Universidad de San Andrés

Inferencia y Estimación

Compresión y Descompresión de Imágenes usando PCA

Catalina Hirsch - 36557
Clara Zavaroni Benoit - 36772
Maylen Antonella Villagrán Cardozo - 36758

6 de septiembre de 2025

Resumen

La realización de este trabajo práctico tiene como objetivo aplicar el método de **Análisis de Componentes Principales (PCA)** en la compresión de imágenes. Se busca reducir el espacio de almacenamiento minimizando la pérdida de información, conservando los datos que contienen la mayor parte de la varianza. Posteriormente, se descomprime la imagen y se compara con la original para evaluar la calidad de la compresión, utilizando métricas objetivas. El desempeño se evalúa en función de la cantidad de componentes principales utilizados. Finalmente, se discuten los resultados y conclusiones obtenidas.

Introducción

En este informe se estudia la compresión y descompresión de imágenes mediante el método de Análisis de Componentes Principales (PCA). Se analizan las propiedades estadísticas de las imágenes, se implementa el proceso de compresión y reconstrucción, y se evalúa el desempeño en función de la calidad de la imagen reconstruida y el ahorro de espacio.

Ejercicio 1: Correlación

El propósito de este ejercicio es analizar el comportamiento de los píxeles vecinos en las imágenes utilizadas. En primer lugar, se cargan las imágenes y se convierten a escala de grises. Luego, se divide cada imagen en bloques de 2x1 píxeles contiguos verticalmente. A continuación, se calcula la correlación entre los píxeles de cada bloque y se almacena en un vector. Finalmente, se realiza un gráfico de dispersión de las correlaciones obtenidas para cada imagen. Los resultados se presentan a continuación:

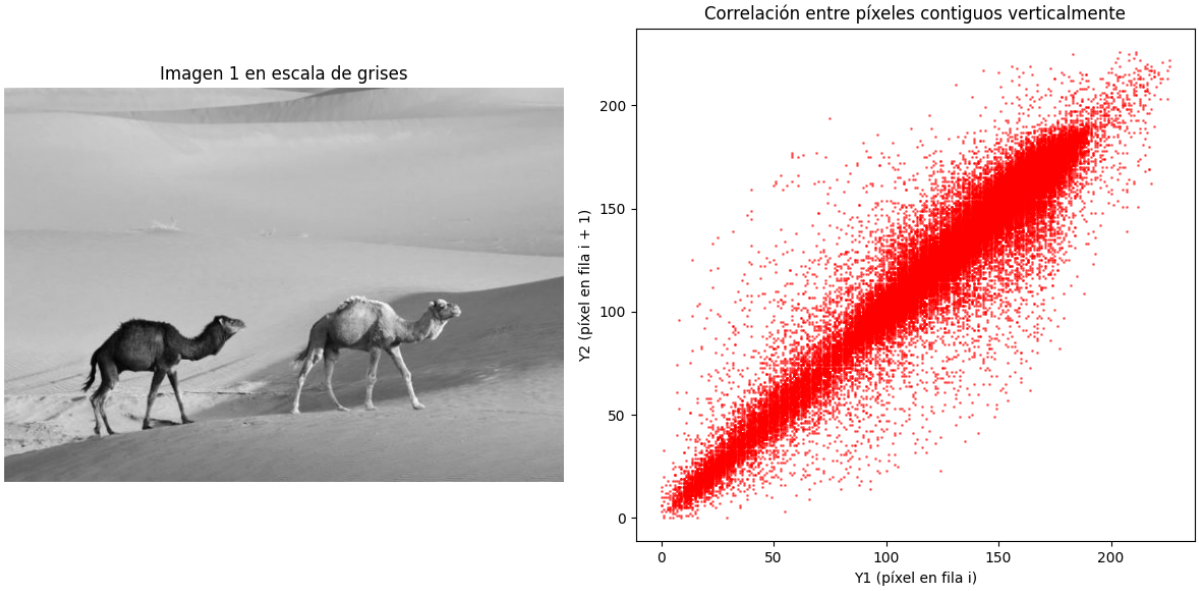

Figura 1. Imagen 1, gráfico de dispersión de la correlación de píxeles contiguos verticalmente

