Ejercicios de R

Curso: Introducción a la Estadística y Probabilidades CM-274

Lecturas Importantes

1. Una guía de aceleración del código de R.
   `http://www.noamross.net/blog/2013/4/25/faster-talk.html`.
2. Herramientas de depuración en R
   `http://www.noamross.net/blog/2013/4/18/r-debug-tools.html`.

Preguntas

1. Usa las funciones `matrix()`, `seq()` y `rep()` para construir la matrices de Henkel 5×5.

$$M = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 4 & 6 & 7 \\ 4 & 5 & 6 & 7 & 8 \\ 5 & 6 & 7 & 8 & 9 \end{bmatrix}$$

   Convierte el código en una función que puede ser usado para construir matrices de dimensión $n \times n$. Usa esa función para mostrar las salida de Matrices de Henkel de orden 10×10 y 12×12.

2. La matriz de Hilbert $n \times n$ tiene a los elementos (i, j) dados por $1/(i + j - 1)$.
   - Escribe una función que muestra una matriz de Hilbert $n \times n$ como salida para entero positivo n.
   - ¿ Son todas las matrices de Hilbert invertibles?.
   - Usa `solve()` y `qr.solve()` para calcular la inversa de las matrices Hilbert, por ejemplo, cuando $n = 10$.

3. Responde las siguientes preguntas
   (a) Escribe una pieza de código R para encontrar el `epsilon` de la máquina.
   (b) Usa *R* para reproducir la siguiente tabla de probabilidad de la distribución normal estándar.

```
> #<  codigo >

> p
   Lugar del primer decimal z
z    0       0.2    0.4     0.6     0.8
0 0.50000 0.57926 0.65542 0.72575 0.78814
1 0.84134 0.88493 0.91924 0.94520 0.96407
2 0.97725 0.98610 0.99180 0.99534 0.99744
3 0.99865 0.99931 0.99966 0.99984 0.99993
4 0.99997 0.99999 0.99999 1.00000 1.00000
```

(c) Usa la aproximación de Riemann

$$I_{LR}(a,b) = h\sum_{i=1}^{n}\phi(x_i)$$

para computar la integral

$$\int_0^2 \phi(x)dx$$

donde $\phi(\cdot)$ es la función densidad de la distribución normal estándar y asumamos conocida (usa la función `dnorm`). El código no debe contener bucles.

4. Un palillo se rompe al azar en 3 piezas. Escribe una función en R que, basada en simulación, calcula y devuelve la probabilidad de que las piezas puedan formar un triángulo.

5. Las siguientes expresiones pueden ser útiles en el siguiente problema: `dchisq(x,n)`, `integrate(f,a,b)$value`, `optimize(f, c(a,b))$ minimum`, `optimize(f, c(a,b), max =TRUE)$maximum`, `uniroot(f, c(a,b))$root`.

   (a) La probabilidad de que una variable aleatoria `chi-cuadrado` se encuentra entre dos valores, a y b, es la integral su densidad de probabilidad entre a y b. Escribe una función de R, que calcula esta probabilidad para la distribución chi-cuadrado con 5 grados de libertad. (Usa las funciones mencionadas).

   (b) Para una distribución `chi-cuadrado` de tres grados de libertad y usando las funciones mencionadas escribe una función que calcule la probabilidad que la variable `chi-cuadrado` de tres grados de libertad pertenece entre los valores `a` y `a +2` (como una función de a).

   (c) Usa las funciones, para calcular el intervalo de longitud 2 el cuál tiene la más alta probabilidad.

6. Considera el siguiente problema : Dado una matriz númerica X, determina el índice de la primera fila de números positivos que no contiene NA. Resuelve el problema usando `for` y la función `apply()`.

