Calcular el pagerank de una matriz dispersa

El **objetivo de esta práctica** es codificar la función ***[x,precision1]=calculo_PR(A,tol,n_iter,x0)*** para calcular el pagerank de la matriz G, y medir las prestaciones numéricas de dicha función.

Se quiere calcular el pagerank de una matriz G, pero como G es una matriz densa y de gran tamaño, por tanto no podemos almacenarla en memoria.

Utilizamos la siguiente estrategia:

“Codificar la función

[x,precision1]=calculo_PR(A,tol,n_iter,x0)

que partiendo de la matriz A (dispersa), utiliza el método de la potencia para calcular x el pagerank de G”.

Siguiendo la notación de clase:

- C la matriz de conectividad, N=size(C,1);
- Nj=sum (C), vector de número de links de salida,
- A es la matriz C reescalada por columnas:
  A(:,k)=0 si Nj(k) = 0 (la columna k de las matrices A y C son nulas).
  A(:,k)=C(:,k)/Nj(k) si Nj(k) ~= 0
- El vector dj es el complementario del vector Nj (dj=1-Nj).
  dj(k)=1 si Nj(k)=0
  dj(k)=0 si Nj(k)=1
- El vector fila e=ones(1,N);
- La matriz G tiene la siguiente expresión:

$$G = \alpha S + (1-\alpha)E = \alpha A + \frac{1}{n} e^T \left[\alpha\, dj + (1-\alpha)\, e\right], \quad \alpha = 0.85$$

A. Rutina auxiliar

1. Matrices dispersas (mdis_A)

Codificar la rutina

A=mdis_A(N)

Tiene como variable de entrada la dimensión N, y como variable de salida una matriz dispersa aleatoria A de tamaño NxN, cuyas columnas sumen 1 o 0. Utilizar la siguiente plantilla:

```
function A=mdis_A(N)
R=5;  % R=número de links entrada/salida de cada nodo
p=randi(N,1,R*N);q=randi(N,1,R*N);
A=sparse(p,q,1,N,N);
Nj=sum(A);
    % Reescalar las columnas de A:
                % A(:,k)=0 si Nj(k)=0
                % A(:,k)=A(:,k)/Nj(k)   si Nj(k) ~= 0
return
```

B. Función calculo_PR

Modificar el método de la potencia para obtener la función

[x,precision1]=calculo_PR(A,tol,n_iter,x0)

que **calcula x el pagerank de la matriz densa G,** utilizando como **variable de entrada la matriz dispersa A.**

El método de la potencia aplicado a la matriz G se puede escribir como el siguiente método iterativo:

$$\begin{cases} k = 0,1,\ldots \\ x^k = x^k / norm(x^k) \\ x^{k+1} = \alpha A x^k + \frac{1}{n} e^T \left[\alpha \; dj + (1-\alpha) \; e \right] x^k \end{cases}$$

Para implementar este método utilizamos la siguiente **estrategia numérica**:

En el cálculo de $x^{k+1} = \alpha A x^k + \frac{1}{n} e^T \left[\alpha \; dj + (1-\alpha) \; e \right] x^k$ tenemos dos sumandos: el primer sumando es una matriz dispersa mientras que el segundo sumando en una matriz densa.

Para evitar calcular $e^T \left[\alpha \; dj + (1-\alpha) \; e \right]$ que es una matriz densa, calculamos:

- el vector fila $v = \left[\alpha \; dj + (1-\alpha) \; e \right]$ es constante en todo el proceso iterativo, y lo calculamos una sola vez antes de iniciar el bucle for/while,
- en cada iteración: calcular primero el escalar $v x^k$, y después calcular el vector columna $\frac{1}{n} e^T v x^k$.

Como se puede observar, el orden de los factores puede ahorrar mucha memoria.

Otras especificaciones para codificar el código:

- Las variables de entrada tol, n_iter y x0 son las mismas que en el método de la potencia.
- Las variables de salida son:
  El vector x es el pagerank de G, esto es, Gx=x y sum(x)= 1.
  La variable de salida precision1 = norm(Gx-x,1), norma 1 del vector Gx-x.

  Nota: No disponemos de la matriz G explícitamente.

Ejecutar la función ***calculo_PR(A)*** para matrices A generadas con la función ***A=mdis_A(N),*** para los valores de N=10^2, 10^3, 10^4, 10^5,…. Completar la siguiente tabla

	Tiempo [seg]	Memoria [MB]	Precisión $\|Gx - x\|_1$
N=			
N=			

Comentar los resultados numéricos obtenidos.