Analizando el comportamiento del gráfico, los puntos se alinean en torno a una recta creciente, lo cual indica una fuerte correlación positiva entre los píxeles. El resultado se debe a que, la imagen presenta suavidad, el color de los píxeles vecinos se asemeja. Por otro lado, los resultados de la segunda imagen son los siguientes:

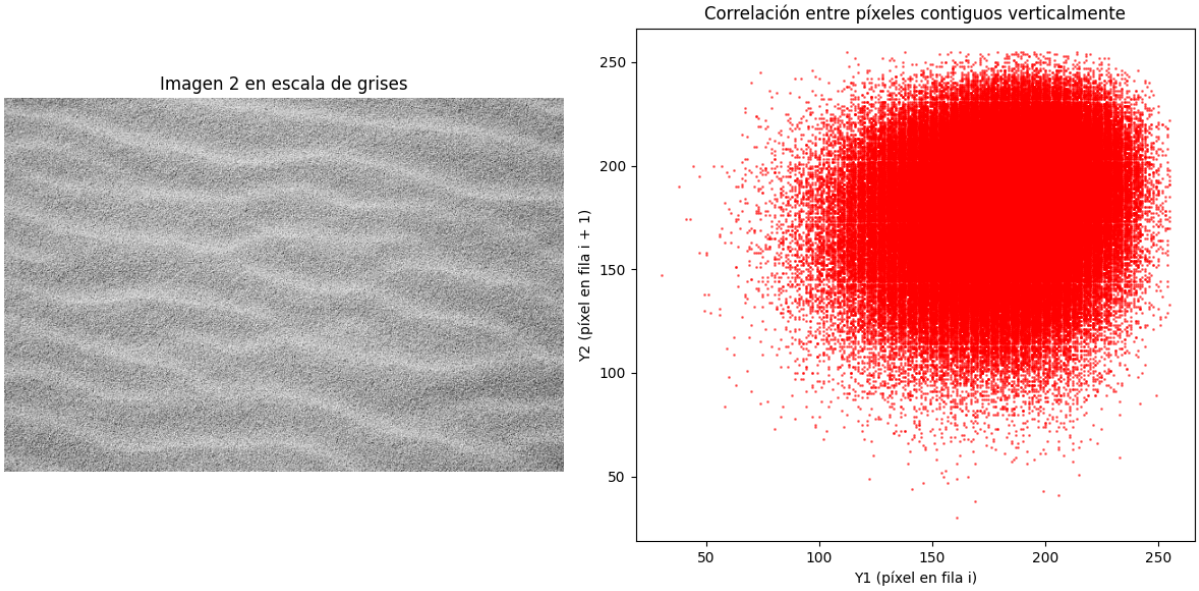

Figura 2. Imagen 2, gráfico de dispersión de la correlación de píxeles contiguos verticalmente

A diferencia del gráfico anterior, los puntos se encuentran más dispersos, implicando una correlación más baja que la imagen anterior. Atribuimos la diferencia al ruido perteneciente a esta imagen, y a la falta de suavidad. Los colores entre los píxeles vecinos no están relacionados.

Luego, se estimó el coeficiente de correlación de cada vector. Los resultados son los presentados a continuación:

1. Para la Imagen 1 se obtuvo, coeficiente de correlación = 0.9789354535355658
2. Para la Imagen 2 se obtuvo, coeficiente de correlación = 0.1460473239944978

Los resultados de la primera imagen apoyan lo intuido anteriormente, al aproximarse al valor 1, indican una fuerte correlación entre los píxeles. Con respecto a la segunda imagen, sus resultados también apoyan lo intuido, al acercarse al valor 0.1, demuestra que los píxeles vecinos no están relacionados.

La diferencia entre ambas imágenes nos indica cuál se podría comprimir con mayor calidad. En primer lugar, la imagen 1 al comprimirse, podría eliminar la redundancia entre píxeles vecinos. Evitando perder la calidad de la imagen. En cambio, en la imagen 2, la falta de redundancia causa la pérdida de calidad al momento de comprimir, ya que se dificulta la predicción de un píxel a partir de su vecino.

Por último, se pide una transformación que descorrelacione las variables. Se utiliza la transformación de descorrelación de PCA:

$$Y = P^T X$$

De esta forma, Y es la proyección de X en el espacio de autovectores de C_X.

