Examen Final Análisis Numérico

Diego Abraham Olvera Mendoz

05 de Enero del 2019

Programas Los programas presentados para los ejercicios 1 al 4 en este examen fueron escritos tratando de seguir los estándares de FORTRAN 77 en la medida de lo posible. El último programa es un script que puede ser ejecutado en software Octave.

1. La distribución normal es muy empleada en estadística y está dada por

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

(a) La función de error se define por

$$\operatorname{erf}\left(x\right)=\frac{2}{\sqrt{\pi}}\int_{0}^{x}e^{-t^{2}}dt$$

Demuestre que la función de distribución acumulativa para la distribución normal

$$F\left(x;\mu,\sigma\right)=\int_{-\infty}^{x}f\left(t;\mu,\sigma\right)dt$$

se puede escribir como

$$F\left(x;\mu,\sigma\right)=\frac{1+\operatorname{erf}\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)}{2}.$$

Demostración
Sustituyendo tenemos

$$F\left(x;\mu,\sigma\right)=\int_{-\infty}^{x}f\left(t;\mu,\sigma\right)dt=\int_{-\infty}^{x}\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{\left(t-\mu\right)^{2}}{2\sigma^{2}}\right)dt=\frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{x}\exp\left(-\left(\frac{t-\mu}{2\sigma}\right)^{2}\right)dt.$$

Realizando un cambio de variable tenemos que $w=\frac{t-\mu}{\sqrt{2}\sigma}$ y $dw=\frac{1}{\sqrt{2}\sigma}dt$.
Despejando $dt=\sqrt{2}\sigma dw$.

Si $t \to -\infty$ entonces el numerador es un número negativo muy grande y $w \to -\infty$.
Si $t = x$ entonces $w = \frac{x-\mu}{\sqrt{2}\sigma}$.

$$F(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} \int\limits_{-\infty}^{\frac{x-\mu}{\sqrt{2}\sigma}} \exp\left(-(w)^2\right)\sqrt{2}\sigma dw$$

$$= \frac{1}{\sqrt{\pi}} \int\limits_{-\infty}^{\frac{x-\mu}{\sqrt{2}\sigma}} \exp\left(-(w)^2\right) dw$$

$$= \frac{1}{\sqrt{\pi}} \int\limits_{-\infty}^{0} \exp\left(-(w)^2\right) dw + \frac{1}{\sqrt{\pi}} \int\limits_{0}^{\frac{x-\mu}{\sqrt{2}\sigma}} \exp\left(-(t)^2\right) dt$$

Resolviendo primero $\frac{1}{\sqrt{\pi}} \int_{-\infty}^{0} \exp\left(-(w)^2\right) dw$ haciendo $w = \frac{z}{\sqrt{2}}$, $dw = \frac{dz}{\sqrt{2}}$,$w^2 = \frac{z^2}{2}$ sustituimos para resolver

$$\frac{1}{\sqrt{\pi}} \int\limits_{-\infty}^{0} \exp\left(-(w)^2\right) dw = \frac{1}{\sqrt{2\pi}} \int\limits_{-\infty}^{0} \exp\left(-\frac{z^2}{2}\right) dz$$

$$= \frac{1}{2}\left(\frac{2}{\sqrt{2\pi}} \int\limits_{-\infty}^{0} \exp\left(-\frac{z^2}{2}\right) dz\right)$$

La integral puede escribirse como $2\int_{-\infty}^{0} \exp\left(-\frac{z^2}{2}\right) dz = \int_{-\infty}^{\infty} \exp\left(-\frac{z^2}{2}\right) dz$ y el resultado es $\int_{-\infty}^{\infty} \exp\left(-\frac{z^2}{2}\right) dz = \sqrt{2\pi}$.
Entonces sustituyendo en el lado izquierdo tenemos

$$\frac{1}{2}\left(\frac{2}{\sqrt{2\pi}} \int\limits_{-\infty}^{0} \exp\left(-\frac{z^2}{2}\right) dz\right) = \frac{1}{2}\frac{1}{\sqrt{2\pi}}\sqrt{2\pi} = \frac{1}{2}.$$

