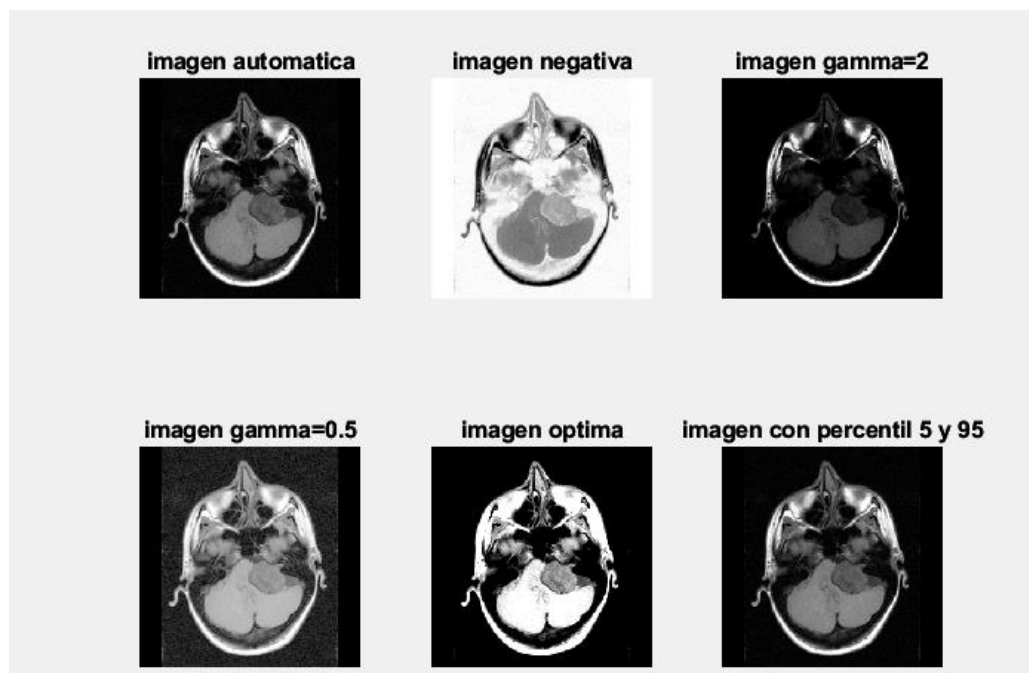


Universidad de Málaga

Materia: Imágenes biomédicas

Profesor: Ignacio Rodríguez Rodríguez

Practica1: Algoritmos básicos de tratamiento digital de imágenes biomédicas



Diego De Pablo

NIE: Y8878615P

Málaga, noviembre de 2023

Introducción:

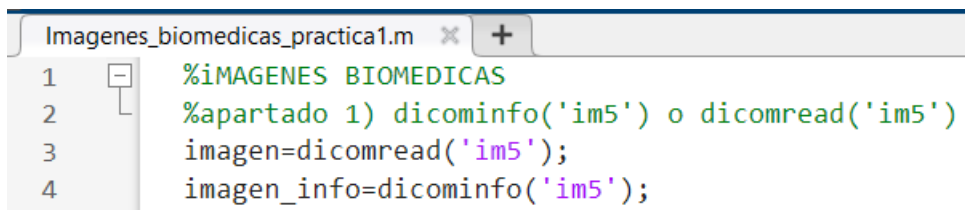
A continuación, se trabajara con la practica 1, que consiste en el manejo de imágenes biomédicas en formato DICOM. Visualización de imágenes biomédicas. Realización de operaciones básicas de tratamiento de imágenes con el objeto de mejorar su visualización, empleando técnicas de modificación de contraste. por temas de simplicidad la memoria del trabajo realizado se hará de carácter más informal. El código de los algoritmos y la valoración crítica de los resultados se harán directamente en orden según lo visto en clase, intentando seguir punto por punto los apartados de la práctica, cabe destacar antes de empezar que esta práctica contiene diversas imágenes que variaran según el alumno, en mi caso es la imagen 5.

GRUPO	Imagen	NOMBRE
B		5 DE PABLO , DIEGO

Contenido de la práctica

1. Abrir la imagen o imágenes que le correspondan, creando una matriz de datos donde almacenar los valores de los píxeles. Una vez abierta, debe comprobar que los metadatos son correctos y determinar los parámetros básicos de la imagen, como el número de filas, columnas o número de bits por píxel. Puede utilizar las funciones `dicomread(...)` o `dicominfo(...)`, del toolbox de Image Processing de Matlab. Describir el tipo de imagen obtenida (Radiología, TC, etc.) y su contenido anatómico (básico).

Codigo de Matlab:



```
1 %IMAGENES BIOMEDICAS
2 %apartado 1) dicominfo('im5') o dicomread('im5')
3 imagen=dicomread('im5');
4 imagen_info=dicominfo('im5');
```

Explicación:

Se descargo la imagen 5 la cual se pasó por las funciones `dicomread` y `dicominfo` respectivamente, `dicomread` leyó la imagen como si fuera una matriz la cual guardamos con el nombre de “imagen” de esta se puede ver una gran cantidad de 0 que nos va informando que la imagen tendrá un fondo negro. Mientras que `dicominfo` nos otorga una gran cantidad de información respecto al archivo, se nos pide destacar:

```
Modality: 'MR'
ModalitiesInStudy: 'MR'
Manufacturer: 'Philips'
```

Tipo de imagen: Resonancia (MR = *Magnetic Resonance*)

Contenido atómico: Cráneo

```
RequestedProcedureDescription: 'Craneo'
```

Otros datos que pueden ser resaltantes:

```
Format: 'DICOM'  
FormatVersion: 3  
Width: 256  
Height: 256  
BitDepth: 12  
ColorType: 'grayscale'
```

Formato “DICOM”

Número de fila 256

Número de columnas 256

Profundidad de bits: 12

Nota: En caso de querer visualizar toda la información obtenida mediante dicominfo, puede bajar al final del documento, se invita a leer todos los datos en vez de quedarse con estos principales.

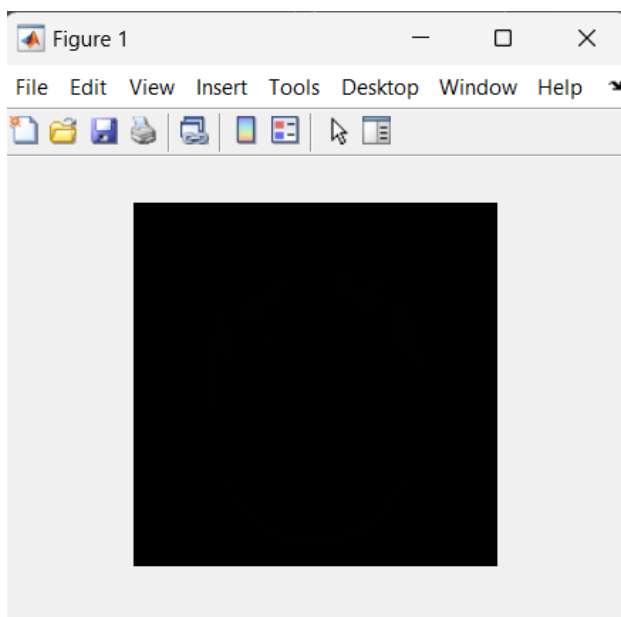


Según los resultados obtenidos apreciamos que la matriz es tiene un tamaño de 256x256 y en formato uint16, por lo que cada pixel que aparece se representa mediante 16 bits y por lo tanto el valor máximo el cual puede alcanzar una celda en la matriz es $2^{16}-1=65535$.

También se puede visualizar la imagen mediante el comando imshow de la siguiente manera:

Codigo de Matlab:

```
imshow(imagen)  
figure(1)
```



```

imagen =

256x256 uint16 matrix

Columns 1 through 19

    0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    0     0     0     0     0     0     0     0     0     0     0     0     0     0     0

```

Explicación:

Al usar `imshow` sale una imagen en negro, porque en Matlab la escala de gris está representada entre el 0 y un número muy grande, la matriz usada tienen valores relativamente cercanos a cero lo que hace que se represente como negro, yendo más específicamente esto dependerán el tipo de dato que cuente la matriz, por ejemplo: para uint16 la escala es [0,65535], donde 0 es negro y el valor máximo es blanco. En nuestro caso la matriz los valores mayoritariamente es 0, lo que nos indica que será de fondo negro, y el número mayor a simple vista no sobrepasa el 500, lo cual en una escala 500 está más cerca del cero que al 65535 lo que hace que prácticamente sea negro

Esto se puede arreglar modificando este intervalo que se hace del siguiente modo:

Código de Matlab:

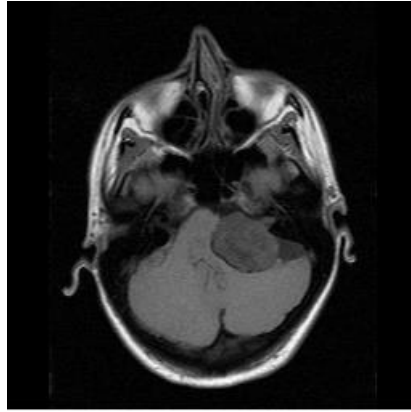
```

max_imagen=max(max(imagen)); %536
figure(2)
imshow(imagen,[0,max_imagen])

```

Al usar `max_imagen` de esta forma obtenemos el valor más grande en la matriz que es 536, en este código se está ajustando el rango de visualización de la imagen para que se extienda desde 0 hasta el valor máximo de la imagen. Esto significa que todos los valores se mapearán linealmente a este rango, lo que ayuda a resaltar las características de la imagen en vez de solo negro.

