Taller - Imágenes Médicas
Manejo de histogramas

William Gómez Roa
wa.gomez@javeriana.edu.co
Bioingeniería y Ciencia de Datos

Carolina Santos Baquero
carolinasantosb@javeriana.edu.co
Bioingeniería

Resumen

En este taller, se abordaron técnicas esenciales de procesamiento de imágenes médicas utilizando MATLAB y Python. Se realizaron ejercicios prácticos que incluyeron normalización, ecualización y umbralización de histogramas, tanto de forma manual como automática. Se compararon los resultados de la umbralización con un único umbral y la umbralización local, evaluando su impacto visual y relevancia para el diagnóstico médico. Además, se aplicó el método de Otsu con la librería ITK en Python para binarizar imágenes médicas, examinando la capacidad de estas operaciones para resaltar información diagnóstica clave. El taller proporcionó una experiencia importante en el ámbito del procesamiento de imágenes médicas y su aplicación en el diagnóstico clínico.

1 Normalización de Histograma

Ejercicio 1:

Normalización lineal del histograma en la imagen butterfly.png en escala de grises en MATLAB

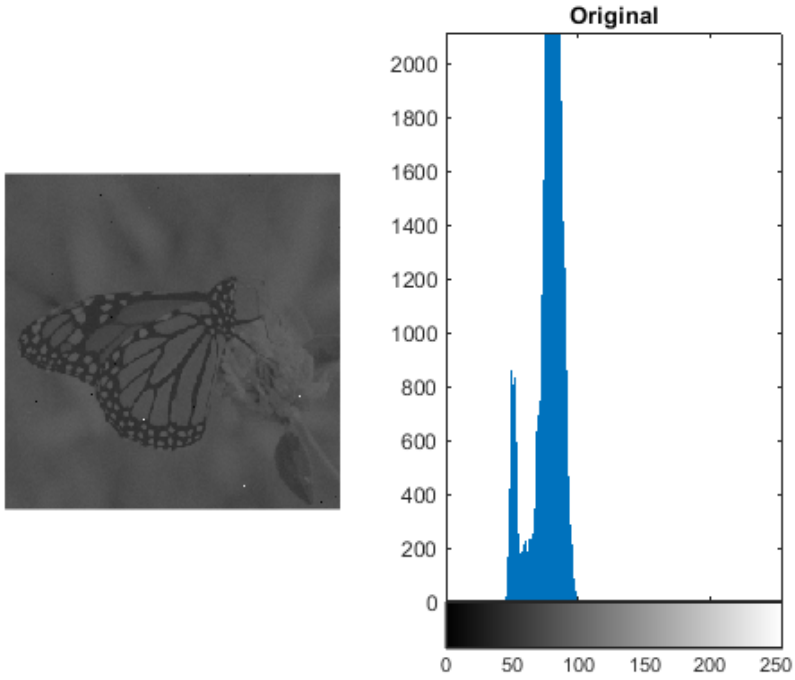

Figure 1: *Imagen original de la mariposa y su histograma*

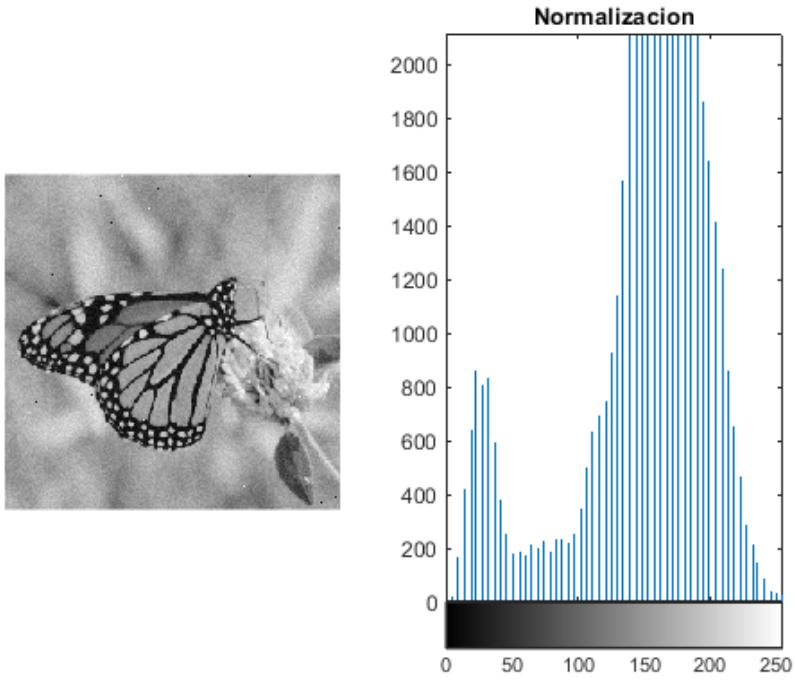

Figure 2: *Imagen normalizada de la mariposa y su histograma*

```
% cargar la imagen
img=imread("butterfly.png");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);
% definir las intensidades en los extremos del rango actual de
   representacion
% es decir, el rango dentro del cual se quiere normalizar
% (se especifican manualmente)
min=45;
max=100;
% trabajar sobre una copia de la imagen con tipo de dato real
% para utilizar mayor precision en los calculos
realimg=double(grayimg);
% aplicar la ecuacion de normalizacion del histograma
normimg=(realimg-min)./(max-min);
% multiplicar por el valor maximo del rango de representacion
normimg=normimg.*255;
% convertir de nuevo a tipo de dato entero para facilitar la visualizacion
entimg=uint8(normimg);
% visualizar la imagen normalizada y su histograma
figure(2)
subplot(1,2,1)
imshow(entimg);
subplot(1,2,2)
imhist(entimg);
% guardar la imagen normalizada en disco
imwrite(entimg,"butterfly_norm.png");
```

2 Ecualización del histograma (manual)

Ejercicio 2:

Ecualización del histograma de una imagen butterfly.png en escala de grises en MATLAB

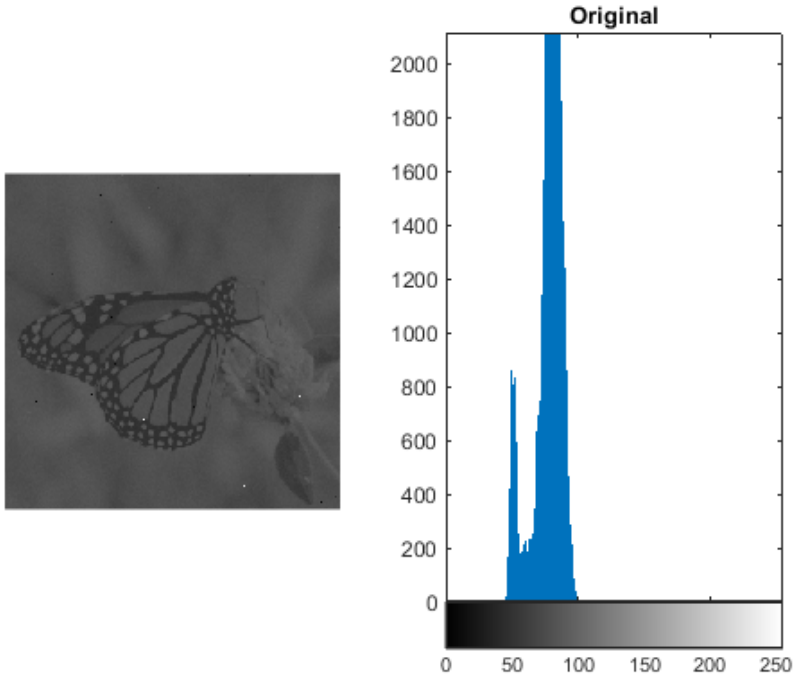

Figure 3: *Imagen original de la mariposa y su histograma*

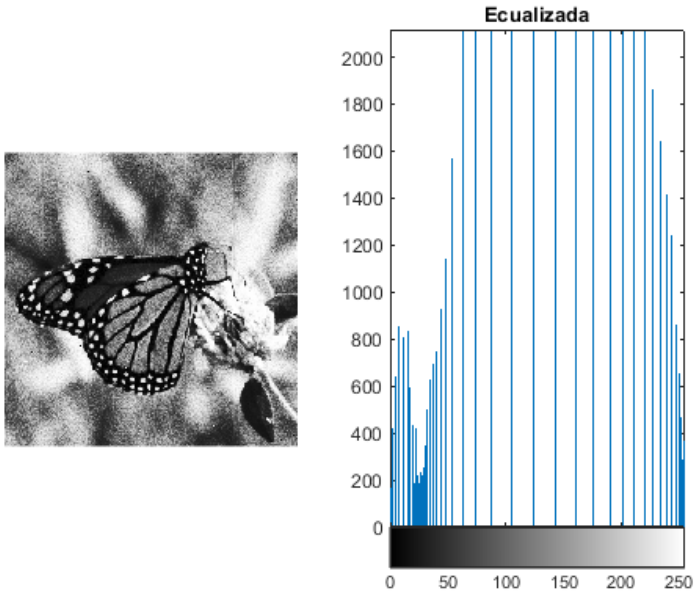

Figure 4: *Imagen ecualizada de la mariposa y su histograma*

```
% cargar la imagen
img=imread("butterfly.png");
```

```
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);
% calcular la frecuencia de cada valor de intensidad
[x,y]=size(grayimg);
Freq=zeros(1,256);
for i=1:x
 for j=1:y
 Freq(grayimg(i,j)+1)=Freq(grayimg(i,j)+1)+1;
 end
end

% calcular la funcion de densidad de probabilidad para cada intensidad
PDF=zeros(1,256);
Total=x*y;
for i=1:256
 PDF(i)=Freq(i)/Total;
end

% calcular la funcion de densidad acumulada para cada intensidad
CDF=zeros(1,256);
CDF(1)=PDF(1);
for i=2:256
 CDF(i)=CDF(i-1)+PDF(i);
end

% multiplicar por el maximo valor de intensidad del rango
Result=zeros(1,256);
for i=1:256
 Result(i)=CDF(i)*(255);
end

% generar la imagen ecualizada
eqimg=uint8(zeros(size(grayimg)));
for i=1:x
 for j=1:y
 eqimg(i,j)=uint8(Result(grayimg(i,j)+1));
 end
```

```
end
% visualizar la imagen ecualizada y su histograma
figure(2)
subplot(1,2,1)
imshow(eqimg);
subplot(1,2,2)
imhist(eqimg);
% guardar la imagen ecualizada en disco
imwrite(eqimg,"butterfly_eq.png");
```

Ejercicio 3: Comparación visualmente de los resultados obtenidos con la mariposa

Las diferencias observadas tanto sobre la imagen como sobre el histograma, es que en la imagen original se puede observar oscura, y no se logra evidencias claramente los detalles de la mariposa, por lo que, al aplicar la normalización en una escala dentro del rango 45 a 100, se logra evidenciar una mejor apariencia visual en términos de contraste y equilibrio, pero no se logra resaltar detalles en áreas específicas. Por otra parte, al aplicar ecualización, se ilustra una mejora en el contraste en comparación a la original, puesto que se distribuyen los valores de los pixeles, resaltándose detalles, mejorando la visibilidad de las características que tiene la mariposa en contacto con la flor, y como este redistribuye los valores de píxeles de manera que los valores más comunes se vuelven menos comunes y viceversa. Esto tiende a aumentar el contraste y destacar los detalles en la imagen, realzando detalles y mejorando el contraste en áreas específicas de una imagen como es en la fotografía original con poca iluminación.

Por otra parte, el histograma de la imagen original muestra cómo se distribuyen los valores de píxeles en la imagen de la mariposa antes de aplicar la normalización y la ecualización, en el que se observa un rango de valores entre los grises oscuros, es decir, que no abarco todo el espectro posible, asimismo, tiene picos y valles, lo que indica variaciones en el contraste. Por otra parte, en la normalización, el histograma se ajustó para abarcar todo el rango posible de valores de píxeles, esto quiere decir, que los valores de píxeles se redistribuyen para que estén dentro de un rango específico entre 45 a 100 en escala de grises, resultando más uniforme y equilibrado en comparación con la imagen original. El histograma resultante de la ecualización tiende a ser más uniforme y plano, lo que significa que los valores de píxeles se han redistribuido para abarcar todo el rango de valores posibles. Los valores que eran más comunes en el histograma original pueden volverse menos comunes y viceversa, teniendo como referencia la imagen original. Comparando solo los histogramas de la normalización y ecualización, se puede evidenciar que la normalización ajusta los valores de píxeles para que se encuentren dentro del rango específico que se le ordena, lo que puede mejorar el contraste y el equilibrio, pero no resalto detalles en la mariposa. En cambio, la ecualización del histograma, redistribuye los valores de píxeles, mejorando el contraste y destacando detalles en la imagen. No podemos decir si la normalización o ecualización es mejor, ya que depende de lo que se esté tratando de lograr y en las instrucciones no lo especifica. La normalización genera el mejor resultado cuando se desea que la imagen se vea más natural y realista en términos de contraste, como se observa con la mariposa, asimismo cuando no se desea resaltar detalles específicos, sino simplemente mejorar el equilibrio, pero por otro lado, la ecualización visualmente hablando genera mejor resultado si quieres mejorar el contraste al resaltar detalles y hacer que las características de la imagen sean más visibles, realzando características específicas de la mariposa y flor, esto hace que la imagen se vea menos natural.

3 Ecualización del histograma (función de Matlab)

Ejercicio 4: Utilizar la función nativa de MATLAB para la ecualización del histograma de la imagen butterfly.png en escala de grises.