Nuevamente, se hace uso de un gráfico de dispersión por imagen, dando los siguientes resultados:

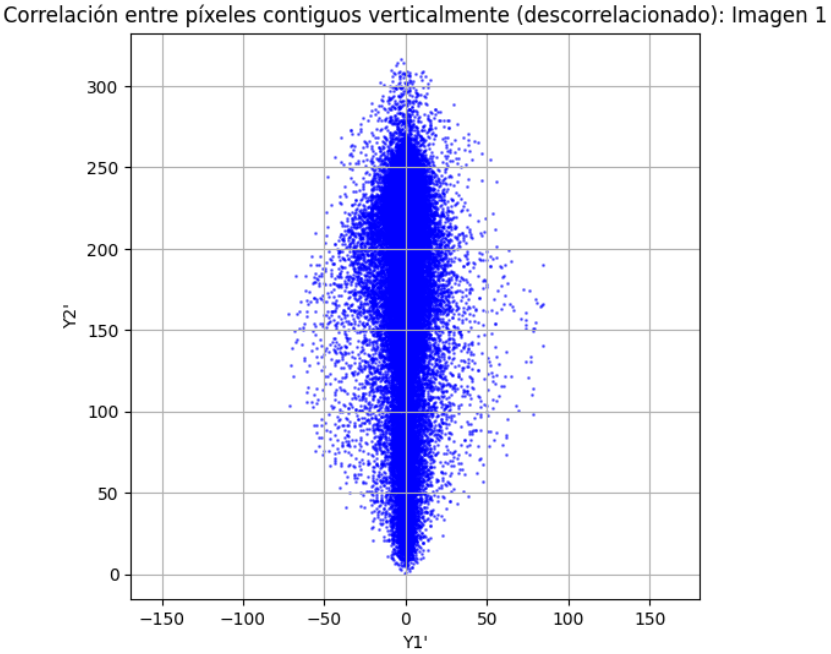

Figura 3. Imagen 1, gráfico de dispersión de la descorrelación de píxeles contiguos verticalmente

Podemos observar en la primera imagen cómo los píxeles se encuentran concentrados en el eje vertical Y2, el proceso de descorrelación resulta exitoso.

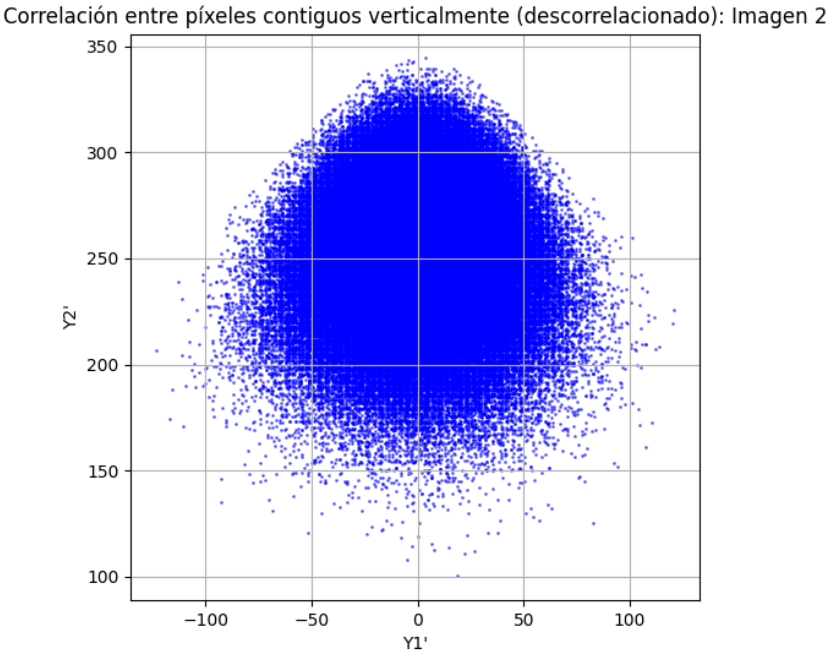

Figura 4. Imagen 2, gráfico de dispersión de la descorrelación de píxeles contiguos verticalmente

Por otro lado, en la segunda imagen, el gráfico se mantiene en forma de una nube, como en la Figura 2, demostrando la falta de correlación inicial.

