Programación: solución de sistemas de ecuaciones lineales usando la descomposición QR

Objetivos. Programar una función que resuelva sistemas de ecuaciones lineales usando funciones programadas anteriormente que realizan la descomposición QR y resuelven sistemas de ecuaciones lineales con matrices triangulares superiores.

Requisitos. Descomposición QR, solución de sistemas de ecuaciones lineales con matrices triangulares superiores.

Se recomienda resolver estos ejercicios antes de la clase práctica.

1. Definición de la descomposición QR en el caso cuadrado. Sea $A \in \mathcal{M}_n(\mathbb{R})$. Un par de matices (Q, R) se llama *descomposición QR* de la matriz A si:

- $QR = \ldots$
- Q es $\ldots$
- R es $\ldots$

Notemos que en el caso de matrices cuadradas no hay diferencia entre la descomposición QR completa y la descomposición QR reducida (delgada).

2. Invertibilidad de la matriz R en la descomposición QR. Supogamos que A es una matriz cuadrada invertible y las matrices Q y R forman su descomposición QR.

- Muestre que la matriz R es invertible.
- ¿Qué podemos afirmar sobre las entradas diagonales de R?

3. Fórmulas matemáticas. Sea $A \in \mathcal{M}_n(\mathbb{R})$ es una matriz cuadrada invertible y sea $b \in \mathbb{R}^n$. Supogamos que (Q, R) es una descomposición QR de A. Exprese la solución del sistema $Ax = b$ en términos de Q, R, b.

$$Ax = b \iff \underbrace{\qquad}_{?} x = b \iff \underbrace{\qquad}_{?} x = \underbrace{\qquad}_{?} b \iff x = \underbrace{\qquad}_{?} \Big(\underbrace{\qquad}_{?} b \Big).$$

Por supuesto, en el algoritmo numérico no es necesario invertir la matriz R. Sería más eficiente resolver el sistema $Rx = \underbrace{\qquad}_{?}$ con una función programada anteriormente que resuelve sistemas de ecuaciones lineales con matrices triangulares superiores.

4. Funciones necesarias programadas anteriormente. Se supone que ya tenemos las siguientes funciones:

- Función `myqr` que realiza la descomposición QR (completa o reducida) con algún método. En el lenguaje MATLAB, la sintaxis es `[Q, R] = myqr(A)`.
- Función `solveut` que resuelve sistemas de ecuaciones lineales con matrices triangulares superiores (las entradas diagonales deben ser no nulas): `x = solveut(U, b)`.
- Función que construye la matriz transpuesta de la matriz dada. En MATLAB se puede usar la operación `'` o `.'`, o sus sinónimos `ctranspose` y `transpose`:

  ```
  B = A'; B = transpose(A);
  ```

- Función que multiplica la matriz dada por el vector dado. En MATLAB se puede usar la operación `*`.

5. Programar una función que resuelva sistemas de ecuaciones lineales usando la descomposición QR.

```
function [x] = solvewithqr(A, b),
   [Q, R] = myqr(A);
   ...
end
```

6. Pruebas con matrices pequeñas.

```
function [x] = test1solvewithqr(),
   n = 4;
   A = 2 * rand(n, n) - ones(n, n); b = 2 * rand(n, 1) - ones(n, 1);
   x = solvewithqr(A, b);
   display(norm(A * x - b));
end
```

7. Pruebas con matrices grandes.

```
function [x] = test2solvewithqr(),
   for n = [100, 200, 400],
      A = ...; b = ...;
      t1 = cputime(); x = solvewithqr(A, b); t2 = cputime();
      display(n); display(norm(A * x - b)); display(t2 - t1);
   end
end
```

8. Medir por separado el tiempo de QR. En la función `test2solvewithqr` sustituir el comando `x = solvewithqr(A, b)` por el código de la función `solvewithqr`, luego medir por separado el tiempo de la ejecución de la función `myqr` y el tiempo de ejecución necesario para la otra parte (solución del sistema triangular superior).