Se podría poner otros valores en lugar de `max_imagen` dependiendo que nos interese ver, al usar `max_imagen` solo normalizamos dependiendo del valor máximo, pero se podría hacer por percentiles para evitar errores de máximos (caso donde por algún error la matriz cuente con un único punto de valor 60.000, seguiría dándonos una imagen en negro, con los percentiles nos aseguramos de visualizar la imagen). Dándonos por resultado la siguiente visualización:



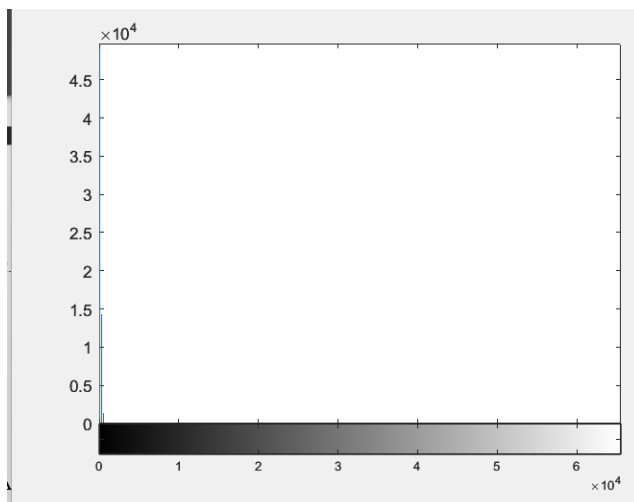
```
%con los percentiles sería
%p1 = prctile(imagen(:), 1);
%p99 = prctile(imagen(:), 99);
%imshow(imagen, [p1, p99]);
```

2. Realizar el cálculo del histograma de dos formas independientes y visualizarlo en un gráfico de forma óptima (de forma que se vean con claridad los picos que corresponden a las zonas de interés de la imagen, que podrán ser identificadas, para lo que deberá definir los valores de los rangos adecuados en los ejes x e y):

1. Utilizando la rutina `imhist(...)` de Matlab (con los parámetros adecuados).
2. Mediante un algoritmo que deberá programar, que obtenga un resultado similar al de la rutina de Matlab (aquí intentamos aprender a programar nosotros mismos lo mismo que hace Matlab).

Nota: bins es el nombre que se da en estadística al número de intervalos (variable aleatoria discreta) en que se divide el rango de la variable (en este caso, el nivel de gris) para calcular el histograma.

Al usar `imhist` seguimos teniendo el problema de las escalas, visualizándose el histograma a duras penas se pueden distinguir 3 contenedores, 1 de tamaño mayor a $4.5 \cdot 10^4$ otro que está cercano al $1.5 \cdot 10^4$ y un último que no llega ni a la mitad del $0.5 \cdot 10^4$, todos estos en el eje vertical, en el eje horizontal vemos que no están distribuidos bien, puesto que están todos muy cercanos al 0, y estamos trabajando con una escala que llega hasta $6 \cdot 10^4$

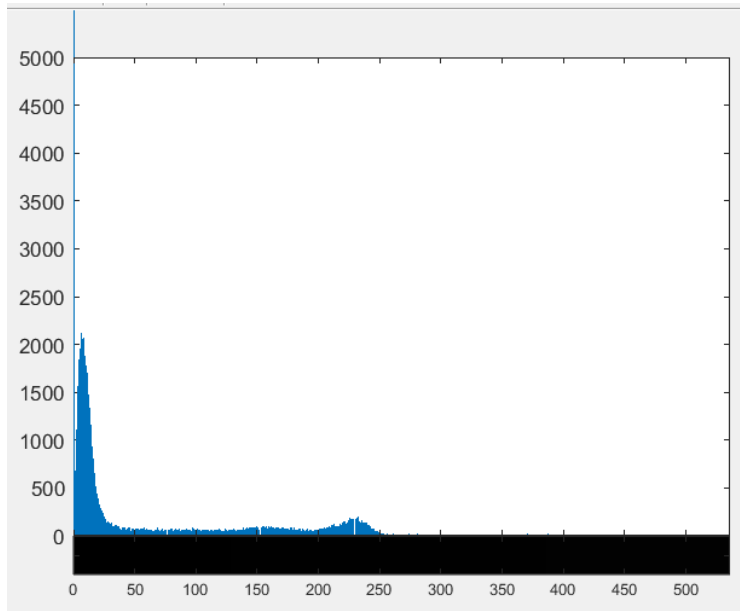


Modificamos el número de bins en el histograma, que actualmente tiene un valor muy bajo, además de poner límites para el plano x y plano y.

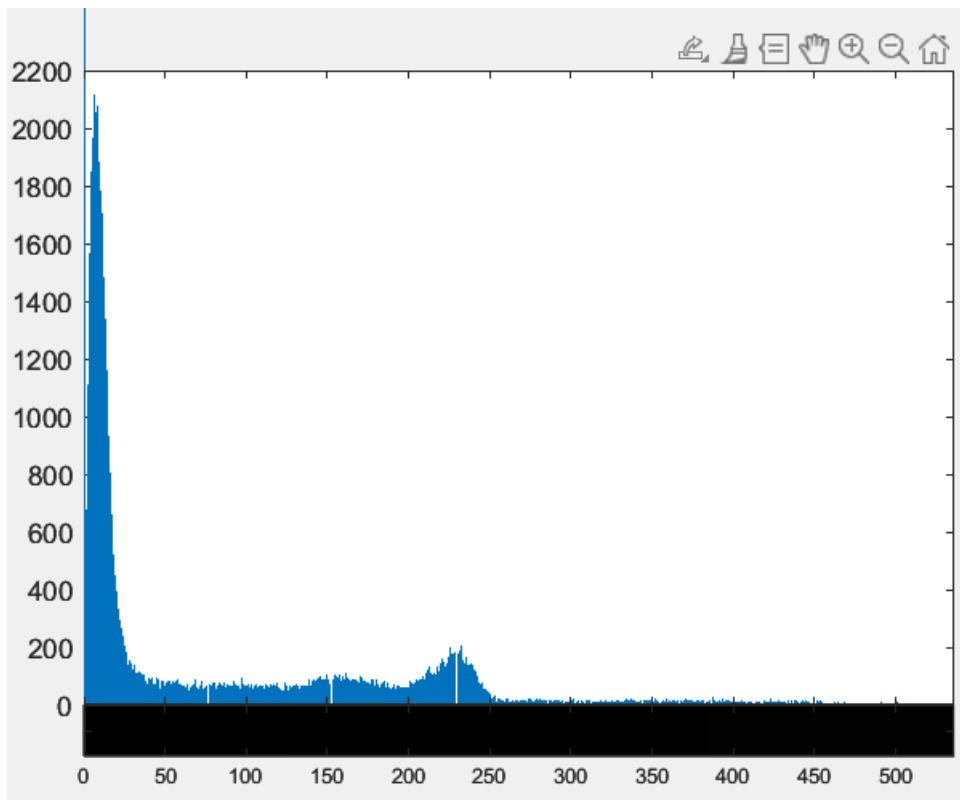
Procedemos a corregir y conseguir unos histogramas más útiles.

Código de Matlab:

```
figure(3)|  
imhist(imagen, 65535)  
axis([0 max_imagen 0 5000])  
%xlim([0,max_imagen])
```



Con `axis([0 max_imagen 0 2200])`



Explicación:

Aumentamos el número de bins para poder ver mejor el histograma, aún así vemos que existe una cantidad exagerada de 0 en comparación a los demás valores, haciendo que el histograma se vea el primer bin muy grande mientras los demás un tamaño moderado, viendo la imagen tiene sentido. Puesto que gran parte de la imagen cuenta con un fondo negro.

Siendo más específico con respecto al código, genera un histograma diferente al anterior. Al usar `imhist(imagen, 65535)`, estás creando un histograma con 65535 bins, lo que permite ver la distribución de todos los posibles valores de píxeles en una imagen `uint16`. Y al usar `axis([0 max_imagen 0 5000])`, se están ajustando los ejes del gráfico para centrarse en el rango de valores de interés.