Ejercicio 2: Compresión

El objetivo de este ejercicio es transformar la información contenida en la imagen, originalmente de alta dimensión, en una representación más compacta que conserve la mayor parte de la información relevante.

El proceso comienza dividiendo la imagen en bloques de 8x8 píxeles, por lo que se tuvo que recortar la imagen de modo que quedara un tamaño múltiplo de 8. Para esto, se utilizó la función `image_to_list_x()`, que recorre cada bloque por columnas y lo convierte en un vector columna de 64 elementos. Así, si la imagen tiene un total de m bloques, se obtiene una colección de vectores $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$, cada uno de dimensión 64×1. Estos vectores se organizan como filas en una matriz de datos X de tamaño $m \times 64$ (en este caso $m = 2537$), donde cada fila representa un bloque vectorizado, una muestra de cada variable, y cada columna corresponde a una posición fija dentro del bloque (una variable).

Para analizar la estructura estadística de los datos, primero se calcula el vector media $\boldsymbol{\mu}$, que contiene el promedio de cada columna de X. Esto se realiza mediante la función `mean_estimator()`. Matemáticamente, esto se expresa como:

$$\boldsymbol{\mu} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i$$

Luego, se centra la matriz de datos restando la media a cada fila, obteniendo así la matriz centrada $X_c = X - \boldsymbol{\mu}^T$. Por como están los datos en la matriz X, la media se resta a cada fila, es decir, a cada bloque.

El siguiente paso consiste en calcular la matriz de covarianza C_X, que describe cómo varían conjuntamente las distintas posiciones dentro de los bloques. Esta matriz se estima mediante la función `cov_estimator()`:

$$C_X = \frac{1}{m-1} X_c^T X_c$$

Cada elemento c_{ij} de C_X representa la covarianza entre la posición i y la posición j dentro de los bloques.

Para aplicar PCA, se utiliza la función `pca_transform()`, que resuelve el problema de autovalores y autovectores de la matriz de covarianza. Es decir, se buscan vectores $\mathbf{v}_j$ y escalares λ_j que satisfagan:

$$C_X \mathbf{v}_j = \lambda_j \mathbf{v}_j$$

Los autovalores λ_j se ordenan de mayor a menor, y se seleccionan los k más grandes, que corresponden a las direcciones de mayor varianza en los datos. Los autovectores asociados se agrupan en la matriz de proyección P:

$$P = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k]$$

La proyección de los datos originales sobre este subespacio de menor dimensión se obtiene multiplicando la matriz centrada X_c por P, resultando en la matriz Y de datos comprimidos y descorrelacionados:

$$Y = X_c P$$

El número de componentes k a conservar se determina en función del factor de compresión deseado, en este caso $s = 80\,\%$ se cumple con $k = 13$. Para visualizar cómo se seleccionan los componentes principales y cuánta información se conserva, se grafica el espectro de autovalores λ_j (ver Figura 5), donde se destacan los autovalores conservados y descartados según el valor de k. En el código, la función `space_saving_k(s, m)` calcula k como:

$$k = \text{máx}\left(1, \text{mín}(m, \lceil m(1-s) \rceil)\right)$$

donde s es el porcentaje de reducción elegido, con valores entre 0 y 1.

Finalmente, para permitir la reconstrucción de la imagen comprimida, se almacenan la matriz Y de datos comprimidos, la matriz de autovectores P y el vector de medias $\boldsymbol{\mu}$. Esto se realiza con la función `save_data()`. La reconstrucción aproximada de la imagen original se realiza aplicando la transformación inversa:

$$\hat{X} = YP^T + \mathbf{1}_m \boldsymbol{\mu}^T$$

Además, se grafican los autovalores λ_j para visualizar cuánta información se conserva y cuánta se descarta al reducir la dimensionalidad, utilizando la función `plot_eigenvalues()`.

