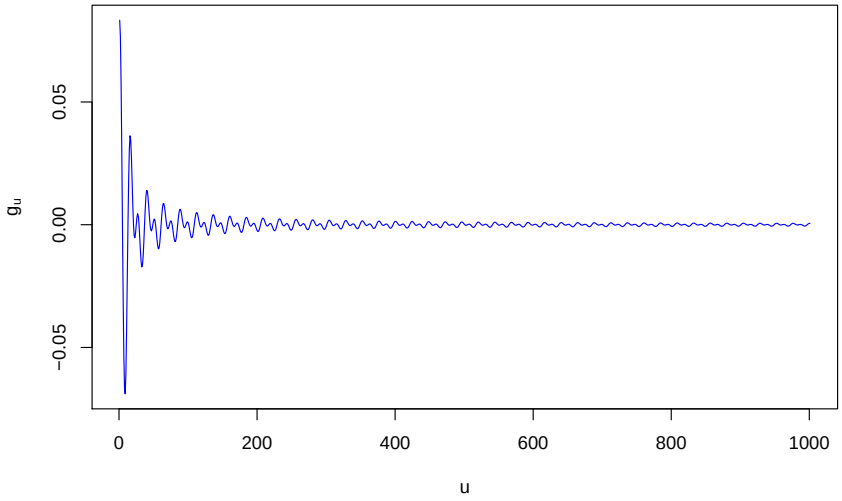


Figura 7.5: Coeficientes del filtro de paso de banda.

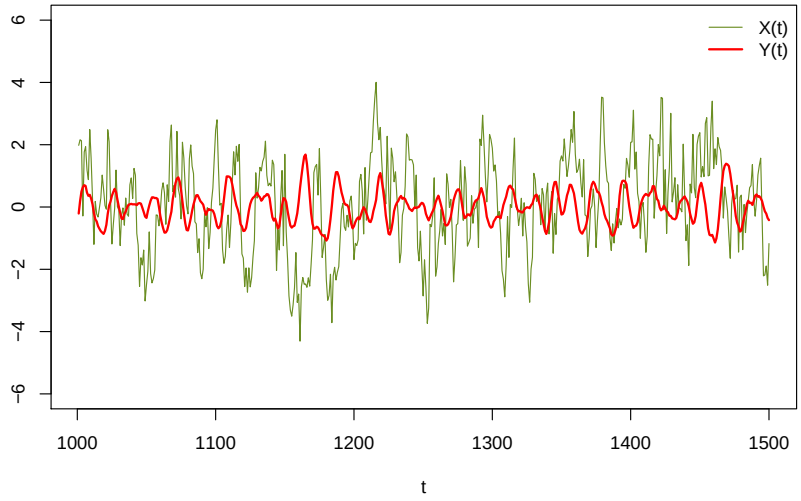


Figura 7.6: Señales original ($X(t)$) y filtrada ($Y(t)$) mediante un fitro paso de banda.

```
## Error: valor ausente donde TRUE/FALSE es necesario
## Error: fit is not an lspec object
## Error: objeto 'fd' no encontrado
```

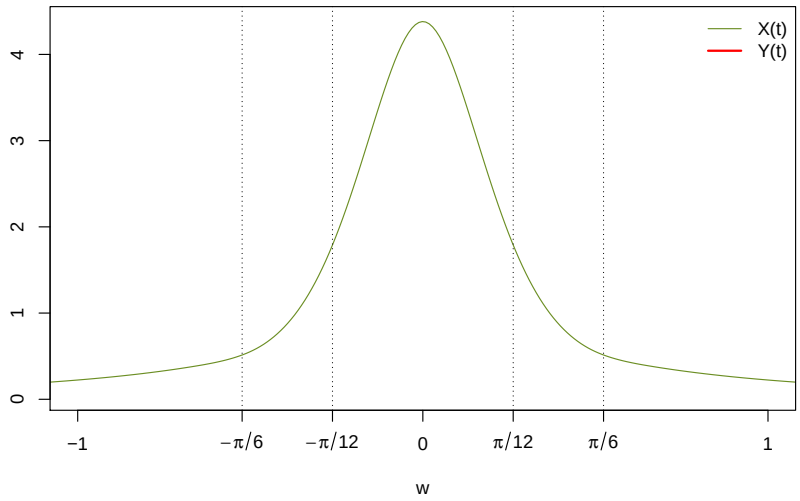


Figura 7.7: Espectros de frecuencias de las señales $X(t)$ e $Y(t)$, esta última filtrada mediante un filtro paso de banda que sólo permite el paso de las frecuencias entre $\frac{\pi}{12}$ y $\frac{\pi}{6}$

Problemas

1. Sea $\{X(t)\ ; t \in \mathbf{Z}\}$ un proceso estacionario de medias móviles de orden 2 (MA(2)), definido por $X(t) = g_0\varepsilon(t) + g_1\varepsilon(t-1) + g_2\varepsilon(t-2)$, siendo $\{\varepsilon(t)\,;\ t \in \mathbf{Z}\}$ un ruido blanco tal que $E[\varepsilon(t)] = 0$ y $[\varepsilon(t)] = \sigma_\varepsilon^2$.

   a) Hallar las funciones de autocovarianza y autocorrelación del proceso $\{X(t)\ ; t \in \mathbf{Z}\}$

   b) Calcular la función de densidad espectral.

2. Consideremos un proceso estacionario $\{X_t;\ t \in \mathbf{Z}\}$ con función de autocovarianza $R(\tau) = 0$ si $\tau \neq 0$ y $R(0) = \sigma^2$

   a) Hallar su función de densidad espectral.

b) Calcular la contribución a la varianza del proceso de las frecuencias mayores de $\pi/2$.

3. Probar (7.12).

4. Probar (7.13).

5. Obtener la expresión del filtro de paso bajo en el que la función de transferencia tiene la forma $\Gamma(\omega) = I_{[0,\omega_0]}(\omega)$.

6. Sea el proceso estacionario AR(1) $X(t) = \beta \cdot X(t-1) + \varepsilon(t)$ donde $\{\varepsilon(t);\ t \in \mathbf{Z}\}$ es un ruido blanco tal que $E[\varepsilon(t)] = 0$ y $var(\varepsilon(t)) = \sigma_\varepsilon^2$, $\forall t \in \mathbb{Z}$.

   a) Calcular su función de autocovarianza.

   b) Hallar su función de densidad espectral.

   c) Sea el filtro lineal $Y(t) = \sum_{u=0}^{\infty} g_u \cdot X(t-u)$. Determinar la sucesión $\{g_u;\ u \in \mathbf{Z}\}$ *h.* de tal forma que la distribución espectral de $\{Y(t);\ t \in \mathbf{Z}\}$ *k.* l. se concentre en $[-\omega_0, \omega_0]$

   d) Hallar la densidad espectral del proceso $\{Y(t);\ t \in \mathbf{Z}\}$

   e) Calcular la función de autocovarianza de $\{Y(t);\ t \in \mathbf{Z}\}$

Apéndice A

El Entorno Estadístico R

A.1. ¿Qué es R?

Podríamos definir R como un *entorno computacional* de código abierto orientado al análisis estadístico de datos. En la web de este programa (www.r-project.org) puede leerse que R incluye :

- Herramientas efectivas para la gestión y almacenamiento de datos.
- Un conjunto de funciones y operadores preparados para actuar directamente sobre vectores o matrices.
- Una gran colección, coherente e integrada, de herramientas para el análisis de datos.
- Procedimientos gráficos para el análisis y visualización de datos, ya sea en pantalla o en papel.

- Un completo y bien desarrollado lenguaje de programación, simple y eficaz, que incluye gran cantidad de funciones predefinidas, instrucciones condicionales, bucles, funciones recursivas definidas por el usuario y procedimientos eficientes de entrada y salida.

R está diseñado en torno a un lenguaje de programación real, y permite a los usuarios añadir funcionalidad adicional mediante la definición de nuevas funciones. Gran parte del sistema está en sí mismo escrito en el lenguaje R, lo que facilita a los usuarios el seguimiento y desarrollo del código. Para las tareas de cómputo intensivo, R puede enlazar en tiempo de ejecución con código C, C + + y Fortran. Los usuarios avanzados pueden incluso escribir código C para manipular objetos R directamente.

Muchos usuarios piensan en R como un paquete estadístico. No obstante, sus diseñadores lo presentan más bien como *un entorno en el que se aplican* los métodos estadísticos. En particular presenta numerosas herramientas orientadas a la simulación, lo que permite obtener resultados incluso en aquellos casos en los que no se dispone de un modelo matemático adecuado para la descripción o resolución del problema en estudio.

Las capacidades de R pueden ampliarse fácilmente a través de la incorporación de *paquetes* o *librerías*. En el momento de la redacción de estas notas (enero 2011) R cuenta con 2782 paquetes disponibles, que cubren una gama muy amplia de la estadística moderna. En su mayor parte, han sido los propios usuarios de R los que, utilizando las capacidades de programación que éste ofrece, han desarrollado estas librerías que incluyen algoritmos avanzados para el análisis de datos, el desarrollo de nuevas técnicas de data mining, o la realización de gráficos de alta calidad -desde los gráficos estadísticos más simples, al tratamiento de mapas o de imágenes de resonancia magnética- demandados por publicaciones científicas, por la empresa o por la administración.

Sin duda, el hecho de que R sea software libre, que puede ser utilizado por el usuario sin coste alguno, unido a la facilidad de programación, han sido las claves para que R se halla convertido en la actualidad en el estándar *de facto* para el desarrollo e intercambio de nuevo software estadístico en la comunidad científico-técnica.

A.2. Ventajas e inconvenientes del uso de R.

Ventajas:

- Es software libre y por tanto su coste es nulo.
- Es multiplataforma: existen versiones para Linux, Mac y Windows. Los procedimientos y análisis desarrollados en una plataforma son inmediatamente ejecutables en otra.
- Implementa una enorme cantidad de métodos estadísticos, desde los más clásicos a los más modernos. Los métodos se organizan en librerías cuyo número se encuentra en constante crecimiento.
- Dispone de una enorme capacidad para combinar, de manera simple, métodos de análisis estándar (regresión, análisis cluster, análisis de series temporales) con análisis desarrollados ad hoc para una situación específica.
- Capacidad para acceder a datos en múltiples formatos. Dispone de librerías para leer datos desde SPSS, SAS, Access, MySQL, Excel,... Asimismo permite también la generación de informes de resultados en diversos formatos.
- Enorme capacidad para manipular y /o modificar datos y funciones.
- Generación de gráficos de alta calidad.
- Existencia de una comunidad de usuarios muy activa, en la que participan estadísticos de renombre.
- Amplia disponibilidad de documentación, tanto en internet como en libros publicados por editoriales de prestigio (Springer, Wiley).
- Facilidad de integración con actividades de formación en técnicas y métodos estadísticos en todos los ámbitos del conocimiento. Su uso es cada vez más generalizado en las universidades, lo que implica que las nuevas generaciones de profesionales ya salen al mercado laboral con formación específica en el manejo de este programa.
- Existencia de extensiones específicas para nuevas áreas como bioinformática, geoestadística, modelos gráficos o análisis de mercados financieros.

- Todos los algoritmos implementados en R pueden ser vistos e interpretados por cualquier usuario, por lo que éste puede saber exactamente qué es lo que hace el ordenador cuando ejecuta un comando.

Inconvenientes:

- Suele señalarse como principal desventaja de R el hecho de que el paquete base no dispone de una interfaz amigable para el usuario; no existe un menú principal en el que el usuario pueda acceder mediante el ratón a submenús para la lectura de datos, la ejecución de procedimientos estadísticos o la generación de gráficos. Estas tareas se llevan a cabo mediante un lenguaje de comandos que puede resultar duro de aprender para el usuario común. No obstante se han desarrollado algunas GUIs (Graphical User Interfaces) que facilitan enormemente esta tarea. En particular la interfaz `R-Commander` desarrollada por John M. Fox en la McMaster University de Canadá presenta un menú para el acceso a los comandos más habituales que, además, muestra el código que emplea R de forma que permite al usuario familiarizarse con el lenguaje.

- El código R es *interpretado,* no *compilado*, por lo que algún algoritmo puede resultar de ejecución lenta, en particular si se realizan tareas de simulación intensiva. Esto no constituye mayor problema para un uso ordinario del programa.

- No dispone de un sistema de base de datos propio, aunque sí cuenta con un formato para el almacenamiento e intercambio de datos. En cualquier caso se han desarrollado paquetes para conectar y acceder a múltiples sistemas de bases de datos (las propias de SAS o SPSS, Access, dBase, Excel, MySQL, ...). Tiene algunas limitaciones en cuanto al uso de la memoria, que dificultan el análisis de bases de datos masivas. No obstante estas limitaciones han ido desapareciendo a medida que se han ido desarrollando ordenadores con mayor capacidad (procesadores de 64 bits, más disponibilidad de memoria y de direccionamiento de la misma). En cualquier caso, salvo que el usuario deba acceder a millones de registros simultáneamente, es difícil que llegue a notar problemas con la memoria.

- En algún caso las nuevas librerías que se incorporan a R pueden tener errores o fallos de implementación. Estos fallos, no obstante, suelen ser detectados por los usuarios, informados a los desarrolladores de las librerías y corregidos en tiempo récord. Debe señalarse, no obstante, que ningún programa (incluso los comerciales) está exento de fallos y el proceso de revisión y corrección de fallos en programas comerciales mediante parches o actualizaciones suele ser notablemente más lento.

- No hay nadie a quien reclamar si algo falla, ni hay un departamento de atención al cliente que nos diga qué podemos hacer si algo va mal, si algún procedimiento nos da un error, o simplemente si no sabemos qué sintaxis utilizar. Pero a cambio existe una comunidad de usuarios organizada en foros y dispuesta a colaborar desinteresadamente en la resolución de problemas.

A.3. Fuentes de información sobre R.

En los últimos años se han publicado cientos de libros de estadística que utilizan R como soporte informático para la puesta en práctica de los conocimientos adquiridos. Algunos de esos libros están disponibles para su descarga en la web. Asimismo múltiples universidades y centros de enseñanza imparten cursos de R y tiene su material también disponible en la web. Como sitios interesantes, la propia página web de R (`www.r-project.org`), en el enlace `Manuals` situado a la izquierda, y a partir de ahí en el enlace `contributed documentation`) dispone de una sección de documentación muy amplia en la que podemos encontrar los siguientes manuales en castellano:

- *"R para Principiantes"* (versión española de "*R for Beginners*", traducido por Jorge A. Ahumada)

- *"Introducción a R"* (versión española de *"An Introduction to R"* por Andrés González and Silvia González)

- *"Gráficos Estadísticos con R"* por Juan Carlos Correa and Nelfi González.

- *"Cartas sobre Estadística de la Revista Argentina de Bioingeniería"* por Marcelo R. Risk.

- *"Introducción al uso y programación del sistema estadístico R"* por Ramón Díaz-Uriarte (transparencias preparadas para un curso de 16 horas sobre R, dirigido principalmente a biólogos y bioinformáticos).
- *"Metodos Estadisticos con R y R Commander"* por Antonio Jose Saez Castillo.

Fuera de la web de R es posible encontrar manuales en español en muchos lugares. Por ejemplo:

- `http://knuth.uca.es/moodle/course/view.php?id=37`: Libro "libre": "*Estadística Básica con R y R-commander*" de A. J. Arriaza Gómez y otros profesores de la Universidad de Cádiz.
- `eio.usc.es/pub/pateiro/files/PubDocentePracticasEstadistica.pdf`: manual de estadística con R (muy orientado a la programación) de Manuel Febrero y otros, de la Universidad de Santiago de Compostela.

También en castellano, las siguientes webs resultan de bastante interés para la formación en R:

- `http://personales.unican.es/gonzaleof/R/index.html`: página web de Francisco J. González, profesor de la Universidad de Cantabria.
- `http://www.ub.edu/stat/docencia/EADB/Curso%20basico%20de%20R.htm`: curso básico de R, ofrecido por el profesor F. Carmona de la Universidad de Barcelona.
- `http://ocw.uc3m.es/estadistica/aprendizaje-del-software-estadis-tico-r-un-ent` curso *on-line* de R ofrecido por el profesor Alberto Muñoz de la Universidad Carlos III de Madrid.