7. El código produce un `gráfico de dispersión`

```
> plot.new()
> plot.window(range(pressure$temperature),
+             range(pressure$pressure))
> box()
> axis(1)
> axis(2)
> points(pressure$temperature, pressure$pressure)
> mtext("temperatura", side=1, line=3)
> mtext("presion", side=2, line=3)
> mtext("Presion de vapor \ncomo una funcion de la Temperatura ",
+       side=3, line=1, font=2)
```

   (a) Describe completamente lo que cada llamada a la función en el código anterior hace,eso incluye una explicación del significado de cada argumento en las llamadas a funciones. Tu respuesta debe incluir una explicación de las diferentes regiones y sistemas de coordenadas creado por este código.

   (b) Describe cómo podría producir el mismo gráfico usando `viewports`, `layouts`, `units` en el sistema gráfico **grid**. Esta descripción debe incluir una mención de las funciones de `grid` que se requieren y lo que estas funciones hacen.

8. (a) Describe las importantes diferencias entre las estructuras fundamentales en R: vectores, matrices, arrays y listas. Usa ejemplos para demostrar las diferencias.

   (b) Explica la diferencia entre las funciones `rbind()`, `cbind()` y `merge()` para combinar estructuras de dos dimensiones en R. Da ejemplos en el uso de funciones.

9. Escribe un programa que usa la función `apply` para calcular las siguientes cantidades desde una matriz almacenada en la variable x

   - El máximo elemento de cada fila de x.
   - La media de los elementos positivos de cada fila de x
   - El primer elemento de cada fila que es mayor que los valores precedentes en la fila o `NA` si ese elemento no existe.

10. Para $n > 2$, la densidad `chi-cuadrado` tiene un máximo valor. Escribe código R, que usa la función `optimise` para localizar el máximo de la densidad para un valor $n > 2$.

11. Escribe una función llamada `norma` que calcula la norma Euclidea de un vector númerico. La norma Euclidea de un vector $x = (x_1, \ldots, x_n)$ es

$$\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2}.$$

    Usa operaciones vectorizadas para calcular la suma. Prueba esta función sobre los vectores $(0, 0, 0, 1)$ y $(2, 5, 2, 4)$ para verificar que el resultado de la función es correcto.

12. Esta pregunta, es acerca de `vectorización` y `reciclaje` en R.

    (a) Define por medio de una función que es `vectorización` en R.

    (b) Define por medio de una función que obedece la reglas de `recycling` en R.

    (c) Considera la función h definida por

$$h(x, y) = \sqrt{x^2 + y^2}$$

    Escribe una función en R, llamada `hypot`, con argumentos x e y que implementa una versión de h que es vectorizada y que cumple las reglas del `recycling`.

13. Escribimos dos funciones `primero` y `ultimo`, que extrae un número específico de elementos desde el inicio y el final de un vector (en el orden que aparecen en el vector). Las funciones deben ser llamadas como siguen

```
primero(x , k)
ultimo(x, k)
```

    donde x es el vector de valores que son extraidos y k especifica el número de elementos a extraer. Si el argumento k es omitido en una de las llamadas, debe tomar por valor por defecto 1.

    (a) Asumiendo que $k \leq$ `length(x)`, escribimos versiones (lo más simples) de las funciones dadas anteriormente.

    (b) Modifica las funciones (a) de manera que si $k >$ `length(x)` entonces estas funciones deberian retornar los valores en x.

    (c) Modifica las funciones (a) de manera que si $k >$ `length(x)` las funciones retornan los k valores, si no hay valores existentes estos deben ser `NA`.

14. Escribe una función para encontrar raices, usando la iteración de Newton

$$x_{i+1} = x_i - f(x_i)/f'(x_i)$$

    a partir de los siguientes pasos

    (a) Comienza con un intervalo y un punto inicial, que puede ser tomado como uno de los extremos del intervalo.

(b) En cada iteración, si el punto encontrado por la la fórmula de Newton se encuentra dentro del intervalom se utiliza para la siguiente iteración y para definir el siguiente intervalo.

(c) Si el punto se encuentra por la fórmula de Newton se encuentra fuera del intervalo, utiliza el punto de bisección.