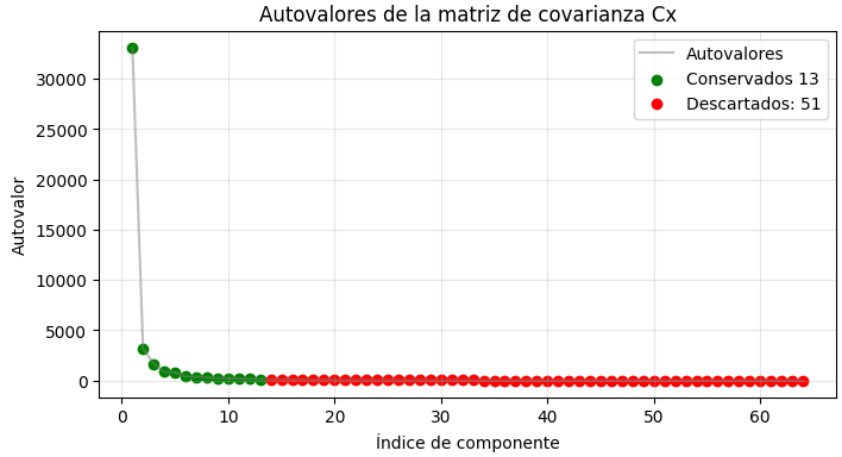

Figura 5. Gráfico de dispersión de los autovalores de la matriz de covarianza para la imagen. En verde se muestran los k autovalores conservados (componentes principales seleccionados) y en rojo los descartados, según el factor de compresión elegido.

Ejercicio 3: Descompresión

Luego de lo realizado en el Ejercicio 2, se realiza la reconstrucción de la imagen comprimida utilizando el proceso inverso del PCA de compresión, para descomprimir. A través del Ejercicio 2, se obtienen los siguientes datos de la imagen:

Vectores y_i (Y), la matriz de autovectores (P) y la media μ_X (μ)

Una vez obtenida la información indicada, se aplica la proyección inversa del PCA, para obtener los vectores x_i aproximados:

$$X = YP^T + \mu$$

donde X es la matriz original, Y es la matriz proyectada, P son los autovectores y μ es la media.

Finalmente, se recorre una matriz vacía, del tamaño necesario de la imagen, completando con los bloques de la imagen reconstruida. De esta manera, obtenemos los siguientes resultados:

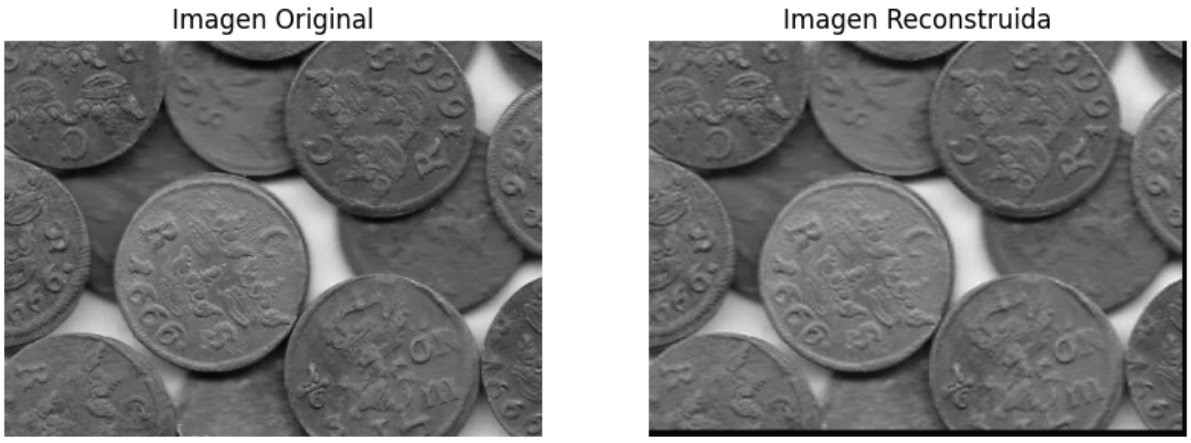

Figura 6. Imagen reconstruida a través del proceso de PCA inverso.

Podemos notar el borde negro de la imagen reconstruida, lo cual sucede al no poder completar bloques de exactamente 8 * 8 al comprimir la imagen. Adicionalmente, podemos observar una diferencia entre las imágenes o pérdida de nitidez, atribuida a la reducción de la dimensionalidad y la eliminación de componentes de menor varianza.