A.4. Descarga e instalación del programa

El procedimiento de instalación de R en Windows es muy simple[1]. Basta acceder a la web `www.r-project.org` (ver figura A.1) y en el recuadro central (`Getting`

[1]En esta sección nos referiremos exclusivamente a la instalación en Windows, por ser el sistema operativo más usado (por el momento). No obstante, la instalación para Mac es prác-

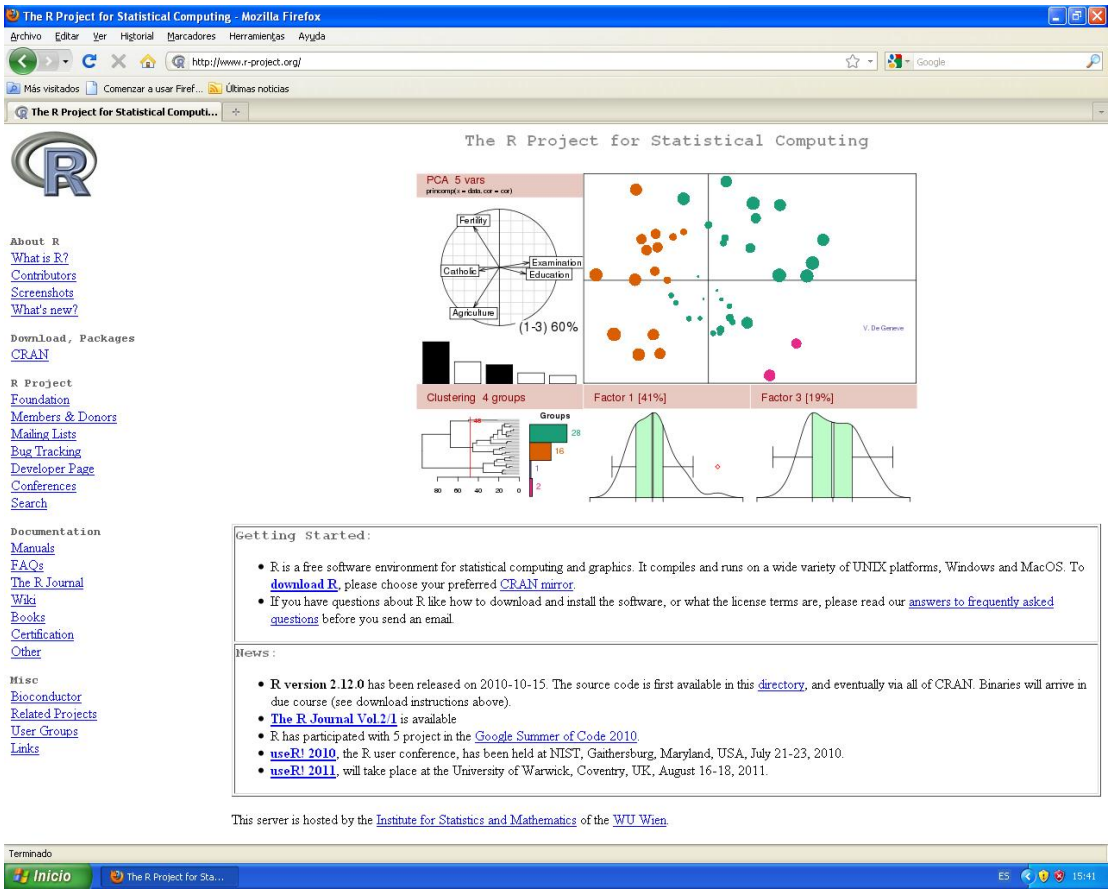


Figura A.1: Pantalla principal de la página web de R-project

`started`) picar en el enlace `download R`. Este enlace nos conducirá a una lista de repositorios distribuidos por todo el mundo desde los que es posible descargar el programa. En nuestro caso podemos elegir el repositorio situado en España. A partir de ahí le indicaremos que deseamos descargar la versión para `Windows`, a continuación seleccionamos el paquete `base` y por último seleccionamos `Download R`. La web nos preguntará si deseamos guardar el archivo `R-xx.yy.z-win.exe` (donde *xx.yy.z* es el número de la última versión disponible de R, la 2.12.1 en el momento de escribir estas líneas). Elegimos la opción `Guardar archivo` y una vez que haya terminado de descargarse lo ejecutamos. Arranca entonces el *Asistente de instalación* que nos hará las preguntas típicas relativas al idioma de instalación,

ticamente idéntica (en este caso nos descargaríamos un archivo.pkg). Para Linux bastará con añadir la dirección de algún *mirror* de R (la lista de mirrors está en la web de R) a la lista de repositorios que utiliza el sistema y proceder a instalar el programa mediante los comandos *sudo apt-get update* y *sudo apt-get install r-base* (en Ubuntu se puede usar también *synaptic* o el nuevo *Centro de Software*).

directorio donde guardar el programa, etc. Podemos elegir todas las opciones que nos ofrece por defecto (pinchando en Siguiente> en las sucesivas ventanas que van apareciendo). Si todo ha ido bien al finalizar el proceso tendremos el icono de R en el escritorio de Windows. Al picar dos veces sobre este icono se iniciará el programa tal como se muestra en la figura A.2.

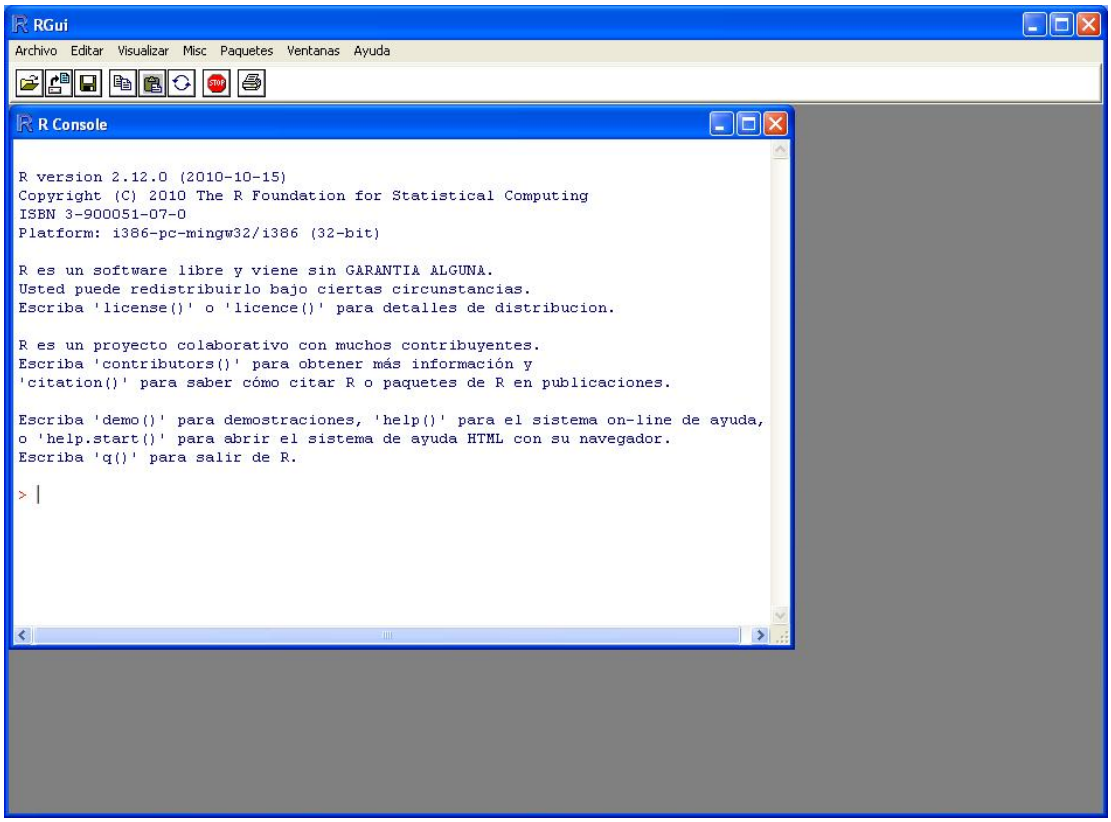


Figura A.2: Entorno R por defecto en Windows

Este entorno dista mucho de los habituales en otros programas estadísticos. No vemos iconos ni múltiples botones que nos permitan hacer cosas. Si recorremos con el ratón los items del menú de la parte superior de la ventana, veremos como se nos van desplegando diversos submenús, en ninguno de los cuales encontramos opciones específicamente orientadas al análisis de datos. ¿Cómo funciona R entonces? ¿Cómo leemos nuestros datos, los procesamos, obtenemos resultados, imprimimos informes, gráficos,...?

La respuesta es que en realidad para hacer todas estas cosas R dispone de su propio lenguaje. Este lenguaje es el que dota a R de la tremenda versatilidad y potencia que posee. Pero también es lo que asusta y aleja a muchos usuarios potenciales

del programa, que ven la sintaxis de este lenguaje como un obstáculo insalvable. Sin embargo, al igual que sucede con el aprendizaje de cualquier idioma, una vez que se conocen algunas palabras y las reglas gramaticales básicas, ya es posible mantener conversaciones sencillas. A partir de ahí, con la práctica y un poquito de empeño poco a poco podremos ir desenvolviéndonos cada vez con mayor soltura.

Para hacernos una idea del lenguaje de R, comencemos nuestra primera sesión con el programa.

A.5. Primera sesión con R.

Se ha seleccionado una muestra aleatoria de diez personas. A cada una de ellas se le ha preguntado su edad y se le ha medido el número de minutos que ha utilizado el móvil durante el miércoles de la semana en que se realizó el estudio. Los resultados se muestran en la tabla A.1.

Edad	22	34	29	25	30	33	31	27	25	25
Tiempo	14.21	10.36	11.89	13.81	12.03	10.99	12.48	13.37	12.29	11.92
Sexo	M	H	H	M	M	H	M	M	H	H

Tabla A.1: Tiempo diario de uso del móvil en una muestra de 10 personas

Vamos a utilizar R para hacer una estadística descriptiva sencilla de estas variables. Una vez arrancado el programa (figura A.2) lo primero que hemos de hacer es introducir los datos. Para ello usamos la función *concatenar,* `c()`, excribiendo lo siguiente en nuestra ventana de R:

```
edad <- c(22, 34, 29, 25, 30, 33, 31, 27, 25, 25)
tiempo <- c(14.21, 10.36, 11.89, 13.81, 12.03, 10.99, 12.48, 13.37, 12.29, 11.92)
sexo <- c("M", "H", "H", "M", "M", "H", "M", "M", "H", "H")
```

Calculemos las medias de la edad y del tiempo de uso del móvil, y contemos cuantos hombres y mujeres tenemos en la muestra:

```
mean(edad)

## [1] 28.1
```

```
mean(tiempo)

## [1] 12.34

table(sexo)

## sexo
## H M
## 5 5
```

Dibujemos ahora un histograma del tiempo de uso del móvil y un diagrama de barras de la distribución de sexos:

```
par(mfrow = c(1, 2))
hist(tiempo, main = "Tiempo de uso del movil", xlab = "Tiempo (min.)")
barplot(table(sexo), main = "Distribuci<U+00F3>n de sexos")
```

El comando `par(mfrow=c(1,2))` tiene como única función configurar la salida de gráficos en 1 fila y 2 columnas, de tal modo que nos muestra los dos gráficos solicitados uno junto a otro. La figura A.3 muestra el resultado de estos comandos.

Por último calculamos la recta de regresión para predecir el tiempo de uso del móvil en función de la edad del sujeto, y la representamos gráficamente (figura A.4):

```
regresion <- lm(tiempo ~ edad)
regresion

##
## Call:
## lm(formula = tiempo ~ edad)
##
## Coefficients:
## (Intercept)         edad
##      19.320       -0.249
```

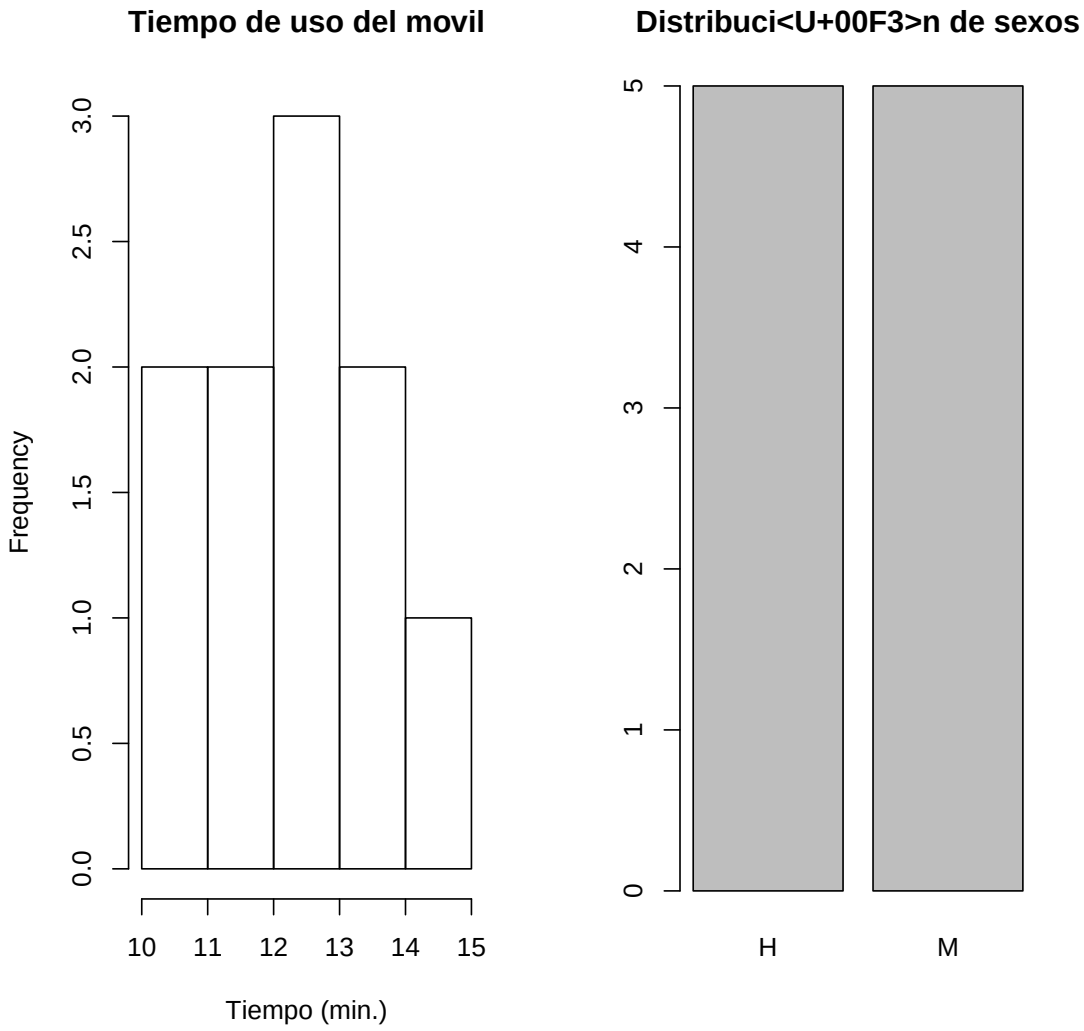


Figura A.3: Ejemplos de histograma y diagrama de barras

```
par(mfrow = c(1, 1))
plot(tiempo, edad)
abline(regresion, col = "red")
```

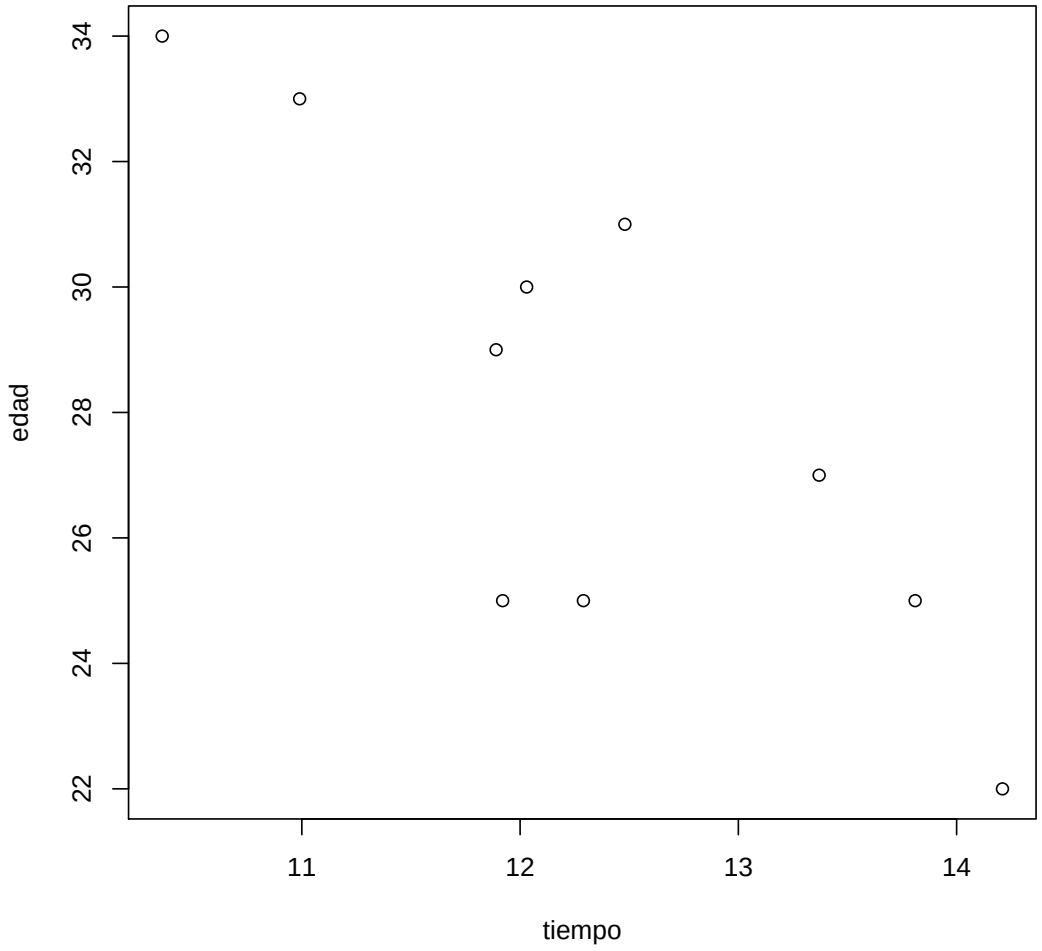


Figura A.4: Gráfico del tiempo de uso del móvil frente a la edad del usuario para los datos de la tabla A.1

A.6. Elementos básicos de R.

Observemos los comandos (instrucciones) que hemos empleado en esta primera sesión con R. Para R todo lo que se maneja en su lenguaje son *objetos*. Los objetos pueden ser de muchas *clases* diferentes. Así, en esta sesión hemos utilizado:

- **Vectores**: un vector es una colección de valores observados en una variable. En nuestro ejemplo, `tiempo` es un vector de *longitud* 10, que contiene los valores `(14.21, 10.36, 11.89, 13.81, 12.03, 10.99, 12.48,`

`13.37, 12.29, 11.92)`. Del mismo modo, `sexo` es también un vector de longitud 10 con los valores `("M", "H", "H", "M", "M", "H", "M", "M", "H", "H")`. El vector `tiempo` es numérico (nótese de paso que el separador decimal en R es el punto), y `sexo` es *alfanumérico* (sus valores pueden ser letras). Los valores de las variables alfanuméricas deben ir entrecomillados.