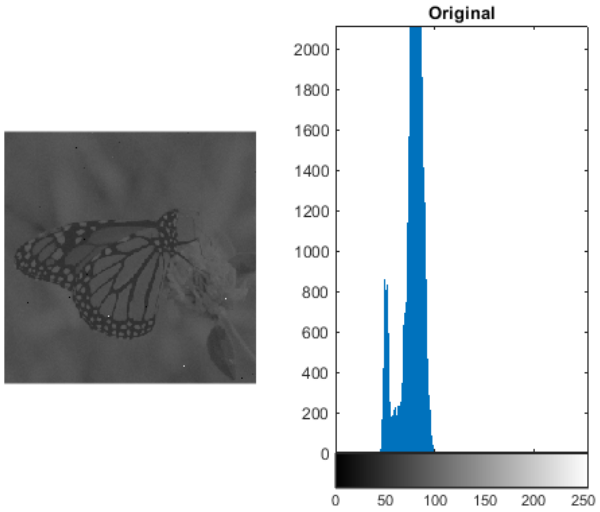

Figure 5: *Imagen original de la mariposa y su histograma*

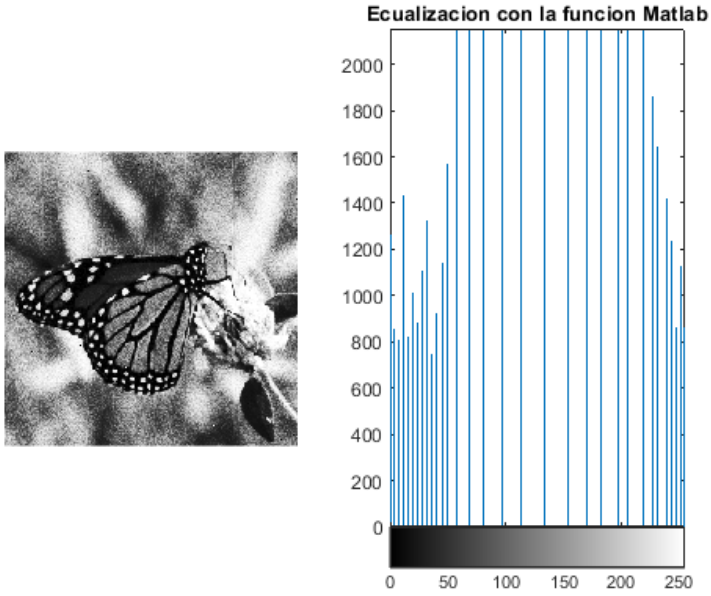

Figure 6: *Imagen ecualizada de la mariposa y su histograma con al funcion en MATLAB*

```
% cargar la imagen
img=imread("butterfly.png");
```

```
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);
% ecualizar el histograma de la imagen usando la funcion propia de MATLAB
eqimg = histeq(grayimg);
% visualizar la imagen ecualizada y su histograma
figure(2)
subplot(1,2,1)
imshow(eqimg);
subplot(1,2,2)
imhist(eqimg);
% guardar la imagen ecualizada en disco
imwrite(eqimg,"butterfly_eqM.png");
```

Ejercicio 5: Comparación del procesos de ecualización manual y la función nativa de MATLAB.

Cuando se realiza la ecualización del histograma de forma manual, se debe calcular y aplicar la transformación en el histograma de la imagen uno mismo. Este proceso implica redistribuir los valores de píxeles para aumentar el contraste y resaltar detalles. La imagen resultante es más brillante y los detalles oscuros se vuelven más visibles. Además, el histograma de la imagen manualmente ecualizada muestra una distribución de valores más uniforme, con una mayor dispersión de valores de píxeles a lo largo del rango. Los valores que eran inicialmente menos comunes se vuelven más comunes, y el histograma tendrá menos picos y valles. Por otra parte, la ecualización con histeq en MATLAB, el proceso de ecualización se realiza automáticamente, reduciendo las líneas de código. La imagen se modifica directamente por la función, teniendo un mayor contraste y los detalles se resaltarán de manera similar a la ecualización manual, en donde no se logra ver mucha diferencia. Ya si comparamos el histograma después de aplicar histeq será similar al resultado de la ecualización manual, pero se puede ver algunas diferencias, ya que se tiene un rango distinto, pero, aun así, la distribución de valores es uniforme con menos picos y valles, lo que indica una mejora en el contraste de la imagen comparando el histograma original. Lo importante es que las dos formas, tiene como finalidad mejorar el contraste y resaltar detalles en la imagen al redistribuir los valores de píxeles, llegando a ser más uniforme en el histograma.

4 Binarización y umbralización usando el método de Otsu

Ejercicio 6: Binarización y umbralización usando el método de Otsu

Estimación automática del umbral que permite posteriormente la binarización y umbralización de la imagen mushrooms.jpg en escala de grises en MATLAB

```
clc,clear, close all;
% cargar la imagen
img=imread("mushrooms.jpg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);

% calcular automaticamente el umbral utilizando el metodo de Otsu
level = graythresh(grayimg);

% Mostrar el valor del umbral en la consola
fprintf('Valor del umbral calculado: %f\n', level);

% binarizar utilizando el umbral calculado
binimg = imbinarize(grayimg,level);
% multiplicar la imagen original por la imagen binarizada para
% obtener la imagen umbralizada
umbimg = uint8(binimg) .* grayimg;

% visualizar y guardar la imagen binarizada
figure(2)
imshow(binimg);
imwrite(binimg,"mushrooms_bin.jpg");
% visualizar y guardar la imagen umbralizada
figure(3)
imshow(umbimg);
imwrite(umbimg,"mushrooms_umb.jpg");
```

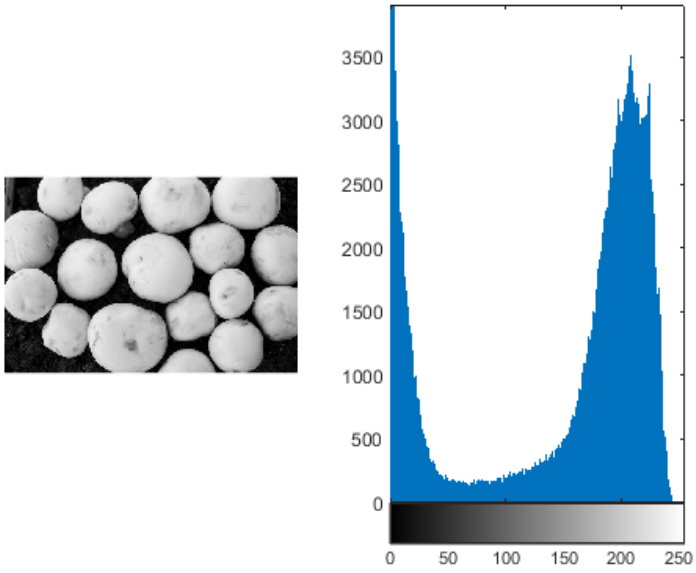

Figure 7: *visualizar la imagen de champiñones original y su histograma*

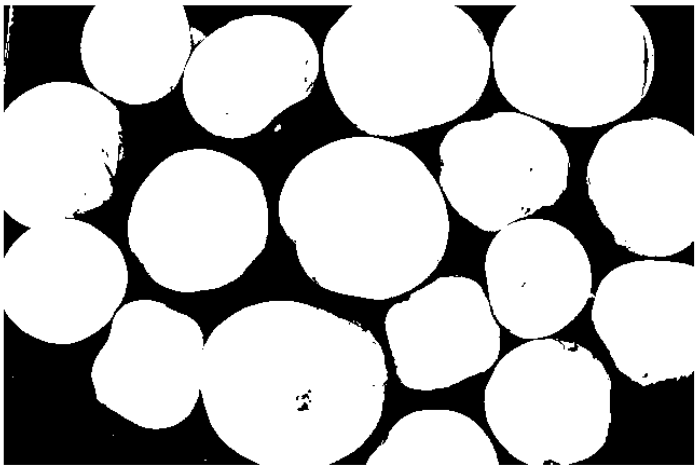

Figure 8: *Imagen binarizada de los champiñones*

Figure 9: *imagen umbralizada de los champiñones*

El valor del umbral calculado es 0.427451

5 Binarización local usando el método de Otsu

Ejercicio 7: Binarización local usando el método de Otsu

Estimación automática de los umbrales locales (con diferentes tamaños de ventana) que permiten la binarización local de la imagen page.jpeg en escala de grises en MATLAB:

```
% cargar la imagen
img=imread("page.jpeg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);

% encontrar los umbrales locales por ventanas de 50x50
loc50img=nlfilter(grayimg,[50 50], @ibimage);

% visualizar y guardar la imagen binarizada localmente
figure(2)
imshow(loc50img);
imwrite(loc50img,"page_bin50.jpeg");
```

```
% encontrar los umbrales locales por ventanas de 30x30
loc30img=nlfilter(grayimg,[30 30], @ibimage);

% visualizar y guardar la imagen binarizada localmente
figure(3)
imshow(loc30img);
imwrite(loc30img,"page_bin30.jpeg");

% funcion ibimage
% encuentra el umbral en cada ventana de la imagen
function f=ibimage(img)
 [x, y]=size(img);
 level=graythresh(img);
 bw=imbinarize(img,level);
 x1=round(x/2);
 y1=round(y/2);
 f=bw(x1, y1);
end
```

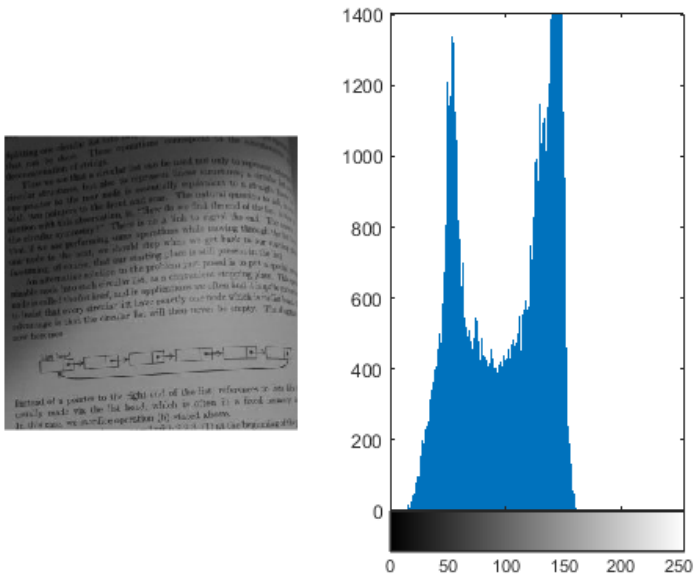

Figure 10: *Imagen original de la pagina y su histograma*

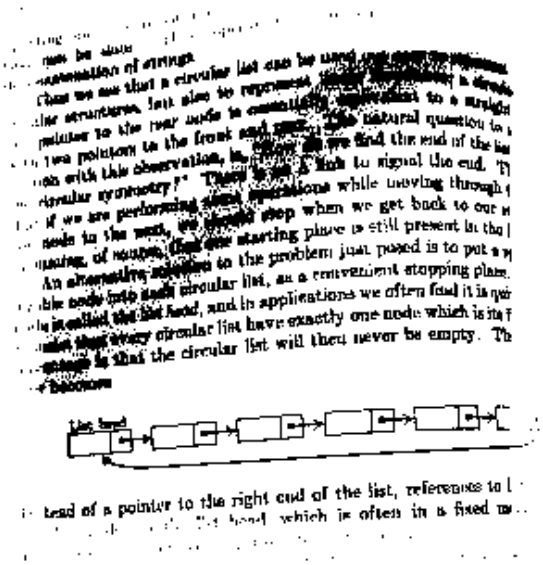

Figure 11: *Imagen de la pagina con binarizada localmente con ventanas de 50x50*

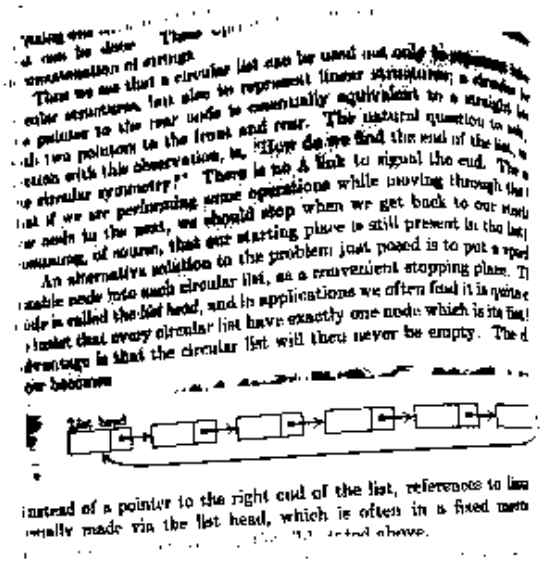

Figure 12: *imagen binarizada localmente con ventanas 30x30*

Ejercicio 8: Comparación de los resultados obtenidos con un único umbral y con umbrales locales de la imagen page.jpeg

```
clc,clear, close all;
% cargar la imagen
img=imread("page.jpeg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);

% calcular automaticamente el umbral utilizando el metodo de Otsu
level = graythresh(grayimg);

% binarizar utilizando el umbral calculado
binimg = imbinarize(grayimg,level);
% multiplicar la imagen original por la imagen binarizada para
% obtener la imagen umbralizada
umbimg = uint8(binimg) .* grayimg;

% visualizar y guardar la imagen binarizada
figure(2)
imshow(binimg);
imwrite(binimg,"page_bin.jpg");
% visualizar y guardar la imagen umbralizada
figure(3)
imshow(umbimg);
imwrite(umbimg,"page_umb.jpg");
```