Ejercicio 4: Medidas de desempeño

El objetivo de este ejercicio es evaluar el desempeño de la compresión de imágenes al variar el espacio ahorrado (S).

$$S = \left(1 - \frac{\text{cantidad de componentes principales } (k)}{\text{cantidad de componentes totales } (m)}\right) \times 100\,\% \tag{1}$$

En primer lugar, para cada porcentaje de espacio ahorrado comprimimos una imagen y la reconstruimos. Luego, para medir el rendimiento del procedimiento, calculamos el error cuadrático medio (MSE) entre la imagen original y la reconstruida.

$$MSE = \frac{1}{N_w N_h} \sum_{i=1}^{N_w} \sum_{j=1}^{N_h} (p_{ij} - \hat{p}_{ij})^2 \tag{2}$$

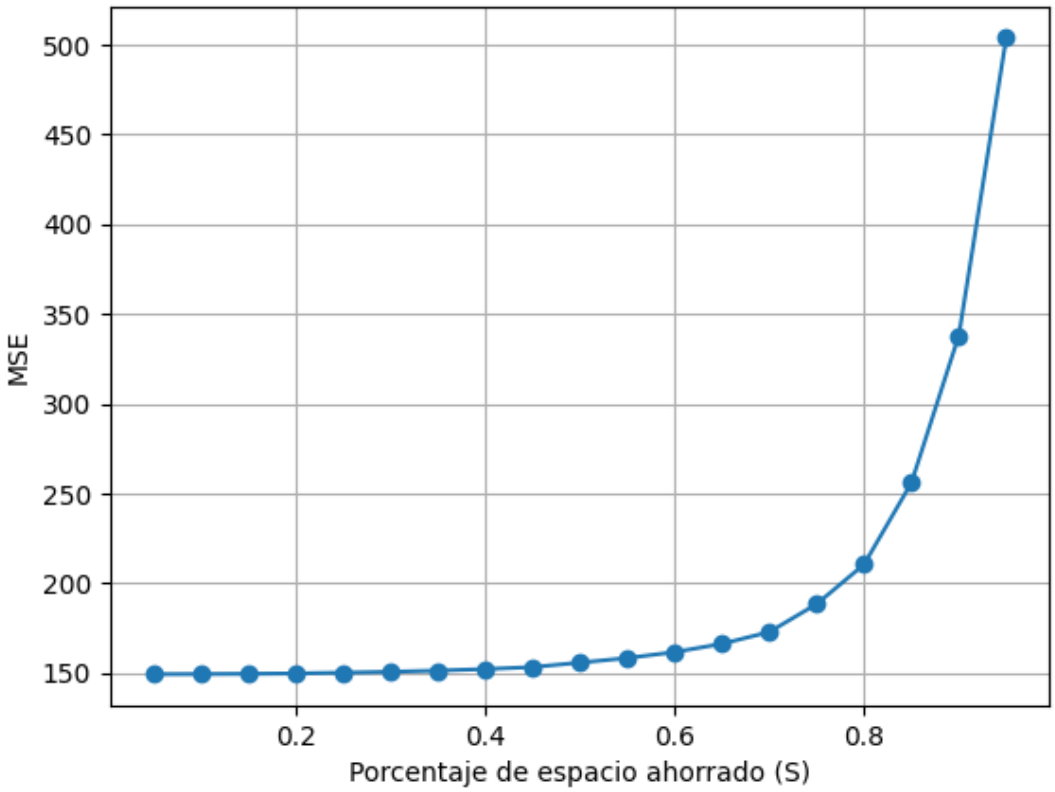

Figura 7. Gráfico MSE vs S

En el gráfico, se observa que el error cuadrático medio aumenta a medida que lo hace el espacio ahorrado. Este resultado es coherente, ya que al descartar un mayor número de componentes principales se pierde más información de la imagen. Sin embargo, esta relación no es proporcional: el error crece considerablemente a partir de un espacio ahorrado del 80 %. Al superar ese porcentaje, la pérdida de información se vuelve significativa, impactando notablemente en la calidad de la imagen.

Para ilustrarlo, incluimos algunas imágenes para diferentes porcentajes de espacio ahorrado.

