Operaciones con el producto interno en GNU Octave

Objetivos. Aprender a usar operaciones del lenguaje GNU Octave que permiten trabajar con el producto interno, calcular la proyección ortogonal de un vector al subespacio generado por una lista ortogonal, ortogonalizar una lista de vectores.

Requisitos. GNU Octave instalado, operaciones con matrices en GNU Octave.

Producto punto, listas ortogonales de vectores

1. Producto punto en GNU Octave. Abra el intérprete de GNU Octave y ejecute los siguientes comandos uno por uno:

```
a = [3; -1; 4]
b = [1; 5; -3]
a'
a' * b
a' * a
norm(a) ^ 2
```

2. Vectores ortogonales y su matriz de Gram. Verifiquemos que tres vectores dados son ortogonales calculando sus productos por pares. Además calculemos los cuadrados de sus normas. Luego formemos una matriz de los tres vectores (columnas) dados y calculemos su matriz de Gram.

```
a = [3; 5; -1; -1]
b = [-3; 3; 5; 1]
c = [2; -1; 2; -1]
[a' * b, a' * c, b' * c]
[a' * a, b' * b, c' * c]
abc = [a b c]
G = abc' * abc
```

Combinación lineal de una lista de vectores ortogonales no nulos

3. Coeficientes de una combinación lineal de vectores ortogonales. Formemos una combinación lineales de los tres vectores ortogonales dados en el ejercicio anterior y mostremos que los coeficientes de esta combinación lineal se pueden recuperar como ciertos cocientes de productos internos.

```
a = [3; 5; -1; -1]
b = [-3; 3; 5; 1]
c = [2; -1; 2; -1]
v = - a + b + 3 * c
la1 = (a' * v) / (a' * a)
la2 = (b' * v) / (b' * b)
la3 = (c' * v) / (c' * c)
```

4. Identidad de Pitágoras–Parseval. Seguimos trabajando con los vectores a, b, c, v del ejercicio anterior y con coeficientes calculados.

```
norm(la1 * a) ^ 2
la1 ^ 2 * (a' * a)
norm(la1 * a) ^ 2 + norm(la2 * b) ^ 2 + norm(la3 * c) ^ 2
la1 ^ 2 * (a' * a) + la2 ^ 2 * (b' * b) + la3 ^ 3 * (c' * c)
v' * v
```

5. Independencia lineal de vectores ortogonales no nulos. Seguimos trabajando con los mismos vectores a, b, c.

```
abc = [a b c]
rank(abc)
```

Proyección ortogonal

6. Proyección ortogonal de un vector al otro. Dados dos vectores a y v en $\mathbb{R}^2$, calculemos $\lambda \in \mathbb{R}$ tal que $a \perp (v - \lambda a)$. Denotemos λa por u y $v - \lambda a$ por w. Verifiquemos que se cumple el teorema de Pitágoras para los vectores u, w, v.

```
a = [-3; 2]
v = [-8; 1]
a' * a
norm(a) ^ 2
a' * v
lambda = (a' * v) / (a' * a)
u = lambda * a
w = v - u
a' * w
[norm(u) ^ 2 + norm(w) ^ 2, norm(v) ^ 2]
```

7. Proyección ortogonal de un vector al subespacio generado por dos vectores ortogonales. Dado un vector $v \in \mathbb{R}^4$ y dos vectores ortogonales $a, b \in \mathbb{R}^4$ calculemos dos vectores $u, w \in \mathbb{R}^4$ tales que $v = u + w$, $u \in \ell(a, b)$, $w \perp a$, $w \perp b$.

```
a = [1; -1; -2; 1]
b = [2; 3; 1; 3]
v = [2; 10; 4; -5]
[a' * a, b' * b, a' * b]
lambda1 = (a' * v) / (a' * a)
lambda2 = (b' * v) / (b' * b)
u = lambda1 * a + lambda2 * b
w = v - a
[a' * w, b' * w]
[norm(u) ^ 2 + norm(w) ^ 2, norm(v) ^ 2]
```

Proceso de ortogonalización de Gram–Schmidt

8. Ortogonalización de Gram–Schmidt. Dados tres vectores $a_1, a_2, a_3 \in \mathbb{R}^3$, construimos tres vectores ortogonales $b_1, b_2, b_3 \in \mathbb{R}^4$ aplicando el algoritmo de Gram–Schmidt a los vectores dados a_1, a_2, a_3. Al final comprobamos que los vectores b_1, b_2, b_3 son ortogonales a pares y calculamos su matriz de Gram.

```
a1 = [-2; 5; 1; -1]
a2 = [-1; 6; 3; 4]
a3 = [-11; 20; 9; 7]
b1 = a1
b1norm2 = b1' * b1
lambda21 = (b1' * a2) / b1norm2
b2 = a2 - lambda21 * b1
b2norm2 = b2' * b2
lambda31 = (b1' * a3) / b1norm2
lambda32 = (b2' * a3) / b2norm2
b3 = a3 - lambda31 * b1 - lambda32 * b2
b3norm2 = b3' * b3
b1b2b3 = [b1 b2 b3]
[b1' * b2, b1' * b3, b2' * b3]
G = b1b2b3' * b1b2b3
```