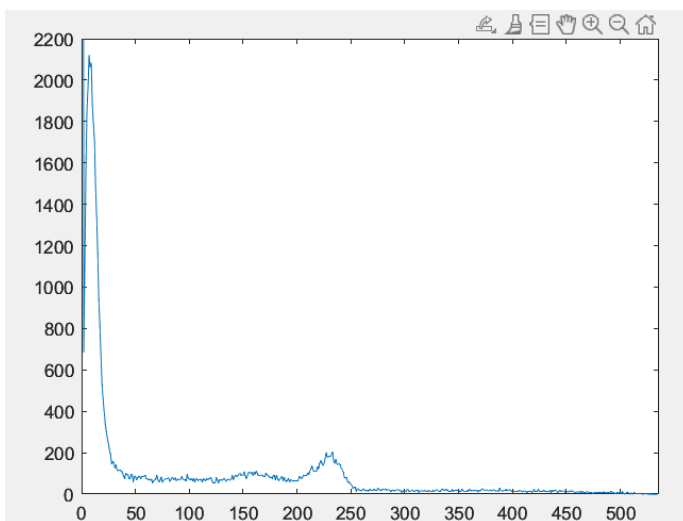
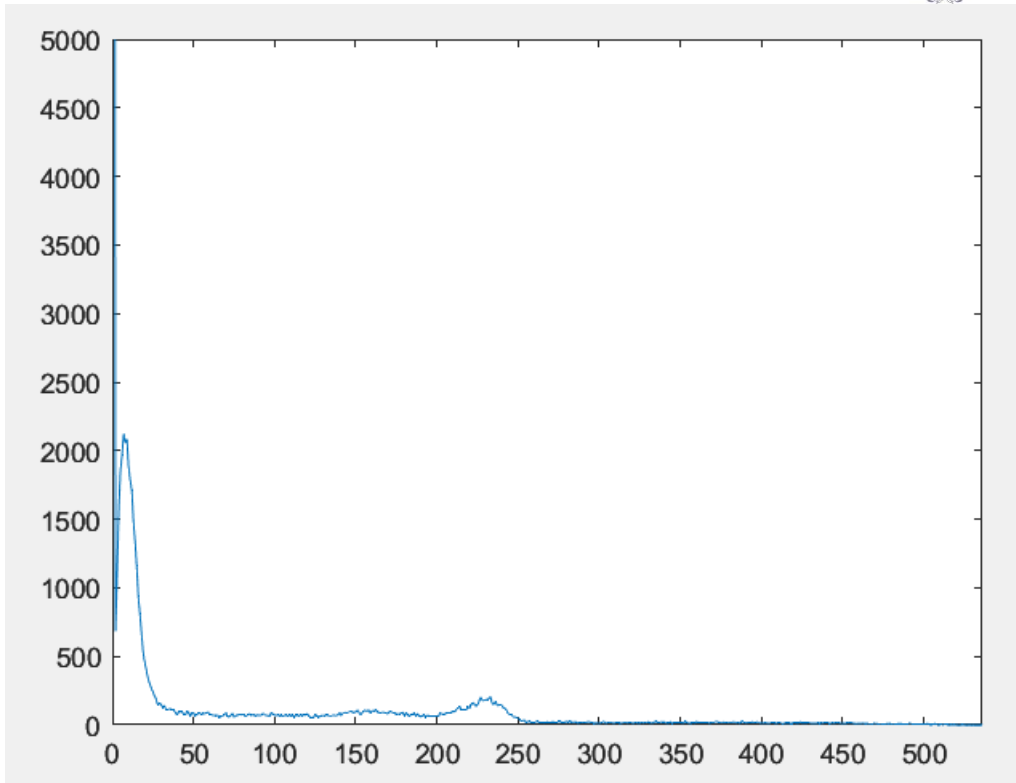
IMPLEMENTACIÓN DEL ALGORITMO:

Código de Matlab:

```
%algoritmo;
i=double(imagen);
[filas, columnas]=size(imagen);
for i=1:65535
    h(i)=0;
end
for i= 1:filas
    for j=1:columnas
        k=imagen(i,j);
        h(k+1) = h(k+1)+1;
    end
end
figure(4)
plot(h)
axis([0 max_imagen 0 5000])
```

Explicación:

Podemos observar que la salida del algoritmo es lo esperado, siendo el punto más superior visto en el histograma, dependiendo de los límites planteados en el `axis` se verá mejor o peor el histograma, en este caso todavía tenemos muchos elementos de valor 0 por el fondo lo que hace que la moda sea 0, con unos 5000 píxeles de ese valor.



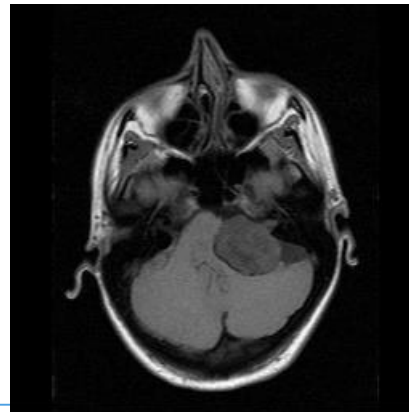
3. Visualizar la parte más significativa de la información contenida en la imagen que te corresponda, mediante la selección de la ventana idónea en cada caso. Para cada imagen (según el número de la imagen que le ha sido asignada), la región de interés es:

1. Estructura ósea
2. Hígado
3. Hueso
4. Tejido graso/microcalcificaciones vasculares
5. Tumor y masa cerebral
6. Estructura ósea
7. Parénquima pulmonar
8. Tejido cerebral
9. Tejido graso
10. Organos

Se recomienda usar el comando `imshow(...)`. Tenga en cuenta que la mayoría de las imágenes médicas tienen un tipo de datos `uint16` que permite más de 256 niveles de gris. La visualización es adecuada si la región de interés se muestra con un contraste adecuado y puede apreciarse con nitidez el órgano o estructura de interés.

Código de Matlab:

```
max_imagen=max(max(imagen)); %536
figure(2)
imshow(imagen,[0,max_imagen])
%con los percentiles sería
%p1 = prctile(imagen(:), 1);
%p99 = prctile(imagen(:), 99);
%imshow(imagen, [p1, p99]);|
....
```



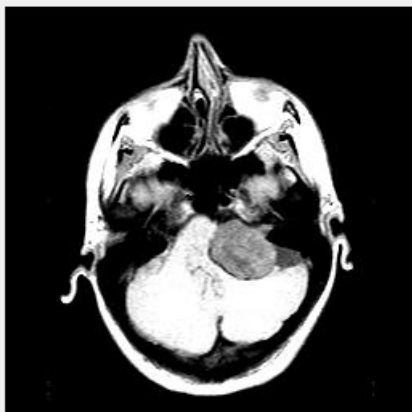
Explicación:

Para este apartado ya me adelanto en el apartado 1 explicando como ajustar mejor la imagen y los distintos medios ya sea por percentiles o por puntos, para este punto ya que conocemos mejor la imagen y sabemos cuales son los puntos de interés, además que confirmamos que nuestra imagen 5 corresponde con tumor y masa cerebral. Para más explicación puede regresar al apartado 1 en su último párrafo considero que hay una buena explicación.

Ahora ya habiendo visto el histograma podemos razonar mejor el porque antes era un cuadrado negro, antes era una escala de 0 al 65553, donde el valor de pixel más lejano era 536, en proporción, todos los valores eran tan cercanos a 0 que se representaban como pixel en negro, con el

%% apartado 3

```
figure(5)
imshow(imagen, [50, 225])%valores donde yo veo mayor diferencia entre el tumor
% y el cerebro
```



diferenciar el tumor

Aquí el cerebro esta muy blanco para

4. Transformar la imagen, de forma que se mantenga la visualización de la parte más significativa de la información de interés (ver apartado anterior) contenida en la imagen. Este tipo de algoritmos modifican el contraste de la imagen, eligiendo una ventana del rango de niveles de gris original y convirtiendo esos niveles de gris en el rango completo de niveles de gris, para mejorar su visualización subjetiva. Para la modificación del contraste, deberá utilizar:

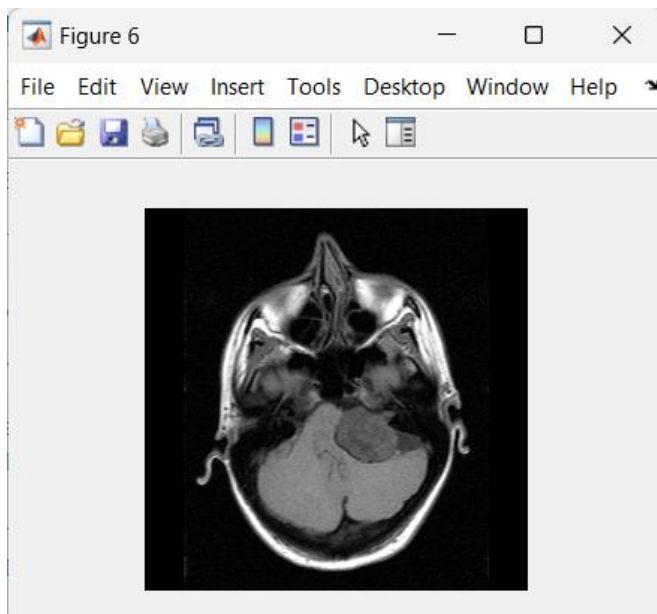
1. La función `imadjust(...)` de Matlab

Los algoritmos a emplear son:

1. Negativo de la imagen.
2. Corrección $\gamma=2$.
3. Ajuste $y=\sqrt{x}$.
4. Ajuste lineal a una ventana óptima (se entiende que eligiendo el rango de niveles de gris de interés para la visualización del órgano o estructura de interés).
5. Ajuste lineal a un rango de entrada comprendido entre los percentiles 5% y el 95% de los niveles de gris. (El rango inferior es el nivel de gris para el cual un máximo del 5% de los píxeles no superan dicho nivel de gris, lo mismo con el rango superior en el 95%). Para calcular los valores de nivel de gris que corresponden a los percentiles 5% y 95%, puede usar el histograma calculado en el apartado 2.