- **Funciones:** una función puede entenderse como un algoritmo o programa encargado de realizar alguna tarea sobre los valores que se le pasen como *argumento*. En nuestro ejemplo hemos usado las funciones:

  - `c()`: función *concatenación*. Se encarga de agrupar en un único vector todos los valores que figuran en el paréntesis.
  - `mean()`: calcula la media de los valores que se le pasen como argumento. Si su argumento es un vector, calcula la media de todos los valores que forman el vector (tal como ha hecho en nuestro ejemplo, calculando las medias de `edad` y `tiempo`.
  - `table()`: calcula la tabla de frecuencias absolutas (número de veces que se observa cada valor) de la variable que se encuentra en el paréntesis.
  - `hist()`: Dibuja un histograma de la variable que se le pase como argumento.
  - `barplot()`: Dibuja un diagrama de barras correspondiente a la *tabla* que se le pasa como argumento.
  - `lm(y ~ x)`: calcula la recta de regresión de `y` sobre `x`.
  - `plot(x,y)`: dibuja la nube de puntos `(x,y)`.
  - `abline()`: dibuja la recta que se especifique en el paréntesis.

- **Operadores:** El símbolo "`<-`" es el operador de asignación. El comando `edad <- c(22, 34,..., 25)` *asigna* al vector `edad` los valores concatenados por la función `c()`. Asimismo el comando `regresion <- lm(tiempo ~ edad)` *asigna* al objeto `regresion` el resultado de calcular la recta de regresión del `tiempo` en función de la `edad`

- **Listas:** una *lista* en R es una agrupación de vectores de distinta longitud y tipo. En nuestro ejemplo, `regresion` es una lista que contiene prácticamente

todo el análisis estadístico de la regresión. Si bien antes cuando le pedimos a R que nos mostrase el valor de `regresion` se limitó a mostrarnos los valores de la pendiente y la ordenada en el origen, en realidad `regresion` contiene muchas más cosas. Si utilizamos la función `str()` podemos ver la estructura detallada de este objeto:

```
str(regresion)

## List of 12
##  $ coefficients : Named num [1:2] 19.32 -0.249
##   ..- attr(*, "names")= chr [1:2] "(Intercept)" "edad"
##  $ residuals    : Named num [1:10] 0.359 -0.508 -0.221 0.704 0.167 ...
##   ..- attr(*, "names")= chr [1:10] "1" "2" "3" "4" ...
##  $ effects      : Named num [1:10] -39.007 2.929 -0.234 0.568 0.186 ...
##   ..- attr(*, "names")= chr [1:10] "(Intercept)" "edad" "" "" ...
##  $ rank         : int 2
##  $ fitted.values: Named num [1:10] 13.9 10.9 12.1 13.1 11.9 ...
##   ..- attr(*, "names")= chr [1:10] "1" "2" "3" "4" ...
##  $ assign       : int [1:2] 0 1
##  $ qr           :List of 5
##   ..$ qr   : num [1:10, 1:2] -3.162 0.316 0.316 0.316 0.316 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:10] "1" "2" "3" "4" ...
##   .. .. ..$ : chr [1:2] "(Intercept)" "edad"
##   .. ..- attr(*, "assign")= int [1:2] 0 1
##   ..$ qraux: num [1:2] 1.32 1.62
##   ..$ pivot: int [1:2] 1 2
##   ..$ tol  : num 1e-07
##   ..$ rank : int 2
##   ..- attr(*, "class")= chr "qr"
##  $ df.residual  : int 8
##  $ xlevels      : Named list()
##  $ call         : language lm(formula = tiempo ~ edad)
##  $ terms        :Classes 'terms', 'formula' length 3 tiempo ~ edad
##   .. ..- attr(*, "variables")= language list(tiempo, edad)
##   .. ..- attr(*, "factors")= int [1:2, 1] 0 1
```

```
##   .. .. ..- attr(*, "dimnames")=List of 2
##   .. .. .. ..$ : chr [1:2] "tiempo" "edad"
##   .. .. .. ..$ : chr "edad"
##   .. ..- attr(*, "term.labels")= chr "edad"
##   .. ..- attr(*, "order")= int 1
##   .. ..- attr(*, "intercept")= int 1
##   .. ..- attr(*, "response")= int 1
##   .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
##   .. ..- attr(*, "predvars")= language list(tiempo, edad)
##   .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "numeric"
##   .. .. ..- attr(*, "names")= chr [1:2] "tiempo" "edad"
##  $ model        :'data.frame': 10 obs. of  2 variables:
##   ..$ tiempo: num [1:10] 14.2 10.4 11.9 13.8 12 ...
##   ..$ edad  : num [1:10] 22 34 29 25 30 33 31 27 25 25
##   ..- attr(*, "terms")=Classes 'terms', 'formula' length 3 tiempo ~ edad
##   .. .. ..- attr(*, "variables")= language list(tiempo, edad)
##   .. .. ..- attr(*, "factors")= int [1:2, 1] 0 1
##   .. .. .. ..- attr(*, "dimnames")=List of 2
##   .. .. .. .. ..$ : chr [1:2] "tiempo" "edad"
##   .. .. .. .. ..$ : chr "edad"
##   .. .. ..- attr(*, "term.labels")= chr "edad"
##   .. .. ..- attr(*, "order")= int 1
##   .. .. ..- attr(*, "intercept")= int 1
##   .. .. ..- attr(*, "response")= int 1
##   .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
##   .. .. ..- attr(*, "predvars")= language list(tiempo, edad)
##   .. .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "numeric"
##   .. .. .. ..- attr(*, "names")= chr [1:2] "tiempo" "edad"
##  - attr(*, "class")= chr "lm"
```

No entraremos aquí a explicar qué es cada uno de los términos que componen el objeto `regresion`. Baste decir que contienen información suficiente para dibujar la recta (es lo que hemos hecho aplicando al objeto `regresion` la función `abline`), hacer predicciones, estimar intervalos de confianza para los coeficientes, etc. R cuenta con funciones encargadas de extraer dicha información en un formato mucho más amigable. A modo de ejemplo podemos citar las funciones:

- `summary()`: nos presenta un resumen de la inferencia para la regresión.
- `confint()`: nos da intervalos de confianza para los coeficientes de la recta.
- `residuals()`: nos muestra los residuos de la regresión.

Invitamos al lector a que ejecute estos comandos en su sesión de R:

```
summary(regresion)
confint(regresion)
residuals(regresion)
```

A.7. Sistemas de ayuda en el entorno R

Hasta ahora hemos sido capaces de ejecutar algunas instrucciones en R y obtener resultados sin mucha dificultad. No obstante, muchas veces nos encontraremos con comandos de los que no sabemos (o no recordamos) cómo es su sintaxis, qué opciones ofrecen, etc. Para ello R ofrece varias posibilidades de ayuda:

- `help()`: es el comando de ayuda por excelencia. Por ejemplo, para obtener ayuda sobre la función `mean` basta con teclear:

```
help(mean)
```

De manera equivalente podríamos usar la sintaxis `?mean`.

- `apropos()`: si queremos utilizar una función en cuyo nombre aparecía el término `norm` pero no recordamos exactamente qué función era, `apropos("norm")` nos devuelve un listado de funciones que contienen dicho témino en su definición.
- `help.search()`: si, en cambio, quisiéramos saber, entre todos los paquetes que R tiene instalados en nuestro sistema, cuáles contienen funciones relacionadas, aunque sea vagamente, con el término `norm` podemos utilizar `help.search("norm")` o, de manera equivalente `??norm`.

- `help.start()`: nos arranca una página html como la que se muestra en la figura A.5 en la que podemos acceder a los manuales originales del programa (en inglés), documentación de referencia sobre los distintos paquetes instalados en nuestro sistema (con información sobre cada una de las funciones que lo componen) y diverso material misceláneo.

- `demo()`: nos hace demostraciones de las posibilidades de uso de distintos paquetes. Por ejemplo: `demo(graphics); demo(persp); demo(lm.glm)`

Figura A.5: Página html de ayuda que se despliega al ejecutar el comando `help.start()`

Por último, si dentro de nuestro entorno R no encontramos ayuda para el problema que nos ocupa, siempre podemos recurrir a la página web de R (fig A.1) y en el menú de la izquierda elegir la opción `search`. Por último, otro sitio interesante donde obtener ayuda es R-Wiki (`http://rwiki.sciviews.org/doku.php`).

A.8. Scripts (archivos de sintaxis de R)

Como hemos podido ver en nuestra primera sesión con R, el uso de este programa requiere el conocimiento de su sintaxis y de las funciones predefinidas. Obviamente ello sólo es posible consultando manuales y aprendiendo qué función es la que hace el análisis estadístico que nos interesa en cada momento. El uso continuado del programa permite que vayamos memorizando estas funciones y las usemos con soltura. Ahora bien, en muchos entornos profesionales -medicina, ciencias sociales, ingeniería,...- la estadística es una herramienta que, si bien resulta indispensable, tampoco se usa todos los días. Ello da lugar a que, enfrentado a un problema de análisis de datos, sea éste nuevo o rutinario, el usuario deba hacer a veces un gran esfuerzo para recordar como arranca el programa, como se leen los datos, como se calcula una media...

Dos son las aproximaciones más usadas para enfrentarse a este problema:

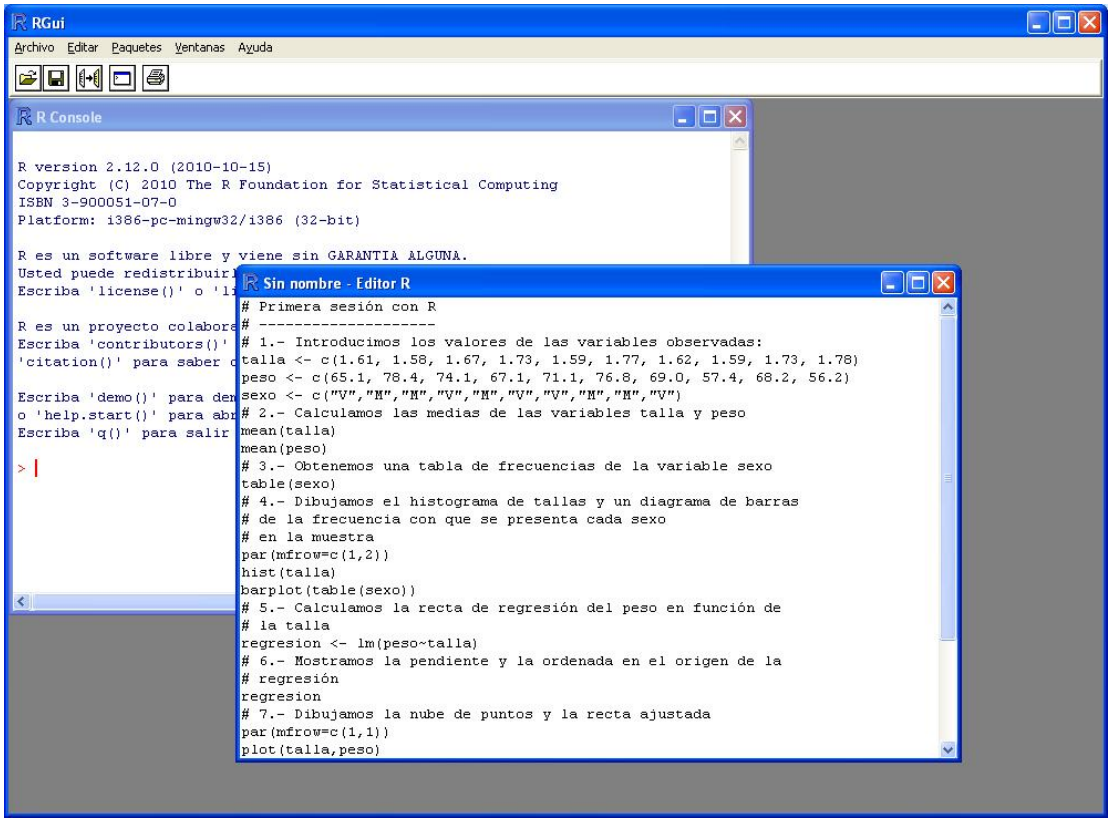


Figura A.6: Ventana de edición de scripts de código de R

1. **Documentar y archivar los procedimientos empleados para cada análisis** en archivos de comandos, llamados *scripts* (guiones). Estos *scripts* pueden reutilizarse con nuevos conjuntos de datos y, en muchas ocasiones, nos sirven de guía para el desarrollo de nuevos procedimientos. Por ello es recomendable en las sesiones de trabajo con R (salvo que sea algo extremadamente simple) no introducir nuestras instrucciones directamente en la linea de comandos, (la que empieza con el símbolo "$>$", y en la que hemos ejecutado el ejemplo visto en la sección anterior), sino abrir un archivo de script e incluir en éste todas las acciones a realizar durante nuestra sesión de trabajo. En dicho archivo podenos incluir lineas de comentario, que deben empezar por el símbolo #, informando de qué es lo que se hace y por qué se hace. Pensemos que aunque ahora sepamos muy bien lo que estamos haciendo, si vamos a volver a empler este código dentro de unos meses (¡a lo mejor sólo dentro de unas semanas!) es muy probable que para ese entonces no recordemos cuál era su función. Es por ello que el guión para nuestra sesión de ejemplo debería quedar, una vez editado, tal como se muestra en la tabla A.2; como vemos, se han insertado lineas de comentario (precedidas por el símbolo #) en las que se explica brevemente qué hacen las distintas partes del código.

```
# --------------------
# Primera sesión con R
# --------------------
# 1.- Introducimos los valores de las variables observadas:
edad <- c(22, 34, 29, 25, 30, 33, 31, 27, 25, 25)
tiempo <- c(14.21, 10.36, 11.89, 13.81, 12.03, 10.99, 12.48,
13.37, 12.29, 11.92)
sexo <- c("M", "H", "H", "M", "M", "H", "M", "M", "H", "H")
# 2.- Calculamos las medias de las variables edad y tiempo
mean(edad)
mean(tiempo)
# 3.- Obtenemos una tabla de frecuencias de la variable sexo
table(sexo)
# 4.- Dibujamos el histograma del tiempo de uso del móvil y un
# diagrama de barras de la frecuencia con que se presenta cada
# sexo en la muestra
par(mfrow=c(1,2))
hist(tiempo)
barplot(table(sexo))
# 5.- Calculamos la recta de regresión del tiempo en función de

# la edad
regresion <- lm(tiempo~edad)
# 6.- Mostramos la pendiente y la ordenada en el origen de la
# regresión
regresion
# 7.- Dibujamos la nube de puntos y la recta ajustada
par(mfrow=c(1,1))
plot(edad, tiempo)
abline(regresion,col="red")
# 8.- Vemos el resumen de la inferencia sobre la regresión
summary(regresion)
# 9.- Intervalos de confianza para los coeficientes
confint(regresion)
# 10.- Obtenemos los residuos de la regresión
residuals(regresion)
```

Tabla A.2: Script de código R para el ejemplo de la sección A.5

Para generar un archivo de script, en el menú de R elegimos la opción *Archivo* → *Nuevo Script*. Se nos abre una ventana de edición en la que podemos escribir nuestras líneas de código, tal como se muestra en la figura A.6.

Una vez que hayamos terminado, guardamos el archivo. Conviene añadirle la extensión *".R"* para que nuestro archivo de código R quede perfectamente identificado para el sistema operativo.
Una vez que se dispone de un *script* éste puede ejecutarse linea a linea simplemente situando el cursor sobre la linea a ejecutar y pulsando CTRL-R, o pulsando el tercer icono de la parte superior izquierda de la pantalla (un icono que muestra dos pantallas con una flecha apuntando de una a otra). Si se desea, se puede ejecutar también todo el código de una vez marcándolo y pulsando CTRL-R, o pulsando en el mismo icono anterior. El comando o comandos que se hayan indicado se ejecutarán y mostrarán su resultado en la ventana principal de R.

2. **Utilizar un entorno de R que disponga de un menú de comandos** al estilo de otros programas estadísticos. R dispone de una librería llamada *R-commander* que facilita un entorno con estas características: el usuario elige el procedimiento en un menú y R-commander se encarga de escribir y ejecutar el código R asociado. Un poco más adelante aprenderemos a instalar y utilizar este entorno.

A.9. Instalación de Editores de Código alternativos: Rstudio.

Tal como hemos visto en la sección anterior, para editar código R podemos utilizar el editor de scripts implementado en el propio programa. Este editor en general va muy bien, pero resulta poco práctico si desarrollamos un código muy largo que incluya, por ejemplo, llamadas a múltiples funciones o generación de nuevas variables u objetos. En estos casos los principales problemas a que se enfrenta el usuario derivan de errores muy simples: paréntesis que se han abierto pero no se han cerrado, comillas mal colocadas, comentarios mal declarados,... En el editor de scripts que incluye R por defecto estos errores resultan difíciles de detectar. Existen otros editores que utilizan diferentes colores para las distintas clases de objetos de R, que autocompletan el código y que incluyen algunos menús de ayuda, facilitándonos mucho la tarea de desarrollar nuestros programas en R y detectar los posibles errores de sintaxis. Describimos a continuación el editor **Rstudio**, que

destaca por sus prestaciones y facilidad de uso.

A.9.1. RStudio

RStudio es un IDE (*Integrated Development Environment*, o Entorno de Desarrollo Integrado) de código abierto para R, lanzado en febrero de 2011. Reúne todas las características que acabamos de citar, a las que se une una muy interesante organización del escritorio de trabajo, que nos muestra un entorno muy compacto con una ventana de edición, otra de resultados, otra para gráficos y una cuarta para la gestión de variables en memoria. Asimismo los menús de ayuda y la descarga de librerías se organizan en pestañas fácilmente accesibles. Cuenta además con la ventaja de ser multiplataforma, lo que significa que existen versiones para Windows, Linux y Mac. En la figura podemos ver el aspecto de RStudio. Este programa puede descargarse libremente de la web http://www.rstudio.org/. La organización del escritorio y la excelente integración con R hacen MUY RECOMENDABLE el uso de este programa.