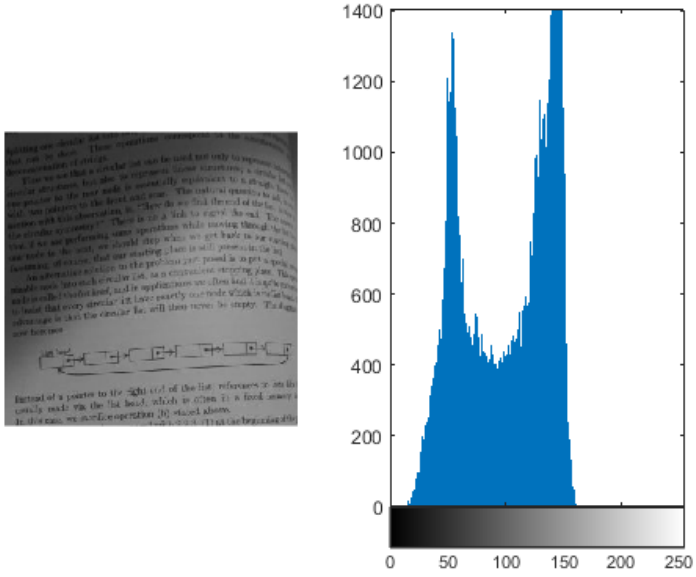

Figure 13: *imagen original de la pagina y su histograma*

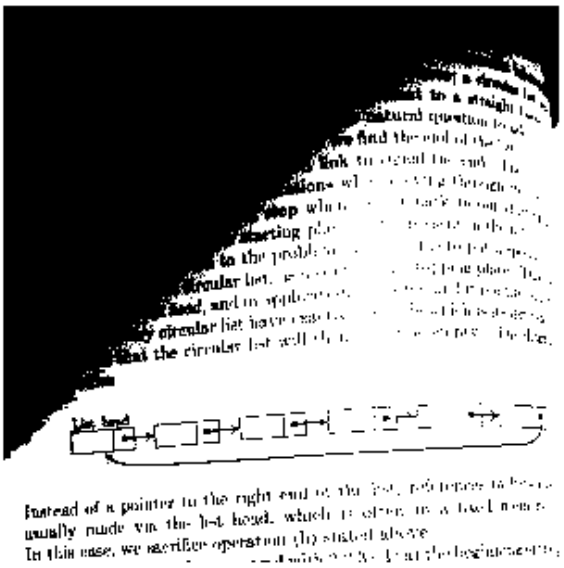

Figure 14: *Imagen binarizada de la pagina*

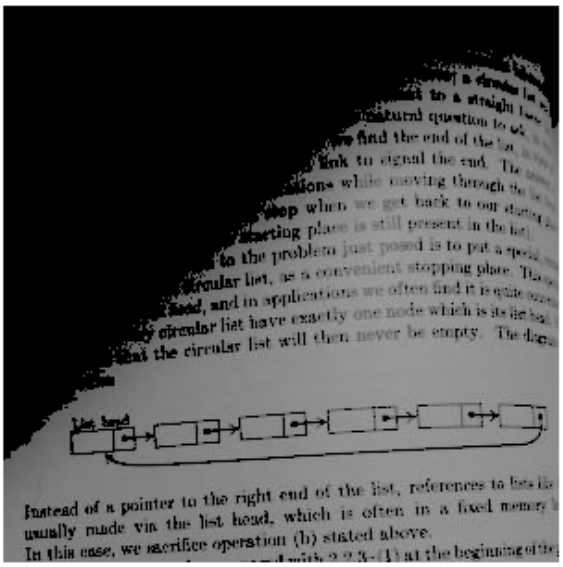

Figure 15: *Imagen umbralizada de la pagina*

La comparación que encontramos entre un único umbral y con umbrales locales, fue que los umbrales locales son más sensibles a las variaciones en los datos, lo que permitió adaptarse mejor a situaciones de la página, donde diferentes partes de los datos tienen características distintas, asimismo captura detalles locales en los datos, en cambio, el del único umbral paso por alto estas variaciones, proporcionando un rendimiento global simple. Observando con la imagen original, la mejor opción en este caso es usar umbrales locales, ya que los datos son heterogéneos y la idea es obtener la mayoría de información que se muestra en la imagen con buena calidad.

Ejercicio 9: Replicando el proceso completo con dos imágenes médicas de diferentes modalidades

1. Normalización del histograma

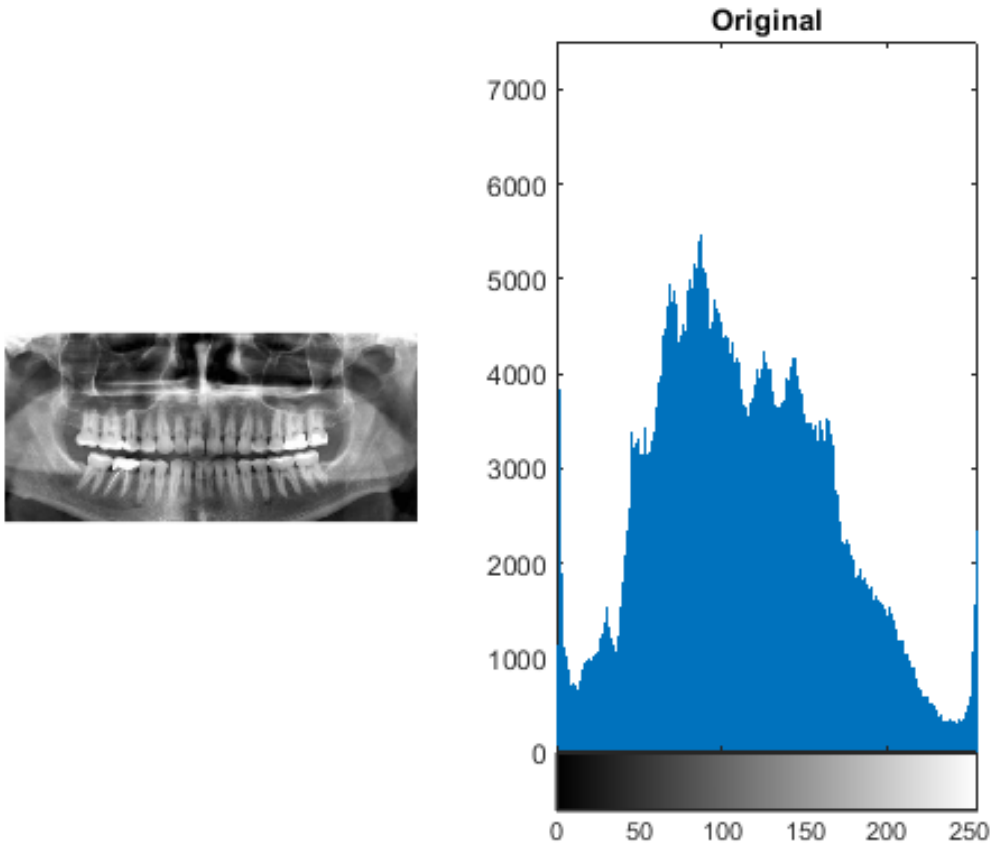

Figure 16: *Imagen original de una imagen de radiografia y su histograma*

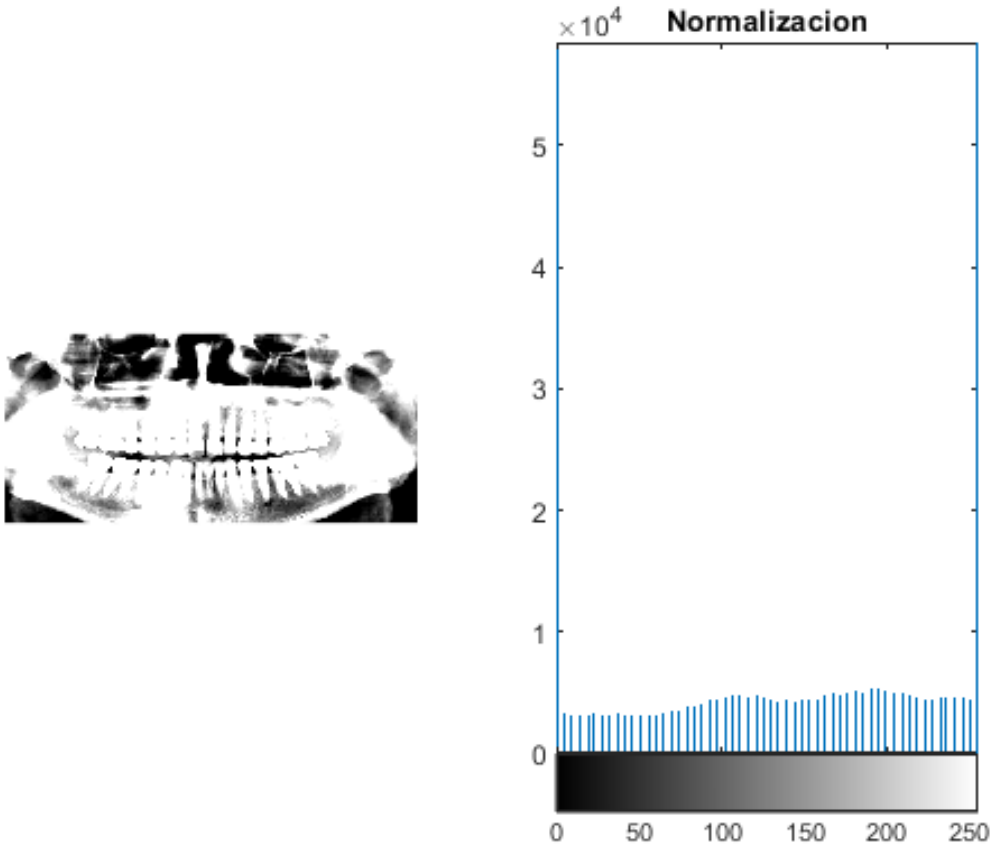

Figure 17: *Imagen con normalización de una imagen de radiografia y su histograma*

```
% cargar la imagen
img=imread("radiografia.jpg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);
title('Original');
% definir las intensidades en los extremos del rango actual de
    representacion
% es decir, el rango dentro del cual se quiere normalizar
% (se especifican manualmente)
min=45;
max=100;
```

```
% trabajar sobre una copia de la imagen con tipo de dato real
% para utilizar mayor precision en los calculos
realimg=double(grayimg);
% aplicar la ecuacion de normalizacion del histograma
normimg=(realimg-min)./(max-min);
% multiplicar por el valor maximo del rango de representacion
normimg=normimg.*255;
% convertir de nuevo a tipo de dato entero para facilitar la visualizacion
entimg=uint8(normimg);
% visualizar la imagen normalizada y su histograma
figure(2)
subplot(1,2,1)
imshow(entimg);
subplot(1,2,2)
imhist(entimg);
title('Normalizacion');
% guardar la imagen normalizada en disco
imwrite(entimg,"radiografia.png");
```

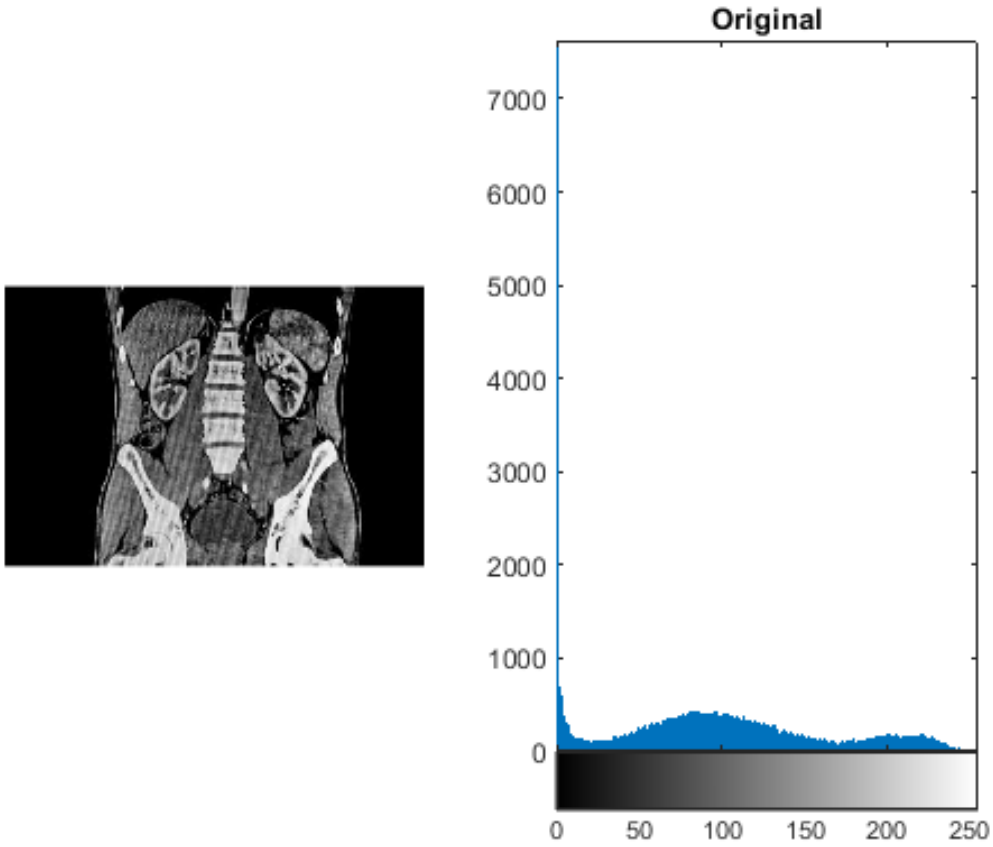

Figure 18: *Imagen original de una imagen de tomografia computacional y su histograma*

