Calcular el pagerank de matrices de gran tamaño.
Calcular e interpretar el pagerank de Stanford Web Matrix

Comandos de Matlab:

- ***tic/toc**.* Comando tic en el inicio de una función y t=toc al final de la función. Al ejecucar la función la variable t contiene el tiempo de ejcución.
- ***Whos A*** para ver la memoria ocupada por la matriz A.

1. Rutinas auxiliares. Prestaciones numéricas de la rutina del CalculoPageRank.

a) Rutina *A=matrizA(N,r)* para generar matrices aleatorias A de dimensión N

Codificar una rutina ***A=matrizA(N,r)*** que genera la matriz A del modelo de forma aleatoria

Variables de entrada:
N=dimensión de la matriz,
p=número medio de links de entrada/salida de cada nodo.

Variable de salida:
A=matriz del modelo dispersa de tamaño NxN (sum(A) debe ser un vector de 1's o 0's).

Plantilla de la rutina:

```
funtion A=matrizA(N,r)

p=randi(N,1,r*N);q=randi(N,1,r*N);A=sparse(p,q,1,N,N);  % A matriz dispersa de tamaño NxN

    % Reescalar la matriz A por columnas (dividir cada columna de A por la suma de sus
elementos). El vector sum(A) debe ser un vector compuesto de 0's y 1's

return
```

Probar la rutina.

>> N=1000,r=20,A=matrizA(N,r);

Comprobar que la matriz A verifica las propiedades de la matriz del modelo.
¿Cuántos nodos de A tienen links de salida? ¿Cuántos nodos de A no tienen links de salida?.
¿Cuánto suman las respuestas de las dos preguntas anteriores?.

b) Prestaciones numéricas de la rutina del CalculoPageRank.

La rutina

[ordenpagerank,tiempo,precision]=CalculoPageRank(N,r,niter)

Es similar a la del apartado 3 de la Práctica 1.

Variables de entrada:
N=dimensión de la matriz,
r=número medio de links de entrada/salida de cada nodo.
niter=número de iteraciones

Variable de salida:
Ordenpagerank es el vector de índices [1:N] ordenado según el pagerank de la matriz G obtenido.

tiempo= es el tiempo de ejecución de la rutina.
$\text{precision}=\left\| Gx-x \right\|_2$ es la precisión obtenida

Plantilla de la rutina:

funtion **[ordenpagerank,tiempo,precision]=CalculoPageRank(N,r,niter)**

A=matrizA(N,r);

Calcular la matriz G, a partir de la matriz A.

Calcular el pagerank de G, con el método de la potencia.

Ordenar las páginas según su pagerank:
ordenpagerank es el vector de índices [1:N] ordenado según el pagerank.
ordenpagerank(1) es el índice de la página de mayor pagerank,
……,
ordenpagerank(end) es el índice de la página de menor pagerank.

Ejecutar **[ordenpagerank,tiempo,precision]=CalculoPageRank(N,r,niter)** para los valores **N=10^3, 10^4, 10^5,…** hasta la capacidad de vuestro ordenador (tiempo de ejecución o memoria).

Si la rutina tarda mas de 60-100 segundos de ejecución (o bien 5-7 minutos de reloj), la podeis cancelar con Crtl-C o Crtl-Z.
Utilizar r=20 y un niter suficiente para obtener una precisión del orden de **10^-12 o menor.**
Con los comandos whos A y tic/toc completar la siguiente tabla

	Tiempo ejecución	Memoria utilizada	Número iteraciones	Precision
N=10^3				
N=10^4				
…				
N=10^7				

Tabla: datos numéricos experimentales

c) Extrapolar los datos obtenidos en la tabla anterior

A partir de los datos numéricos experimentales obtenidos en la tabla anterior se desea ajustar el tiempo de ejecución y la memoria utilizada, respecto de la dimensión de la matriz.

¿Qué tipo de función ajustará mejor los datos de **tiempo de ejecución** (un polinomio, función exponencial, etc.) en el sentido de los mínimos cuadrados?

Para el tipo de función elegida, calcular la que ajuste a los datos de la tabla (tiempo de ejecución) en el sentido de los mínimos cuadrados.

Utilizar el ajuste obtenido para extrapolar los datos experimentales y completar la tabla extrapolada.

¿Cuántos siglos/millones de siglos de ejecución necesito para ejecutar la rutina con N=10^10?

¿Qué tipo de función ajustará mejor los datos de **memoria utilizada** (un polinomio, función exponencial, etc.) en el sentido de los mínimos cuadrados?