15. Escribe código R que calcula el valor de la función

$$f(x,y) = \begin{cases} \sqrt{x} + \sin(y), & x \geq 0 \\ y + \cos(x), & \text{en otros casos.} \end{cases}$$

16. El modelo de `Regresión Lineal Simple` se ajusta a una respuesta y_i mediante una función lineal de una variable predictor x_i.

$$\widehat{y_i} = a + bx_i \text{ para } (i = 1, \ldots, n).$$

Por lo general, los `mínimos cuadrados` son utilizados para estimar los parámetros desconocidos a y b, pero a veces se utiliza la `menor desviación absoluta`. Esto requiere la elección de a y b a fin de minimizar

$$Q(a,b) = \sum_{i=1}^{n} |y_i - \widehat{y_i}|.$$

(a) Implementa una función que calcule $Q(a,b)$. Debes definir una función de un solo argumento el cúal es un vector cuyos primer elemento es a y el segundo elemento b.

(b) Explica como usa R la función `optim` para obtener el mejor ajuste de valores de a y b.

17. Los químicos a veces hacen tres mediciones paralelas de una cantidad y los procesan como sigue

- Primero, calcula la media de las tres observaciones.
- Se descarta la observación más alejada de la media.
- Finalmente, se reporta la media de las dos observaciones restantes.

Escribe una función que toma un único argumento x, que contiene tres valores númericos y retorna los valores descritos.

18. Sea una definición de la raiz cuadrada, definida por el método de Newton

```
> r =
+   function(x, eps = 1e-10) {
+     g = 1
+     while(abs(1 - g^2/x) > eps)
+       g = .5 * (g + x/g)
+     g
+   }
```

pero esta función trabaja para escalares, pero no cuando se pasa un vector

```
>r(c(1,2))
[1] 1
Warning message:
In while (abs(1 - g^2/x) > eps) g = 0.5 * (g + x/g) :
  the condition has length > 1 and only the first element will be used
```

Una manera de resolver este problema es dividir el cálculo en dos partes; una para calcular la raíz cuadrada para valores escalares y la otra usa un bucle sobre un vector

```
> s.r =
+   function(x, eps = 1e-10) {
+     g = 1
+     while(abs(1 - g^2/x) > eps)
+     g = .5 * (g + x/g)
+   g
+ }
```

```
>  r =
+   function(x, eps = 1e-10) {
+     ans = numeric(length(x))
+     for(i in seq(along = x))
+       ans[i] = s.r(x[i])
+   ans
+ }
```

Calculando ahora los valores

```
>  r(c(1,2))
```

La estrategia de usar bucles funciona, pero tienden a ser ineficientes debido a los cálculos que se llevan a cabo elemento a elemento. Una estrategia alternativa es llevar a cabo el cálculo de vectores en lugar de escalares. En este caso particular, podemos cambiar el cálculo para que funcione con vectores como sigue

- Cambia la inicialización de g de forma que sea un vector y no un escalar

  `g = rep(1, length = length(x))`

- Cambiar la prueba para que los cambios de **g** continuen hasta que todos los elementos de la respuesta se hayan calculado con una suficiente precisión.

  `while(any(abs(1 - g^2/x)) > eps))`

Esto continúa mejorando las aproximaciones de la raíz cuadrada hasta que todos ellos han alcanzado el nivel de exactitud. Lleva esto a la práctica implementando los cambios en `r` y probando la función resultante.

La estrategia de la sección anterior conlleva el cálculo de la raíz cuadrada incluso después de que las raíces cuadradas se han determinado para elementos de x. Estos cálculos adicionales pueden evitarse manteniendo un registro de los elementos de x cuyas raíces no se calculan con la precisión suficiente y sólo realizando los cálculos para esos elementos.

```
n.d = abs(1 - g^2/x)) > eps
```

Esto puede ser hecho como parte de la prueba

```
while(any((n.d = abs(1 - g^2/x))) > eps))
```

Dentro del bucle, los cambios pueden llevarse a cabo sólo en el subconjunto de **g** que necesita ser actualizado, es decir,`g[n.d]`. Las actualizaciones se llevan a cabo utilizando sólo los elementos correspondientes de **g** y x , es decir, `g[n.d]` y `x [n.d]`.