Hoja Entrega Tarea 2

APELLIDOS, NOMBRE: GARCÍA MARTÍNEZ, MÁXIMO
APELLIDOS, NOMBRE:

Se han de incluir los comandos y códigos empleados en cada uno de los apartados. Los resultados no se darán por válidos si no son precedidos por dichos códigos.

1. Leer y ver imagen. Obtener matriz A de datos. Estudio matriz A:

1.1. Leer imagen y crear matriz A. Dimensión de A.

```
function [A, im] = obtenermatriz(path)
    im = imread(path);
    imagesc(im);colormap(gray);
    A = double(im);
end
```

El anterior código, convierte una imagen a una matriz llamada A. La dimensión de está matriz es 321 x 617, es decir, 198057 elementos. La dimensión de la matriz y de la imagen coinciden.

1.2.- Calcular el vector s=(s_k) de valores singulares de la matriz A. Representar gráficamente el vector s (en una escala adecuada).

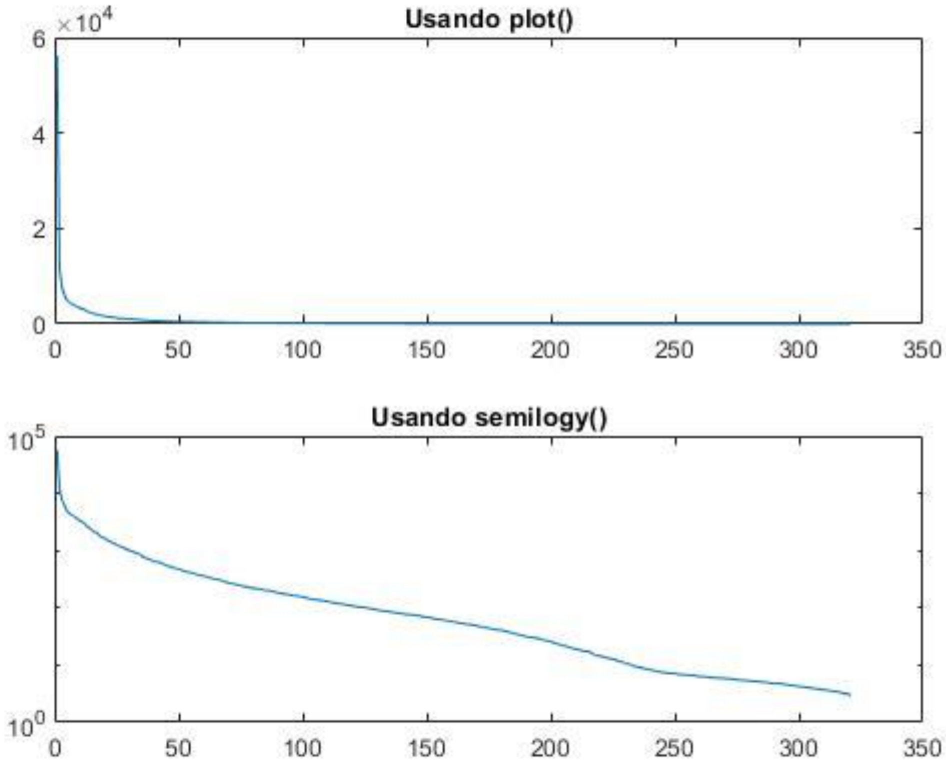
El código para esta parte ha sido el siguiente:

```
function valoressingulares(A)
    [U,S,V] = svd(A);
    s = diag(S);

    %Primera gráfica
    subplot(2,1,1);
    plot(1:321, s);
    title('Usando plot()');
    subplot(2,1,2);
    semilogy(1:321, s);
    title('Usando semilogy()');

    %Segunda gráfica
    cont_inf = s/s(1);
    subplot(1,1,1);
    plot(cont_inf(1:21));
    title('Valores singulares en proporción al primer valor singular');
end
```

En la primera parte obtenemos los vectores y valores singulares haciendo uso del comando de MATLAB svd(). Posteriormente convertimos la matriz de valores singulares a un vector de valores singulares y como en nuestro caso tiene m > n, no tendremos ningún 0 en este vector y podemos usar directamente diag(S). En el caso, m < n, es decir, una imagen vertical, tendríamos que eliminar los valores del vector diag(S) que se encontrasen a partir del índice m inclusive.