Para el tipo de función elegida, calcular la que ajuste a los datos de la tabla (memoria utilizada) en el sentido de los mínimos cuadrados.

Utilizar el ajuste obtenido para extrapolar los datos experimentales y completar la tabla extrapolada.

¿Cuántos millones de ordenadores como el mio necesito para almacenar en memoria una matriz de tamaño N=10^10?

	Tiempo estimado [horas/siglos]	Memoria estimada [GB/TB]
N=10^8		
N=10^9		
N=10^10		

Tabla: datos extrapolados

2. Codificar la rutina para calcular el pagerank de la matriz de pagerank G a partir de la matriz A

El objetivo es calcular el pagerank de la una matriz G de gran tamaño. La memoria del ordenador no tiene capacidad para almacenar la matriz G. Calculamos el pagerank de G utilizando como variable de entrada la matriz A.

Codificar la función

[pagerank,precision,tiempo]=calculo_PR(A,niter)

Variables de entrada:

La matriz dispersa *A* y el número de iteraciones *niter*.

Variables de salida:

pagerank es el orden de las páginas según el pagerank de la matriz G , la precisión obtenida (precisión = $\|Gx - x\|_2$) y el **tiempo** utilizado.

Proceso:

A partir de la matriz A de entrada, el código calcula el pagerank de G, pero NO dispone de la matriz G explícitamente.

A partir de la matriz A, calculamos los vectores Nj, dj y v. Ver las diapositivas de la presentación de clase.
Utilizar el código del método de la potencia.
Utilizar el parámetro $\alpha = 0.85$.

[pagerank,precision,tiempo]=calculo_PR(A,niter)

$$
\begin{cases}
v = \left[\alpha\, dj + (1-\alpha)\, e\right] \\
k = 1 : n\_iter \\
x = x / norm(x) \\
x = \alpha A x + \frac{1}{N} e^T (v x) \\
end \\
\text{precision } = \left\| Gx - x \right\|_2 \\
\text{Obs: Calcular Gx 'sin utilizar' G}
\end{cases}
$$

Probar la rutina:

Ejecutando los comandos:

>>N=1e+4,r=20,A=matrizA(N,r);n_iter=100;[pagerarank,precision,tiempo]=calculo_PR(A,niter)*;*

Calcular la matriz G explícitamente.
Comprobar que el pagerank calculado con la rutina calculo_PR a partir de la matriz A, coincide con el pagerank de la matriz G.

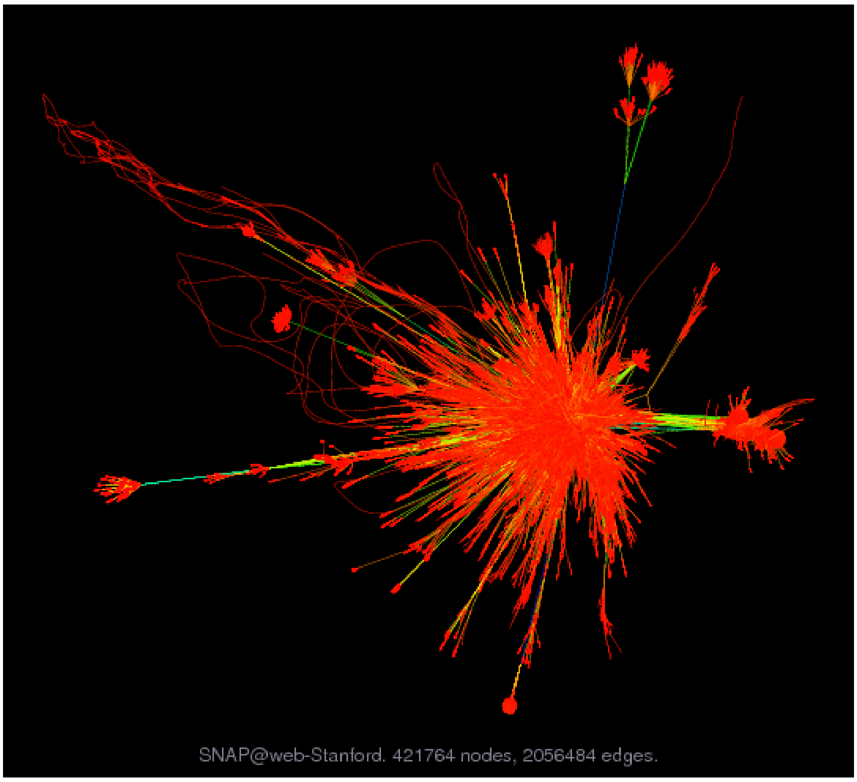
3. Calcular e interpretar el pagerank de Stanford Web Matrix

Calcular el **pagerank de Stanford Web Matrix** (281903 nodos y 2.3 millones de links). Visualizar y ordenar los nodos en función de su pagerank.