Continuando con la integral del lado derecho

$$= \frac{1}{2} + \frac{1}{2}\left(\frac{2}{\sqrt{\pi}} \int\limits_{0}^{\frac{x-\mu}{\sqrt{2}\sigma}} \exp\left(-(t)^2\right) dt\right)$$

$$= \frac{1}{2} + \frac{1}{2}\left(\operatorname{erf}\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)\right)$$

$$= \frac{1+\operatorname{erf}\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)}{2} = F\left(x;\mu,\sigma\right)$$

(b) El nombre del programa es *NCDF.f.* Pueden usarse los siguientes valores $x = 1.644853$, $\mu = 0$, $\sigma = 1$ y obtenemos $F\left(x;\mu,\sigma\right) = 0.9499$.

(c) El nombre del programa es *INVNCDF.f.* Pueden usarse los valores obtenidos en el inciso anterior y obtendremos x ingresando $p = F\left(x;\mu,\sigma\right) = 0.9499$ cuando el programa lo solicite.

2. Demuestre que las dos raíces x_1y x_2 de $ax^2+bx+c=0$ están dadas por

$$x_1 = \frac{-b \pm \sqrt{b^2-4ac}}{2a}, \qquad x_2 = \frac{c}{ax_1}$$

Demostración
Si x_2 es solución de $ax^2+bx+c=0$ entoces

$$a\left(\frac{c}{ax_1}\right)^2 + b\left(\frac{c}{ax_1}\right) + c = \frac{ac^2}{a^2x_1^2} + \frac{bc}{ax_1} + c = \frac{c^2+bcx_1+acx_1^2}{ax_1^2} = 0$$

con $x_1 = \frac{-b+sgn(b)\sqrt{b^2-4ac}}{2a}$ la ecuacion anterior se convierte en:

$$\frac{c^2 + bc\left(\frac{-b+sgn(b)\sqrt{b^2-4ac}}{2a}\right) + ac\left(\frac{-b+sgn(b)\sqrt{b^2-4ac}}{2a}\right)^2}{a\left(\frac{-b+sgn(b)\sqrt{b^2-4ac}}{2a}\right)^2}$$

Desarrollando las potencias y productos tenemos:

$$\frac{c^2 + \left(\frac{-b^2c+sgn(b)bc\sqrt{b^2-4ac}}{2a}\right) + ac\left(\frac{b^2-2sgn(b)b\sqrt{b^2-4ac}+sgn(b)b^2-4ac}{(2a)^2}\right)}{a\left(\frac{b^2-2sgn(b)b\sqrt{b^2-4ac}+sgn(b)b^2-4ac}{(2a)^2}\right)}$$

$$= \frac{c^2 - \frac{b^2c}{2a} + \frac{sgn(b)bc\sqrt{b^2-4ac}}{2a} + \frac{ab^2c}{4a^2} - \frac{sgn(b)2abc\sqrt{b^2-4ac}}{4a^2} + \frac{sgn(b)ab^2c}{4a^2} - \frac{4a^2c^2}{4a^2}}{a\left(\frac{b^2-2sgn(b)b\sqrt{b^2-4ac}+sgn(b)b^2-4ac}{(2a)^2}\right)}$$

$$= \frac{c^2 - c^2 + \frac{b^2c}{4a} + \frac{b^2c}{4a} - \frac{b^2c}{2a} + \frac{sgn(b)bc\sqrt{b^2-4ac}}{2a} - \frac{sgn(b)2bc\sqrt{b^2-4ac}}{4a}}{a\left(\frac{b^2-2sgn(b)b\sqrt{b^2-4ac}+sgn(b)b^2-4ac}{(2a)^2}\right)}$$

$$= \frac{0}{a\left(\frac{-b+sgn(b)\sqrt{b^2-4ac}}{2a}\right)^2} = 0.$$

Estas fórmulas se emplean para evitar errores numéricos cuando la diferencia entre $|b|$ y el discriminante $\sqrt{b^2-4ac}$ es muy pequeña, por lo que

$$\frac{-b+sgn(b)\sqrt{b^2-4ac}}{2a} \neq 0.$$

El programa que hace uso de estas formulas es *QuadraticEq.f*. La tolerancia que debe cumplir la diferencia para hacer uso de las formulas expustas es de 0.01.Ingresando los coeficientes $a = \frac{1}{10} \quad , b = 5 \quad , c = \frac{1}{100}$ el programa utiliza la formula alterntiva para calcular x_2. El resultado es

$$x_1 = -49.9979... \qquad \text{y} \qquad x_2 = -2.0000...E - 03.$$

El programa también es capaz de calcular raíces complejas utilizando la formlua cuadrática $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$ y no la formula alternativa.