Posteriormente dibujamos una gráfica para ver los valores que va tomando el vector de valores singulares. He realizado dos gráficas: una usando el comando plot(), pero como se puede observar en la *gráfica 1,* no se puede ver bien, por eso en la segunda gráfica se ha usado una escala logarítmica para poder ver mejor como los valores van decreciendo.

Hoja Entrega Tarea 2



Gráfica 1. Vector de valores singulares con 321 elementos.

- r=Rango(A). Justificar la respuesta.

El valor del rango de A coincide con el número de valores singulares (distintos a cero), es decir, 321. Al realizar la llamada a svd(), se encarga de eliminar aquellos vectores linealmente dependientes. Este valor se puede obtener de dos formas:

- Usando la función rank() de Matlab.
- Contar los números distintos a cero de la diagonal de la matriz S (en nuestro caso no hay, como se ha explicado anteriormente).

Por otra parte, se puede obtener el rango de A obteniendo el mínimo entre m y n, siendo m y n las dimensiones de la imagen de entrada y puesto a ser una fotografía, la matriz que se forma a partir de ella no contendrá ningún vector dependiente de otro y, por lo tanto, el rango de la matriz será el máximo posible, es decir, el mínimo entre n y m.

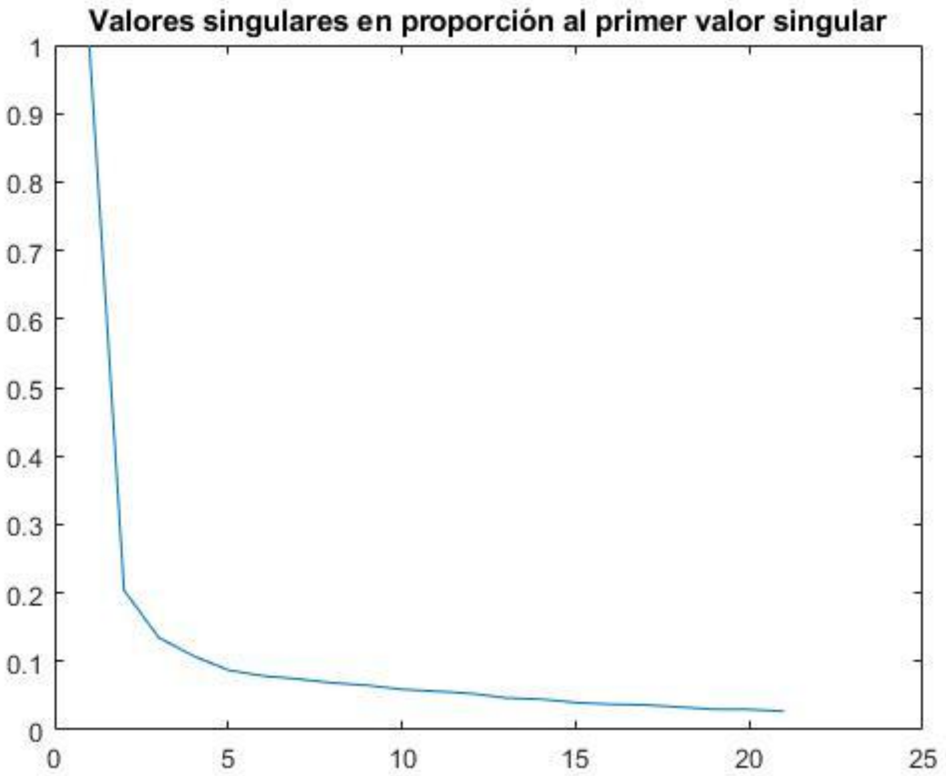
La anterior afirmación, parte de la base que, en una matriz formada a partir de una fotografía, no contendrá filas o columnas de pixeles dependientes de otros y la probabilidad de que eso ocurra es casi nula. Sobre todo, teniendo en cuenta que: el tamaño de la imagen es muy grande y que la fotografía se ha generado de algo natural y no por ordenador y es menos probable que haya filas o columnas de pixeles que sean dependientes entre ellos.

- Representar gráficamente el vector cont_inf=s/s_1. Mostrar las 21 primeras componentes de dicho vector ¿A partir de qué valor de k aproximadamente los valores singulares se reducen a menos de la mitad de s_1? Para el valor de k calculado dar el valor de s_{k+1}.