Código de Matlab:

```
%% apartado 4
%Cargamos la imagen y la ajustamos automáticamente
imagen=dicomread('im5');
imagen_ajustada=imadjust(imagen);
figure(6)
imshow(imagen_ajustada) %Visualización de la imagen
%J = imadjust(I,[LOW_IN; HIGH_IN],[LOW_OUT; HIGH_OUT],gamma=1)
```



Explicación:

Aquí se muestra el uso de `imadjust` para darnos nuestra imagen automáticamente, posteriormente se explicará sobre el negativo, corrección $\gamma=2$, igual que otros puntos, pero primero quería dejar constancia de como es el ajuste por defecto y también explicar a continuación como funciona `imadjust` en Matlab:

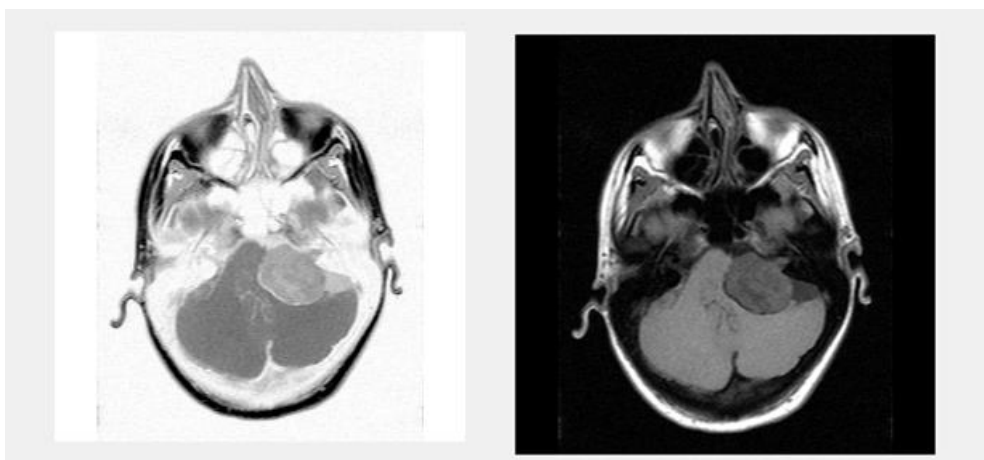
La función `imadjust` normaliza la matriz, ahora tiene los órdenes ajustados para que se pueda ver, esta función hace una transformación lineal. Los parámetros que se usan son `J = imadjust(Imagen,[LOW_IN; HIGH_IN],[LOW_OUT; HIGH_OUT],gamma=1)` `I` sería la imagen, en cuanto a **LOW_IN** y **HIGH_IN**: Los límites inferior y superior de los valores de intensidad de la imagen que se desea ajustar. Si no se especifican, por defecto se utiliza el 1% inferior y superior de todos los valores de píxeles (es decir parecido a tomar desde el percentil 1% al 99%). También tenemos **low_out** y **high_out** son los límites inferior y superior de los valores de intensidad de la imagen resultante. Si no se especifican, se podría decir que es una especie de normalización donde los valores de salida por defecto van del 0 al 1. Y para finalizar la función **gamma**: El valor del parámetro gamma utilizado para ajustar la curva que describe la relación entre los valores de intensidad en la imagen original y los valores de intensidad en la imagen resultante. Si no se especifica, por defecto se utiliza un valor gamma de 1, el gamma es el exponente de x para la gráfica $y=x^{\text{gamma}}$, en la transformada lineal

Código de Matlab:

```
%% apartado 4
%Cargamos la imagen y la ajustamos automaticamente
imagen=dicomread('im5');
imagen_ajustada=imadjust(imagen);
figure(6)
imshow(imagen_ajustada) %Visualizacion de la imagen
%J = imadjust(I,[LOW_IN; HIGH_IN],[LOW_OUT; HIGH_OUT],gamma=1)
```

Negativo de la imagen

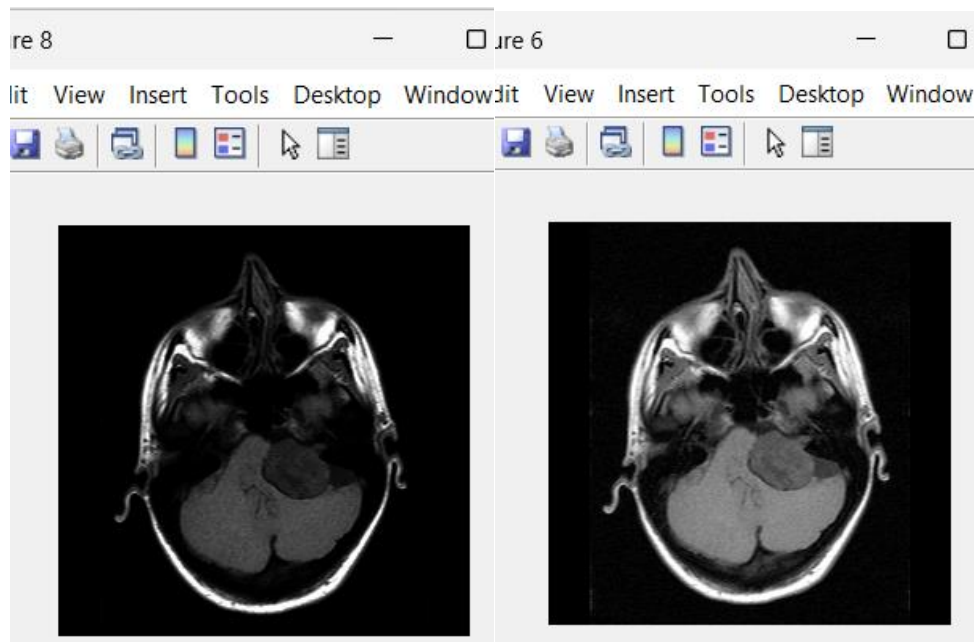
```
%negativo de la imagen:
%J = imadjust(I,[LOW_IN; HIGH_IN],[LOW_OUT; HIGH_OUT],gamma=1) se consigue
%cambiando el low_out por el high_out y visceversa, en este caso:
imagen_negativo = imadjust(imagen_ajustada,[0; 1],[1; 0],1);
figure(7)
imshow(imagen_negativo)
```



Explicación:

se consigue cambiando el low_out por el high_out y viceversa, en este caso hace que los valores que eran 0 por ende negro ahora mostraran un pixel blanco. La imagen de la izquierda es el negativo y el de la derecha la original.

```
%Correccion gamma =2:
%J = imadjust(I,[LOW_IN; HIGH_IN],[LOW_OUT; HIGH_OUT],gamma=2) se consigue
imagen_gamma2 = imadjust(imagen_ajustada,[0; 1],[0; 1],2);
figure(8)
imshow(imagen_gamma2)|
```



Explicación:

Cuando se aplica la función `imadjust` en MATLAB con un valor de gamma igual a 2, se ajusta la curva de relación entre los valores de intensidad en la imagen original y los valores de intensidad en la imagen resultante. En particular, los valores de intensidad más bajos se comprimen aún más, mientras que los valores de intensidad más altos se expanden aún más. Esto resulta en una imagen con un mayor contraste y detalles más nítidos.

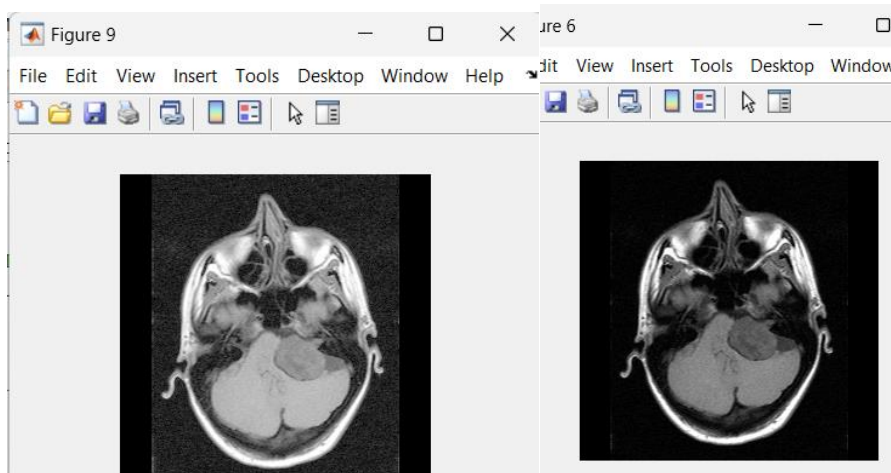
Ahora en nuestra imagen teniendo un gamma=2 se puede observar que la imagen no mejora mucho, se vuelve más oscura, como ya vimos lo que sucede es que los valores de intensidad en la imagen original estén concentrados en el extremo del rango de valores de intensidad. Esto puede resultar en una imagen con una apariencia más oscura después de aplicar la función `imadjust` con un valor de gamma alto. Se puede corregir, pero entiendo que lo mejor sería ver las demás funciones sin modificar tanto los valores de la imagen para ver cual ajuste automatico es mejor.