1. Buscar en Google: ‘stanford web matrix’. Acceder a la página:

https://www.cise.ufl.edu/research/sparse/matrices/SNAP/web-Stanford.html

Descargar download as a MATLAB mat-file, file size: 6 MB.



Ejecutar los comandos (tarda unos minutos, que no se os olvide el ; para que no muestre el contenido de A por pantalla).

```
load web-Stanford.mat;
Problem
A=Problem.A;
whos
spy(A);title('Gráfica de la dispersión de la matriz A')
```

a) Dar los comandos necesarios para conocer las **características de la matriz A**.

- ¿Es cuadrada?. ¿Cuántos nodos tiene la red?.
- Tipo de matriz (dispersa, completa).
- Tamaño en memoria de A.
- Número de elementos no nulos de A.
- Dar el comando:>> B=A-1; ¿Qué sucede?. ¿Por qué?.
- ¿Cuántos elementos de A son 1's? ¿Cuántos son 0's?. ¿Cuántos son distintos de 1's y de 0's?.
- Mostrar el contenido de A de las filas 1:1000 y las columnas 1:1000. ¿Cuántos elementos no nulos hay en esa submatriz 1000x1000?.
- ¿Qué nodo tiene el mayor número de links de salida? ¿Cuántos links de salida tiene?.
- La matriz A ¿qué tipo de matriz es de las descritas en las diapositivas (C, A, S o G) según el modelo?.
  ¿Es una matriz de conectividad C? ¿Por qué?.
  ¿Es una matriz de transición A? ¿Por qué?.
  ¿Es una matriz de transición modificada S? ¿Por qué?.
  ¿Es una matriz de Google G? ¿Por qué?.
- Calcular el índice de dispersión de A (número de elementos no nulos/número total de elementos).
- Calcular el número medio de links de salida: ¿cada página cuantos línks de salida de media tiene?.

- **Sin utilizar el número total de nodos**: ¿Cuántos nodos sin salida tiene la red?, ¿cuántos nodos con salida tiene la red?.
  Comprobar si se verifica la siguiente relación:

  número de nodos totales = número de nodos sin salida + número de nodos con salida.

b) Calcular el pagerank de Stanford Web Matrix.

Utilizar la función [pagerarank,precision,tiempo]=calculo_PR(A,niter); para calcular el pagerank, de la Stanford Web Matrix. Utilizar un niter suficiente para obtener una precisión menor de 1e-12. Completar la siguiente tabla:

	Tiempo [seg]	Memoria [MB]	Nº iteraciones	Precisión $\left\|Gx - x\right\|_2$
N=281903				

c) Visualizar y ordenar los resultados.

- Visualizar el pagerank obtenido con el comando *bar(pagerank)*.
- Utilizar el comando *sort* para ordenar los elementos del vector pagerank.
- Construir tabla de dimensión 2xN que contenga los nodos y los pageranks **ordenados de mayor a menor.**
  Dar el comando *fprintf* para extraer de la tabla los 20 nodos con los mayores pagerank. El resultado debe ser algo similar a:

```
Orden   1  Nodo  xxxx  Pagerank  0.yyyy
Orden   2  Nodo  xxxx  Pagerank  0.yyyy
Orden   3  Nodo  xxxx  Pagerank  0.yyyy
Orden   4  Nodo  xxxx  Pagerank  0.yyyy
Orden   5  Nodo  xxxx  Pagerank  0.yyyy
```

- Repetir el comando anterior para extraer de la tabla los 20 nodos con los menores pagerank.
- ¿Hay varios nodos que tienen igual pagerank, y ese pagerank es el menor?. ¿Cuántos nodos tienen el menor pagerank y cuáles son?. Si es posible, identificar estos nodos y visualizarlos con el comando *fprintf*.
- ¿Hay varios nodos que tienen igual pagerank, y ese pagerank es el mayor?. ¿Cuántos nodos tienen el mayor pagerank y cuáles son?. Si es posible, identificar estos nodos y visualizarlos con el comando *fprintf*.