Implementación en FORTRAN 77 del método de ortonormalización de Gram–Schmidt en $\mathbb{R}^n$

Pablo Santamaría

v0.2 (Marzo 2008)

Sea S el subespacio de $\mathbb{R}^n$ generado por el conjunto de m vectores linealmente independientes $\{v_1, v_2, \ldots, v_m\}$ $(m \leq n)$. El método de ortonormalización de Gram-Schmidt construye una base *ortornomal* $\{w_1, w_2, \ldots, w_m\}$ para S a partir de la base anterior según el procedimiento:

$$w_1 = \frac{v_1}{\|v_1\|},$$
$$w_j = \frac{v_j - \sum_{l=1}^{j-1} \langle w_l | v_j \rangle w_l}{\left\| v_j - \sum_{l=1}^{j-1} \langle w_l | v_j \rangle w_l \right\|} \qquad j = 2, \ldots, m,$$

siendo $\langle v|u\rangle = v^t u$ el producto interno canónico de $\mathbb{R}^n$ y $\|v\| = \sqrt{\langle v|v\rangle}$ la norma asociada. Una implementación algorítmica de este procedimiento puede plantearse como sigue.

Dados $v_1, v_2, \ldots, v_m$
Para $j = 1, 2, \ldots, m$
    Tomar $w_j = v_j$
    Para $l = 1, 2 \ldots, j-1$
        Calcular $r_{lj} = \langle w_l | v_j \rangle$
        Tomar $w_j = w_j - r_{lj} w_l$
    Calcular $r_{jj} = \|w_j\|$
    Tomar $w_j = \frac{w_j}{r_{jj}}$

Sin embargo, esta implementación directa del método no constituye una buena implementación *numérica*, por cuanto al utilizar aritmética de punto flotante los vectores sucesivos calculados se apartan de la ortogonalidad debido a los errores de redondeo involucrados. Una implementación matemáticamente equivalente, pero numéricamente más estable procede como sigue

Dados $v_1, v_2, \ldots, v_m$
Para $j = 1, 2, \ldots, m$
    Tomar $w_j = v_j$
    Para $l = 1, 2 \ldots, j-1$
        Calcular $r_{lj} = \langle w_l | w_j \rangle$
        Tomar $w_j = w_j - r_{lj} w_l$
    Calcular $r_{jj} = \|w_j\|$
    Tomar $w_j = \frac{w_j}{r_{jj}}$

Para la implementación computacional de este procedimiento disponemos los m vectores v_j, cada uno de n componentes, como las columnas de una matriz A cuyas dimensiones lógicas son, pues, $n \times m$. Como es usual, la dimensión física de la matriz será $n_{max} \times n_{max}$. La siguiente subrutina implementa el método. Obsérvese que los vectores ortonormalizados vuelven en la misma matriz A, cuyos valores originales, son pues, destruidos.

```
      SUBROUTINE GRAMSCHMIDT(A,N,M,NMAX)

      IMPLICIT NONE
      INTEGER NMAX,N,M
      DOUBLE PRECISION A(NMAX,*)
      INTEGER I,J,L
      DOUBLE PRECISION PI

      DO J=1,M
          DO L=1,J-1
*             ----------------------------
*             CALCULAR PI = <W_L|W_J>
*             ----------------------------
              PI = 0.0D0
              DO I=1,N
                  PI = PI + A(I,L)*A(I,J)
              END DO
*             ----------------------------
*             TOMAR W_J = W_J - PI*W_J
*             ----------------------------
              DO I=1,N
                  A(I,J) = A(I,J)-PI*A(I,L)
              END DO
          END DO
*         ----------------------------
*         CALCULAR NORMA DE W_J
*         ----------------------------
          PI = 0.0D0
          DO I=1,N
              PI = PI + A(I,J)**2
          END DO
          PI = SQRT(PI)
*         ----------------------------
*         NORMALIZAR W_J
*         ----------------------------
          DO I=1,N
              A(I,J) = A(I,J)/PI
          END DO
      END DO
      RETURN
      END
```

Espero que este apunte les haya resultado útil, y como siempre, cualquier duda o sugerencia envíenla por *e-mail* a `pablo@fcaglp.unlp.edu.ar`.