Ajuste $y = \sqrt{x}$

```
%Ajuste y=sqrt(x), es decir gamma=0,5
%J = imadjust(I,[LOW_IN; HIGH_IN],[LOW_OUT; HIGH_OUT],gamma=0.5) se consigue
imagen_sqrtgamma = imadjust(imagen_ajustada,[0; 1],[0; 1],0.5);
figure(9)
imshow(imagen_sqrtgamma)
```

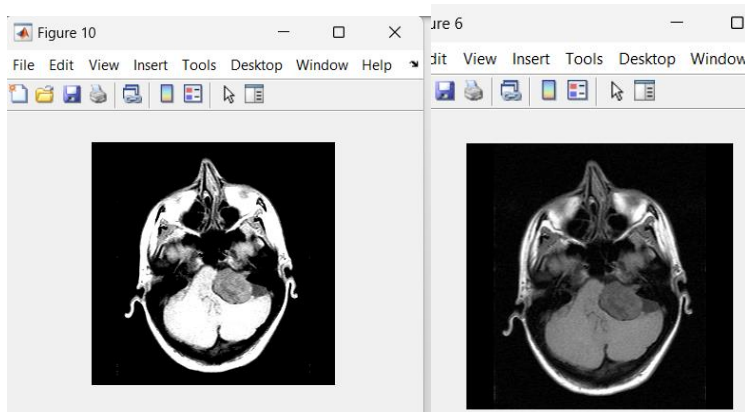
Cuando se aplica la función `imadjust` en MATLAB con un valor de `gamma` igual a 0.5, se ajusta la curva de relación entre los valores de intensidad en la imagen original y los valores de intensidad en la imagen resultante. En particular, los valores de intensidad más bajos se expanden aún más, mientras que los valores de intensidad más altos se comprimen aún más. Esto resulta en una imagen con un menor contraste y detalles menos nítidos.

Al contrario del caso anterior ahora nos da una imagen con más



4. Ajuste lineal a una ventana óptima (se entiende que eligiendo el rango de niveles de gris de interés para la visualización del órgano o estructura de interés).

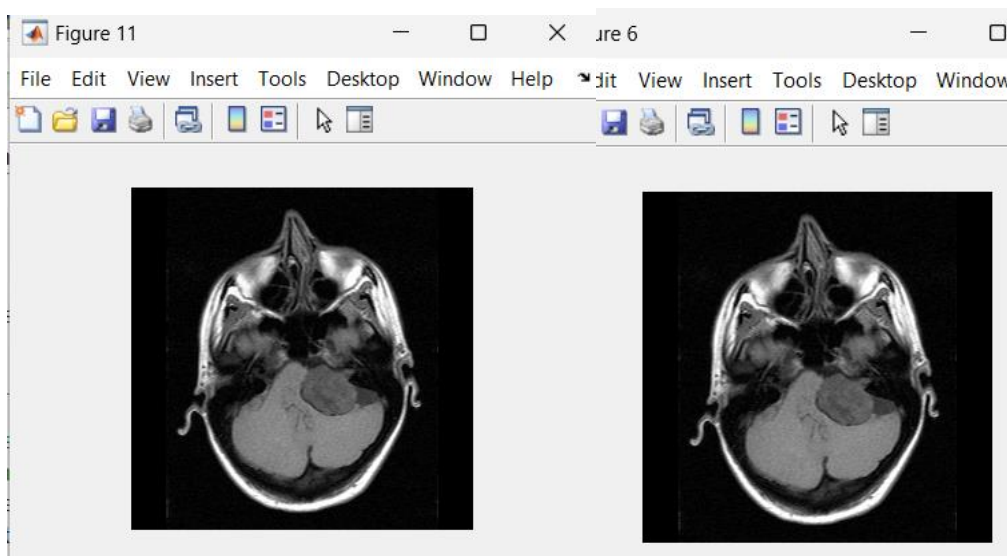
```
%%
% imshow(imagen, [50, 225])
v=[50,225];
v=v/65535;
imagen_optima= imadjust(imagen,[v(1); v(2)],[0; 1]);
figure(10)
imshow(imagen_optima)
```



A la derecha se ve la imagen ajustada con los valores encontrados en el apartado 3

5. Ajuste lineal a un rango de entrada comprendido entre los percentiles 5% y el 95% de los niveles de gris. (El rango inferior es el nivel de gris para el cual un máximo del 5% de los píxeles no superan dicho nivel de gris, lo mismo con el rango superior en el 95%). Para calcular los valores de nivel de gris que corresponden a los percentiles 5% y 95%, puede usar el histograma calculado en el apartado 2.

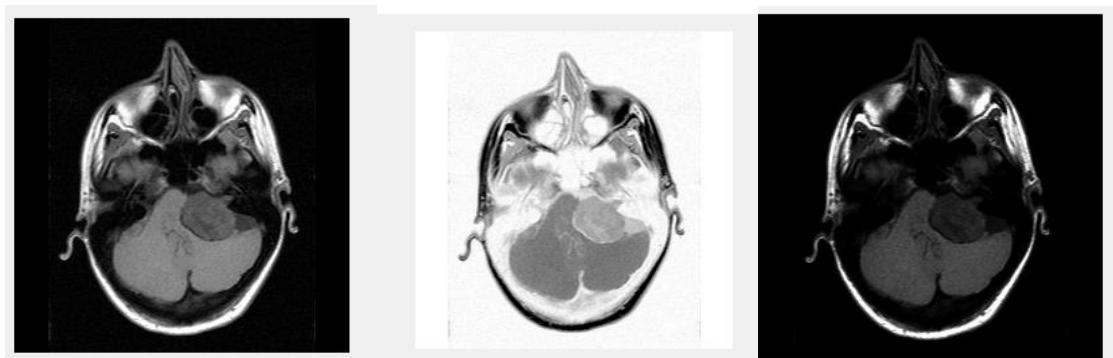
```
%con los percentiles se podría calcular con stretchlim(imagen,[0.01,0.99])
%o con p1 = prctile(imagen(:), 1) y p99 = prctile(imagen(:), 99);
[p] = stretchlim(imagen,[0.01,0.99]);
imagen_nueva= imadjust(imagen,[p(1);p(2)],[0; 1]);
figure(11)
imshow(imagen_nueva)
```

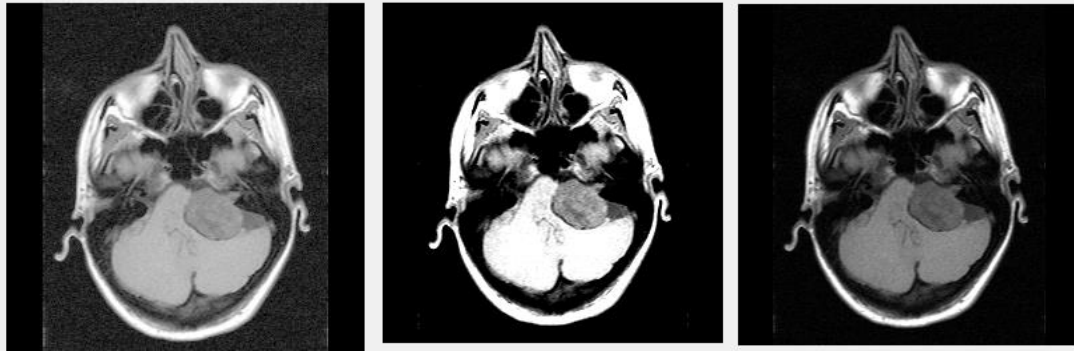


Este no cambia mucho, pues el ajuste automatico agarra los percentiles del 1 al 99, se puede ver que la imagen de la izquierda tiene más nitidez en algunos blancos, esto es porque se acercan más al limite.