A.10. Instalación de librerías adicionales en R

Como ya hemos señalado, una de las grandes ventajas de R respecto a otros programas de análisis de datos es su capacidad de crecer mediante la incorporación de nuevas librerías que añaden nuevas funciones a las ya existentes en el programa. En esta sección, a modo de ejemplo, vamos a aprender a descargar, instalar y utilizar la librería R-Commander. Otras librerías de R se descargan e instalan mediante el mismo procedimiento.

A.10.1. Instalación de la librería R-Commander

Ya mencionamos en una sección anterior que una alternativa cómoda para utilizar R sin necesidad de conocer su lenguaje es emplear un sistema de menús al estilo de otros paquetes estadísticos, de tal forma que a través de dichos menús sea posible importar o exportar datos, elegir el tipo de análisis a realizar, elegir las representaciones gráficas, etc. En estos momentos R cuenta con la librería *R-Commander*, que crea un entorno de trabajo con un menú de estas características.

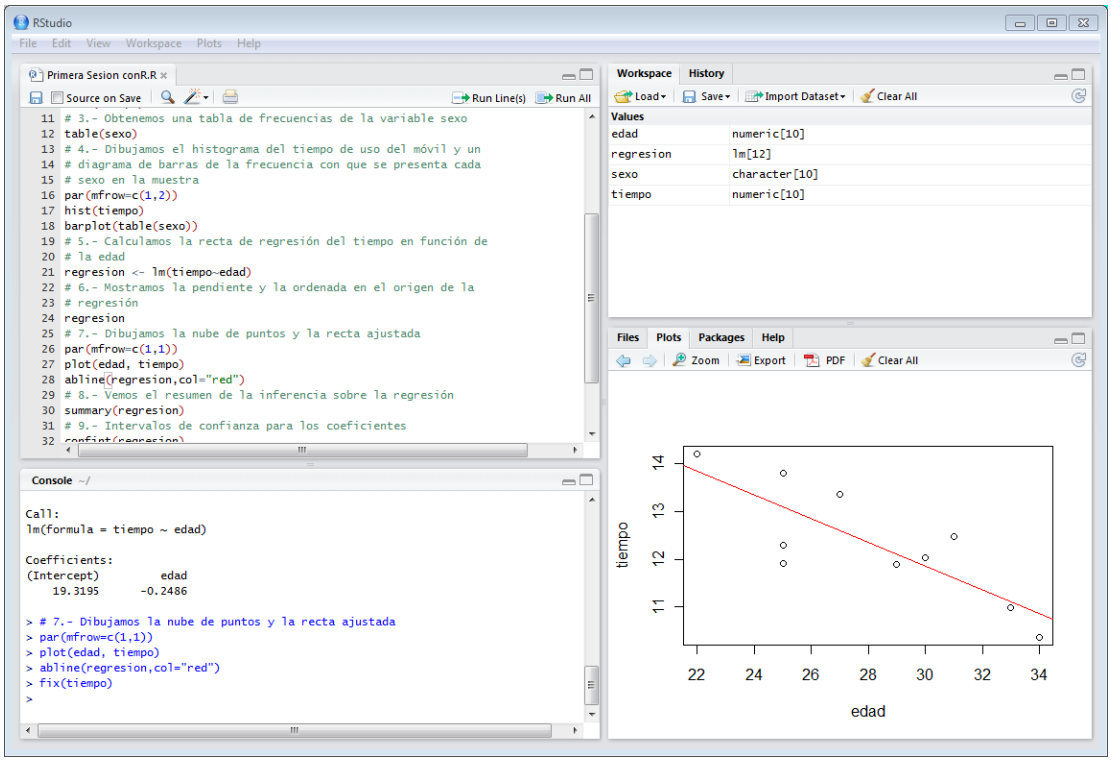


Figura A.7: Entorno Integrado de Desarrollo para R que ofrece el programa RStudio. Pueden verse la ventana de edición, la ventana de resultados, la ventana de inspección de variables y la ventana de gráficos.

La gran ventaja de este entorno, aparte de facilitar el acceso a los procedimientos estadísticos, es que cuando el usuario elige un procedimiento del menú, en la pantalla se muestra el código R que lleva a cabo dicho procedimiento, de tal forma que aún sin querer vamos aprendiendo el lenguaje R.

Para instalar R-Commander (al igual que para instalar cualquier otra librería), simplemente arrancamos R, y en el menú superior elegimos la opción[2] *Paquetes* →*Instalar Paquete(s)*, tal como se muestra en la figura A.8.

R nos pregunta entonces desde qué repositorio[3] deseamos descargar la librería.

[2]Conviene distinguir entre los conceptos *"Cargar paquete"* e *"Instalar paquete"*, que aparecen agrupados en el mismo menú. *"Instalar paquete"* significa que deseamos descargarnos una librería desde el sitio de R en internet e instalarla físicamente en nuestro ordenador. Es lo que debemos hacer la primera vez que queremos usar una nueva librería. *"Cargar paquete"*, en cambio, presupone que la librería ya ha sido instalada con anterioridad y que ahora pretendemos cargarla en memoria para que R pueda utilizarla durante esta sesión.

[3]Un *repositorio* es un lugar en el que se encuentran almacenados todos los programas que componen R, tanto el sistema base, como las librerías, como el código fuente. Ahora mismo

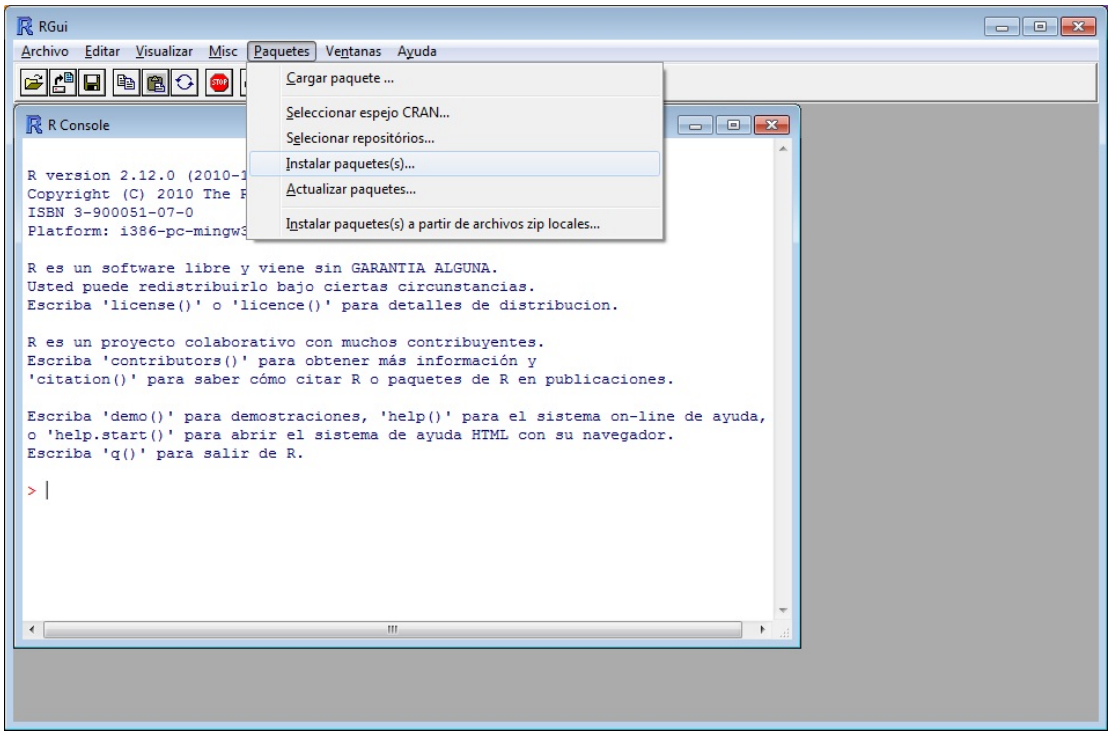


Figura A.8: Instalación de paquetes en R

En principio, aunque todos los repositorios funcionan como *espejos* (todos tienen el mismo contenido), hay ciertas adaptaciones locales en lo referido al idioma. Así, si elegimos el repositorio de España (o de otro país de lengua castellana), la versión de R-Commander que se descarga viene con el sistema de menús casi completamente traducido al español.

Tras elegir repositorio, se nos muestra un menú como el de la figura A.9, en el que seleccionamos el paquete que deseamos instalar, en este caso, Rcmdr. Una vez que pulsemos OK comienza la descarga. Si todo va bien, ésta concluirá con el mensaje siguiente:

```
Rcmdr' successfully unpacked and MD5 sums checked

The downloaded packages are in C:\Users\...\Rtmprp5FkF\downloaded_packages
```

De esta forma ya tenemos el *paquete* Rcmdr descargado, descomprimido y convertido en librería instalada en nuestro ordenador. Una cuestión importante a tener

existen múltiples repositorios repartidos por todo el planeta, que se mantienen constantemente actualizados.

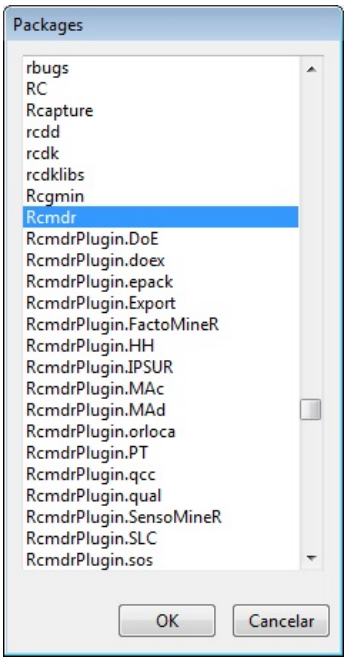


Figura A.9: Menú de selección de librerías disponibles para instalar en R

en cuenta es que *tener una librería físicamente instalada en nuestro ordenador no significa que esté automáticamente disponible para su uso con R*. Para ello R debe cargarla en memoria y ello se lleva a cabo mediante la función `library()`. Por tanto, en R tecleamos:

> library(Rcmdr)

La primera vez que ejecutemos R-Commander nos informará de que le faltan algunos paquetes para su correcto funcionamiento[4] y nos pedirá permiso para instalarlos[5]. Le decimos que sí y esperamos unos minutos mientras realiza la descarga e instalación de esos paquetes. Al finalizar arrancará el entorno R-Commander tal como se muestra en la figura A.10.

Hay que indicar que R-Commander pertenece al pequeño grupo de librerías que arrancan entornos adicionales (con sistemas de menús propios), por lo que el

[4]R-Commander ha ido incorporando a su sistema de menús opciones que se encuentran en otras librerías. Dependiendo de la configuración del sistema de cada usuario, al instalar Rcmdr, el programa detecta qué librerías están ya instaladas en el ordenador del usuario y descarga el resto.

[5]Conviene aclarar, no obstante, que la mayoría de las librerías de R son autocontenidas, y no requieren de la instalación de paquetes adicionales, como ocurre con R-commander.

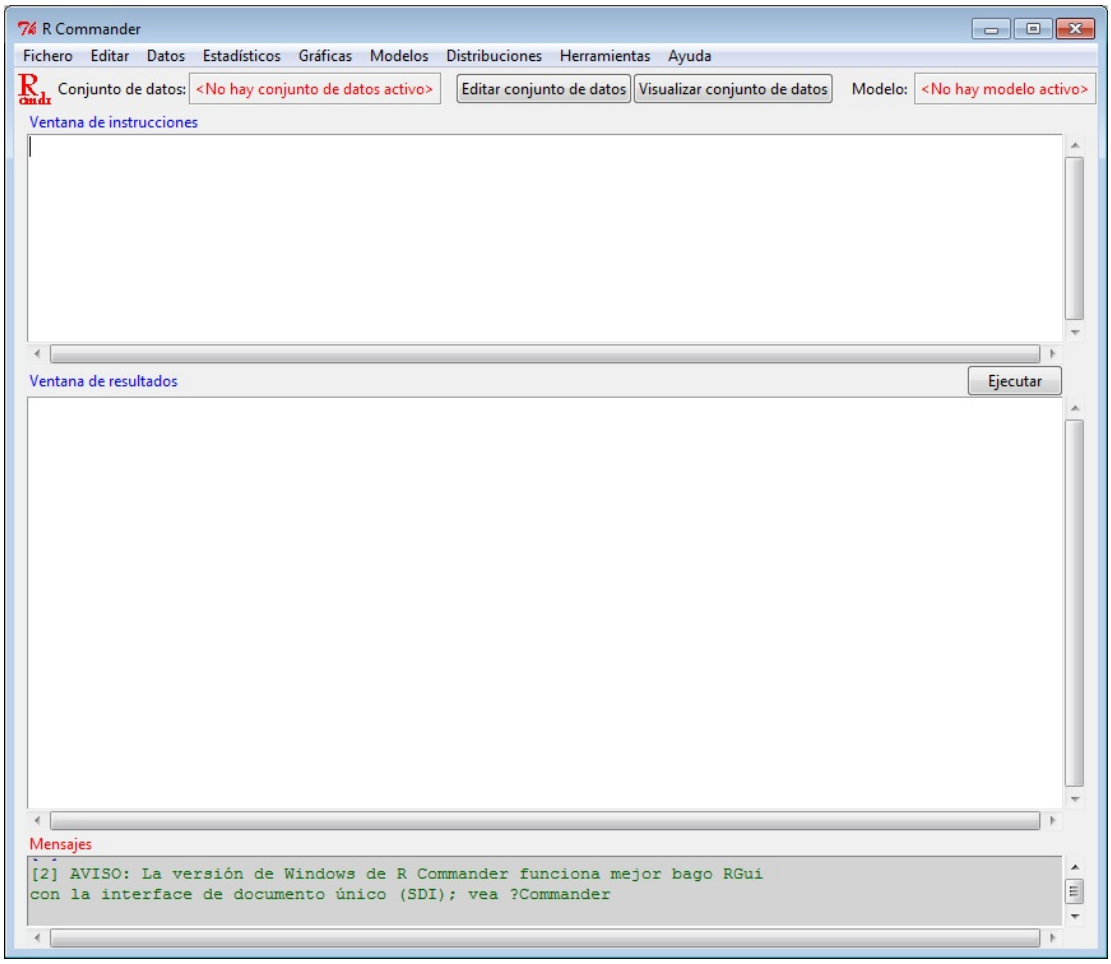


Figura A.10: Entorno R-Commander

comando anterior `library(Rcmdr)` produce el efecto claramente visible de hacer aparecer una nueva ventana en pantalla. La mayoría de librerías de R son en realidad colecciones de funciones y la ejecución del comando `library(nombre-del-paquete)` no causa ningún efecto visible, salvo que a partir del momento de su ejecución todas las funciones que contiene el paquete están listas para su uso. Si ejecutamos el comando `help.start()`, elegimos la opción `Packages` y a continuación seleccionamos el nombre del paquete que hemos instalado, R nos mostrará la lista de funciones que contiene dicho paquete, dándonos además la opción de seleccionar cada función para obtener ayuda sobre su funcionamiento.

En Windows el paquete Rcmdr funciona mejor con la interfaz SDI (Single Document Interface). Por defecto, R se instala con la interfaz MDI (Multiple Document Interface), que permite tener simultáneamente varias ventanas abiertas

dentro de R. Sin embargo, como efecto secundario de esta interfaz, las ventanas de R-Commander en las que nos pide información asociada al procedimiento que hayamos puesto en marcha, se quedan ocultas bajo la ventana principal de R, lo que dificulta su manejo. La manera más sencilla de evitar este efecto es la siguiente:

1. Hacer una copia del icono R que se encuentra en el escritorio (picar con el botón derecho del ratón sobre el icono, elegir *copiar*, y pegar en otro lugar del escritorio). Al nuevo icono podemos ponerle el nombre R-SDI, por ejemplo.

2. Picar con el botón derecho del ratón sobre el nuevo icono y seleccionar *Propiedades.* Veremos que en la cajita *Destino* dice algo así como *"C:\Program Files\R\R-2.12.0\bin\i386\Rgui.exe".* Añadir - *-sdi* a continuación, de tal manera que quede de la forma: *"C:\Program Files\R\R-2.12.0\bin\i386\Rgui.exe" - -sdi.*

3. Ahora cuando arranquemos R desde el icono R-SDI, y ejecutemos `library(Rcmdr)`, las distintas ventanas de R-Commander se presentarán correctamente.

A.10.2. Instalar paquetes desde archivos zip locales.

En ocasiones necesitamos instalar librerías de R sin estar conectados a internet. En tal caso, si disponemos del paquete que contiene la librería en formato comprimido zip (lo que presupone que nosotros mismos hemos descargado dicho paquete previamente de internet, o que nos ha sido facilitado de alguna otra manera, por ejemplo en un CD), podemos proceder muy sencillamente a su instalación simplemente eligiendo la opción:

Paquetes → Instalar paquete(s) a partir de archivos zip locales...

del menú principal de R. En la ventana que nos muestra sólo tendremos que elegir el directorio y el paquete que deseamos instalar. Si deseamos instalar varios podemos seleccionarlos todos a la vez.[6]

[6]Con la instalación de paquetes a partir de archivos zip locales nos encontramos a veces con problemas derivados de *dependencias* entre paquetes. Muchos paquetes de R dependen de funciones que se encuentran en otros paquetes. Si el paquete que estamos instalando a partir de un archivo local depende de algún otro paquete del que no disponemos, la instalación de nuestro paquete se completará, pero algunas funciones de las que contiene (y en algún caso puede que todas) no funcionarán adecuadamente.

A.11. Objetos en R: factores, vectores, matrices, data.frames, listas.