3. El algoritmo de Gauss-Jordan utilizado en el programa *GaussJ.f* es el propuesto en *Numerical Recipies in Fortran 77.*
Ingresando el sistema

$$A\mathbf{x} = \mathbf{b}$$

$$\begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}$$

El programa solicita la dimensión n de la matriz A cuadrada y cada elemento a_{ij}. El número de vectores columna solución para este ejercicio es 1 y los elementos son solicitados como b_{i1}. La solución al sistema que muestra el programa es:

$$\begin{array}{l} x_1 = 1 \\ x_2 = 1 \\ x_3 = 0 \end{array}.$$

4. Las ecuaciones del algoritmo de Crout factorizan una matriz cuadrada

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1j} \\ a_{21} & a_{22} & \dots & a_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ a_{i1} & a_{i2} & \dots & a_{ij} \end{bmatrix}, \qquad i = j = 1, 2, \cdots, N$$

en dos matrices $A = LU$ donde L es una matriz triangular inferior y U es una matriz triangular superior.
Si $N = 4$ tenemos

$$L = \begin{bmatrix} \alpha_{11} & 0 & 0 & 0 \\ \alpha_{21} & \alpha_{22} & 0 & 0 \\ \alpha_{31} & \alpha_{32} & \alpha_{33} & 0 \\ \alpha_{41} & \alpha_{42} & \alpha_{43} & \alpha_{44} \end{bmatrix} \text{ y } U = \begin{bmatrix} \beta_{11} & \beta_{12} & \beta_{13} & \beta_{14} \\ 0 & \beta_{22} & \beta_{23} & \beta_{24} \\ 0 & 0 & \beta_{33} & \beta_{34} \\ 0 & 0 & 0 & \beta_{44} \end{bmatrix}.$$

La factorización LU de Crout nos permite hacer $\alpha_{ii} = 1,\ i = 1, 2, \ldots N$. Al tener los valores de los elementos de la diagonal de L podemos entonces resolver el sistema de N^2 ecuaciones para factorizar

$$A = LU = \begin{bmatrix} \beta_{11} & \beta_{12} & \beta_{13} & \beta_{14} \\ \alpha_{21} & \beta_{22} & \beta_{23} & \beta_{24} \\ \alpha_{31} & \alpha_{32} & \beta_{33} & \beta_{34} \\ \alpha_{41} & \alpha_{42} & \alpha_{43} & \beta_{44} \end{bmatrix}$$

haciendo los siguientes pasos:

- Asignar $\alpha_{ii} = 1, \qquad i = 1, 2, \ldots N$.
- Para cada $j = 1, 2, 3, \ldots, N$ hacer los siguientes procedimientos.
  - Primero, para $i = 1, 2, \ldots, j$ calculamos

    $$\beta_{ij} = a_{ij} - \sum_{k=1}^{i-1} \alpha_{ik}\beta_{kj}.$$

    Cuando $i = 1$ el termino de la sumatoria es igual a 0.
  - Segundo, para $i = j+1, j+2, \ldots, N$ resolvemos

    $$\alpha_{ij} = \frac{1}{\beta_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} \alpha_{ik}\beta_{kj} \right).$$

  - Realizar ambos cálculos antes de avanzar al siguiente valor de j.