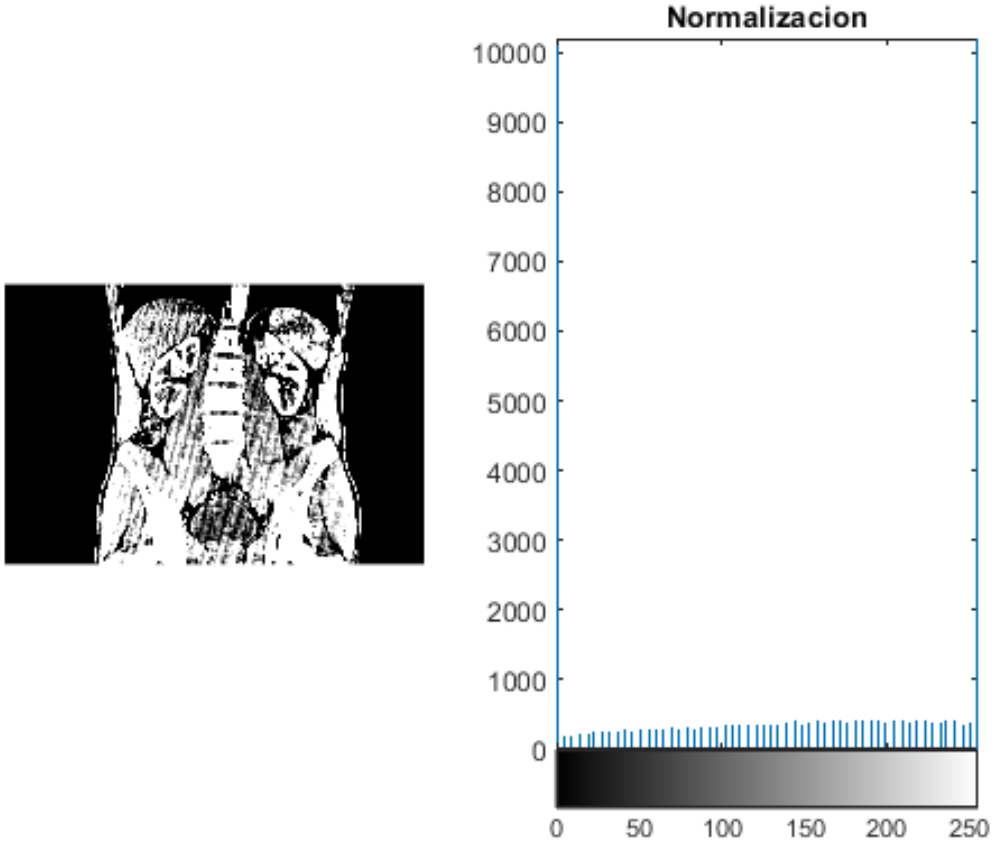

Figure 19: *Imagen con normalización de una imagen de tomografia computacional y su histograma*

```
% cargar la imagen
img=imread("abdomen.jpg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);
title('Original');
% definir las intensidades en los extremos del rango actual de
    representacion
% es decir, el rango dentro del cual se quiere normalizar
% (se especifican manualmente)
min=45;
max=100;
```

```
% trabajar sobre una copia de la imagen con tipo de dato real
% para utilizar mayor precision en los calculos
realimg=double(grayimg);
% aplicar la ecuacion de normalizacion del histograma
normimg=(realimg-min)./(max-min);
% multiplicar por el valor maximo del rango de representacion
normimg=normimg.*255;
% convertir de nuevo a tipo de dato entero para facilitar la visualizacion
entimg=uint8(normimg);
% visualizar la imagen normalizada y su histograma
figure(2)
subplot(1,2,1)
imshow(entimg);
subplot(1,2,2)
imhist(entimg);
title('Normalizacion');
% guardar la imagen normalizada en disco
imwrite(entimg,"abdomen.png");
```

2. Ecualización del histograma (ejercicios 2)

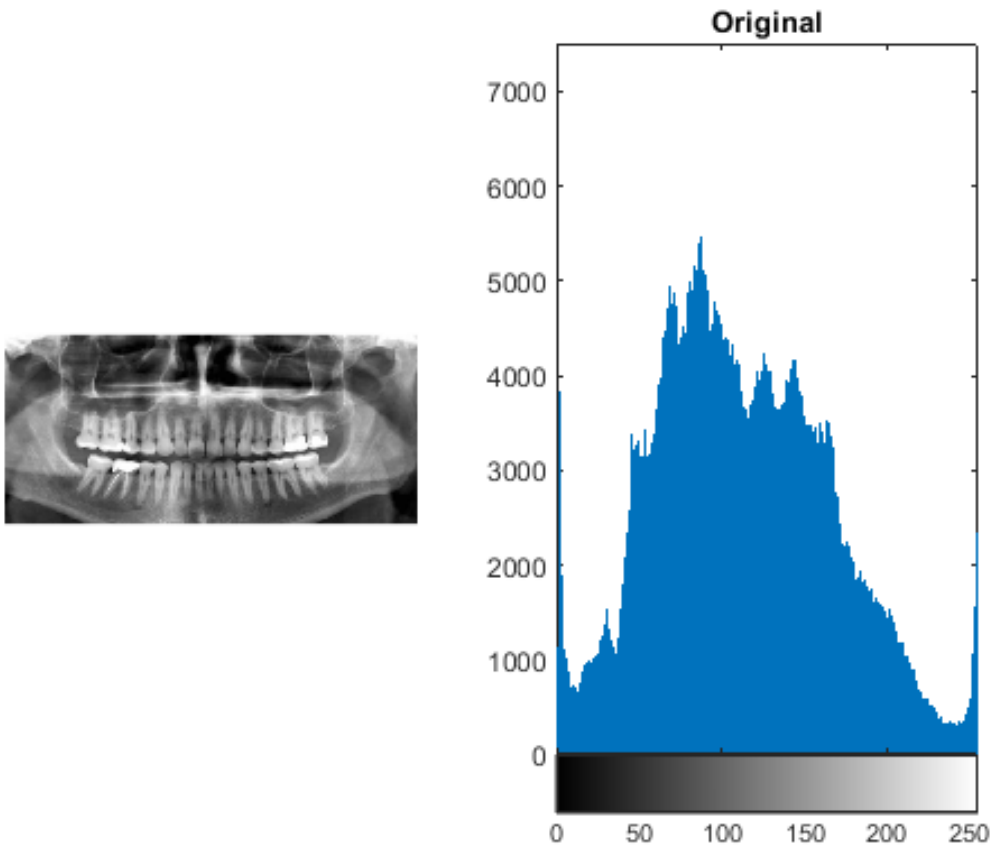

Figure 20: *Imagen original de una imagen de radiografia y su histograma*

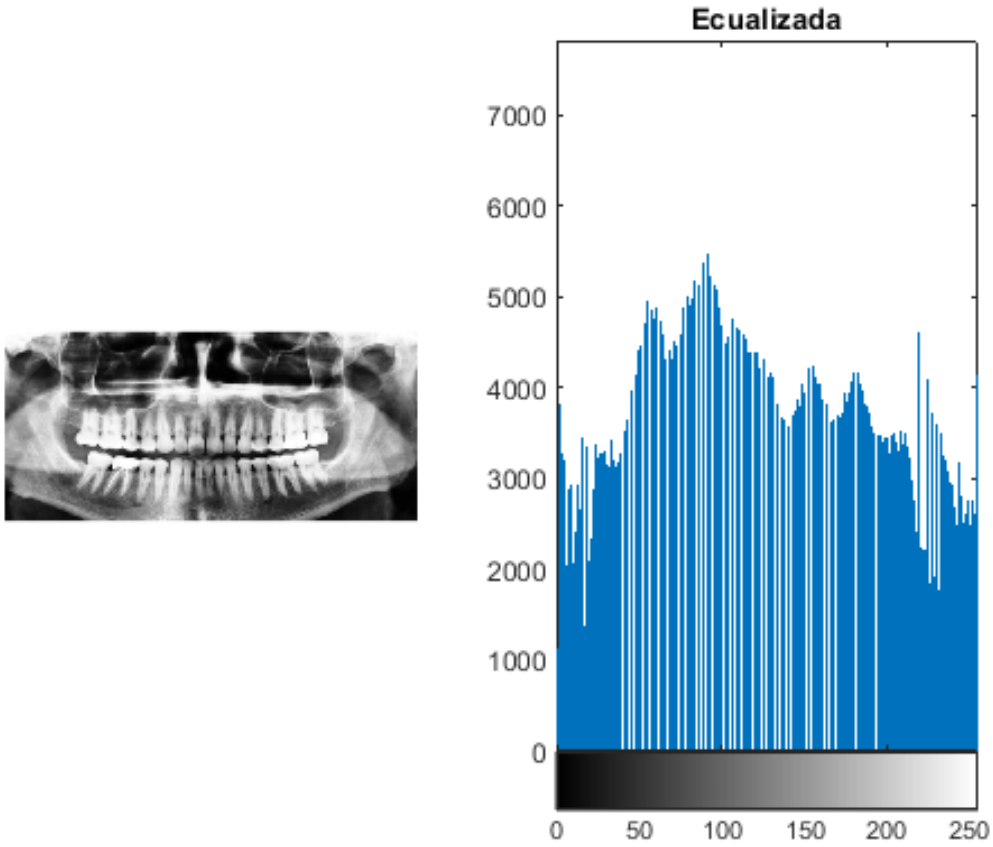

Figure 21: *Imagen con ecualizada de una imagen de radiografia y su histograma*

```
% cargar la imagen
img=imread("radiografia.jpg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);
title('Original');
% calcular la frecuencia de cada valor de intensidad
[x,y]=size(grayimg);
Freq=zeros(1,256);
for i=1:x
 for j=1:y
 Freq(grayimg(i,j)+1)=Freq(grayimg(i,j)+1)+1;
```

```
 end
end

% calcular la funcion de densidad de probabilidad para cada intensidad
PDF=zeros(1,256);
Total=x*y;
for i=1:256
 PDF(i)=Freq(i)/Total;
end

% calcular la funcion de densidad acumulada para cada intensidad
CDF=zeros(1,256);
CDF(1)=PDF(1);
for i=2:256
 CDF(i)=CDF(i-1)+PDF(i);
end

% multiplicar por el maximo valor de intensidad del rango
Result=zeros(1,256);
for i=1:256
 Result(i)=CDF(i)*(255);
end

% generar la imagen ecualizada
eqimg=uint8(zeros(size(grayimg)));
for i=1:x
 for j=1:y
 eqimg(i,j)=uint8(Result(grayimg(i,j)+1));
 end
end
% visualizar la imagen ecualizada y su histograma
figure(2)
subplot(1,2,1)
imshow(eqimg);
subplot(1,2,2)
imhist(eqimg);
title('Ecualizada');
% guardar la imagen ecualizada en disco
imwrite(eqimg,"radiografia.png");
```

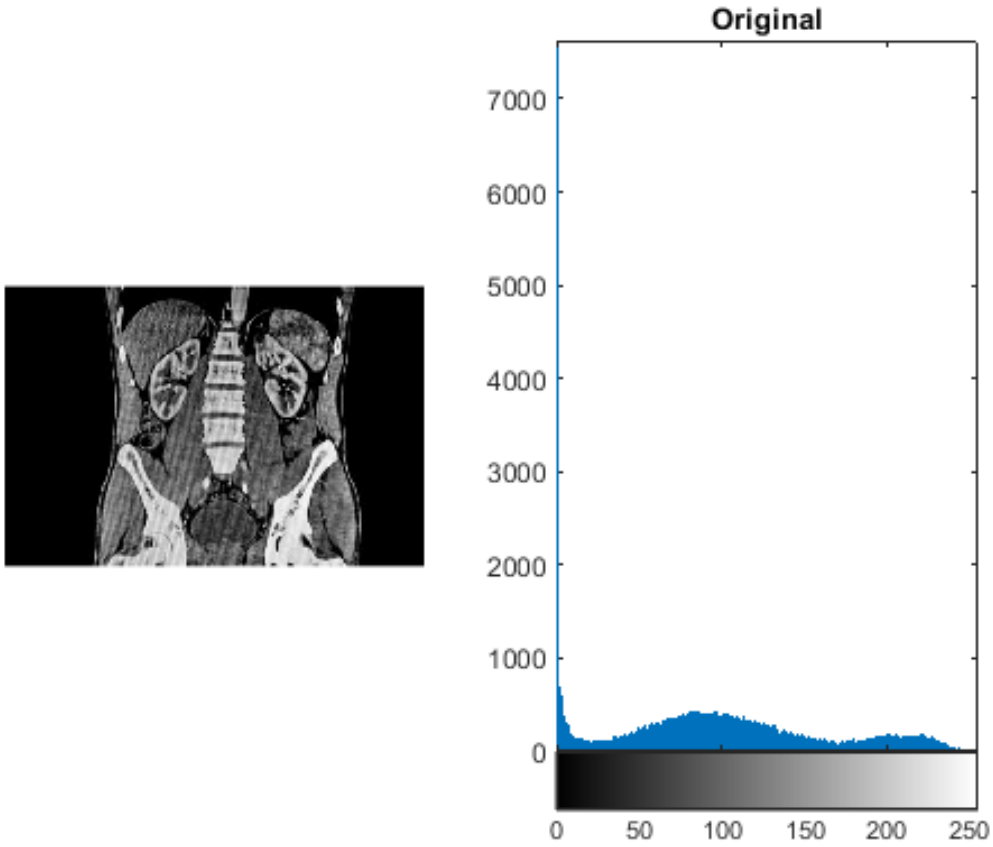

Figure 22: *Imagen original de una imagen de tomografia computacional y su histograma*