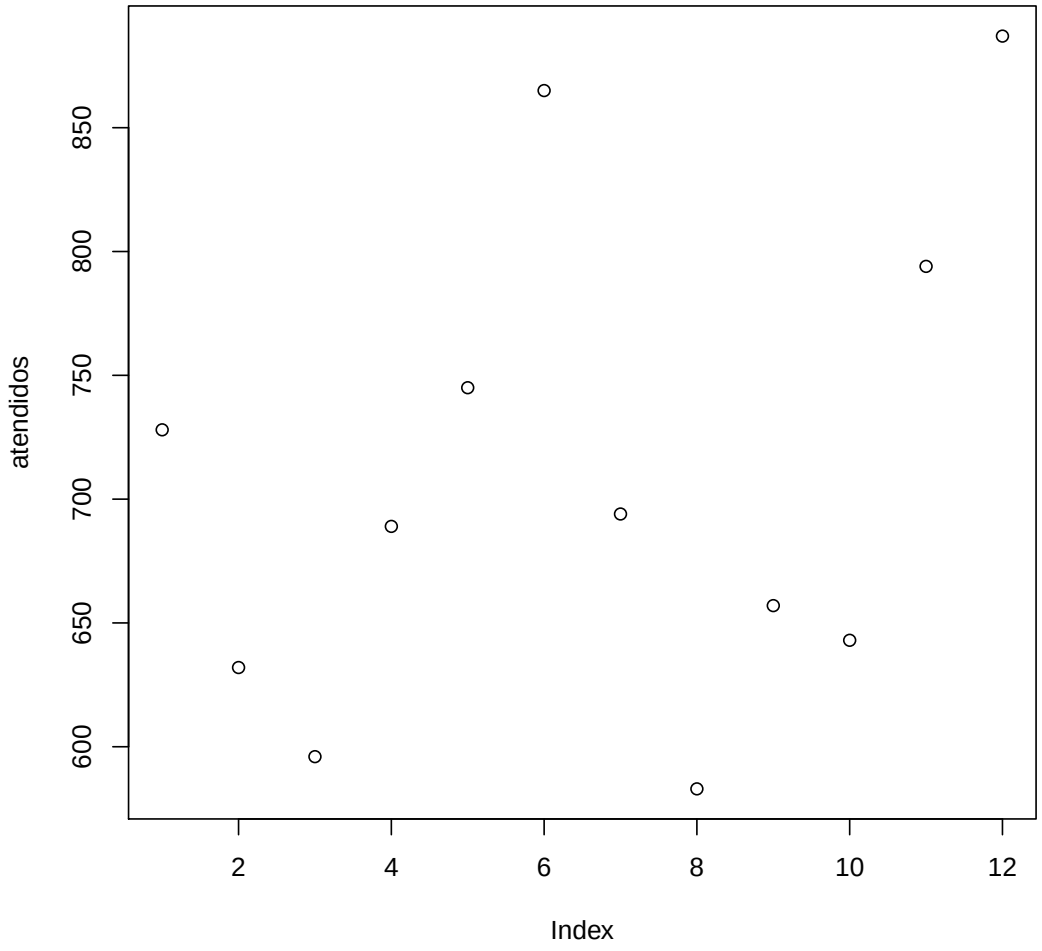
A.11.1. Programación orientada a objetos en R.

Para realizar un uso eficiente del programa R es preciso entender y aprender a manipular bien las clases de los distintos objetos que maneja, ya que R utiliza *programación orientada a objetos*. Esto significa que *una misma función hace cosas distintas según la clase del objeto que recibe como argumento*, pudiendo incluso no hacer nada (o dar error) si se le pasan argumentos de una clase inadecuada. A modo de ejemplo, veamos como la función `plot()` puede mostrar distintos gráficos según la clase del objeto a representar. Para ello, supongamos que el siguiente vector representa el número de personas atendidas mensualmente en el servicio de urgencias de un centro de salud durante el pasado año (datos de enero a diciembre):

```
atendidos <- c(728, 632, 596, 689, 745, 865, 694, 583, 657, 643, 794, 887)
class(atendidos)

## [1] "numeric"
```

La función `class()` nos devuelve la *clase* del objeto `atendidos`, que como vemos es un vector numérico. Podemos obtener una representación gráfica de este vector simplemente mediante:

```
plot(atendidos)
```



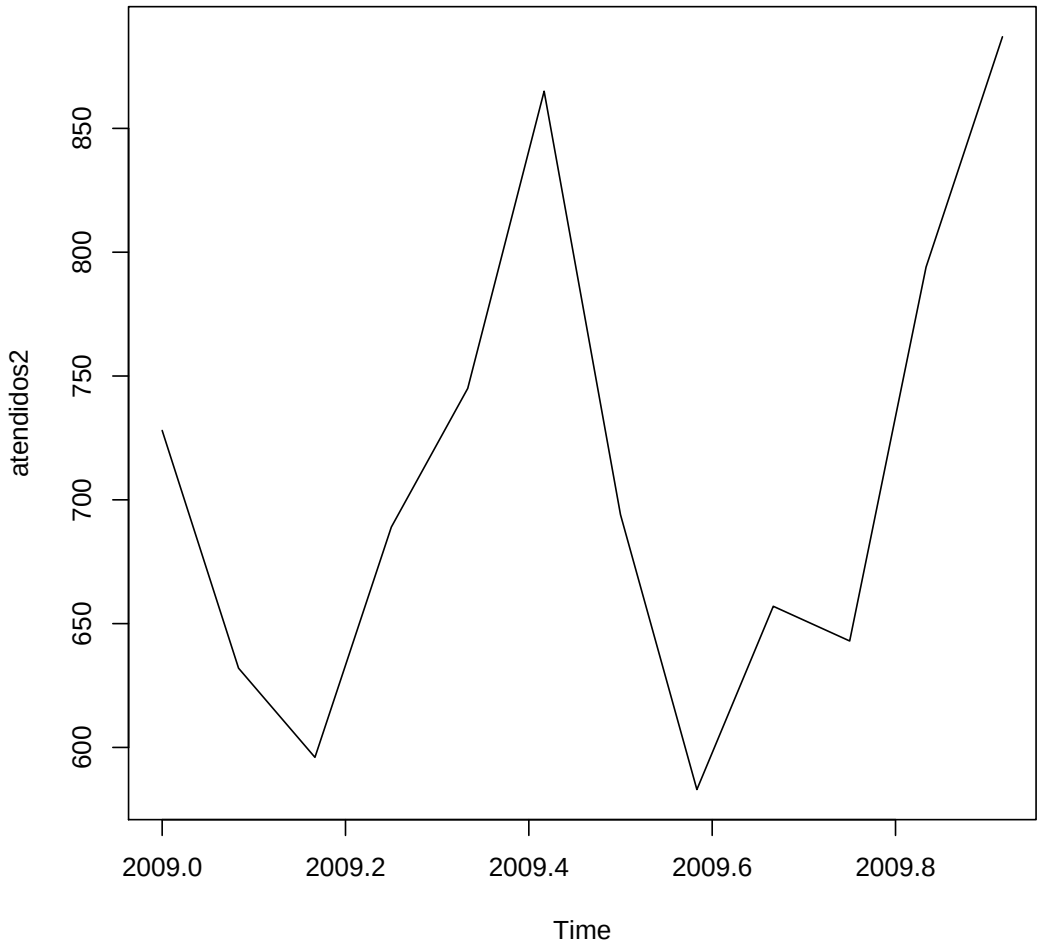
Ahora convertimos estos datos en *serie temporal* mediante la función `ts()`, indicando que esta serie comienza en enero del año 2009 y tiene una frecuencia de 12 observaciones por año (esto es, una por mes):

```
atendidos2 <- ts(atendidos, frequency = 12, start = c(2009, 1))
atendidos2

##      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 2009 728 632 596 689 745 865 694 583 657 643 794 887

class(atendidos2)
```

```
## [1] "ts"
```

Como podemos ver, la clase del objeto `atendidos2` es `ts` (*time series*). Podemos comprobar que si aplicamos la misma función `plot()` a `atendidos2`, el gráfico obtenido es distinto que cuando se aplica a `atendidos`, aún cuando los datos sean exactamente los mismos:

```
plot(atendidos2)
```



A.11.2. Factores

Un *factor* es una variable categórica. En R los factores se utilizan habitualmente para realizar clasificaciones de los datos, estableciendo su pertenencia a los grupos o categorías determinados por los niveles (valores) del factor. Un factor puede ser de tipo numérico o de tipo carácter[7]. A modo de ejemplo, la variable `sexo` mostrada en la tabla A.1 puede ser considerada un factor, ya que establece para cada sujeto de la tabla su pertenencia a una de las dos categorías *Hombre* o *Mujer*. Para que R reconozca al sexo como factor, una vez introducidos los datos utilizamos la función:

```
sexo <- factor(sexo)
sexo

##  [1] M H H M M H M M H H
## Levels: H M
```

con lo que hemos convertido `sexo` en un factor con dos niveles `V` y `H`. En muchos casos, los niveles del factor son poco ilustrativos de su significado. La siguiente sintaxis especifica explícitamente los niveles del factor (*levels*) y asigna etiquetas (*labels*) a cada uno de ellos:

```
sexo <- factor(sexo, levels = c("H", "M"), labels = c("Hombre", "Mujer"))
```

Estas etiquetas aparecerán en los resultados de los procedimientos estadísticos donde apareza el factor, aclarando su significado. Por ejemplo, si pedimos a R que nos construya la tabla de frecuencias de sexos, en lugar de `H` o `M` nos mostrará los términos *Hombre* o *Mujer:*

```
table(sexo)

## sexo
## Hombre  Mujer
##      5      5
```

[7] Independientemente de que el factor sea numérico o carácter, sus valores son siempre almacenados internamente por R como números enteros, con lo que se consigue economizar memoria.

A.11.3. Vectores. Asignación y selección de los valores de un vector.

Ya hemos visto en nuestra primera sesión (ver tabla A.2) que el operador para asignar valores a un vector es `<-`. Este operador puede utilizarse de varias formas equivalentes:

```
edad <- c(22, 34, 29, 25, 30, 33, 31, 27, 25, 25)
```

También puede utilizarse indistintamente el símbolo "=":

```
edad = c(22, 34, 29, 25, 30, 33, 31, 27, 25, 25)
```

Si deseamos asignar a una variable una sucesión de valores consecutivos podemos utilizar el operador `":"`. Así para asignar al vector x los valores de 1 a 10 procederíamos del siguiente modo:

```
x = 1:10
x
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

Podemos construir secuencias de números más complejas mediante la función `seq()`:

- Sucesión de valores de 1 a 20 de 2 en 2:

```
x = seq(1, 20, by = 2)
x
```

```
##  [1]  1  3  5  7  9 11 13 15 17 19
```

- Sucesión de 8 valores equiespaciados entre 1 y 20:

```
y = seq(1, 20, length = 8)
y

## [1]  1.000  3.714  6.429  9.143 11.857 14.571 17.286 20.000
```

Es posible acceder al valor que ocupa la posición k dentro de un vector x refiriéndonos a él como `x[k]`:

```
x[3]

## [1] 5

y[5]

## [1] 11.86
```

Podemos acceder también simultáneamente a varios valores dentro de un vector. Por ejemplo, si deseamos ver del segundo al quinto de los valores observados en la variable `edad`:

```
edad[2:5]

## [1] 34 29 25 30
```

Y si quisiéramos ver sólo los valores primero, tercero y séptimo:

```
edad[c(1, 3, 7)]

## [1] 22 29 31
```

A.11.4. Selección condicionada.

La función `which()` nos da las posiciones, dentro de un vector, de los valores que cumplen cierta condición. Por ejemplo:

```
which(edad > 25)

## [1] 2 3 5 6 7 8
```

nos indica que los valores del vector `edad` mayores que 25 son los que ocupan las posiciones 2, 3, 5, 6, 7 y 8. Podemos ver cuáles son concretamente esos valores mediante:

```
edad[which(edad > 25)]

## [1] 34 29 30 33 31 27
```

Esta expresión puede simplificarse; si no utilizamos `which()` obtenemos exactamente el mismo resultado:

```
edad[edad > 25]

## [1] 34 29 30 33 31 27
```

Se puede realizar también la selección de valores de un vector condicionando por los valores de otro vector. Por ejemplo, si las diez edades del ejemplo anterior corresponden a personas cuyo sexo viene dado por:

```
sexo <- c("M", "H", "H", "M", "M", "H", "M", "M", "H", "H")
```

podríamos seleccionar la edad de las mujeres simplemente mediante:

```
edad[sexo == "M"]

## [1] 22 25 30 31 27
```

o podríamos seleccionar el sexo de aquellos que han hablado más de 13 minutos diarios mediante:

```
sexo[tiempo > 13]
```

```
## [1] "M" "M" "M"
```

A.12. Aritmética vectorial.

R utiliza los operadores aritméticos elementales: suma (+), resta (-), multiplicación (*) y división (/). Cuenta asimismo con un catálogo muy completo de funciones matemáticas. Por citar unas pocas: logaritmo neperiano (*log*), exponencial (*exp*), seno (*sin*), coseno (*cos*), valor absoluto (*abs*), parte entera (*floor*), redondeo (*round*).

Una característica importante de R es su capacidad para la *aritmética vectorial*. Esto significa que cuando una función o un operador se aplica a un vector, dicha operación o función se ejecuta sobre todos los componentes del vector. Veamos algunos ejemplos:

- Necesitamos transformar la variable `tiempo` a escala logarítmica y guardar el resultado en una variable llamada `logtime`. En R esto es tan simple como hacer lo siguiente:

```
options(width = 60)
```

```
logtime = log(tiempo)
logtime
```

```
##  [1] 2.654 2.338 2.476 2.625 2.487 2.397 2.524 2.593 2.509
## [10] 2.478
```

- Queremos *tipificar* los valores de la variable tiempo (restarles su media y dividir por su desviación típica). La variable tiempo tipificada se obtiene fácilmente en R mediante:

```
ztiempo = (tiempo - mean(tiempo))/sd(tiempo)
ztiempo

##  [1]  1.5626 -1.6459 -0.3709  1.2292 -0.2542 -1.1209  0.1208
##  [8]  0.8626 -0.0375 -0.3459
```

- Queremos pasar la variable *tiempo* que está en minutos, a segundos. Si hacemos:

```
tiempo_seg = 60 * tiempo
tiempo_seg

##  [1] 852.6 621.6 713.4 828.6 721.8 659.4 748.8 802.2 737.4
## [10] 715.2
```

comprobamos que se han multiplicado por 60 todos los valores de tiempo.

A.13. Matrices

Hay varias maneras de definir una matriz en R. Si es una matriz pequeña podemos utilizar la siguiente sintaxis:

```
A = matrix(nrow = 3, ncol = 3, c(1, 2, 3, 4, 5, 6, 7, 8, 9),
    byrow = TRUE)
```

Con el argumento `nrow` hemos indicado el número de filas de nuestra matriz, con `ncol` el número de columnas; a continuación hemos puesto los valores que forman la matriz (los valores del 1 al 9), y le hemos pedido a R que use esos valores para rellenar la matriz A *por filas* (`byrow=TRUE`). La matriz A así construida es:

```
A
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
```

Si disponemos de varios vectores de la misma longitud que queremos utilizar como filas (o columnas) de una matriz, podemos utilizar la función `rbind()` para *unirlos por filas* o la función `cbind()` para *unirlos por columnas*, como vemos en el siguiente ejemplo:

```
vector1 = c(1, 2, 3, 4)
vector2 = c(5, 6, 7, 8)
vector3 = c(9, 10, 11, 12)
M1 = cbind(vector1, vector2, vector3)  # Unimos por columnas
M1
```

```
##      vector1 vector2 vector3
## [1,]       1       5       9
## [2,]       2       6      10
## [3,]       3       7      11
## [4,]       4       8      12
```

```
M2 = rbind(vector1, vector2, vector3)  # Unimos por filas
M2
```

```
##         [,1] [,2] [,3] [,4]
## vector1    1    2    3    4
## vector2    5    6    7    8
## vector3    9   10   11   12
```

Se pueden seleccionar partes de una matriz utilizando los índices de posición $[fila, columna]$ entre corchetes. El siguiente ejemplo ilustra la forma de hacerlo:

```
A[2, 3]  # Se selecciona el valor de la fila 2, columna 3
```

```
## [1] 6
```

```
A[2, ]  # Se selecciona la fila 2 completa
```

```
## [1] 4 5 6
```

```
A[, 3]  # Se selecciona la columna 3 completa
```

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

```
A[1, 2:3]  # Se seleccionan el segundo y tercer valor de la fila 1
```

```
## [1] 2 3
```

A.13.1. Operaciones con matrices

La función `diag()` extrae la diagonal principal de una matriz:

```
diag(A)
```

```
## [1] 1 5 9
```

También permite crear matrices diagonales:

```
diag(c(1, 2, 3, 4))
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    2    0    0
## [3,]    0    0    3    0
## [4,]    0    0    0    4
```

Hay que tener cierto cuidado con los operadores aritméticos básicos. Si se suman una matriz y una constante, el efecto es que dicha constante se suma a todos los elementos de la matriz. Lo mismo ocurre con la diferencia, la multiplicación y la división:

```
M = matrix(nrow = 2, c(1, 2, 5, 7), byrow = F)
M
```

```
##      [,1] [,2]
## [1,]    1    5
## [2,]    2    7
```

```
M + 2
```

```
##      [,1] [,2]
## [1,]    3    7
## [2,]    4    9
```

```
M - 2
```

```
##      [,1] [,2]
## [1,]   -1    3
## [2,]    0    5
```

```
M * 2
```

```
##      [,1] [,2]
## [1,]    2   10
## [2,]    4   14
```

```
M/2
```

```
##      [,1] [,2]
## [1,]  0.5  2.5
## [2,]  1.0  3.5
```

Asimismo, si a una matriz se le suma un vector cuya longitud sea igual al número de filas de la matriz, se obtiene como resultado una nueva matriz cuyas columnas son la suma de las columnas de la matriz original más dicho vector. Lo mismo ocurre con la diferencia, la multiplicación y la división:

```
v = c(3, 4)
M + v
```

```
##      [,1] [,2]
## [1,]    4    8
## [2,]    6   11
```

```
M - v
```

```
##      [,1] [,2]
## [1,]   -2    2
## [2,]   -2    3
```

```
M * v
```

```
##      [,1] [,2]
## [1,]    3   15
## [2,]    8   28
```

```
M/v
```

```
##        [,1]  [,2]
## [1,] 0.3333 1.667
## [2,] 0.5000 1.750
```

La suma o resta de matrices de la misma dimensión se realiza con los operadores + y -; el producto de matrices (siempre que sean compatibles) se realiza con el símbolo `%*%`:

```
M + M
```

```
##      [,1] [,2]
## [1,]    2   10
## [2,]    4   14
```

```
M - M
```

```
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    0
```

```
M %*% M
```

```
##      [,1] [,2]
## [1,]   11   40
## [2,]   16   59
```

Una fuente de posibles errores en el cálculo matricial, cuando se utilizan matrices de la misma dimensión, es utilizar los operadores `*` y `/` ya que no realizan el producto matricial, sino que multiplican las matrices término a término:

```
M * M
```

```
##      [,1] [,2]
## [1,]    1   25
## [2,]    4   49
```

```
M/M
```

```
##      [,1] [,2]
## [1,]    1    1
## [2,]    1    1
```

A.13.2. Potencia de una matriz.

R no dispone en su paquete base de una función para calcular la potencia n-ésima de una matriz. No obstante el paquete *expm* impementa el operador `%^%` construido con este objetivo. Su uso es muy simple: si queremos calcular la matriz A^n bastará con utilizar el comando `A %^% n`. Debemos recordar cargar primero el paquete *expm:*

```
library(expm)
A = matrix(1:4, nrow = 2)
A

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

A %*% A

##      [,1] [,2]
## [1,]    7   15
## [2,]   10   22

A %^% 2

##      [,1] [,2]
## [1,]    7   15
## [2,]   10   22

A %^% 5

##      [,1] [,2]
## [1,] 1069 2337
## [2,] 1558 3406
```

A.13.3. Sistemas de ecuaciones lineales. Matrices inversas.

R dispone de la función `solve()` para resolver sistemas lineales de ecuaciones. En particular, si el sistema a resolver es de la forma $Ax = b$ (donde b podría ser también una matriz), su solución es $x = A^{-1}b$, que en R se obtiene mediante `solve(A,b)`. En particular, si b es la matriz identidad, la función anterior devuelve la matriz inversa de A:

```
A = matrix(ncol = 3, c(2, 4, 3, 5, 7, 1, 2, 2, 3), byrow = TRUE)
A
```

```
##      [,1] [,2] [,3]
## [1,]    2    4    3
## [2,]    5    7    1
## [3,]    2    2    3
```

```
I = diag(1, nrow = 3)  # Matriz diagonal de dimension 3
solve(A, I)  # Matriz inversa de A
```

```
##         [,1]    [,2]    [,3]
## [1,] -0.7308  0.2308  0.6538
## [2,]  0.5000  0.0000 -0.5000
## [3,]  0.1538 -0.1538  0.2308
```

Podemos obviar la referencia a la matriz identidad `I`. Si utilizamos solamente `solve(A)` obtenemos también la inversa de `A`:

```
solve(A)
```

```
##         [,1]    [,2]    [,3]
## [1,] -0.7308  0.2308  0.6538
## [2,]  0.5000  0.0000 -0.5000
## [3,]  0.1538 -0.1538  0.2308
```

A.14. Data.frames

El término *data.frame* es difícil de traducir al castellano. Podría traducirse como *Hoja de datos* o *Marco de datos.* Los *data.frames* son una clase de objetos especial en R. Normalmente, cuando se realiza un estudio estadístico sobre los sujetos u objetos de una muestra, la información se organiza precisamente en un *data.frame:* una hoja de datos, en los que cada fila corresponde a un sujeto y cada columna a una variable. Así, el conjunto de datos que ya vimos en la tabla A.1 de la sección A.5 constituye un data.frame tal como se muestra en la tabla A.3.

edad	tiempo	sexo
22	14.21	M
34	10.36	H
29	11.89	H
25	13.81	M
30	12.03	M
33	10.99	H
31	12.48	M
27	13.37	M
25	12.29	H
25	11.92	H

Tabla A.3: Datos de la tabla A.1 organizados en forma de *data.frame*

La estructura de un *data.frame* es muy similar a la de una matriz. La diferencia es que una matriz sólo admite valores numéricos, mientras que en un *data.frame* podemos incluir también datos alfanuméricos. El siguiente ejemplo nos muestra como crear un *data.frame* con las variables de la tabla A.3:

```
edad <- c(22, 34, 29, 25, 30, 33, 31, 27, 25, 25)
tiempo <- c(14.21, 10.36, 11.89, 13.81, 12.03, 10.99, 12.48,
    13.37, 12.29, 11.92)
sexo <- c("M", "H", "H", "M", "M", "H", "M", "M", "H", "H")
misdatos <- data.frame(edad, tiempo, sexo)
misdatos

##    edad tiempo sexo
## 1    22  14.21    M
```

```
## 2    34  10.36    H
## 3    29  11.89    H
## 4    25  13.81    M
## 5    30  12.03    M
## 6    33  10.99    H
## 7    31  12.48    M
## 8    27  13.37    M
## 9    25  12.29    H
## 10   25  11.92    H

str(misdatos)

## 'data.frame': 10 obs. of  3 variables:
##  $ edad  : num  22 34 29 25 30 33 31 27 25 25
##  $ tiempo: num  14.2 10.4 11.9 13.8 12 ...
##  $ sexo  : Factor w/ 2 levels "H","M": 2 1 1 2 2 1 2 2 1 1

names(misdatos)

## [1] "edad"   "tiempo" "sexo"
```

En este ejemplo hemos creado un *data.frame* llamado `misdatos` que contiene a las tres variables `edad`, `tiempo` y `sexo`. La función `str()` nos muestra la estructura de este objeto, confirmándonos que es un *data.frame* de tres variables con 10 observaciones cada una. Nos informa además de que las dos primeras variables son numéricas y la tercera, el sexo, es un *factor* con dos valores, *"V"* y *"M"*. La función `names()` por su parte, nos devuelve los nombres de las variables contenidas en *misdatos.*

Como veremos en la siguiente sección, cuando desde R leemos datos situados en un fichero externo (un fichero de texto, una hoja excel, un archivo de datos de SPSS,...), estos datos se importan en un *data.frame*, sobre el cual podemos realizar nuestro estudio estadístico.

El acceso a los datos que se encuentran en un *data.frame* es muy similar al acceso a los datos de una matriz que ya vimos en la sección anterior. Sin embargo, para los

data.frames R dispone de algunas funciones que facilitan la tarea de seleccionar o filtrar datos. Así por ejemplo, si queremos ver sólo los datos de los sujetos 3 a 6, escribiríamos:

```
misdatos[3:6, ]

##   edad tiempo sexo
## 3   29  11.89    H
## 4   25  13.81    M
## 5   30  12.03    M
## 6   33  10.99    H
```

Si queremos seleccionar los datos de edad (primera columna), podemos tratar a `misdatos` igual que si fuese una matriz:

```
misdatos[, 1]

##  [1] 22 34 29 25 30 33 31 27 25 25
```

Aunque también podemos referirnos a la columna por su nombre:

```
misdatos$edad

##  [1] 22 34 29 25 30 33 31 27 25 25
```

Nótese que en este caso hemos de utilizar el nombre del *data.frame* (en este caso `misdatos`) seguido del símbolo `$` y del nombre de la variable que nos interesa (`edad`).

La función `subset()` nos permite seleccionar una parte del *data.frame*. Por ejemplo, si deseamos quedarnos sólo con los hombres utilizaríamos:

```
hombres = subset(misdatos, sexo == "H")
hombres

##    edad tiempo sexo
```

```
## 2    34  10.36    H
## 3    29  11.89    H
## 6    33  10.99    H
## 9    25  12.29    H
## 10   25  11.92    H
```

Podemos elaborar selecciones más complejas; por ejemplo:

- Sujetos que sean hombres y tengan más de 30 años (la condición "y" se especifica mediante el símbolo "&"):

```
mayores = subset(misdatos, sexo == "H" & edad > 30)

mayores

##   edad tiempo sexo
## 2   34  10.36    H
## 6   33  10.99    H
```

- Hombres que tengan menos de 25 años y hayan hablado por el móvil más de 12 minutos diarios:

```
jov_habladores = subset(misdatos, sexo == "H" & edad < 20 & tiempo >
    12)

jov_habladores

## [1] edad   tiempo sexo
## <0 rows> (or 0-length row.names)
```

- Sujetos que tengan menos de 25 o más 30 años (la condición "o" se expresa mediante la línea vertical "|"):

```
extremos = subset(misdatos, edad < 25 | edad > 30)

extremos

##   edad tiempo sexo
## 1   22  14.21    M
## 2   34  10.36    H
## 6   33  10.99    H
## 7   31  12.48    M
```

Podemos seleccionar además un subconjunto de variables del *data.frame*. Por ejemplo, si nos interesan solo la edad y el tiempo de uso del móvil de los hombres de la muestra:

```
hombres = subset(misdatos, sexo == "H", select = c(edad, tiempo))
hombres

##    edad tiempo
## 2    34  10.36
## 3    29  11.89
## 6    33  10.99
## 9    25  12.29
## 10   25  11.92
```

A.15. Listas

La *lista* es la estructura de datos más compleja que maneja R. Podemos entender una lista como un contenedor de objetos que pueden ser de cualquier clase: números, vectores, matrices, funciones, data.frames, incluso otras listas. Una lista puede contener a la vez varios de estos objetos, que pueden ser además de distintas dimensiones.

Ya hemos visto una lista en la sección A.6 cuando pedimos a R que nos mostrase la estructura del objeto `regresion`. Podemos crear ahora una lista que conten-

ga el *data.frame* `misdatos`, la matriz `A`, el vector `x=c(1,2,3,4)` y la constante `e=exp(1)`:

```
MiLista <- list(misdatos, A, M = M, x = c(1, 2, 3, 4), e = exp(1))
MiLista

## [[1]]
##    edad tiempo sexo
## 1    22  14.21    M
## 2    34  10.36    H
## 3    29  11.89    H
## 4    25  13.81    M
## 5    30  12.03    M
## 6    33  10.99    H
## 7    31  12.48    M
## 8    27  13.37    M
## 9    25  12.29    H
## 10   25  11.92    H
##
## [[2]]
##      [,1] [,2] [,3]
## [1,]    2    4    3
## [2,]    5    7    1
## [3,]    2    2    3
##
## $M
##      [,1] [,2]
## [1,]    1    5
## [2,]    2    7
##
## $x
## [1] 1 2 3 4
##
## $e
## [1] 2.718
```

```
MiLista$misdatos

## NULL

MiLista[[1]]

##    edad tiempo sexo
## 1    22  14.21    M
## 2    34  10.36    H
## 3    29  11.89    H
## 4    25  13.81    M
## 5    30  12.03    M
## 6    33  10.99    H
## 7    31  12.48    M
## 8    27  13.37    M
## 9    25  12.29    H
## 10   25  11.92    H

MiLista$M

##      [,1] [,2]
## [1,]    1    5
## [2,]    2    7

MiLista$x

## [1] 1 2 3 4

MiLista$e

## [1] 2.718
```

Como vemos, para acceder a los objetos que forman parte de una lista, basta con añadir su nombre a continuación del de la lista, separados por el símbolo `$`, o bien

con el índice de posición dentro de la lista con doble corchete `[[]]`. Nótese que los objetos `misdatos` y `A` no tienen nombre dentro de la lista, por lo que hemos de referirnos a ellos como `MiLista[[1]]` o `MiLista[[2]]`. Sin embargo, el objeto `M` sí que tiene nombre. Para que un objeto dentro de una lista tenga nombre, éste debe declararse explícitamente en la construcción de la lista, tal como se hizo con `M`, `x` o `e`.

R utiliza las listas, sobre, todo como salida de los distintos procedimientos estadísticos. Así, por ejemplo, al realizar un contraste de las medias de dos poblaciones, R nos calcula, entre otras cosas, la diferencia de medias muestrales, el valor del estadístico de contraste, el p-valor del test y el intervalo de confianza para la diferencia observada. Todos estos términos forman parte de una lista. La sintaxis para comparar, por ejemplo, el tiempo medio de uso del móvil entre hombres y mujeres a partir de nuestros datos sería:

```
t.test(tiempo ~ sexo, data = misdatos)

##
##  Welch Two Sample t-test
##
## data:  tiempo by sexo
## t = -3.133, df = 7.854, p-value = 0.01427
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.9378 -0.4422
## sample estimates:
## mean in group H mean in group M
##           11.49           13.18
```

Si guardamos el resultado de este contraste en el objeto `contraste`, podemos observar que tiene estructura de lista:

```
contraste = t.test(tiempo ~ sexo, data = misdatos)
str(contraste)

## List of 9
```

```
##  $ statistic  : Named num -3.13
##   ..- attr(*, "names")= chr "t"
##  $ parameter  : Named num 7.85
##   ..- attr(*, "names")= chr "df"
##  $ p.value    : num 0.0143
##  $ conf.int   : atomic [1:2] -2.938 -0.442
##   ..- attr(*, "conf.level")= num 0.95
##  $ estimate   : Named num [1:2] 11.5 13.2
##   ..- attr(*, "names")= chr [1:2] "mean in group H" "mean in group M"
##  $ null.value : Named num 0
##   ..- attr(*, "names")= chr "difference in means"
##  $ alternative: chr "two.sided"
##  $ method     : chr "Welch Two Sample t-test"
##  $ data.name  : chr "tiempo by sexo"
##  - attr(*, "class")= chr "htest"
```

A.16. Acceso a datos almacenados en archivos externos.

En las secciones anteriores hemos visto como introducir datos en R utilizando la función de concatenación `c()`. Obviamente esta forma de crear conjntos de datos sólo es eficaz si el número de valores a introducir es pequeño. Si hemos de manejar bases de datos muy grandes resulta más conveniente utilizar programas específicamente diseñados para ello: gestores de hojas de cálculo como Excel u OpenOffice Calc, bases de datos como MySQL o Access, o incluso los editores de datos que proporcionan SPSS o SAS. Para acceder a estos datos desde R existen fundamentalmente dos maneras:

- Que el programa que hemos utilizado para crear la base de datos la exporte a un formato *"libre"*, normalmente texto ASCII plano, que puede ser leido directamente por R.
- Que carguemos en R alguna librería que sea capaz de entender el formato en que se han guardado los datos. Como veremos en esta sección, R dispone

de librerías que le permiten cargar directamente datos guardados en los formatos específicos de los programas citados más arriba.

A.16.1. Establecimiento del directorio de trabajo en R

Cuando vamos a utilizar R para leer datos de ficheros externos, es conveniente decirle a R desde el principio cuál es el directorio donde están los datos. Habitualmente, éste será el *directorio de trabajo*: no sólo contiene los datos, sino también guardaremos en él los resultados de nuestro trabajo, los gráficos que podamos generar, etc. Si nuestro directorio de trabajo es, por ejemplo, `c:\usuario\Mis Documentos\Mis Trabajos R`, utilizaremos la función `setwd()` (*set working directory*) para indicar a R que es en ese directorio donde se realizarán las operaciones de entrada/salida de datos/resultados durante nuestra sesión de trabajo:

```
setwd("c:/usuario/Mis Documentos/Mis Trabajos R")
```

Es importante tener en cuenta que:

- Las barras que separan los distintos subdirectorios que componen la ruta son de la forma "/" y no la habitual "\" de Windows[8]. Alternativamente, se puede utilizar también doble barra "\\".
- El nombre del directorio debe especificarse entre comillas.

El directorio de trabajo puede modificarse en cualquier momento de nuestra sesión. Basta con ejecutar de nuevo la función `setwd()` indicando un nuevo directorio.

A.16.2. Lectura y escritura de datos en formato.csv

El formato `csv` (siglas de *Comma Separated Values*) es un formato estándar para el almacenamiento de datos. Básicamente un archivo.*csv* es un archivo ASCII (texto plano) en el que cada fila corresponde a un caso, esto es, a un sujeto u

[8]Esto es herencia del origen de R en sistemas Unix/Linux, en los que se usa la barra de la forma "/" para separar los directorios de una ruta. Obviamente, si hemos instalado R en Linux (o Mac), esta manera de escribir la ruta de un directorio es la natural.

objeto sobre el que se han observado varias variables; y en cada fila, los valores de las distintas variables se introducen separados por *punto y coma*[9]. Asimismo los valores de variables alfanuméricas deben escribirse entre comillas. Usualmente, en la primera fila de un archivo.csv se escriben los nombres de las variables que contiene, entre comillas y separados también por punto y coma.

La figura A.11 muestra el contenido de un archivo.csv que contiene los cinco primeros datos de edad, tiempo de uso del móvil y sexo ya vistos en la tabla A.1.