El código de esta parte es el siguiente

```
cont_inf = s/s(1);
plot(cont_inf(1:21));
title('Valores singulares en proporción al primer valor singular');
```

Hoja Entrega Tarea 2



Gráfica 2. Vector de valores singulares desde el 1 hasta el 21 en proporción al primer valor singular.

A partir de k=2, el valor de s2 se reduce a menos de la mitad de s1.
El vector singular es un vector de pesos. En este caso, el valor singular 1 (S1) tiene el peso de 1 (porque es el que usamos de referencia), pero el valor singular 2 tiene solo un 20% del peso que tiene S1. El valor 3, aproximadamente un peso 15% del peso de S1 y así continuamente.

Por otro lado, también se puede observar que los pesos siempre están ordenados de mayor a menor, siendo el primer elemento el más importante y el último elemento el que menos.

Los valores del vector de valores singulares van siendo más parecidos a 0, pero siempre serán mayores a 0. La diferencia entre los elementos contiguos es cada vez menor, siendo al principio la diferencia en torno a 10^5 y posteriormente la diferencia es de 10^1, como se puede observar en la *Gráfica 1 (semilogy).* Esta característica está totalmente relacionada con los pesos en el vector S.

El objetivo de SVD, es reducir el tamaño de una matriz manteniendo la mayor cantidad de información necesaria y es la persona la encargada de decidir cuanto se quiere perder. Cuanta menos cantidad de datos se quiera perder mayor será la k.

Si obtuviésemos un SVD de la submatriz A con k=2, el error que se tendría sería igual que el valor singular s3.

- Completar Tabla 1:

Rango r	s_1	k	s_{k+1}	s_{k+1}/ s_1	$s_{11}/ s_1,\ s_{21}/ s_1,\ s_{31}/ s_1$
321	5.6271e+04	2	1.1463e+04	0.2037	0.0564 0.0271 0.0174

Hoja Entrega Tarea 2

2. Aproximaciones de rango bajo. Ver aproximaciones.

2.1. Construir las matrices $A_k = \sum_{i=1}^{k} s_i u_i v_i'$ de rango k, con k=10, 20, 30.

```
[U10, S10, V10, im10] = kfunction(A, 10);
[U20, S20, V20, im20] = kfunction(A, 20);
[U30, S30, V30, im30] = kfunction(A, 30);

function [Uk, Sk, Vk, imk] = kfunction(A, k)
    [Uk, Sk, Vk] = svds(A, k);
    imk = Uk * Sk * Vk';
end
```

Está función recibe como parámetro la matriz de entrada A (la imagen) y el k al que se quiere reducir. Se podría haber calculado el U, S, V total y luego realizar submatrices, de estas mismas, y desperdiciar el resto de los valores que no se van a usar, pero MATLAB ya tiene integrada una función que te calcula el SVD a partir de un k y evitar operaciones de cómputo innecesarias. Es por eso que se usa svds().
Por otra parte, también se devuelve la matriz con valores del tipo uint8 para poder ser dibujada posteriormente.

2.2. Completar los datos pedidos en Tabla 2:

k	Rango(A_k)	Valor singular dominante de A_k (s_{k+1})	Sk+1 / s1
10	10	5.6271e+04	0.0564
20	20	5.6271e+04	0.0271
30	30	5.6271e+04	0.0174

¿Relación entre los valores de la columna tercera de la tabla anterior y los valores singulares de A? ¿Relación entre los valores de la columna tercera de la Tabla 2 y los de la columna última de la Tabla 1? Justificar las respuestas.