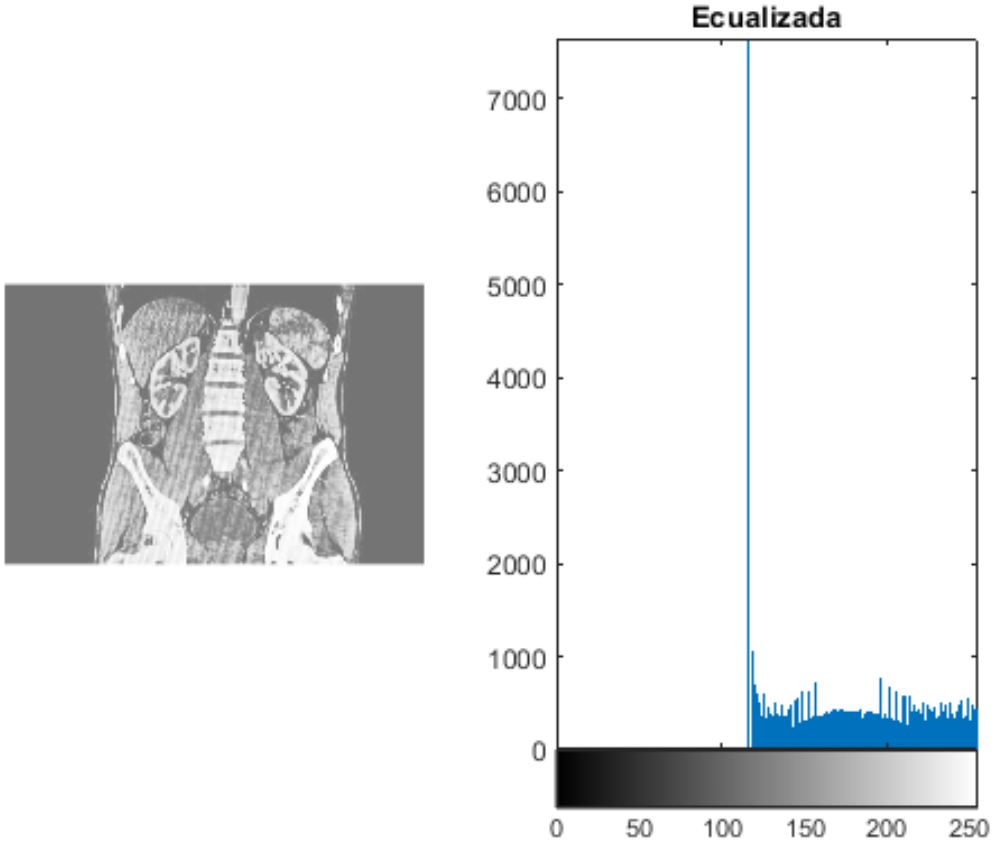

Figure 23: *Imagen con ecualizada de una imagen de tomografia computacional y su histograma*

```
% cargar la imagen
img=imread("abdomen.jpg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);
title('Original');
% calcular la frecuencia de cada valor de intensidad
[x,y]=size(grayimg);
Freq=zeros(1,256);
for i=1:x
 for j=1:y
 Freq(grayimg(i,j)+1)=Freq(grayimg(i,j)+1)+1;
```

```
 end
end

% calcular la funcion de densidad de probabilidad para cada intensidad
PDF=zeros(1,256);
Total=x*y;
for i=1:256
 PDF(i)=Freq(i)/Total;
end

% calcular la funcion de densidad acumulada para cada intensidad
CDF=zeros(1,256);
CDF(1)=PDF(1);
for i=2:256
 CDF(i)=CDF(i-1)+PDF(i);
end

% multiplicar por el maximo valor de intensidad del rango
Result=zeros(1,256);
for i=1:256
 Result(i)=CDF(i)*(255);
end

% generar la imagen ecualizada
eqimg=uint8(zeros(size(grayimg)));
for i=1:x
 for j=1:y
 eqimg(i,j)=uint8(Result(grayimg(i,j)+1));
 end
end
% visualizar la imagen ecualizada y su histograma
figure(2)
subplot(1,2,1)
imshow(eqimg);
subplot(1,2,2)
imhist(eqimg);
title('Ecualizada');
% guardar la imagen ecualizada en disco
imwrite(eqimg,"abdomen.png");
```

3. Ecualización del histograma (ejercicios 3)

Las diferencias observadas en la radiografia tanto sobre la imagen como sobre el histograma, es que en la imagen original se puede observar oscura, y no se logra evidencias claramente los detalles de la radiografia, por lo que, al aplicar la normalización en una escala dentro del rango 45 a 100, se logra evidenciar una mejor apariencia visual en términos de contraste y equilibrio, pero no se logra resaltar detalles en áreas específicas, ya que se genero un exceso de brillo, lo que hace que no se pueda ver claramente los dientes. Por otra parte, al aplicar ecualización, se ilustra una mejora en el contraste en comparación a la original, puesto que se distribuyen los valores de los pixeles, resaltándose detalles, mejorando la visibilidad de las características que tiene la radiografia, y como este redistribuye los valores de píxeles de manera que los valores más comunes se vuelven menos comunes y viceversa. Esto tiende a aumentar el contraste y destacar los detalles en la imagen, realzando detalles y mejorando el contraste en áreas específicas de una imagen como es en la fotografía original con poca iluminación.

Por otra parte, el histograma de la imagen original muestra cómo se distribuyen los valores de píxeles en la imagen de la radiografia antes de aplicar la normalización y la ecualización, en el que se observa un rango de valores distribuidos en toda la escala, asimismo, tiene picos y valles, lo que indica variaciones en el contraste. Por otra parte, en la normalización, el histograma se ajustó para abarcar todo el rango posible de valores de píxeles, esto quiere decir, que los valores de píxeles se redistribuyen para que estén dentro de un rango específico entre 45 a 100 en escala de grises, resultando más uniforme y equilibrado en comparación con la imagen original. El histograma resultante de la ecualización tiende a ser más uniforme y plano, lo que significa que los valores de píxeles se han redistribuido para abarcar todo el rango de valores posibles. Los valores que eran más comunes en el histograma original pueden volverse menos comunes y viceversa, teniendo como referencia la imagen original. Comparando solo los histogramas de la normalización y ecualización, se puede evidenciar que la normalización ajusta los valores de píxeles para que se encuentren dentro del rango específico que se le ordena, lo que puede mejorar el contraste y el equilibrio, pero no resalto detalles en la radiografia. En cambio, la ecualización del histograma, redistribuye los valores de píxeles, mejorando el contraste y destacando detalles en la imagen. La ecualización general el mejor resultado, ya que visualmente hablando genera mejor resultado, puesto que mejora el contraste al resaltar detalles de los dientes y hace que las características de la imagen sean más visibles, realzando características específicas de la radiografia, en este caso, se ve la imagen más natural, en comparación con la normalización que se observa muy brillante, ocultando detalles que tiene la imagen.

Por otra parte, las diferencias observadas tanto sobre la imagen como sobre el histograma en la tomografia computacional, es que en la imagen original se puede observar oscura, y no se logra evidencias claramente los detalles de la tomografia computacional, por lo que, al aplicar la normalización en una escala dentro del rango 45 a 100, se logra evidenciar una mejor apariencia visual en términos de contraste y equilibrio, pero no se logra resaltar detalles en áreas específicas. Por otra parte, al aplicar ecualización, se ilustra una mejora en el contraste en comparación a la original, puesto que se distribuyen los valores de los pixeles, resaltándose detalles, mejorando la visibilidad de las características que tiene la tomografia, y como este redistribuye los valores de píxeles de manera que los valores más comunes se vuelven menos comunes y viceversa. Esto tiende a aumentar el contraste y destacar los detalles en la imagen, realzando detalles y mejorando el contraste en áreas específicas de una imagen como es en la fotografía original con poca iluminación.

Por otra parte, el histograma de la imagen original muestra cómo se distribuyen los valores de píxeles en la imagen de la tomografia antes de aplicar la normalización y la ecualización, en el que se observa un rango de valores con mayor intensidad entre los grises oscuros, asimismo, tiene picos y valles, lo que indica

variaciones en el contraste. Por otro lado, en la normalización, el histograma se ajustó para abarcar todo el rango posible de valores de píxeles, esto quiere decir, que los valores de píxeles se redistribuyen para que estén dentro de un rango específico entre 45 a 100 en escala de grises, resultando más uniforme y equilibrado en comparación con la imagen original En este caso, la normalización es el mejor resultado, ya que genera la imagen más natural y realista en términos de contraste, mejorando el equilibrio, y este hace que se mas detalles de la imagen en comparación con la ecualización.

4. Ecualización del histograma (ejercicios 4)

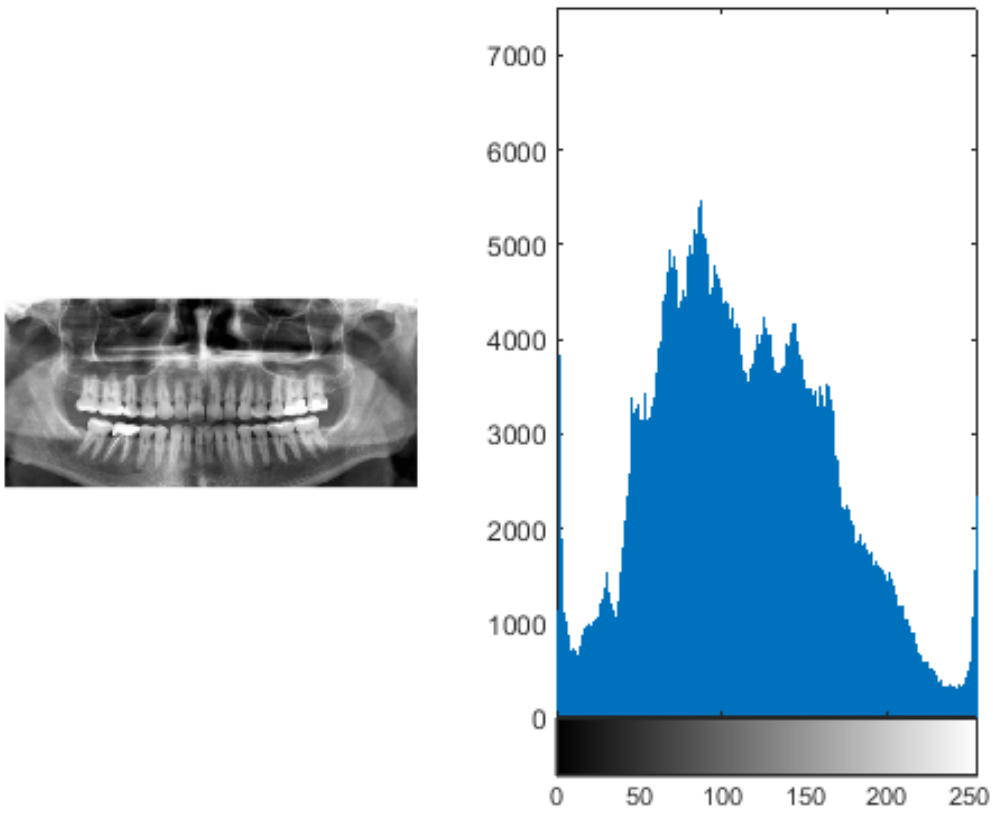

Figure 24: *Imagen original de una imagen de radiografia y su histograma*

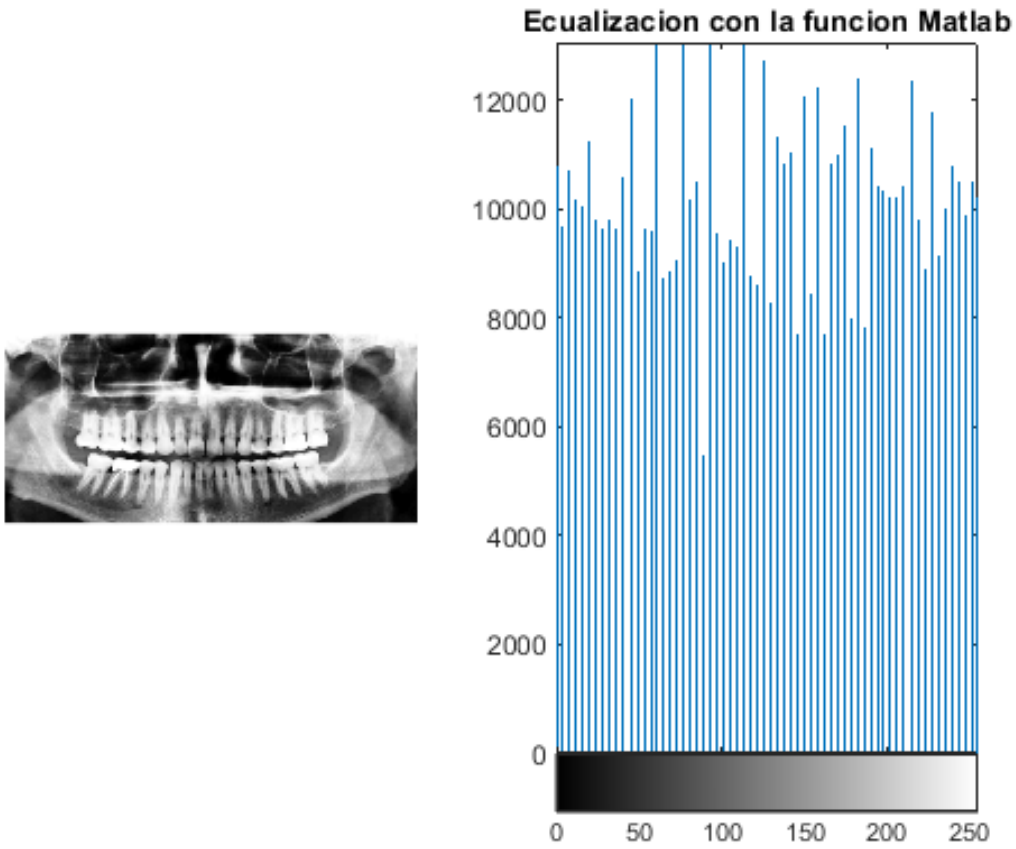

Figure 25: *Imagen con ecualizada de una imagen de radiografia usando la función de Matlab y su histograma*

```
% cargar la imagen
img=imread("radiografia.jpg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);
% ecualizar el histograma de la imagen usando la funcion propia de MATLAB
eqimg = histeq(grayimg);
% visualizar la imagen ecualizada y su histograma
figure(2)
subplot(1,2,1)
imshow(eqimg);
subplot(1,2,2)
```

```
imhist(eqimg);
title('Ecualizacion con la funcion Matlab');
% guardar la imagen ecualizada en disco
imwrite(eqimg,"radiografia.png");
```