El programa que utiliza estas formulas para factorizar A y resolver el sistema $A\mathbf{x} = \mathbf{b}$ es *CROUT.f*.
El programa realiza un ejemplo con una matriz

$$A = \begin{bmatrix} 1 & 1 & 0 & 3 \\ 2 & 1 & -1 & 1 \\ 3 & -1 & -1 & 2 \\ -1 & 2 & 3 & -1 \end{bmatrix}.$$

Utilizando la subrutina *ludcmp* obtenemos

$$A = LU = \begin{bmatrix} 1 & 1 & 0 & 3 \\ 2 & -1 & -1 & -5 \\ 3 & 4 & 3 & 13 \\ -1 & -3 & 0 & -13 \end{bmatrix}.$$

Resolvemos el sistema haciendo substitucion hacia atras a U y substitución hacia adelante a L en la subrutina *lubksb* que devuelve

$$\mathbf{x} = \begin{bmatrix} 3 \\ -1 \\ 0 \\ 2 \end{bmatrix}.$$

5. Haga un programa que resuelva sistemas de ecuaciones sobredeterminados. Consideremos la matriz rectangular

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1j} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2j} \\ \vdots & \vdots & \ddots & \dots & \vdots \\ a_{i1} & a_{i2} & a_{i3} & \dots & a_{ij} \end{bmatrix}, \quad \text{con} \quad \begin{array}{c} i = 1, 2, \dots, m \\ j = 1, 2, \dots n \end{array}$$

Se dice que un sistema $A\mathbf{x} = \mathbf{b}$ es sobredetermindo cuando $m > n$. Es decir cuando se tienen más ecuaciones que incógnitas. Para este tipo de sistemas no existe una solución, lo que se busca es un vector solución que de la mejor aproximacion para cada una de las ecuaciones del sistema $A\mathbf{x} = \mathbf{b}$. El problema lineal de mínimos cuadrados se acerca a la optimizacion encontrando un vector x con la mínima norma del vector residual $\mathbf{r} = \mathbf{b} - A\mathbf{x}$.
La solución de mínimos cuadrados de $A\mathbf{x} = \mathbf{b}$ es el vector $\mathbf{x}$ que hace de $\|\mathbf{r}\|_2 = \|\mathbf{b} - A\mathbf{x}\|_2$ un mínimo

$$\min_{\mathbf{x} \epsilon \mathbb{R}^{\ltimes}} \|\mathbf{b} - A\mathbf{x}\|^2$$

Realizando la factorización QR de A, la solucion al sistema se obtiene resolviendo

$$R\mathbf{x} = Q^t\mathbf{b}.$$

No fue posible realizar un programa en FORTRAN para solucionar sistemas sobredeterminados. El principal problema fue la factorización QR utilizando la transformacion de Householder. Se optó por este método de ortonnormalización ya que el método de Gram-Schmidt no es recomendable para métodos númericos por problemas de acumulación de redondeo.

Se presenta un script *SimpleMinsqrtWithQR.m* de Octave para la solucion de sistemas sobredeterminados. Se hace uso de la función *qr(A)* que utiliza subrutinas de LAPACK para calcular la factorización QR.
Como ejemplo resolvemos el sistema $A\mathbf{x} = \mathbf{b}$con

$$A = \begin{bmatrix} 3 & -1 \\ 4 & 2 \\ 0 & 1 \end{bmatrix} \quad \text{y} \quad \mathbf{b} = \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}$$

La factorización $A = QR$ es

$$Q = \begin{bmatrix} -0.60000 & 0.71554 & 0.35777 \\ -0.80000 & -0.53666 & -0.26833 \\ 0.00000 & -0.44721 & 0.89443 \end{bmatrix} \quad R = \begin{bmatrix} -5.00000 & -1.00000 \\ 0.00000 & -2.23607 \\ 0.00000 & 0.00000 \end{bmatrix}$$

$$Q^t\mathbf{b} = \begin{bmatrix} -1.60000 \\ -1.52053 \\ 0.35777 \end{bmatrix}$$

El vector óptimo a ser solución se obtiene realizando una division izquierda en el script

$$\mathbf{x} = R \setminus Q^t\mathbf{b}.$$

En Octave, si el sistema no es cuadrado, o si la matriz de coeficientes es singular, la division izquierda calcula una solución de norma mínima

$$\mathbf{x} = \begin{bmatrix} 0.18400 \\ 0.68000 \end{bmatrix}.$$