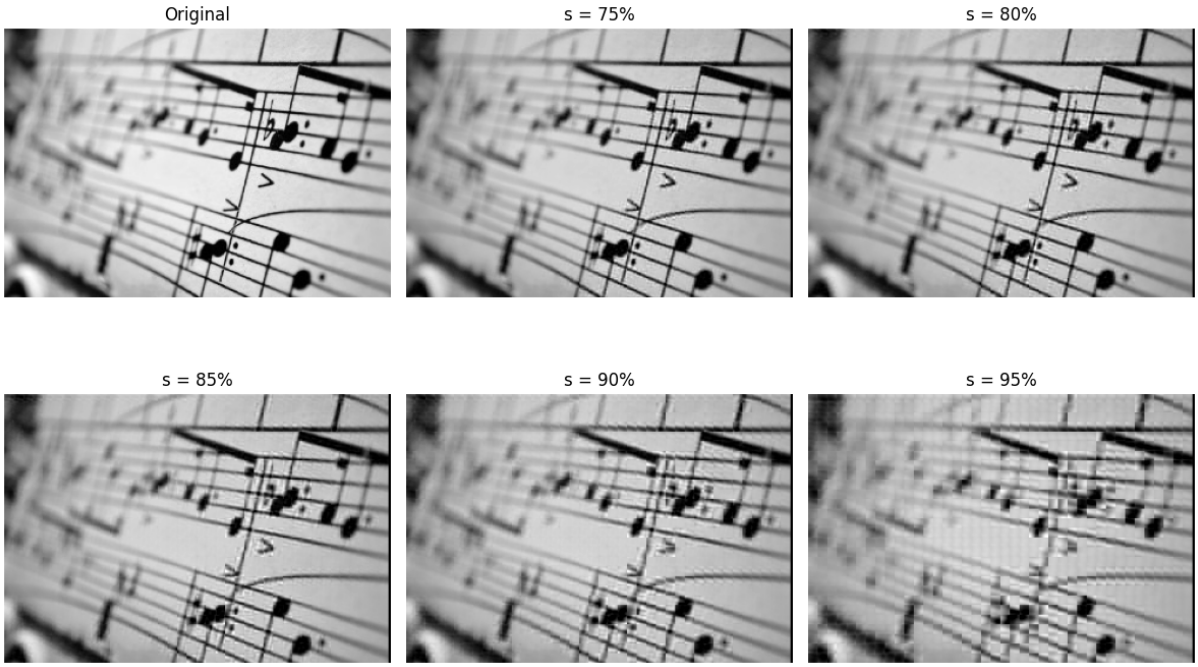

Figura 8. Imagen original vs las reconstruidas para diferentes S.

Vemos que a medida que S aumenta, la imagen reconstruida pierde nitidez. La reducción de componentes principales se ve en una disminución de la calidad de la imagen.

Conclusiones

En este trabajo práctico implementamos y evaluamos la compresión de imágenes usando PCA (Análisis de Componentes Principales). Este permite reducir la dimensión de los datos al descartar las componentes de menor varianza, quedándose con las componentes principales, es decir aquellas necesarias para la comprensión de la imagen. De esta manera, se ahorra espacio sin comprometer significativamente la calidad de la imagen.

Los resultados mostraron que, a medida que aumenta el porcentaje de espacio ahorrado, la calidad de la imagen reconstruida decae progresivamente, especialmente a partir de la reducción de 80 %. Esto se refleja en un incremento notable del MSE. Sin embargo, consideramos que para valores intermedios de compresión (cercanos al 70 %), obtenemos un buen balance entre nitidez de la imagen y reducción de memoria.

Los problemas a lo largo de este trabajo práctico fueron mínimos, consistindo principalmente en la eleccion de funciones de librerías para el cálculo de autovalores y autovectores, y en la implementación de las funciones de compresión y descompresión. Por lo que se tuvo que comprender el formato de los datos con los que trabajan estas librerías, llevando lo teorico a la práctica.

En conclusión, la aplicación de PCA para la compresión de imágenes nos permitió observar de primera mano cómo la reducción de dimensionalidad ayuda en el ahorro de almacenamiento, sin comprometer demasiado la calidad de la imagen.