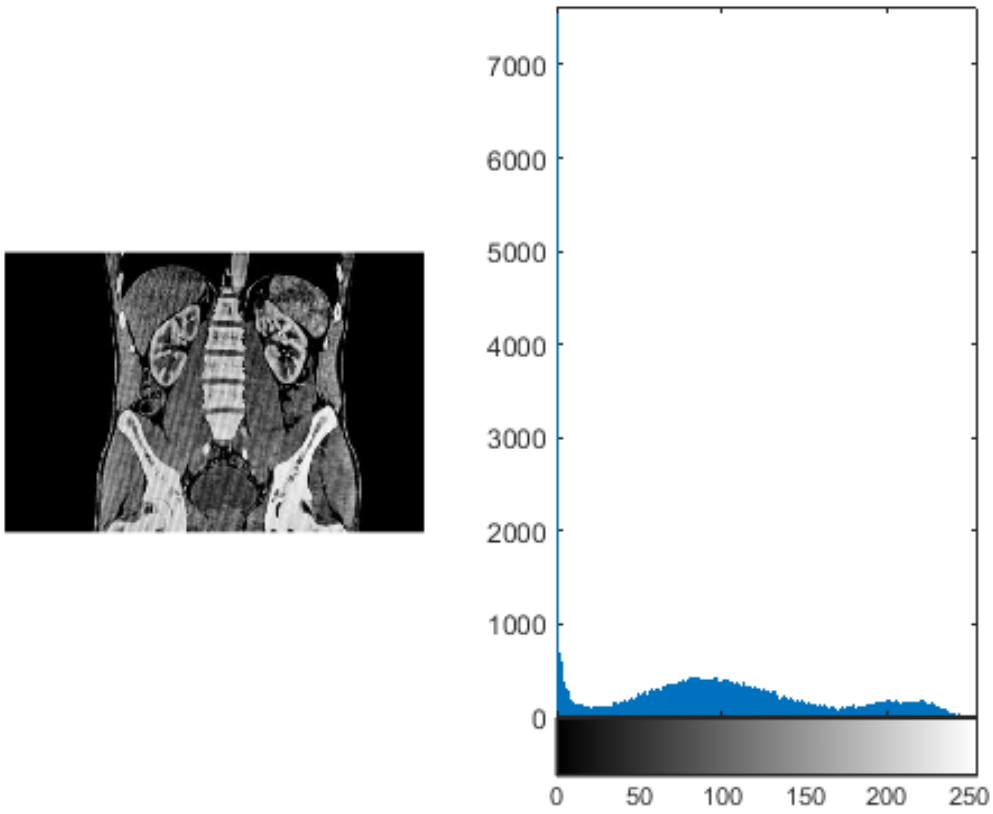

Figure 26: *Imagen original de una imagen de tomografia computacional y su histograma*

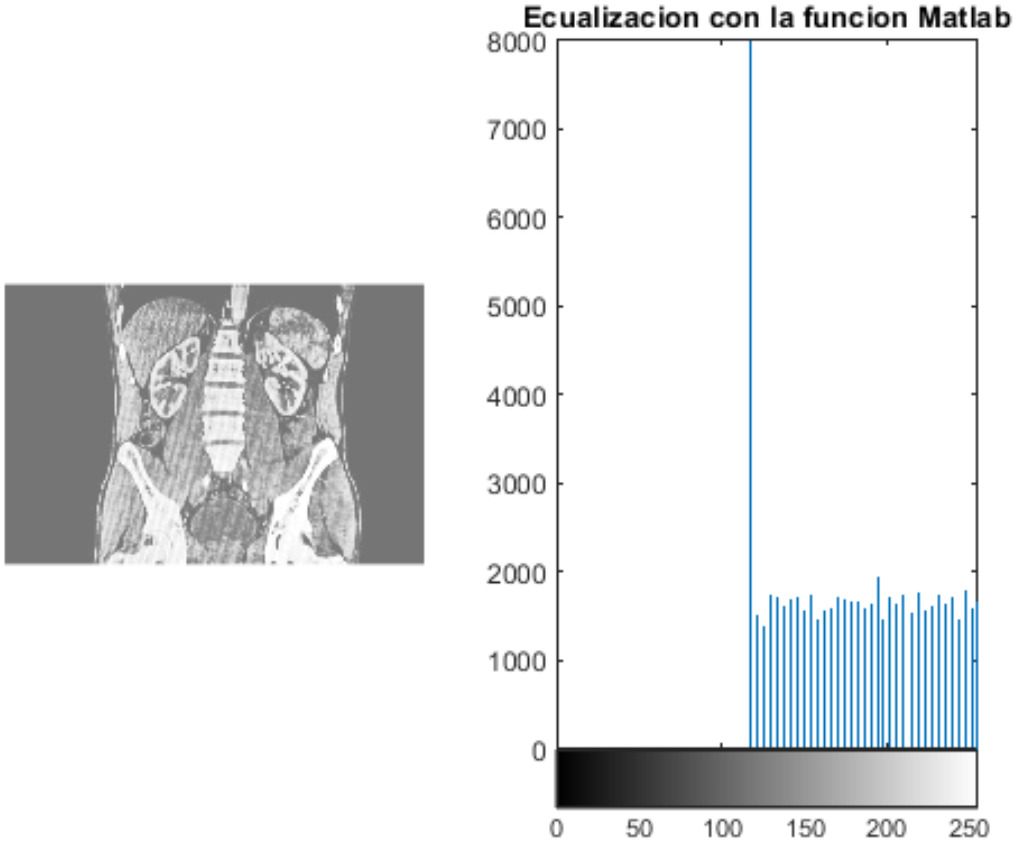

Figure 27: *Imagen con ecualización de una imagen de tomografia computacional usando la funcion de Matlab y su histograma*

```
% cargar la imagen
img=imread("abdomen.jpg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);
% ecualizar el histograma de la imagen usando la funcion propia de MATLAB
eqimg = histeq(grayimg);
% visualizar la imagen ecualizada y su histograma
figure(2)
subplot(1,2,1)
imshow(eqimg);
```

```
subplot(1,2,2)
imhist(eqimg);
title('Ecualizacion con la funcion Matlab');
% guardar la imagen ecualizada en disco
imwrite(eqimg,"abdomen.png");
```

5. Ecualización del histograma (ejercicios 5)

Cuando se realiza la ecualización del histograma de forma manual tanto para la radiografia y tomografia computacional, se debe calcular y aplicar la transformación en el histograma de la imagen uno mismo. Este proceso implica redistribuir los valores de píxeles para aumentar el contraste y resaltar detalles. La imagen resultante es más brillante y los detalles oscuros se vuelven más visibles. Además, el histograma de la imagen manualmente ecualizada muestra una distribución de valores más uniforme, con una mayor dispersión de valores de píxeles a lo largo del rango. Los valores que eran inicialmente menos comunes se vuelven más comunes, y el histograma tendrá menos picos y valles. Por otra parte, la ecualización con histeq en MATLAB, el proceso de ecualización se realiza automáticamente, reduciendo las líneas de código. La imagen se modifica directamente por la función, teniendo un mayor contraste y los detalles se resaltarán de manera similar a la ecualización manual, en donde no se logra ver mucha diferencia.

Ya si comparamos el histograma después de aplicar histeq será similar al resultado de la ecualización manual, pero se puede ver algunas diferencias, ya que se tiene un rango distinto, pero, aun así, la distribución de valores es uniforme con menos picos y valles, lo que indica una mejora en el contraste de la imagen comparando el histograma original. Lo importante es que las dos formas, tiene como finalidad mejorar el contraste y resaltar detalles en la imagen al redistribuir los valores de píxeles, llegando a ser más uniforme en el histograma.

6. Binarización y umbralización usando el método de Otsu

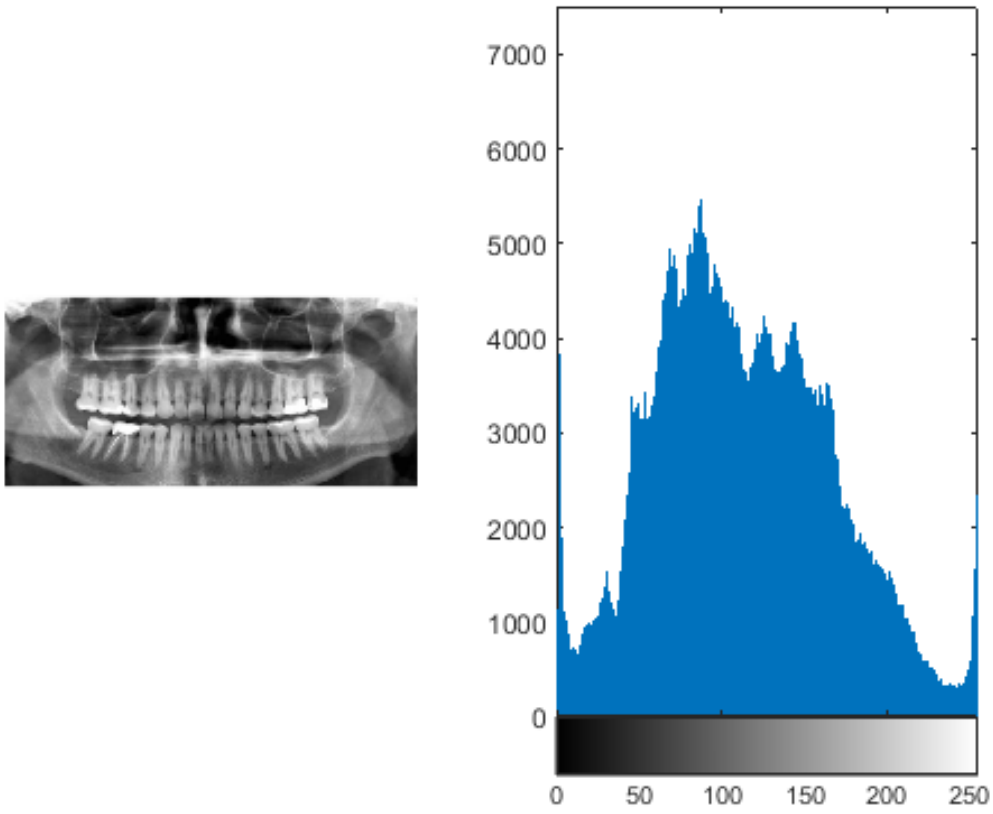

Figure 28: *Imagen original de una imagen de radiografía y su histograma*

Figure 29: *Imagen binarizada de una imagen de radiografia*

Figure 30: *Imagen umbralizada de una imagen de radiografia*

```
clc,clear, close all;
% cargar la imagen
img=imread("radiografia.jpg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
```

```
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);

% calcular automaticamente el umbral utilizando el metodo de Otsu
level = graythresh(grayimg);

% Mostrar el valor del umbral en la consola
fprintf('Valor del umbral calculado: %f\n', level);

% binarizar utilizando el umbral calculado
binimg = imbinarize(grayimg,level);
% multiplicar la imagen original por la imagen binarizada para
% obtener la imagen umbralizada
umbimg = uint8(binimg) .* grayimg;

% visualizar y guardar la imagen binarizada
figure(2)
imshow(binimg);
imwrite(binimg,"radiografia_bin.jpg");
% visualizar y guardar la imagen umbralizada
figure(3)
imshow(umbimg);
imwrite(umbimg,"radiografia_umb.jpg");
```

El valor del umbral calculado es de 0.462745

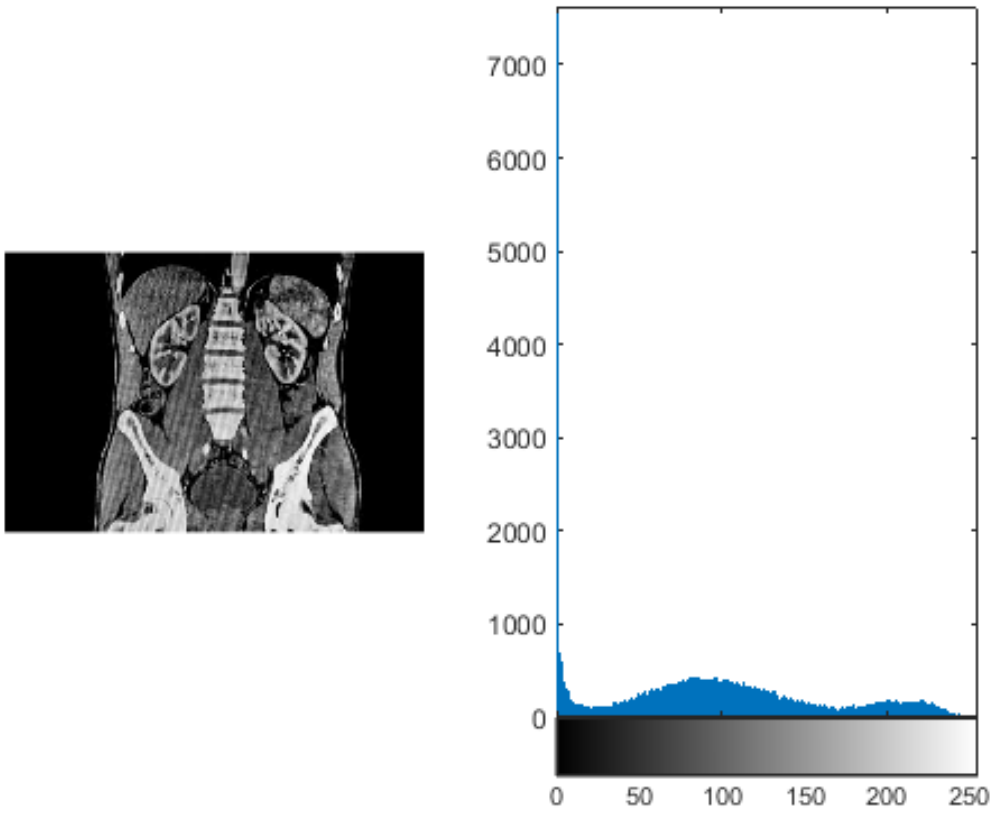

Figure 31: *Imagen original de una imagen de tomografia computacional y su histograma*