```
"edad"; "tiempo"; "sexo"
22; 14.21; “M”
34; 10.36; “H”
29; 11.89; “H”
25; 13.81; “M”
30; 12.03; “M”
```

Figura A.11: Contenido de un archivo en formato.csv

Si tuviésemos que utilizar un editor de texto (el *Notepad* de Windows, por ejemplo, o *gedit* en Linux) para escribir datos en formato *csv,* es obvio que no habríamos avanzado mucho respecto a introducir directamente los datos en la consola de R con la función de concatenación. Sin embargo muchos programas tipo hoja de cálculo -en particular *Excel* y *OpenOffice Calc*[10]- permiten guardar los datos en este formato, sin que tengamos que preocuparnos de escribir los puntos y comas o las comillas, ya que el programa lo hace por nosotros; asimismo si con uno de estos programas abrimos un archivo.csv, dicho archivo se abre como hoja de cálculo, con lo que su manipulación (introducción de datos, correcciones,...) resulta muy sencilla.

Para leer con R un archivo *csv* podemos utilizar la función `read.table()`. Si el fichero anterior se llama `usomovil.csv`, (y ya nos hemos situado en el directorio de trabajo mediante `setwd()` la sintaxis para su lectura[11] es:

[9]En realidad en la definición original de los ficheros csv, los valores de las variables se separan por comas. Ahora bien, en España y otros países la coma se reserva para separar los decimales por lo que, para evitar confusiones, en los ficheros csv “españoles” las variables deben separarse por punto y coma.

[10]Señalemos que estos dos programas, si bien son los más populares, no son los únicos que manejan hojas de cálculo y son capaces de generar ficheros csv. Podemos citar también *Gnumeric*, integrado en Gnome Office de Linux, *Numbers*, integrado en iWork de Apple, *Lotus 1-2-3,* integrado en Lotus SmartSuite, *Corel Quattro Pro,* integrado en WordPerfect, o *KSpread,* integrado en KOffice, también de Linux.

[11]Señalemos que a veces podremos encontrarnos con ficheros *csv* al estilo anglosajón, con

```
movildata <- read.table(file = "usomovil.csv", header = TRUE,
    sep = ";", dec = ",")
```

De esta forma, los datos del archivo se asignan a un *data.frame* con el nombre `movildata`. En esta función hemos utilizado cuatro argumentos:

- `file`: es el nombre del archivo. Debe escribirse entre comillas.
- `header`: es TRUE si los nombres de las variables están escritos en la cabecera (primera linea) del archivo. En caso contrario, se declara como FALSE.
- `sep`: es el símbolo que se usa para separar las variables. En nuestro caso es el punto y coma (";"). Debe ir entre comillas.
- `dec`: es el símbolo que se usa para separar los decimales. En nuestro caso es la coma (","). También va entre comillas.

Si como resultado de nuestro trabajo se ha generado un conjunto de datos llamado `results` y queremos guardarlo en el fichero `resultado.csv`, bastará con utilizar la sintaxis:

```
write.table(results, file = "resultado.csv", sep = ";", dec = ",",
    row.names = FALSE, col.names = TRUE)
```

Hemos especificado `col.names=TRUE` para que se escriban los nombres de las variables en la cabecera de las columnas, y `row.names=FALSE` para que *no* se escriban los nombres o números de fila.

A.16.3. Las funciones `attach()` y `detach()`

Una vez que los datos de un archivo han sido leidos y convertidos en un *data.frame*, su manipulación puede realizarse tal como hemos visto en la sección A.14. En particular, para acceder a los datos de una variable, su nombre ha de ir precedido por el nombre del *data.frame* y separado de éste por el símbolo $. Esto puede

los decimales separados por puntos y las variables por comas. En tal caso en nuestra función `read.table()` deberemos especificar `dec="."` y `sep=","`.

resultar tedioso si vamos a llevar a cabo un cierto número de procedimientos estadísticos utilizando estas variables y constantemente hemos de referirnos a ellas de esta forma. Para resolver este problema podemos usar la función `attach()`, que activa el *accceso directo* a las variables por su nombre.

Supongamos que, como se indicó en la sección anterior, hemos leido el archivo `usomovil.csv` y hemos asignado su contenido al *data.frame* `movildata`. Si ahora ejecutamos:

```
attach(movildata)
```

podremos acceder a las variables utilizando directamente su nombre `edad`, `tiempo` o `sexo` sin que sea necesario utilizar `movildata$edad`, `movildata$tiempo` o `movildata$sexo`. Una vez que hayamos terminado de analizar estos datos, podemos desactivar el acceso directo mediante:

```
detach(movildata)
```

Debe tenerse en cuenta que si estamos trabajando con varios *data.frames* que contienen variables de igual nombre, las variables a las que se tiene acceso directo son las que corresponden al último *data.frame* sobre el que se haya realizado el `attach()`. Así, por ejemplo, si tenemos un *data.frame* `mvdata1` con edad, tiempo y sexo de una muestra de sujetos, y otro *data.frame* `mvdata2` con datos de las mismas variables medidos en una segunda muestra, la sintaxis:

```
attach(mvdata1)
attach(mvdata2)
mean(tiempo)
```

mostraría el tiempo medio de uso del móvil por parte de los sujetos de la segunda muestra. Si ahora escribimos:

```
detach(mvdata2)
mean(tiempo)
```

la linea `detach(mvdata2)` desactiva el acceso directo a las variables de `mvdata2`; como no se ha realizado el `detach()` de `mvdata1`, la linea `mean(tiempo)` nos

mostrará el tiempo medio de uso del móvil de los sujetos de la muestra 1 que es la que ahora está "activa".

A.16.4. Lectura de datos en formato.sav de SPSS

Para leer archivos de datos generados por SPSS (usualmente con la extensión.*sav*), hemos de cargar la librería[12] `foreign`:

```
library(foreign)
```

y a continuación proceder a la lectura del archivo en cuestión. Si el archivo que queremos leer se llama `traffic.sav` escribiremos:

```
trafico <- read.spss(file = "traffic.sav", to.data.frame = TRUE)
```

De esta forma los datos del archivo se almacenan en `trafico`. Es recomendable utilizar el argumento `to.data.frame=TRUE` para que el objeto `trafico` en el que hemos puesto los datos tenga estructura de *data.frame*. En caso contrario su estructura sería la de una *lista*, lo que puede dar lugar a algunos efectos colaterales indeseados.

A.16.5. Lectura/Escritura de datos en formato.xls y.xlsx de Excel

Como ya hemos señalado en la sección A.16.2, Excel (en cualquiera de sus versiones) permite guardar las hojas de cálculo en formato csv, que R lee directamente a través de la función `read.table()`. Ahora bien, al guardar en este formato, Excel sólo guarda la hoja activa, con lo que si tenemos un Libro Excel con muchas hojas de cálculo, tendríamos que guardarlas por separado en múltiples archivos csv, lo que puede resultar trabajoso (amén de lo confuso que podría llegar a resultar un directorio lleno de archivos csv). Por ello en estos casos resulta ventajoso utilizar alguna librería que permita, desde dentro de R, acceder al Libro Excel y seleccionar directamente cada hoja en la que estamos interesados.

[12] Si no tenemos esta librería deberemos instalarla tal como hemos visto en la sección A.10.

Hay tres librerías que permiten que R lea directamente datos en los formatos que usa Excel. Aclaremos en primer lugar que a partir de Excel 2007 el formato por defecto de los archivos generados por este programa es.xlsx. En las versiones anteriores, el formato es.xls. Para las versiones anteriores a la 2007, las librerías que permiten acceder a los archivos.xls son `xlsReadWrite` y `RODBC`. Para acceder a archivos.xlsx de Excel 2007 y posteriores, la librería a utilizar es `xlsx`. No trataremos aquí esta última librería, ya que su funcionamiento depende de que el usuario tenga Java instalado en su equipo[13]. Si tenemos Excel 2007 instalado en nuestro sistema y queremos utilizar los archivos que genera para leerlos con R, tenemos la opción de guardarlos como.csv o como archivos.xls de Excel 2003.

A.16.5.1. La librería RODBC

Esta librería hace uso de ODBC (*Open DataBase Connectivity*), un controlador estándar de acceso a bases de datos desarrollado por Microsoft, que permite que otras aplicaciones accedan a datos almacenados en Access o Excel[14]. Si bien el uso de esta librería permite realizar incluso consultas complejas a bases de datos, aquí nos limitaremos a señalar como se utiliza para leer y guardar datos en Excel.

Supongamos que en el directorio `c:\usuario\Desktop` tenemos un archivo llamado `pruebaExcel.xls`, que contiene los datos de cinco variables en la primera hoja de cálculo[15]. Cada variable ocupa una columna y cada caso una fila. En la primera fila del archivo están los nombres de las variables. Para leer este archivo desde R utilizaremos la siguiente sintaxis:

```
library(RODBC)
setwd("c:/usuario/Desktop")
conex.in <- odbcConnectExcel(xls.file = "pruebaExcel.xls")
sqlTables(conex.in)
```

[13] En la web `www.java.com` pueden encontrarse enlaces para la descarga e instalación de java en Windows.

[14] En realidad ODBC tiene un uso más amplio ya que en general permite la conexión con sistemas de bases de datos que usen SQL (*Structured Query Language,* o *Lenguaje de Consulta Estructurado*), tales como MySQL, PostgreSQL, DB2, Oracle o SQLite.

[15] Cuando se guarda un archivo en Excel, por defecto se crea un "libro" con tres hojas de cálculo llamadas "Hoja1", "Hoja2" y "Hoja3". Obviamente el usuario puede cambiar estos nombres y eliminar o añadir hojas de cálculo a su libro. En este ejemplo asumimos que el archivo contiene tres hojas con sus nombres por defecto.

```
misdatos <- sqlFetch(conex.in, "Hoja1$")
odbcClose(conex.in)
```

Las dos primeras lineas simplemente especifican que cargamos la librería `RODBC` en memoria, y nos situamos en el directorio de trabajo `c:\usuario\Desktop` donde se encuentra el archivo que vamos a leer. La linea:

```
conex.in <- odbcConnectExcel(xls.file = "pruebaExcel.xls")
```

indica que creamos una "*conexión*" con el archivo `pruebaExcel.xls`, y que dicha conexión se llama `conex.in` (podemos usar cualquier otro nombre). Crear una *conexión* significa simplemente abrir un canal de comunicación entre R y el contenido del archivo Excel. La función `sqlTables(conex.in)` que encontramos en la linea siguiente, usa dicha conexión para mostrar la lista de hojas de cálculo que contiene el libro `pruebaExcel.xls`. La siguiente linea:

```
misdatos <- sqlFetch(conex.in, "Hoja1$")
```

"lee" la hoja 1 (la función `sqlTables()` utilizada en la linea anterior nos ha mostrado que el nombre de dicha hoja es `Hoja1$`) a través de la conexión `conex.in` y guarda los datos en el *data.frame* `misdatos`. Por último, si no vamos a leer nada más en el archivo Excel cerramos la conexión mediante la función `odbcClose(conex.in)`.

De manera similar, si deseamos guardar el *data.frame* `results` en el archivo Excel `resultados.xls`, deberemos abrir una conexión, guardar los datos y cerrar la conexión. Para ello utilizaremos la sintaxis:

```
conex.out <- odbcConnectExcel(xls.file = "OutExcel.xls", readOnly = FALSE)
sqlSave(conex.out, results, rownames = F)
odbcClose(conex.out)
```

Nótese que en este caso, al abrir la conexión mediante `odbcConnectExcel` hemos especificado la opción `readOnly=FALSE`. Esto indica a R que la conexión *no* es de sólo lectura, ya que vamos a escribir en el archivo.

A.16.5.2. La librería xlsReadWrite

Esta librería proporciona las funciones `read.xls()` y `write.xls()` que facilitan notablemente la lectura y escritura de archivos Excel. Parte de esta librería usa software que no es de código abierto, por lo que en realidad se compone de dos paquetes: uno que contiene el código abierto, y que R instala normalmente a través del menú *Paquetes* →*Instalar Paquetes,* y otro que se descarga[16] e instala automáticamente al ejecutar por primera vez la función:

```
library(xlsReadWrite)
```

Si queremos leer los datos que están en la primera hoja del archivo `pruebaExcel.xls` y asignarlos al *data.frame* `misdatos` la sintaxis a utilizar es simplemente:

```
misdatos <- read.xls(file = "pruebaExcel.xls", sheet = "Hoja1")
```

Asimismo, si queremos guardar el contenido del *data.frame* `misdatos` en el archivo Excel `resultado.xls` la sintaxis es:

```
write.xls(misdatos, file = "resultado.xls")
```

A.16.6. Lectura/Escritura de datos en formato.Rdata de R

Como muchos otros programas para el análisis estadístico, R dispone de su propio formato compacto para el almacenamiento de datos. Si, a lo largo de una sesión de trabajo, hemos generado los objetos `misdatos`, `regre`, `x` e `y` (pueden ser objetos de cualquier clase*: data.frames, listas, vectores,...*) podemos guardarlos en el archivo `SesionDeTrabajo.Rdata` mediante la sintaxis:

```
save(misdatos, regre, x, y, file = "SesionDeTrabajo.Rdata")
```

Este formato de archivo sólo resulta legible desde R. Si deseamos leer este fichero para trabajar nuevamente con los datos que contiene, la sintaxis a emplear es:

[16] directamente desde la web del autor de la librería.

```
load(file = "SesionDeTrabajo.Rdata")
```

Las variables que contiene este archivo se cargan directamente en el actual "entorno" o "ambiente" (*environment*) de trabajo, con lo cual ya podemos acceder a ellas por su nombre. Nótese la diferencia con la lectura de datos desde archivos .csv,.xls. o.sav: cuando se emplean `read.table()`, `read.spss()` o `read.xls()`, los datos normalmente se asignan a un *data.frame* que debemos *activar* mediante `attach()`. Sin embargo, cuando se usa `load(nombre-de-archivo.Rdata)` los *data.frames* que se guardaron mediante `save(nombre-de-archivo.Rdata)` se cargan directamente en memoria.

Conviene aclarar algo más este concepto; cuando arrancamos R, automáticamente se asigna un espacio en la memoria del ordenador en el que se irán almacenando los objetos que creemos durante nuestra sesión de trabajo; este espacio de memoria recibe el nombre de "*parent enviroment*" o *entorno padre*. R permite asignar otros espacios de memoria, distintos del entorno padre, en los que se pueden crear nuevos objetos. Esto tiene interés en aquellas ocasiones (normalmente vinculadas a la programación) en las que es necesario trabajar con dos variables *distintas* que compartan el *mismo nombre*. Así, si creamos un entorno llamado `nuevoEspacio` podríamos tener datos de una variable `x` en el *entorno padre* y datos (distintos) de otra variable también llamada `x` en el entorno `nuevoEspacio` sin que ambas se confundan.

Imaginemos ahora que hemos iniciado una nueva sesión de trabajo y que durante el curso de la misma hemos creado dos nuevas variables `x` e `y`. Si ahora mediante la función `load(file="SesionDeTrabajo.Rdata")` cargamos el archivo `SesionDeTrabajo.Rdata` creado anteriormente (y que, recordemos, contiene unas variables `x` e `y` distintas de las que acabamos de crear), nuestras nuevas `x` e `y` son borradas y sustituidas por las contenidas en `SesionDeTrabajo.Rdata`. Para evitar este efecto, es recomendable (salvo que estemos seguros de que no vamos a borrar nada) ejecutar el código siguiente:

```
nuevoEspacio = new.env()
load(file = "SesionDeTrabajo.Rdata", NuevoEspacio)
misObjetos = as.list(NuevoEspacio)
names(misObjetos)
```

En la primera linea, mediante la función `new.env()` hemos creado un nuevo *entorno de trabajo* llamado `NuevoEspacio` en memoria; la segunda linea lee los objetos que están en `SesionDeTrabajo.Rdata` y los coloca en el *entorno* `nuevoEspacio`; a continuación, la función `as.list()` convierte el contenido de este entorno en una *lista* que hemos llamado `MisObjetos`. Por último, la función `names()` nos muestra los nombres de los objetos contenidos en dicha lista. Para acceder a esos objetos, como siempre, habremos de preceder su nombre por el nombre de la lista y el símbolo `$` (`MisObjetos$x`, `MisObjetos$y`, etc).

A.17. Creación de funciones por parte del usuario: programación elemental en R.

En las secciones anteriores hemos podido ver como prácticamente todos los procedimientos en R se ejecutan mediante funciones. La gran versatilidad de R deriva del hecho de que cualquier usuario puede programar nuevas funciones, incrementando así paulatinamente las posibilidades del programa.

De una manera muy sucinta podemos definir una función como un fragmento de código que se encarga de realizar una o varias acciones. Estas acciones pueden ser de naturaleza muy variopinta: realizar cálculos, elaborar gráficos, mostrar mensajes, leer o guardar datos, interactuar con el sistema operativo,...

A.17.1. Estructura de una función.

La declaración (definición o construcción) de una función de R consta de cuatro partes:

- **Nombre:** La única condición que debe cumplir el nombre de una función para estar bien definido es empezar por una letra y no contener espacios ni símbolos reservados (“,+,-, etc.). En el nombre de una función sí que pueden utilizarse el punto (.) y el guión bajo (_). A continuación del nombre debe especificarse “`<- function`” (o de modo alternativo “`= function`”), para informar a R de que estamos definiendo una función.

- **Argumentos:** son los valores que recibe la función y con los que operará para obtener algún resultado. Una función puede carecer de argumentos, si la acción a realizar no depende de ningún valor que tenga que pasarle el usuario. Los argumentos deben especificarse entre paréntesis.
- **Código:** es el conjunto de instrucciones de R necesarias para que la función cumpla su cometido.
- **Resultado:** En el caso de que el objetivo de la función sea realizar algún tipo de cálculo, la función debe terminar devolviendo el resultado. El resultado se especifica entre paréntesis a continuación del comando `return`. No todas las funciones tienen por qué devolver un resultado (por ejemplo, las que se encargan sólo de imprimir un mensaje)

Ejemplo 1: Como ejemplo veamos como declarar una función que nos devuelva simplemente la suma de dos valores:

```
suma <- function(a, b) {
    total = a + b
    return(total)
}
```

Podemos identificar fácilmente en esta función las cuatro partes de las que hablábamos más arriba: el nombre de la función (`suma`) y su declaración como una función (`<-function`), seguido de sus argumentos, `(a,b)`. A continuación, entre llaves `{}`, está el *cuerpo* de la función que contiene el código a ejecutar (en este caso se crea una variable llamada `total` en la que se guarda el valor de `a+b`), y se devuelve el resultado mediante el comando `return(total)`.

Para ejecutar una función basta con escribir su nombre seguido de los valores de sus argumentos entre paréntesis. Por ejemplo, para sumar 3 y 5 con nuestra función bastaría con escribir:

```
suma(3, 5)
```

```
## [1] 8
```

Ejemplo 2: Vamos a construir ahora una función que nos permita leer fácilmente archivos excel utilizando la librería RODBC. Recordemos que leer archivos excel con esta librería resultaba algo complejo, puesto que es necesario "abrir" una conexión con el libro excel, leer los datos en la hoja adecuada y finalmente "cerrar" la conexión. Podemos incluir todos estos pasos en una función a la que sólo le pasemos como argumentos el nombre del libro excel y el nombre de la hoja donde están los datos:

```
read.xls <- function(libro, hoja) {
    require(RODBC)
    conex.in <- odbcConnectExcel(xls.file = libro)
    misdatos <- sqlFetch(conex.in, paste(hoja, "$"))
    odbcClose(conex.in)
    return(misdatos)
}
```

De esta forma, para leer la hoja 1 del archivo `prueba.xls` simplemente escribiríamos:

```
nuevosDatos <- read.xls("prueba.xls", "Hoja1")
```

Por cierto, notemos el uso de la función `require()`. Esta función es muy similar a `library()`. Recordemos que `library(nombre-del-paquete)` carga directamente en memoria el paquete cuyo nombre hemos especificado. La función `require(nombre-del-paquete)`, en cambio, sólo carga el paquete en caso de que no haya sido cargado previamente.

A.17.2. Un programa sencillo en R.

Supongamos que los tiempos entre llegadas de los paquetes de datos que acceden a un router siguen una distribución de Weibull con parámetro de escala λ_e y parámetro de forma κ_e. Asimismo, el tiempo que tarda el router en transmitir cada paquete, una vez que éste accede a la CPU del dispositivo, sigue también una distribución de Weibull de parámetros de escala y forma respectivos λ_s y κ_s. Se supone además que los tiempos entre llegadas son independientes entre sí, e

independientes de los tiempos de servicio que son, a su vez, también independientes entre sí. También se asume que el router dispone de memoria suficiente para almacenar todos los paquetes en espera de ser retransmitidos. En estas condiciones se desea simular el funcionamiento del sistema para estimar las características del tiempo de espera de los paquetes en el router, así como la evolución del tamaño de la cola de paquetes en espera.

Para llevar a cabo una simulación resulta siempre aconsejable descomponer las tareas a realizar en distintas funciones. De esta forma, cada función se puede probar por separado, lo que facilita la detección de posibles errores. No siempre es sencillo decidir qué funciones construir y como hacerlo, y el desarrollo de programas compactos y robustos requiere indiscutiblemente cierta práctica. En este caso, vamos a construir seis funciones:

1. Una función que se encargue de simular los tiempos entre llegadas y los tiempos de servicio. R dispone de la función `rweibull()` para simular valores independientes de la distribución de Weibull. Como el espacio de almacenamiento del buffer es tal que no se pierden paquetes[17], es evidente que una vez que un paquete llega al sistema:

   - Si el sistema está vacío, su instante de salida se calcula como su instante de llegada más su tiempo de servicio.
   - Si el sistema está ocupado, su instante de salida se calcula como el instante de salida del paquete anterior más su tiempo de servicio.

   El tiempo en el sistema es la diferencia entre el instante de salida y el instante de llegada; el tiempo en cola es la diferencia entre en tiempo en el sistema y el tiempo de servicio. La siguiente función se encarga de realizar todos estos cálculos.

```
tiemposSistema = function(n, kE, lambdaE, kS, lambdaS) {
    entreLlegadas = rweibull(n, kE, lambdaE)
    tllegada = cumsum(entreLlegadas)
```

[17] El hecho de que el buffer sea infinito nos permite también calcular de manera muy simple todos los instantes de llegada y salida, ya que no es preciso que la función revise continuamente si hay espacio o no para admitir una nueva llegada. Como no se pierden paquetes sabemos *a priori* que todos los paquetes que acceden al sistema son atendidos.

```
    tservicio = rweibull(n, kS, lambdaS)
    tsalida = numeric(n)
    tsalida[1] = tllegada[1] + tservicio[1]
    for (k in 2:n) tsalida[k] = max(tllegada[k], tsalida[k - 1]) + tservicio[k]
    tsistema = tsalida - tllegada
    tcola = tsistema - tservicio
    return(data.frame(entreLlegadas, tllegada, tsalida, tservicio, tsistema,
        tcola))
}
```

En este código hemos utilizado la función `cumsum()`, que toma un vector $(t, t_2, \ldots, t_n)$ y lo transforma en $(T_1, T_2, \ldots, T_n)$, siendo $T_k = \sum_{i=1}^{k} t_i$. De esta forma, si los t_i son los tiempos entre llegadas de los sucesivos paquetes, T_k es el instante de llegada del k-ésimo paquete. Hemos utilizado también la función `numeric(n)`, que crea un vector de ceros de dimensión n, y nos permite inicializar los tiempos de salida. Si bien en R no es estrictamente necesario inicializar la dimensión de los vectores a utilizar en una función, sí que resulta conveniente hacerlo: la memoria se reserva una única vez, lo que permite una utilización más eficiente de la memoria y una ejecución más rápida del código. Por último, hemos utllizado también un bucle `for` cuya sintaxis, como podemos ver, es muy simple.

2. Una función que, a partir de los instantes de llegada y salida de cada paquete se encargue de reconstruir la evolución del sistema. Para ello bastará con ordenar todos los instantes de entrada/salida de menor a mayor: cada vez que hay una entrada el número de paquetes en el sistema se incrementa en una unidad; cada vez que hay una salida, se decrementa en una unidad. Asimismo, el número de paquetes en cola será 0 si el sistema está vacío o contiene un único paquete y será $k - 1$ si el sistema contiene k paquetes. La siguiente función realiza estos cálculos y devuelve ordenada la lista de eventos de entrada/salida al sistema, consignando para cada evento cuantos paquetes hay en el sistema y cuántos en cola; identifica además a cada paquete con su número secuencial de acceso: cuando este valor es positivo, representa la llegada y cuando es negativo, la salida:

```
evolSistema = function(tllegada, tsalida) {

    n = length(tllegada)

    eventos = data.frame(rbind(cbind(tllegada, 1, 1:n), cbind(tsalida, -1, -(1:n))))

    names(eventos) = c("t", "inout", "cliente")

    o = order(eventos$t)

    eventos = eventos[o, ]

    eventos$nSistema = cumsum(eventos$inout)

    eventos$nCola = ifelse(eventos$nSistema <= 1, 0, eventos$nSistema - 1)

    row.names(eventos) = NULL

    eventos = rbind(c(0, 0, 0), eventos[, -2])

    return(eventos)

}
```

En este código hemos utilizado la función `length(tllegada)` que nos devuelve la longitud del vector `tllegada`; hemos utilizado también la función `order(eventos$t)` que nos devuelve las posiciones que ocupan los valores de `eventos$t` en una lista ordenada de mayor a menor; a continuación, el comando `eventos=eventos[o,]` coloca las fijas de la matriz `eventos` en ese orden. El comando `ifelse(condición,valor-si,valor-no)` construye un nuevo vector cuyos valores son `valor-si` o `valor-no` según que los valores del vector original cumplan o no la condición especificada.

3. Una función que lleva a cabo una estadística descriptiva simple (media y desviación típica) de los tiempos entre llegadas, tiempos de servicio, tiempos en el sistema y tiempos en cola.

```
describeTiempos = function(cola) {

    scola = subset(cola, sel = c(entreLlegadas, tservicio, tsistema, tcola))

    media = apply(scola, 2, mean)

    desvTip = apply(scola, 2, sd)

    return(rbind(media, desvTip))

}
```

Vemos en esta función el uso de la función `apply(scola,2,mean)`. De una manera muy simple "aplica" la función `mean()` a todas las columnas (el 2 indica columnas, 1 indicaría filas) del *data.frame* `scola`. Como las columnas de *este data.frame* son las variables que representan los distintos tiempos medidos, en un sólo comando hemos calculado la media de todos los tiempos de interés.

4. Una función que dibuja los histogramas de los tiempos de mayor interés en el estudio de la cola: tiempo entre llegadas, tiempos de servicio, tiempos en el sistema y tiempos en cola.

```
graficoTiempos = function(cola) {

    scola = subset(cola, sel = c(entreLlegadas, tservicio, tsistema, tcola))

    with(scola, {

        xlim1 = range(c(entreLlegadas, tservicio))

        xlim2 = range(c(tsistema, tcola))

        par(mfrow = c(2, 2))

        hist(entreLlegadas, xlim = xlim1)
```

```
        hist(tsistema, xlim = xlim2)

        hist(tservicio, xlim = xlim1)

        hist(tcola, xlim = xlim2)

    })

}
```

Nótese en esta función el uso del comando `with(scola,acciones)`. Este comando realiza las acciones especificadas leyendo las variables del *data.frame* `scola`

5. Una función que calcula la media y la desviación típica del tamaño de la cola:

```
describeCola = function(sistema) {

    cola = with(sistema, evolSistema(tllegada, tsalida))

    n = nrow(cola)

    medNumCola = with(cola, sum(diff(t) * nCola[-n])/t[n])

    mnc2 = with(cola, sum(diff(t) * nCola[-n]^2)/t[n])

    sdNumCola = sqrt((n/(n - 1)) * (mnc2 - medNumCola^2))

    desc = c(medNumCola, sdNumCola)

    names(desc) = c("medNumCola", "sdNumCola")

    return(desc)

}
```

6. Una función que realiza un gráfico de la evolución de la cola a lo largo del tiempo:

```
graficoCola = function(sistema) {

    cola = with(sistema, evolSistema(tllegada, tsalida))

    par(mfrow = c(1, 1))

    with(cola, plot(stepfun(t[-1], nCola), xlab = "Tiempo", ylab = "Numero en cola",
        main = "Tama<U+00F1>o de       cola", do.points = F))

}
```

Por último, lo que podríamos llamar nuestro "*programa principal*" se reduce a tres lineas: una para calcular los tiempos en el sistema, otra para realizar la estadística descriptiva de estos tiempos y una tercera para realizar la estadística descriptiva del tamaño de la cola:

```
sistema1 = tiemposSistema(1000, 23.5, 0.77, 3.6, 0.82)
describeTiempos(sistema1)

##         entreLlegadas tservicio tsistema  tcola
## media          0.7528    0.7431   2.0630 1.3200
## desvTip        0.0398    0.2243   0.9633 0.9489

describeCola(sistema1)

## medNumCola  sdNumCola
##      1.751      1.324
```

Si ejecutamos además las dos lineas siguientes:

```
graficoTiempos(sistema1)
graficoCola(sistema1)
```

obtenemos las figuras A.12 y A.13, que muestran los gráficos resultantes de esta simulación.

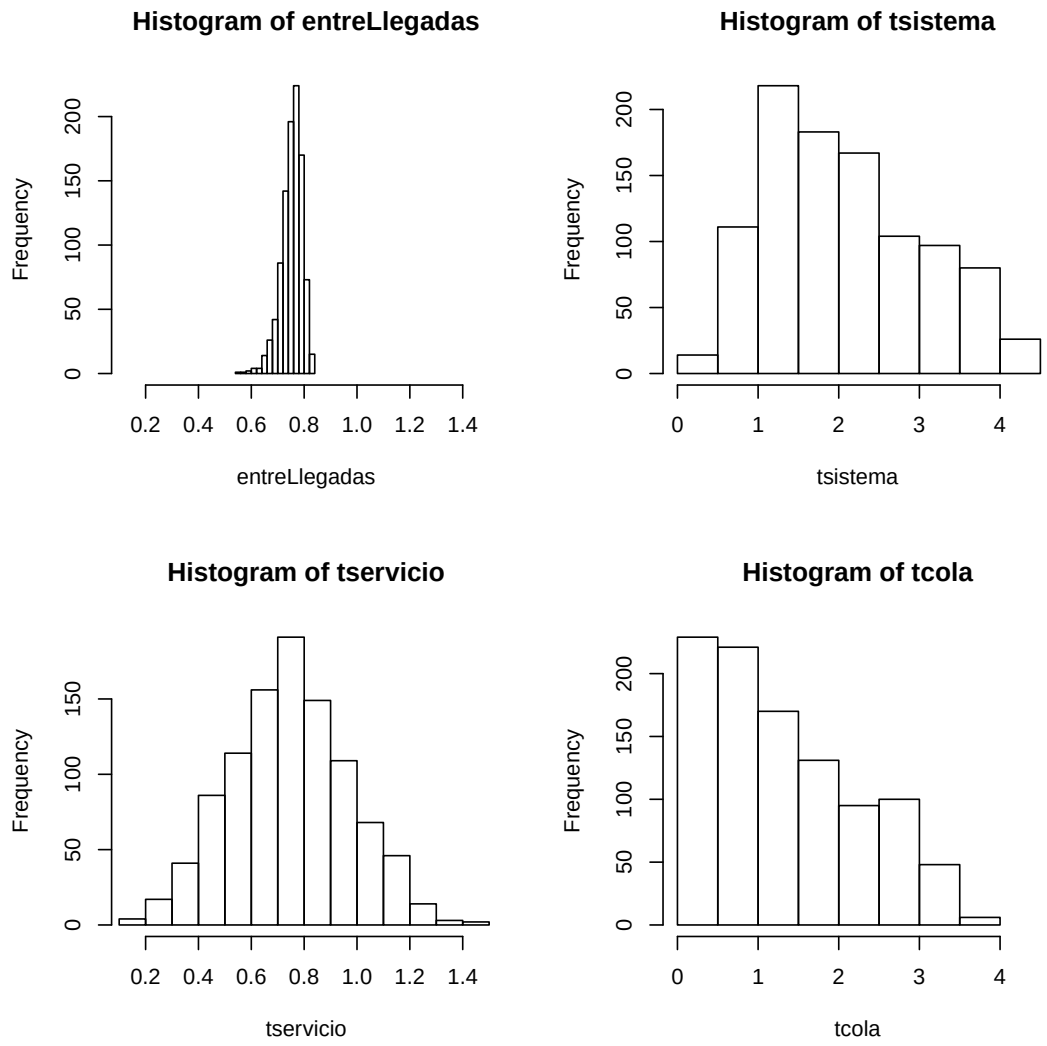


Figura A.12: Histograma de los distintos tiempos calculados en la simulación del router.

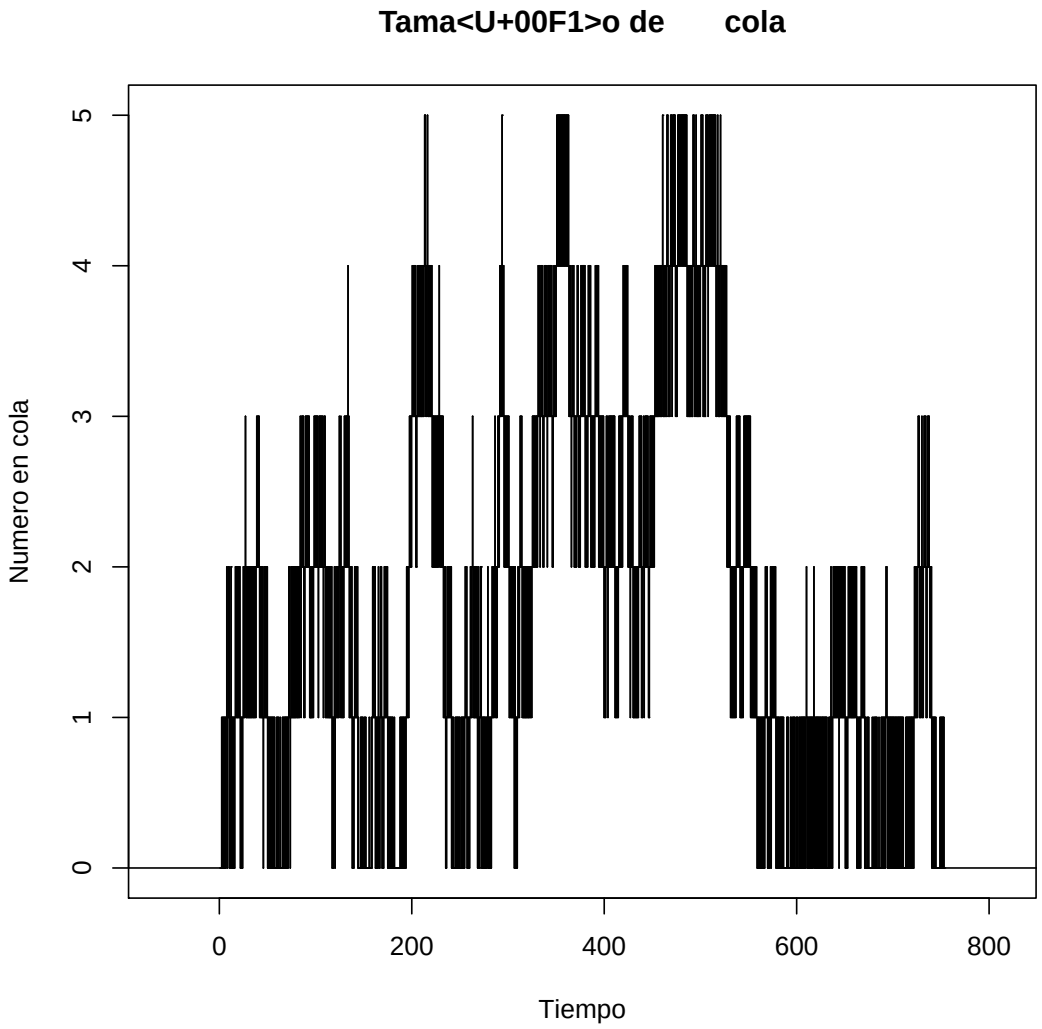


Figura A.13: Evolución del tamaño de la cola