eman ta zabal zazu

Universidad del País Vasco

Euskal Herriko Unibertsitatea

TAREA 2

EJERCICIOS[1]

Puedes a tu elección trabajar en R, S-PLUS o MATLAB.

1. En clase vimos como la media aritmética de un subconjunto de n observaciones tomadas de entre N puede calcularse en función de la n-ésima observación y la media aritmética de las precedentes:

$$\overline{X}_n = \left(\frac{n-1}{n}\right) \overline{X}_{n-1} + \frac{X_n}{n} \tag{1}$$

   Demuestra la relación análoga:

$$S_n^2 = \left(\frac{n-1}{n}\right) S_{n-1}^2 + \frac{1}{n}(X_n - \overline{X}_n)^2.$$

2. Escribe una función que a partir de un vector de números proporcione su media aritmética y varianza muestral calculadas según las fórmulas anteriores.

3. Escribe una función que realice inversión de matrices cuadradas generales utilizando eliminación gaussiana. Para simplificar al máximo, puedes hacer uso de almacenamiento adicional: para invertir

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n-1,1} & a_{n-1,2} & \dots & a_{n-1,n} \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix}$$

   puedes recurrir a formar

$$(A|I_n) = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} & 1 & 0 & \dots & 0 \\ a_{21} & a_{22} & \dots & a_{2n} & 0 & 1 & \dots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ a_{n-1,1} & a_{n-1,2} & \dots & a_{n-1,n} & 0 & 0 & \dots & 0 \\ a_{n1} & a_{n2} & \dots & a_{nn} & 0 & 0 & \dots & 1 \end{pmatrix}$$

   y reducirla mediante operaciones por filas hasta que tenga la forma $(I_n|B)$, en cuyo momento tendrás $B = A^{-1}$. También para simplificar puedes prescindir de pivotado parcial o total. Compara los resultados devueltos por tu función con los de las funciones de librería de R (`solve`) o MATLAB (`inv`).

[1] Copia de estos enunciados y, en general, de todo el material escrito repartido en clase, está disponible en `http://etdx01.bs.ehu.es`.

4. Cuenta el número de operaciones aritméticas (sumas, multiplicaciones, divisiones) realizadas en el cómputo anterior. ¿Cuál es su orden?

5. Haciendo uso de la técnica de solución de sistemas de ecuaciones lineales de matriz triangular por retro-sustitución *(back substitution)* bosqueja un algoritmo de inversión de matrices triangulares más efectivo que el de eliminación gaussiana. Cuenta las operaciones que ahorras. *Optativo:* prográmalo y muestra su funcionamiento.

AYUDAS, SUGERENCIAS Y COMPLEMENTOS

1. Puede resultarte de interés en el cálculo del número de operaciones requeridas en la eliminación gaussiana la fórmula:

$$\sum_{k=1}^{n} k^2 = \frac{n}{6}(n+1)(2n+1)$$

   (ver [2], 1.2.3.1-2, pág. 34).

2. Matrices especiales o estructuradas pueden requerir bastante menos esfuerzo en la inversión que matrices generales. Por ejemplo, matrices triangulares, tridiagonales, o con estructura de Toeplitz. Estas últimas, son muy frecuentes en análisis de series temporales, y se caracterizan por tener iguales sus infra- y supra-diagonales:

$$\begin{pmatrix} \rho_0 & \rho_1 & \rho_2 & \cdots & \rho_q \\ \rho_1 & \rho_0 & \rho_1 & \cdots & \rho_{q-1} \\ \rho_2 & \rho_1 & \rho_0 & \cdots & \rho_{q-2} \\ \vdots & \vdots & \vdots & \ldots & \vdots \\ \rho_{q-1} & \rho_{q-2} & \rho_{q-3} & \cdots & \rho_1 \\ \rho_q & \rho_{q-1} & \rho_{q-2} & \cdots & \rho_0 \end{pmatrix}$$

   Pueden invertirse con $O(q^2)$ operaciones (algoritmo de Levinson).

3. *Incluso* para matrices ordinarias hay algoritmos de orden asintóticamente mejor que el de eliminación gaussiana, como el de Schönhague-Strassen (ver [1], Cap. 6), que invierte matrices en $O(n^{2,81})$ operaciones. Acontece sin embargo que, para lograr dicho orden asintótico, llevar "las cuentas" supone una carga tal que anula las ventajas de algoritmo para n realista.

4. Ahora que lo sabes hacer. . . te interesará utilizar las funciones de librería del programa que utilices (R, MatLab, etc.) Aunque escribieras una función de inversión, descomposición LU, etc. técnicamente muy buena, probablemente no llegarías a acercarte a la eficiencia de las funciones de librería de los programas citados. Uno de los motivos es que funciones como éstas, que son bloques constructivos de otras muchas, acostumbran a estar escritas en un lenguaje compilado (Fortran, C, C++ o incluso ensamblador). En un lenguaje *interpretado* (R y MatLab lo son) un intérprete "traduce" el código que tu escribes *cada vez* que se ejecuta. En un lenguaje *compilado* esta traducción se hace de una vez para siempre obteniéndose un ejecutable en código directamente explotable por la máquina, y por ende mucho más rápido de ejecutar.

Referencias

[1] A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, Ma., 1974.

[2] A. Jeffrey. *Handbook of Mathematical Formulas and Integrals*. Academic Press, 1995.

[3] R.A. Thisted. *Elements of Statistical Computing*. Chapman & Hall, New York, 1988.