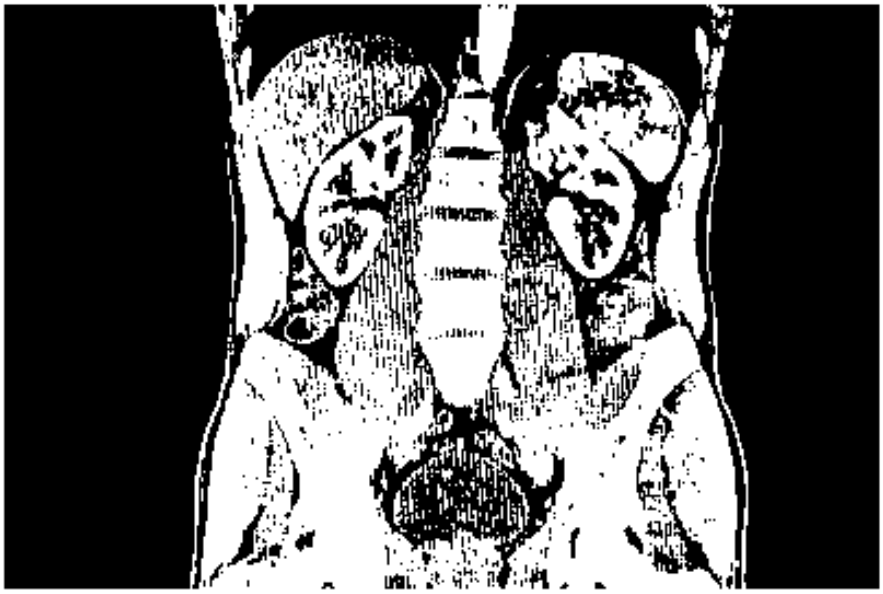

Figure 32: *Imagen binarizada de una imagen de tomografia computacional y su histograma*

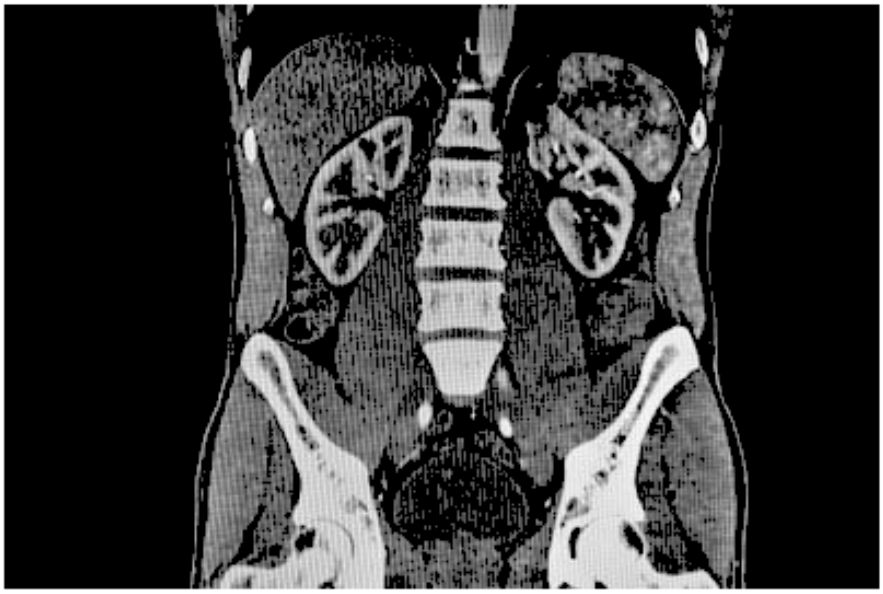

Figure 33: *Imagen umbralizada de una imagen de tomografia computacional y su histograma*

El valor del umbral calculado es de 0.290196

```
clc,clear, close all;
% cargar la imagen
img=imread("abdomen.jpg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);

% calcular automaticamente el umbral utilizando el metodo de Otsu
level = graythresh(grayimg);

% Mostrar el valor del umbral en la consola
fprintf('Valor del umbral calculado: %f\n', level);

% binarizar utilizando el umbral calculado
binimg = imbinarize(grayimg,level);
```

```
% multiplicar la imagen original por la imagen binarizada para
% obtener la imagen umbralizada
umbimg = uint8(binimg) .* grayimg;

% visualizar y guardar la imagen binarizada
figure(2)
imshow(binimg);
imwrite(binimg,"abdomen_bin.jpg");
% visualizar y guardar la imagen umbralizada
figure(3)
imshow(umbimg);
imwrite(umbimg,"amdomen_umb.jpg");
```

7. Binarización local usando el método de Otsu (ejercicios 7)

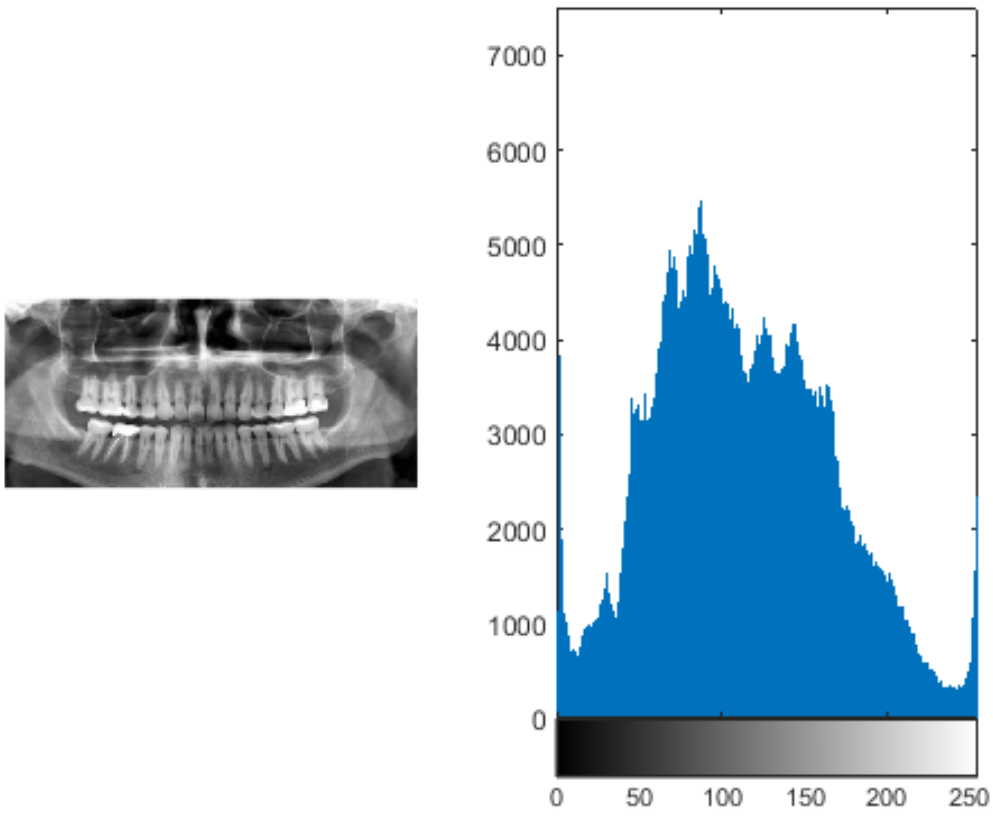

Figure 34: *Imagen original de una imagen de radiografia y su histograma*

Figure 35: *Imagen binarizada localmente 50x50 de una imagen de radiografia y su histograma*

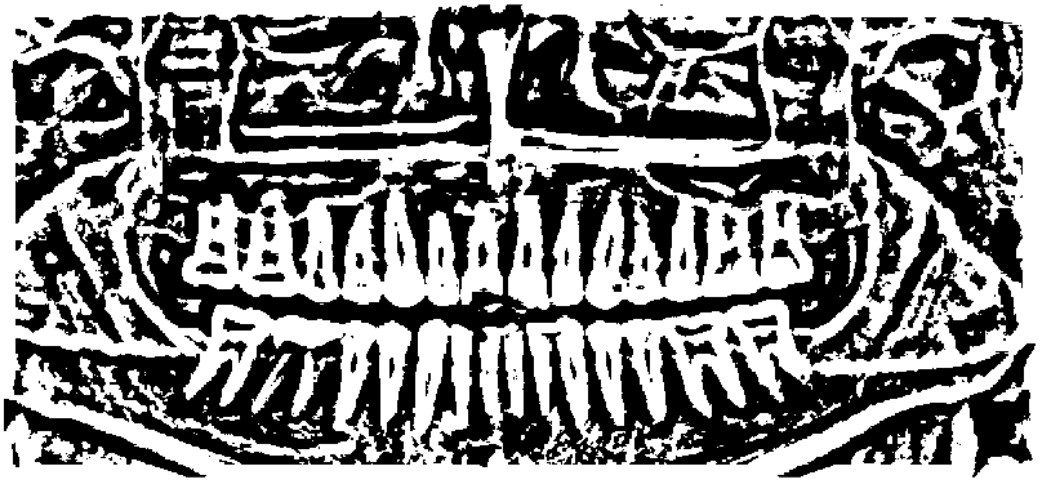

Figure 36: *Imagen binarizada localmente 30x30 de una imagen de radiografia y su histograma*

```
% cargar la imagen
img=imread("radiografia.jpg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
```

```
subplot(1,2,2)
imhist(grayimg);

% encontrar los umbrales locales por ventanas de 50x50
loc50img=nlfilter(grayimg,[50 50], @ibimage);

% visualizar y guardar la imagen binarizada localmente
figure(2)
imshow(loc50img);
imwrite(loc50img,"radiografia_bin50.jpeg");

% encontrar los umbrales locales por ventanas de 30x30
loc30img=nlfilter(grayimg,[30 30], @ibimage);

% visualizar y guardar la imagen binarizada localmente
figure(3)
imshow(loc30img);
imwrite(loc30img,"radiografia_bin30.jpeg");

% funcion ibimage
% encuentra el umbral en cada ventana de la imagen
function f=ibimage(img)
 [x, y]=size(img);
 level=graythresh(img);
 bw=imbinarize(img,level);
 x1=round(x/2);
 y1=round(y/2);
 f=bw(x1, y1);
end
```

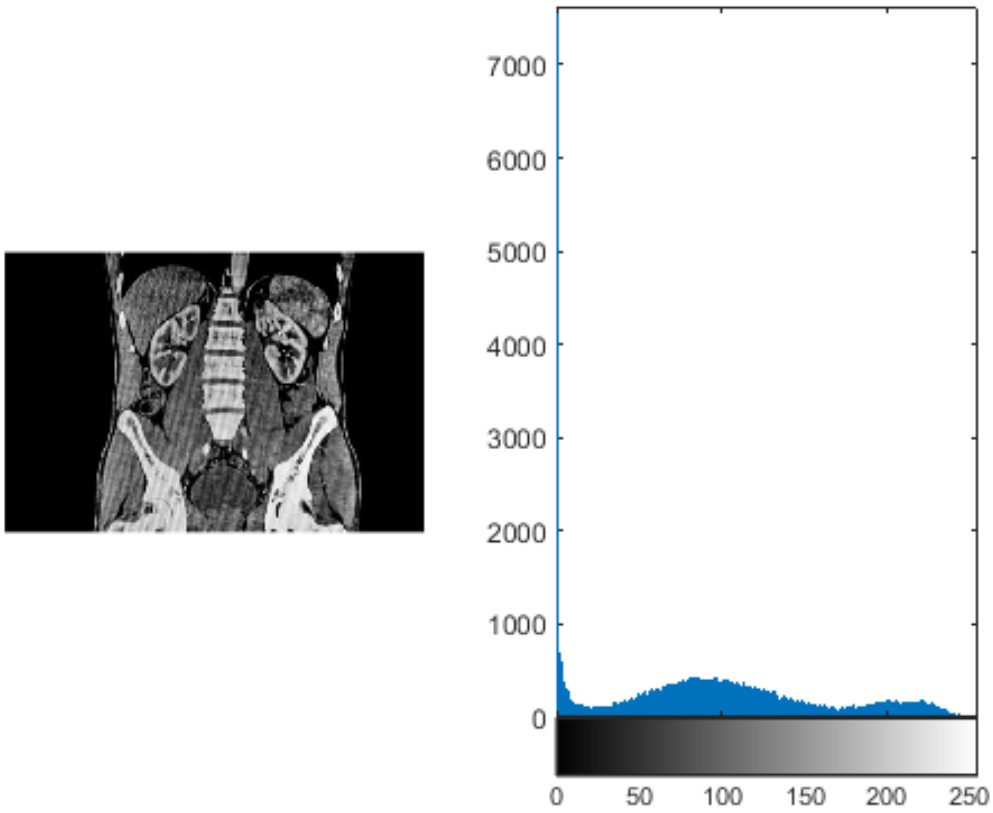

Figure 37: *Imagen original de una imagen de tomografia computacional y su histograma*