2. De los cinco algoritmos anteriores, indique cuál es que ofrece una mejor visualización, según su criterio subjetivo.

Vamos a juntar las imágenes para compararlas de una mejor manera, del siguiente orden de izquierda a derecha, arriba abajo: 1) ajuste automatico, 2) negativa, 3) corrección gamma2, 4) corrección gama=0,5, 5) imagen optima y 6) imagen percentil 5% y 95%:

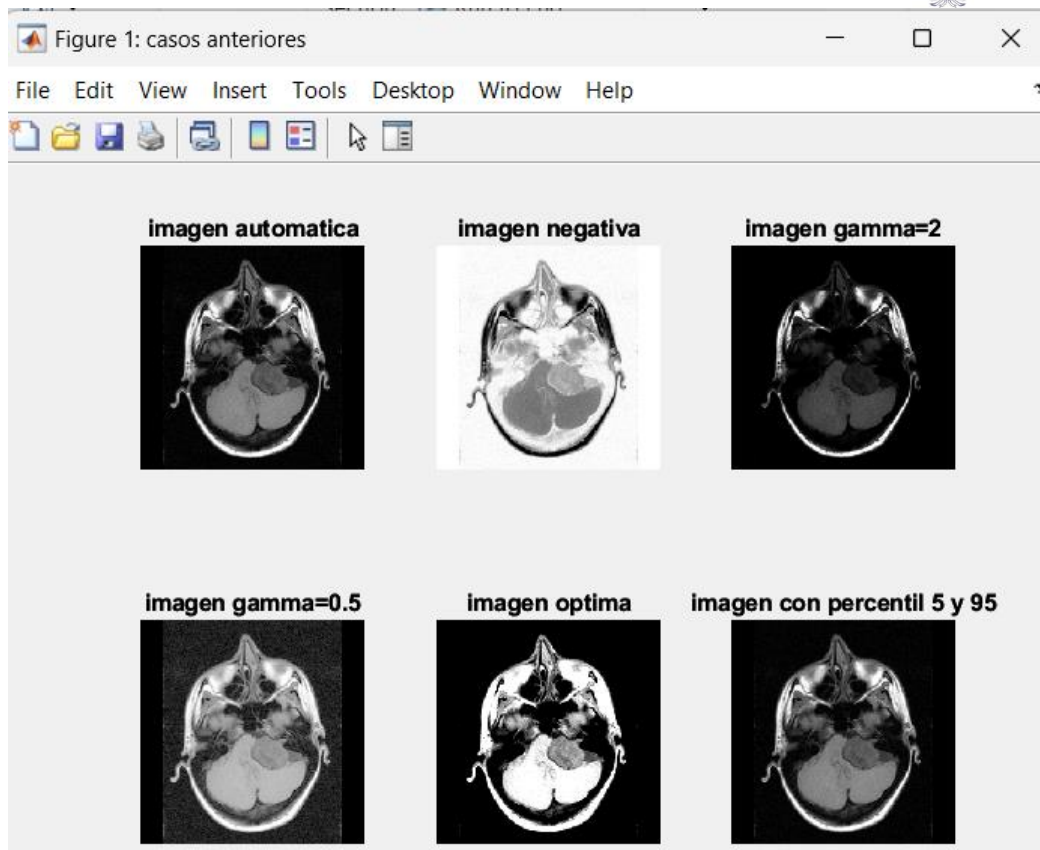




En cuanto a ofrecer una mejor visualización según mi criterio es la corrección gamma 0,5. A pesar de que se note un poco de ruido, realmente considero que trabaja bien con esta imagen donde se puede diferenciar cada componente del cerebro y el tumor sin tener que llegar a extremos con un contraste demasiado fuerte como sucede con otras imágenes. Igual también estoy considerando una mejor visualización desde el punto de vista humano, donde puedo ver la forma del tumor y del resto del cerebro sin problema, capaz si mi interés es una mejor visualización para el entrenamiento de una ia o probar un algoritmo, lo mejor sería una imagen sin tanto ruido y con más contraste a pesar de que se pierda algunos sectores de la imagen.

Esto también se puede hacer en Matlab con el siguiente código:

```
%%
figure('Name', 'casos anteriores')
subplot(2,3,1);
imshow(imagen_ajustada)
title('imagen automatica')
subplot(2,3,2);
imshow(imagen_negativo)
title('imagen negativa')
subplot(2,3,3);
imshow(imagen_gamma2)
title('imagen gamma=2')
subplot(2,3,4);
imshow(imagen_sqrtgamma)
title('imagen gamma=0.5')|
subplot(2,3,5);
imshow(imagen_optima)
title('imagen optima')
subplot(2,3,6);
imshow(imagen_nueva)
title('imagen con percentil 5 y 95')
```



3. Deberá programar un algoritmo propio que obtenga un resultado similar al de la rutina de Matlab imadjust (sólo la opción, de entre las cinco, con la que se obtiene la mejor visualización).

A continuación el código del algoritmo hecho desde cero, y luego una comparación entre el ajuste automatico y mi algoritmo propio

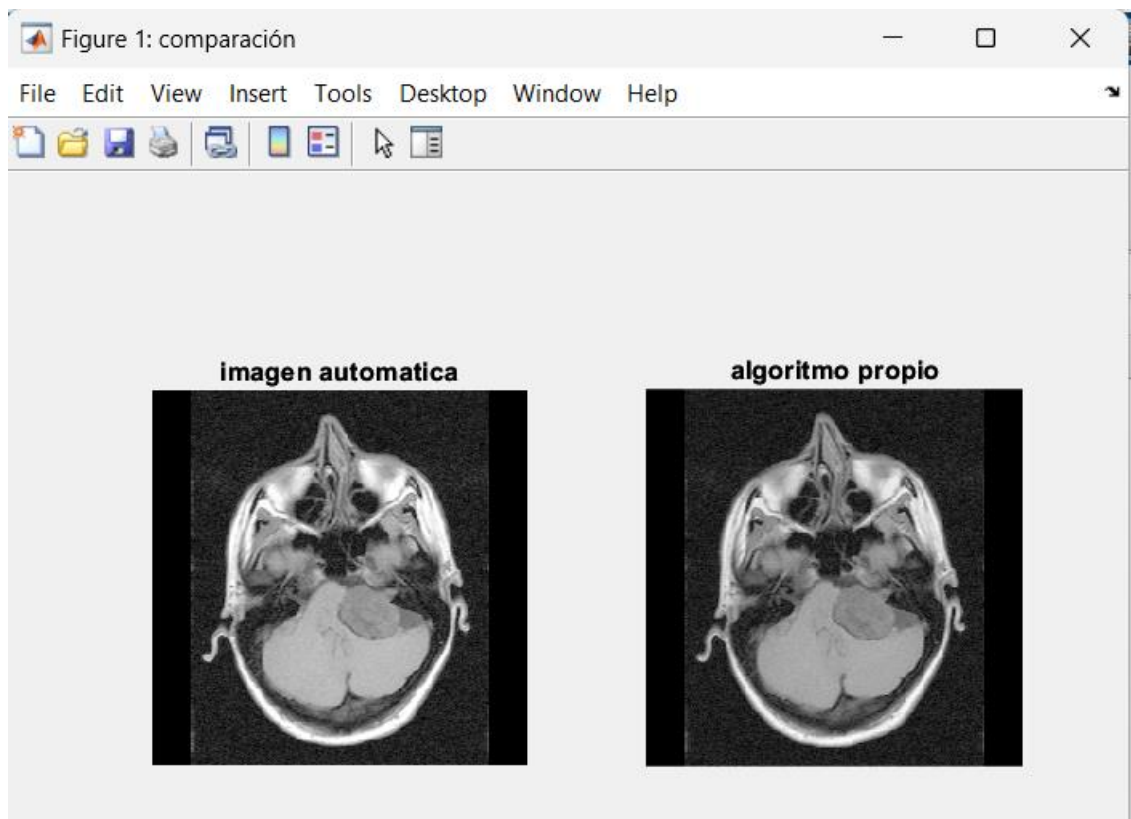
```
function resultado=ajuste_gamma_sqrt(imagen,low_in, high_in)
if nargin < 2
    low_in = 0;
end
if nargin < 3
    high_in = 1;
end

imagen=double(imagen);
imagen = mat2gray(imagen);
[filas, columnas]=size(imagen);
resultado=zeros(filas,columnas);
for i=1:filas
    for j=1: columnas
        pixel=imagen(i,j);
        if (pixel >= low_in && pixel <= high_in)
            resultado(i,j)=((pixel-low_in)/(high_in-low_in))^0.5;
        else if pixel>high_in
            resultado(i,j)=65535;
        else
            resultado(i,j)=0;
        end
    end
end
end
end
```


Para compararlas usamos el siguiente código

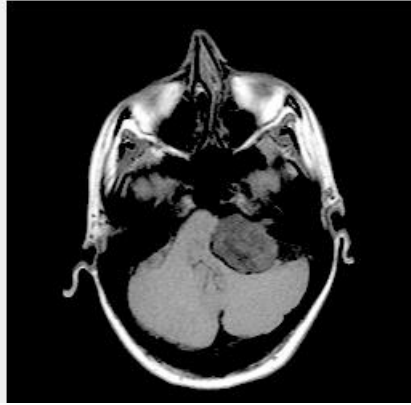
```
clear variables
clc
close all

imagen=dicomread('im5');
imagen_ajustada=imadjust(imagen);
figure('Name', 'comparación')
subplot(1,2,1);
imagen_sqrtgamma = imadjust(imagen_ajustada,[0; 1],[0; 1],0.5);
imshow(imagen_sqrtgamma)
title('imagen automatica')
subplot(1,2,2);
imshow(ajuste_gamma_sqrt(imagen))
title('algoritmo propio')
```



Podemos ver que cumple con lo deseado, al ser las dos imágenes prácticamente iguales, algo bueno de escribir tu propio algoritmo es que este deja mayor control para elegir que deseas modificar, en este caso programe que automáticamente el `low_in` y `high_in` que son tomados como los límites inferiores y superiores fueran 0 y 1 respectivamente, pero en caso de querer llamar a la función y cambiar dichos valores se pueda hacer simplemente poniendo los valores luego de darle la imagen.