En la tabla anterior, tenemos 3 matrices que se han calculado a partir de A. Se han calculado a partir de k=10, k=20 y k=30. El rango de Ak, coincidirá con el valor de k, no hay vectores independientes. El rango será siempre el mayor posible.
Por otro lado, se puede ver que el valor singular dominante para los diferentes Ak, es el mismo. Esto se debe a que las tres matrices tienen los 10 primeros valores iguales. Es decir, Ak, se calculaba de la siguiente forma:

$$A_k = \sum_{i=1}^{k} s_i u_i v_i'$$

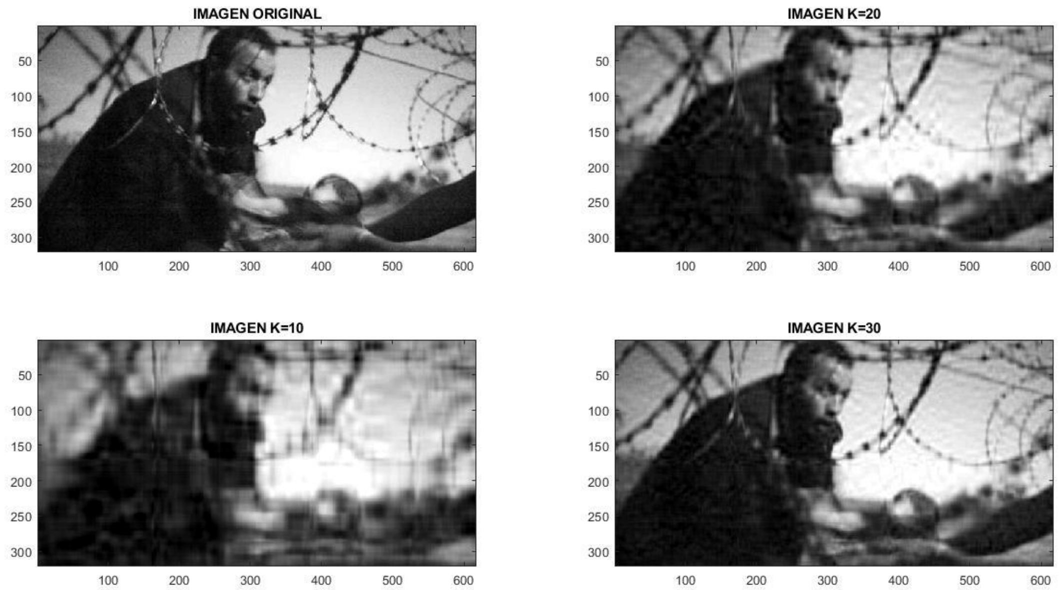
Como se puede observar los primeros elementos siempre van a ser los mismos. Lo que varía entre los distintos Ak es el error.

Como se puede ver en la tabla, a mayor k, menor error. El error es calculado por la norma euclídea de la diferencia entre A y Ak. Esto es lo mismo que decir el error que es el mayor valor singular que no se toma, es decir, S_{k+1}. El error también se podría calcular con la norma de Frobenius.

A mayor k, menor es el error, esto es debido que si usamos k=10 y k=20, la “precisión” de k=20 va a ser la misma que la que tenía k=10 más lo valores que hay entre el elemento con índice 10 y 20. En resumen, la precisión se podría decir que es acumulativa.

3. Vemos el resultado que producen las aproximaciones anteriores. Estudiamos los errores y la compresión obtenida.

3.1. Visualizar las imágenes aproximadas imk a partir de las matrices Ak. ¿Se reconoce la imagen original en dichas imágenes? Comentar los resultados.



Como era de esperar, según k es mayor, la imagen se puede visualizar con mejor calidad. La diferencia notable está entre la imagen que se ha calculado a partir de k=10 y la imagen con k=20, la resolución mejora y se puede ver más detalles de la imagen. La diferencia entre la imagen con k=20 y k=30 es la reducción de ruido en k=30. A pesar de ello, si se compara la imagen k=30 con la imagen original se puede observar que aún hay mucho ruido.
Realizando pruebas a parte, si se usa k=90, a simple vista y en tamaño pequeño (no a escala), no se puede ver ninguna diferencia entre la imagen original y la imagen final. Es decir, reducir la cantidad de ruido presente en la imagen final puede ser muy costoso a nivel computacional. El valor k=90 no es casualidad, si se ve en la *gráfica 1*, a partir de x=90 el valor se mantiene muy cerca de 0 hasta x=321.
El código usado para reconstruir las imágenes ha sido:

```
dibujarimg(im, im10, im20, im30);

function dibujarimg(im1, im2, im3, im4)
    subplot(2,2,1);
    imagesc(im1);colormap(gray);
    title('IMAGEN ORIGINAL');

    subplot(2,2,3);
    imagesc(im2);colormap(gray);
    title('IMAGEN K=10');

    subplot(2,2,2);
    imagesc(im3);colormap(gray);
    title('IMAGEN K=20');

    subplot(2,2,4);
    imagesc(im4);colormap(gray);
    title('IMAGEN K=30');
end
```