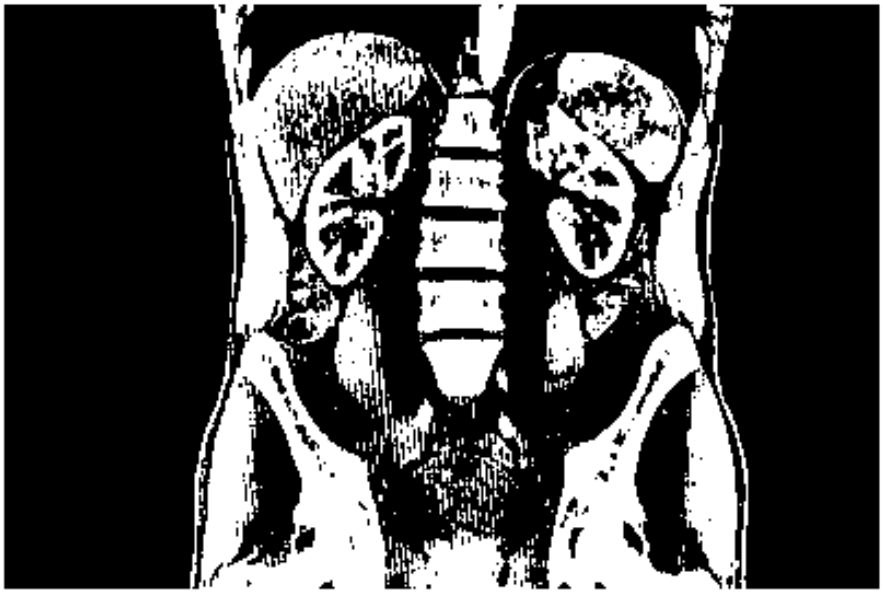

Figure 38: *Imagen binarizada localmente 50x50 de una imagen de tomografia computacional y su histograma*

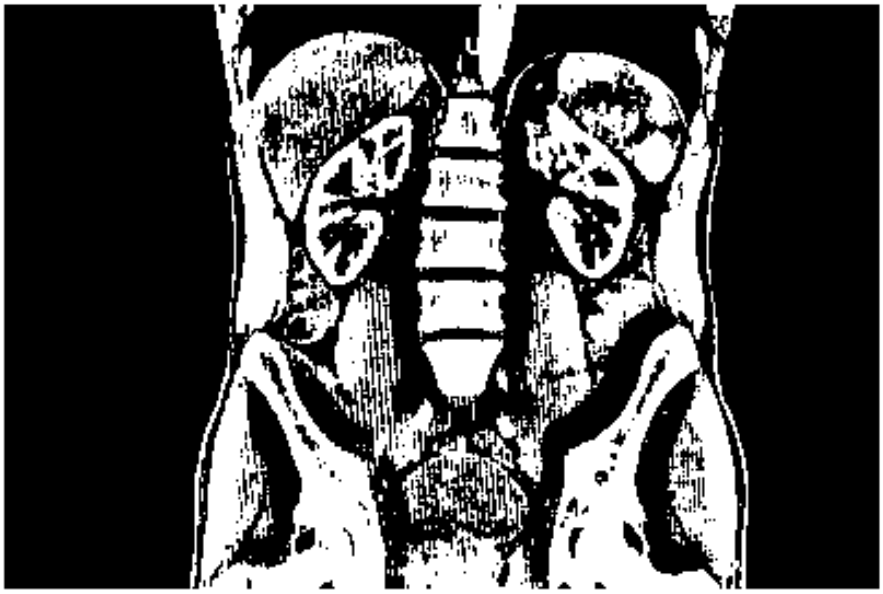

Figure 39: *Imagen binarizada localmente 30x30 de una imagen de tomografia computacional y su histograma*

```
% cargar la imagen
img=imread("abdomen.jpg");
% convertir a escala de grises
grayimg=im2gray(img);
% visualizar la imagen y su histograma
figure(1)
subplot(1,2,1)
imshow(grayimg);
subplot(1,2,2)
imhist(grayimg);

% encontrar los umbrales locales por ventanas de 50x50
loc50img=nlfilter(grayimg,[50 50], @ibimage);

% visualizar y guardar la imagen binarizada localmente
figure(2)
imshow(loc50img);
imwrite(loc50img,"abdomen_bin50.jpeg");

% encontrar los umbrales locales por ventanas de 30x30
loc30img=nlfilter(grayimg,[30 30], @ibimage);
```

```
% visualizar y guardar la imagen binarizada localmente
figure(3)
imshow(loc30img);
imwrite(loc30img,"abdomen_bin30.jpeg");

% funcion ibimage
% encuentra el umbral en cada ventana de la imagen
function f=ibimage(img)
 [x, y]=size(img);
 level=graythresh(img);
 bw=imbinarize(img,level);
 x1=round(x/2);
 y1=round(y/2);
 f=bw(x1, y1);
end
```

8. Binarización local usando el método de Otsu (ejercicios 8)

La comparación que encontramos entre un único umbral y con umbrales locales, fue que la imagen como tal, tenia datos relativamente homogéneos y no muestran variaciones significativas en diferentes partes del conjunto de datos, lo que hizo que las dos imagenes se vieran bien. Ademas, las partes de interés en las imagenes están claramente definidos y tienen bordes nítidos, por lo que un unico humbral fue adecuado para separarlos del fondo sin necesidad de umbrales locales. Las áreas de transición entre objetos y fondo son suaves y no requieren un tratamiento especial. En este caso es mejor usar solo un unico humbral, ya si se ve variaciones como se observo en la pagina, es mejor usar hacer umbrales locales.

¿Estas operaciones permiten mejorar las imágenes utilizadas? ¿Permiten resaltar información que pueda ser usada posteriormente para el diagnóstico?

Si, ya que mejora la calidad de la imagen, haciendo que se vean mas detalles de lo que se quiere observar, ayudando a encontrar los diagnosticos.

6 Binarización usando el método de Otsu (librería ITK)

Ejercicio 10:

Código en Python con un programa simple utilizando la librería ITK, en el cual se carga una imagen médica MRLiverTumor.nii.gz y se binariza utilizando el método de Otsu, almacenando el resultado obtenido en disco, e imprimiendo en pantalla el valor del umbral calculado automáticamente

```
import itk

# definir la imagen como volumen (3 dimensiones)
ImageType = itk.Image[itk.UC, 3]
# definir un lector para la imagen de entrada e inicializarlo
```

```
reader = itk.ImageFileReader[ImageType].New()
# definir el algoritmo de Otsu que binarizara la imagen
otsuFilter = itk.OtsuThresholdImageFilter[ImageType,ImageType].New()
# definir un escritor para la imagen de salida e inicializarlo
writer = itk.ImageFileWriter[ImageType].New()
# asignar al lector el nombre del archivo de entrada
reader.SetFileName("MRLiverTumor.nii.gz")
# asignar al escritor el nombre del archivo de salida
writer.SetFileName("MRLiverTumor_bin.nii.gz")
# conectar el algoritmo al lector para tomar la imagen de entrada
otsuFilter.SetInput(reader.GetOutput())
# conectar el escritor al algoritmo para tomar la imagen de salida
writer.SetInput(otsuFilter.GetOutput())
# ejecutar la linea de procesamiento completa
writer.Update()
# imprimir en pantalla el valor del umbral automatico
print(otsuFilter.GetThreshold())

import numpy as np
import matplotlib.pyplot as plt

# Convertir las imagenes ITK a matrices de NumPy para visualizacion
imagen_original_np = itk.array_from_image(reader.GetOutput())
imagen_binarizada_np = itk.array_from_image(otsuFilter.GetOutput())

# Visualizar la imagen original
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.imshow(imagen_original_np[:, :, imagen_original_np.shape[2] // 2],
   cmap='gray')
plt.title("Imagen Original")

# Visualizar la imagen binarizada
plt.subplot(1, 2, 2)
plt.imshow(imagen_binarizada_np[:, :, imagen_binarizada_np.shape[2] // 2],
    cmap='gray')
plt.title("Imagen Binarizada")

# Titulo de la figura en su totalidad
plt.suptitle("MRI Liver Tumor", fontsize=16)

plt.show()
```

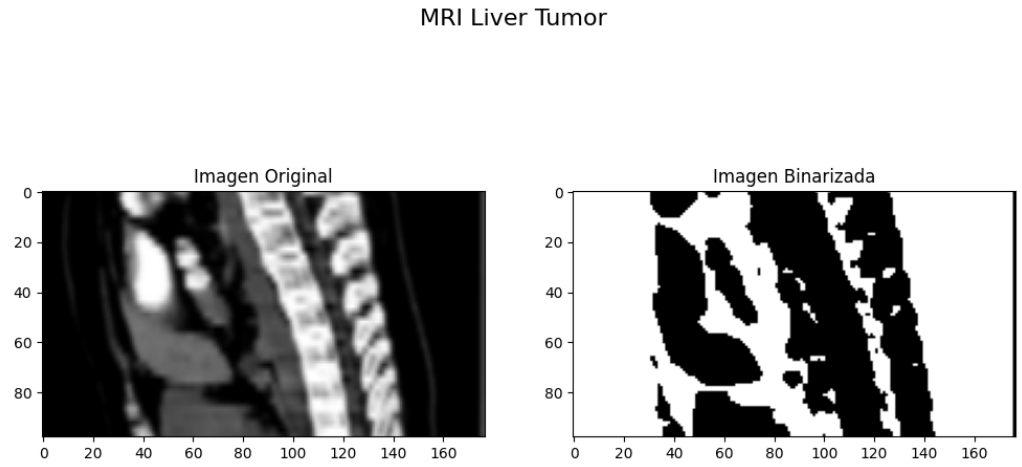

Figure 40: *Imagen MRI de tumor de cerevro binarizada con el método Otsu de la librería ITK*

El Umbral encontrado para esta imagen de Tumor de Hígado fue: 61

En este caso, la umbralización no mejoró la visualización de los tumores en la imagen (Region blanca sobre el Hígado). La técnica resultó en una pérdida significativa de detalles, lo que dificulta la identificación precisa del tumor, el cúal es más identificable en la imágen original.

Ejercicio 11:

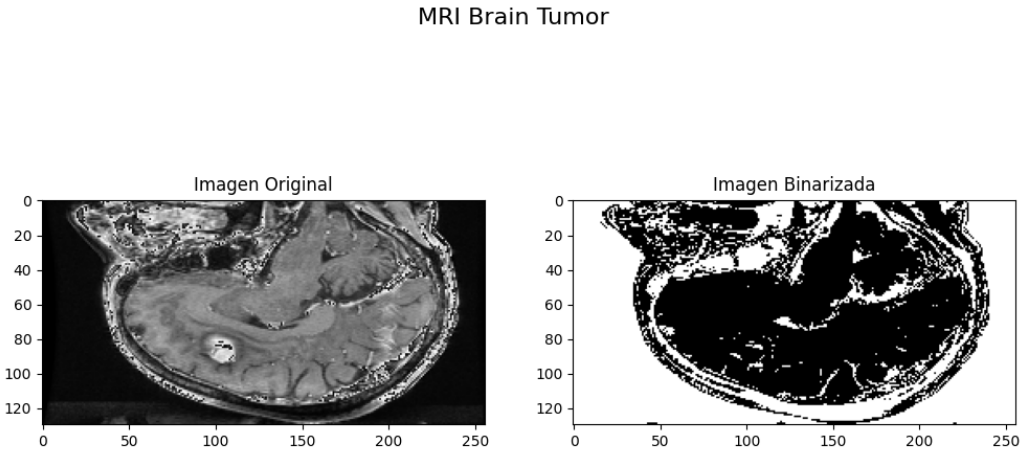

Figure 41: *Imagen MRI de tumor de hígado binarizada con el método Otsu de la librería ITK*

El Umbral encontrado para esta imagen de Tumor de Cerebro fue: 75

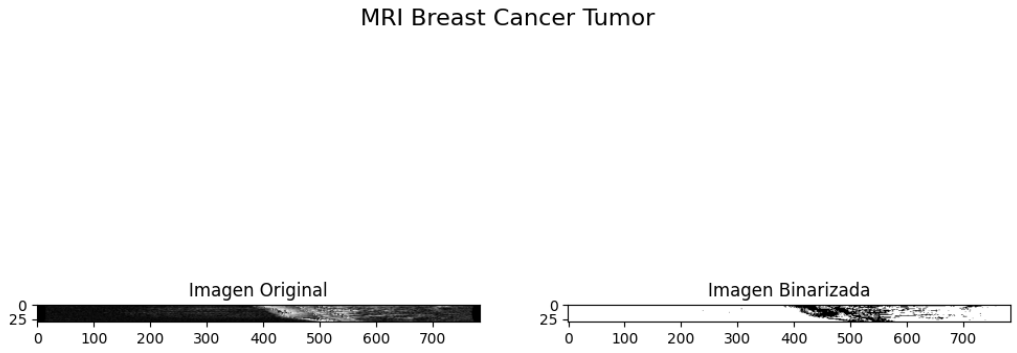

Figure 42: *Imagen MRI de tumor de Cancer de Mama binarizada con el método Otsu de la librería ITK*

El Umbral encontrado para esta imagen de Tumor de Cancer de Mama fue: 97

En ambos casos, el método OtsuThresholdImageFilter en ITK de umbralización, no mejoró la visualización de los tumores en la imagen (Region Clara). La técnica resultó en una pérdida significativa de detalles, lo que dificulta la identificación precisa del tumor, el cúal es más identificable en la imágen original.