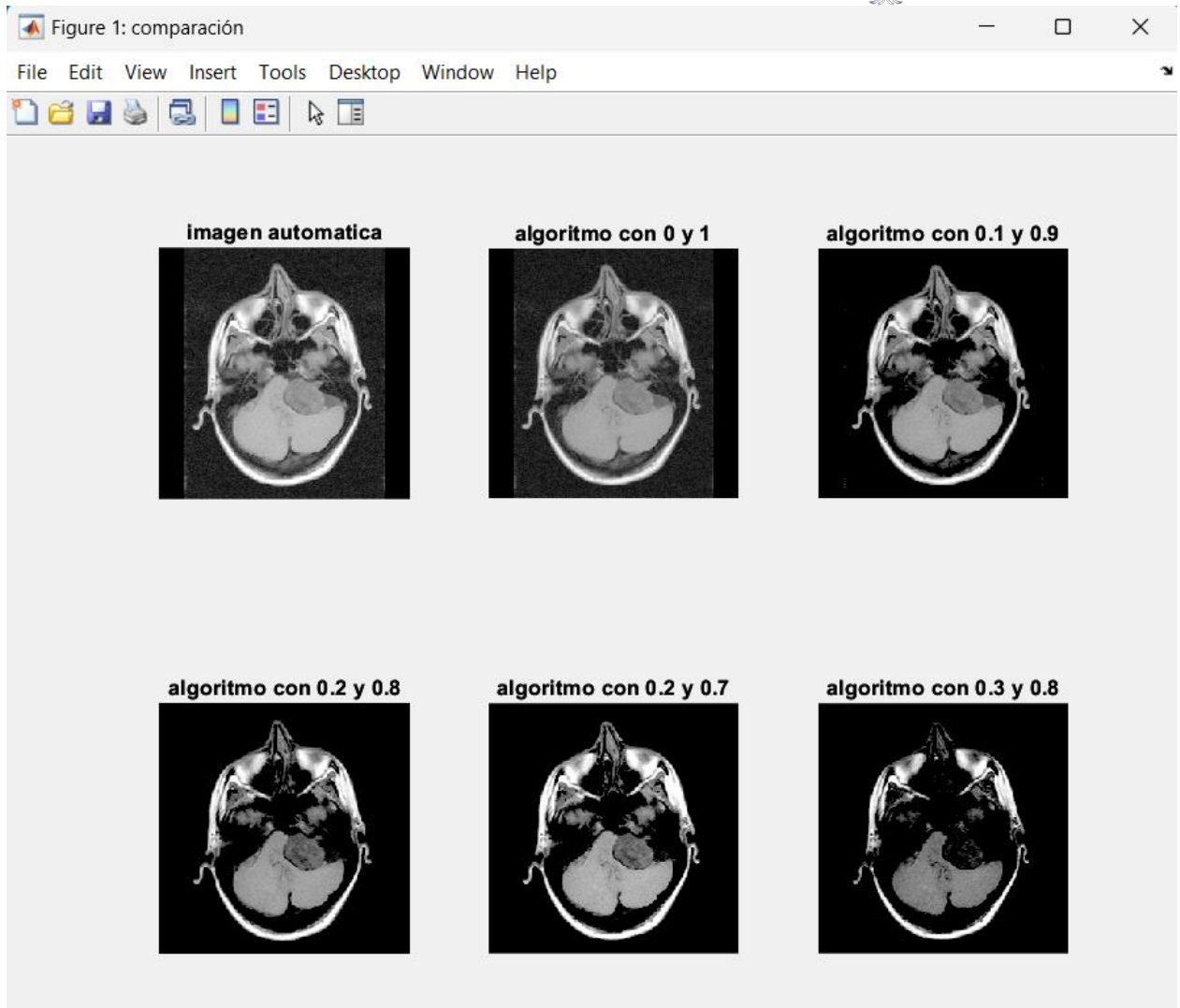
Usando: `imshow(ajuste_gamma_sqrt(imagen,0.2,0.8))`



Obtenemos

Y si queremos seguir comparando más específicamente los valores podremos usar el siguiente código

```
%%
imagen=dicomread('im5');
imagen_ajustada=imadjust(imagen);
figure('Name', 'comparación')
subplot(2,3,1);
imagen_sqrtgamma = imadjust(imagen_ajustada,[0; 1],[0; 1],0.5);
imshow(imagen_sqrtgamma)
title('imagen automatica')
subplot(2,3,2);
imshow(ajuste_gamma_sqrt(imagen))
title('algoritmo con 0 y 1')
subplot(2,3,3);
imshow(ajuste_gamma_sqrt(imagen,0.1,0.9))
title('algoritmo con 0.1 y 0.9')
subplot(2,3,4);
imshow(ajuste_gamma_sqrt(imagen,0.2,0.8))
title('algoritmo con 0.2 y 0.8')
subplot(2,3,5);
imshow(ajuste_gamma_sqrt(imagen,0.2,0.7))
title('algoritmo con 0.2 y 0.7')
subplot(2,3,6);
imshow(ajuste_gamma_sqrt(imagen,0.3,0.8))
title('algoritmo con 0.3 y 0.8')
```



5. Guardar todos los resultados de imágenes finales obtenidas para la memoria de prácticas. Use la función `imwrite(...)`.
 1. En formato png, con el mismo tipo de datos que la imagen original.
 2. En formato jpg, tras reescalar la imagen al tipo uint8 (256 niveles de gris).
 3. Adjuntar un fichero comprimido a la memoria con los ficheros de imagen obtenidos.

Código de Matlab:

En esta ocasión hacemos una tarea tan sencilla como guardar las fotos finales obtenidas para tenerlas en el archivo de entrega de la práctica, tal dice ahí, cabe acotar que entendemos como resultados finales como las imágenes pedidas en cada apartado desde el apartado 3, ya que estas son las que muestran más fielmente como es trabajar con imágenes biomédicas que a fin y a cabo es nuestro interés en esta práctica.

Para guardar las imágenes usamos el comando **imwrite ()** dándole la imagen y el nombre que tendrá el archivo (es prefer, en el caso de jpg también tendremos que pasar la imagen a formato uint8. Entiendo que la practica lo pide así porque el formato jpg pierde mucha calidad de imagen y mejor manejar menores datos y no perder tanta información.

%% 5 guardar las imagenes

```
imwrite(imagen,"imagen_original.png");  
imwrite(imagen_ajustada,"imagen_ajustada.png");  
imwrite(imagen_negativo,"imagen_negativo.png");  
imwrite(imagen_gamma2,"imagen_gamma2.png");  
imwrite(imagen_sqrtgamma,"imagen_sqrtgamma.png");  
imwrite(imagen_optima,"imagen_optima4.png");  
imwrite(imagen_nueva,"imagen_percentiles5_95.png");
```

%% 5.2 uint8(y jpg

```
imwrite(uint8(imagen),"imagen_original.jpg");  
imwrite(uint8(imagen_ajustada),"imagen_ajustada.jpg");  
imwrite(uint8(imagen_negativo),"imagen_negativo.jpg");  
imwrite(uint8(imagen_gamma2),"imagen_gamma2.jpg");  
imwrite(uint8(imagen_sqrtgamma),"imagen_sqrtgamma.jpg");  
imwrite(uint8(imagen_optima),"imagen_optima4.jpg");  
imwrite(uint8(imagen_nueva),"imagen_percentiles5_95.jpg");
```

%%

A continuación, se darán las conclusiones, luego el código usado en toda la práctica y el resto de información de dicominfo que no fue utilizada en el primer apartado.

Conclusión

Las **imágenes biomédicas** son imágenes que se utilizan en el ámbito de la salud para la diagnosis, el tratamiento y el seguimiento de pacientes. Son una herramienta fundamental para los profesionales médicos, ya que les permiten visualizar el interior del cuerpo humano y diagnosticar enfermedades. Estas imágenes son manejadas con el formato **DICOM**, que es un estándar internacional para el almacenamiento y la transmisión de imágenes biomédicas. Este formato permite que las imágenes puedan ser compartidas entre diferentes sistemas informáticos y dispositivos médicos, por eso su manejo es fundamental para un ingeniero de la salud.

La práctica ha permitido adquirir los conocimientos y las habilidades necesarias para manipular imágenes biomédicas en formato DICOM. Se ha aprendido a visualizar imágenes biomédicas y a realizar operaciones básicas de tratamiento de imágenes para mejorar su visualización. Se puede hacer énfasis a la gran importancia de utilizar el comando "imadjust()" para aumentar el contraste de las imágenes de tumor y masa cerebral obtenidas de resonancia magnética.

