Programación: producto diádico

Objetivos. Programar una función que calcule el producto diádico de dos vectores dados.

Requisitos. Programación de funciones, ciclos `for`, entradas de vectores.

1. Definición del producto diádico. Dados dos vectores $a \in \mathbb{R}^m$, $b \in \mathbb{R}^n$, su *producto diádico* $a \otimes b$ se define de la siguiente manera:

$$a \otimes b = \Big[a_j b_k\Big]_{j,k=1}^{m,n}.$$

En otras palabras, $a \otimes b$ es una matriz $m \times n$, cuya entrada (j, k) es

$$(a \otimes b)_{j,k} = a_j b_k.$$

En inglés su usan las palabras *outer product*, *dyadic product*, *tensorial product*.

2. Ejemplo. Escribir el producto diádico $a \otimes b$ de los vectores

$$a = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}, \qquad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}.$$

Mostrar que el resultado coincide con el producto matricial

$$ab^\top = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \begin{bmatrix} b_1 & b_2 \end{bmatrix}.$$

3. Problema: producto diádico. En algún lenguaje de programación escribir una función que calcule la matriz $ab^\top$, donde $a \in \mathbb{R}^m$ y $b \in \mathbb{R}^n$ son dos vectores dados.

Entrada: $a \in \mathbb{R}^m$, $b \in \mathbb{R}^n$.

Salida: la matriz $ab^\top$.

Solución en el lenguaje Matlab (guardar en el archivo `outerproduct.m`):

```
function [r] = outerproduct(a, b),
   m = length(a);
   n = length(b);
   r = zeros(m, n);
   for j = 1 : m,
      for k = 1 : n,
         r(j, k) = a(j) * b(k);
      end
   end
end
```

4. Análisis de complejidad. Calcular el número de operaciones de multiplicación en el algoritmo anterior, suponiendo que $m = n$. La respuesta es un polinomio de la variable n.

5. Pruebas con vectores pequeños. En algún lenguaje de programación escribir un programa que llame a la función del ejercicio anterior, aplicándola a los siguientes datos:

$$a = \begin{bmatrix} 3 \\ -4 \\ 1 \end{bmatrix}, \qquad b = \begin{bmatrix} 2 \\ 5 \end{bmatrix}.$$

Solución en el lenguaje de Matlab (guardar en el archivo `testouterproduct1.m`):

```
function [] = testouterproduct1(),
   a = [3; -4; 1]; b = [2; 5];
   display(outerproduct(a, b));
   display(a * b');
end
```

6. Pruebas con vectores grandes aleatorios. En algún lenguaje de programación escribir un programa que genere vectores de tamaños grandes: $n = 10$, $n = 10^2$, $n = 10^3$, aplique a estos vectores la función del Ejercicio 3 y mida el tiempo de ejecución. Solución en el lenguaje de Matlab (guardar en el archivo `testouterproduct2.m`):

```
function [] = testouterproduct2(),
   for n = [512, 1024, 2048],
      display(n);
      a = rand(n, 1);
      b = rand(n, 1);
      t1 = cputime();
      p = outerproduct(a, b);
      t2 = cputime();
      display(t2 - t1);
   end
end
```

¿Cómo se cambia el tiempo de ejecución al multiplicar n por 2?