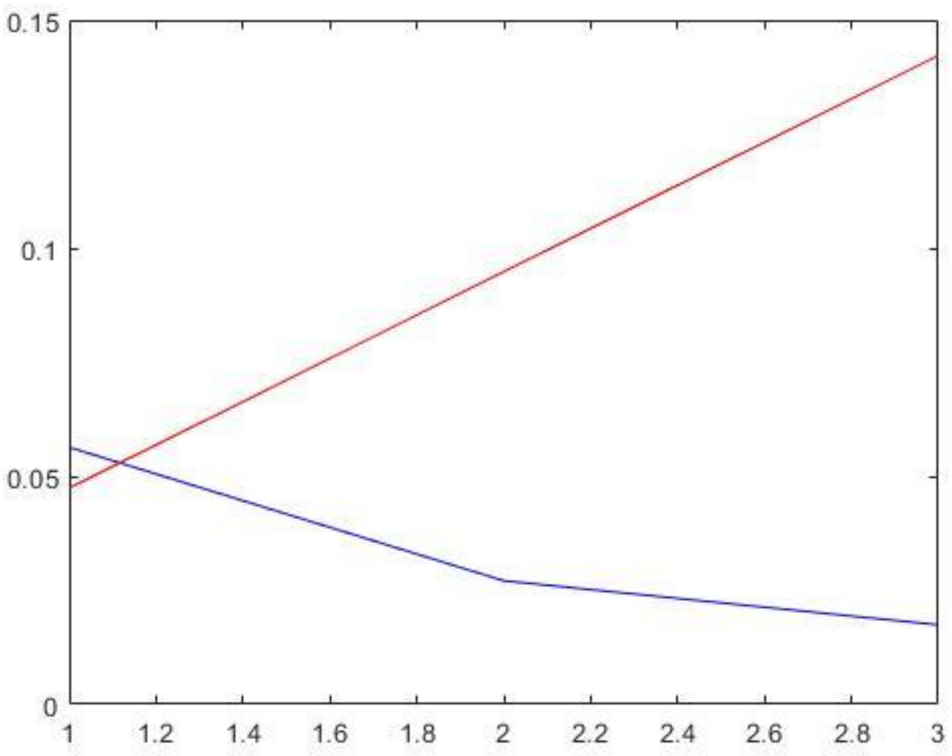
3.2. Rellenar Tabla 3 para las aproximaciones de rango bajo Ak:
El código para obtener los valores de esta tabla es:

```
function [err, gan] = error_ganancia(A, Ak, k)
    err = norm(A - Ak) / norm(A);
    [n, m] = size(A);
    gan = k*(n+m+1)/(n*m);
end
```

donde A, es la matriz de inicio, Ak es la matriz obtenida a partir de k.

k	Error_relativo_k	Ganancia_k
10	0.0564	0.0474
20	0.0271	0.0948
30	0.0174	0.1422

- Representar en una misma gráfica los vectores de las dos últimas columnas de la tabla anterior (2ª columna azul, 3ª columna rojo). ¿Qué relación hay entre las últimas columnas de la Tabla 2 y la Tabla 3?



El código para dibujar las gráficas es:

```
[err10, gan10] = error_ganancia(A, U10 * S10 * V10', 10);
[err20, gan20] = error_ganancia(A, U20 * S20 * V20', 20);
[err30, gan30] = error_ganancia(A, U30 * S30 * V30', 30);

subplot(1,1,1);
```

```
plot([1, 2, 3], [err10, err20, err30], 'b', [1, 2, 3], [gan10, gan20, gan30],
'r');
```

Las ganancias se pueden ver que crece linealmente mientras que el error decrece. El valor del error nunca será menor a 0 y cada vez los errores serán menores. A mayor k, el error será menor.

4. Conclusiones del ejercicio.

Las imágenes se pueden reducir usando este método y pudiendo reducir la cantidad de datos, pero al mismo tiempo no reducir la cantidad de detalles. Este ejercicio muestra cómo reducir el tamaño de la imagen de forma práctica usando el método SVD.