Además de la aplicación del comando "imadjust()" para aumentar el contraste de las imágenes de tumor y masa cerebral obtenidas de resonancia magnética, la práctica también incluye la aplicación específica de algoritmos de mejora de contraste. Estos algoritmos son los siguientes:

- **Negativo de la imagen:** Este algoritmo invierte los valores de píxeles de la imagen, de modo que los píxeles más oscuros se convierten en los más claros y viceversa.
- **Corrección gamma:** Este algoritmo multiplica los valores de píxeles de la imagen por un factor constante, que se denomina factor gamma. El factor gamma puede utilizarse para aumentar o disminuir el contraste de la imagen.
- **Ajuste $y=\sqrt{x}$:** Este algoritmo aplica una transformación cuadrática a los valores de píxeles de la imagen. Esta transformación puede utilizarse para aumentar el contraste de la imagen en los valores de píxeles más bajos.
- **Ajuste lineal a una ventana óptima:** Este algoritmo selecciona un rango de niveles de gris de interés y aplica una transformación lineal a los valores de píxeles de la imagen en ese rango. Esta transformación puede utilizarse para mejorar la visualización de las estructuras de interés en la imagen.
- **Ajuste lineal a un rango de entrada comprendido entre los percentiles 5% y el 95%:** Este algoritmo aplica una transformación lineal a los valores de píxeles de la imagen en un rango de entrada comprendido entre los percentiles 5% y el 95%. Este rango de entrada se calcula utilizando el histograma de la imagen.

Por ultimo agradecer acompañar hasta aquí a continuación el código.

Bibliografía

R.C. Gonzalez, R.E. Woods; Digital Image Processing, 3ed, Prentice-Hall 2007.

R.C. Gonzalez, R.E. Woods, S.L. Eddins; Digital Image Processing using Matlab, 2ed, Prentice-Hall 2009.

W.K. Pratt; Digital Image Processing, 4ed, Wiley 2007.

Guion de practica ingeniería de la salud UMA 2024

Y gracias especiales al profesor Ignacio que estuvo en todo momento para solventar dudas y explicar los apartados de la practica

Codigo

```
%IMAGENES BIOMEDICAS
```

```
%%
```

```
%apartado 1) dicominfo('im5') o dicomread('im5')  
imagen=dicomread('im5');  
imagen_info=dicominfo('im5');
```

```
imshow(imagen)  
figure(1)
```

```
%apartado 2) imhist(imagen), xlim([0,2000])  
%%
```

```
max_imagen=max(max(imagen)); %536  
figure(2)  
imshow(imagen,[0,max_imagen])  
%con los percentiles sería  
%p1 = prctile(imagen(:), 1);  
%p99 = prctile(imagen(:), 99);  
%imshow(imagen, [p1, p99]);  
%%
```

```
figure(3)  
[~, ~]=imhist(imagen);  
imhist(imagen, 65535)  
axis([0 max_imagen 0 2200])  
%xlim([0,max_imagen])  
%%  
%algoritmo::  
i=double(imagen);  
[filas, columnas]=size(imagen);  
for i=1:65535  
    h(i)=0;  
end  
for i= 1:filas  
    for j=1:columnas  
        k=imagen(i,j);  
        h(k+1) = h(k+1)+1;  
    end  
end  
figure(4)
```

```

plot(h)
axis([0 max_imagen 0 2200])
%% apartado 3

figure(5)
imshow(imagen, [50, 225])%valores donde yo veo mayor diferencia entre el
tumor
% y el cerebro

%% apartado 4
%Cargamos la imagen y la ajustamos automaticamente
imagen=dicomread('im5');
imagen_ajustada=imadjust(imagen);
figure(6)
imshow(imagen_ajustada) %Visualizacion de la imagen
%J = imadjust(I,[LOW_IN; HIGH_IN],[LOW_OUT; HIGH_OUT],gamma=1)
%%

%negativo de la imagen:
%J = imadjust(I,[LOW_IN; HIGH_IN],[LOW_OUT; HIGH_OUT],gamma=1) se consigue
%cambiando el low_out por el high_out y visceversa, en este caso:
imagen_negativo = imadjust(imagen_ajustada,[0; 1],[1; 0],1);
figure(7)
imshow(imagen_negativo)
%%

%Correccion gamma =2:
%J = imadjust(I,[LOW_IN; HIGH_IN],[LOW_OUT; HIGH_OUT],gamma=2) se consigue
imagen_gamma2 = imadjust(imagen_ajustada,[0; 1],[0; 1],2);
figure(8)
imshow(imagen_gamma2)

%%

%Ajuste y=sqrt(x), es decir gamma=0,5
%J = imadjust(I,[LOW_IN; HIGH_IN],[LOW_OUT; HIGH_OUT],gamma=0.5) se consigue
imagen_sqrtgamma = imadjust(imagen_ajustada,[0; 1],[0; 1],0.5);
figure(9)
imshow(imagen_sqrtgamma)
%%
% imshow(imagen, [50, 225])
v=[50,225];
v=v/65535;
imagen_optima= imadjust(imagen,[v(1); v(2)],[0; 1]);
figure(10)
imshow(imagen_optima)
%%
%con los percentiles se podría calcular con stretchlim(imagen,[0.01,0.99])
%o con p1 = prctile(imagen(:), 1) y p99 = prctile(imagen(:), 99);
[p] = stretchlim(imagen,[0.01,0.99]);
imagen_nueva= imadjust(imagen,[p(1);p(2)],[0; 1]);
figure(11)
imshow(imagen_nueva)

%%
figure('Name', 'casos anteriores')
subplot(2,3,1);
imshow(imagen_ajustada)
title('imagen automatica')

```

```
subplot(2,3,2);
imshow(imagen_negativo)
title('imagen negativa')
subplot(2,3,3);
imshow(imagen_gamma2)
title('imagen gamma=2')
subplot(2,3,4);
imshow(imagen_sqrtgamma)
title('imagen gamma=0.5')
subplot(2,3,5);
imshow(imagen_optima)
title('imagen optima')
subplot(2,3,6);
imshow(imagen_nueva)
title('imagen con percentil 5 y 95')

%%
%programar algoritmo de la funcion gamma 0,5

imagen=dicomread('im5');
imagen_ajustada=imadjust(imagen);
figure('Name', 'comparación')
subplot(1,2,1);
imagen_sqrtgamma = imadjust(imagen_ajustada,[0; 1],[0; 1],0.5);
imshow(imagen_sqrtgamma)
title('imagen automatica')
subplot(1,2,2);
imshow(ajuste_gamma_sqrt(imagen))
title('algoritmo propio')

%%
imagen=dicomread('im5');
imagen_ajustada=imadjust(imagen);
figure('Name', 'comparación')
subplot(2,3,1);
imagen_sqrtgamma = imadjust(imagen_ajustada,[0; 1],[0; 1],0.5);
imshow(imagen_sqrtgamma)
title('imagen automatica')
subplot(2,3,2);
imshow(ajuste_gamma_sqrt(imagen))
title('algoritmo con 0 y 1')
subplot(2,3,3);
imshow(ajuste_gamma_sqrt(imagen,0.1,0.9))
title('algoritmo con 0.1 y 0.9')
subplot(2,3,4);
imshow(ajuste_gamma_sqrt(imagen,0.2,0.8))
title('algoritmo con 0.2 y 0.8')
subplot(2,3,5);
imshow(ajuste_gamma_sqrt(imagen,0.2,0.7))
title('algoritmo con 0.2 y 0.7')
subplot(2,3,6);
imshow(ajuste_gamma_sqrt(imagen,0.3,0.8))
title('algoritmo con 0.3 y 0.8')

%% 5 guardar las imagenes
imwrite(imagen,"imagen_original.png");
imwrite(imagen_ajustada,"imagen_ajustada.png");
imwrite(imagen_negativo,"imagen_negativo.png");
imwrite(imagen_gamma2,"imagen_gamma2.png");
```



```
imwrite(imagen_sqrtgamma,"imagen_sqrtgamma.png");  
imwrite(imagen_optima,"imagen_optima4.png");  
imwrite(imagen_nueva,"imagen_percentiles5_95.png");
```

```
%% 5.2 uint8( y jpg
```

```
imwrite(uint8(imagen),"imagen_original.jpg");  
imwrite(uint8(imagen_ajustada),"imagen_ajustada.jpg");  
imwrite(uint8(imagen_negativo),"imagen_negativo.jpg");  
imwrite(uint8(imagen_gamma2),"imagen_gamma2.jpg");  
imwrite(uint8(imagen_sqrtgamma),"imagen_sqrtgamma.jpg");  
imwrite(uint8(imagen_optima),"imagen_optima4.jpg");  
imwrite(uint8(imagen_nueva),"imagen_percentiles5_95.jpg");
```

```
%%
```

```
% algoritmo de mejora gamma = 0.5
```

```
function resultado=ajuste_gamma_sqrt(imagen,low_in, high_in)  
if nargin < 2  
    low_in = 0;  
end  
if nargin < 3  
    high_in = 1;  
end  
  
imagen=double(imagen);  
imagen = mat2gray(imagen);  
[filas, columnas]=size(imagen);  
resultado=zeros(filas,columnas);  
for i=1:filas  
    for j=1: columnas  
        pixel=imagen(i,j);  
        if (pixel >= low_in && pixel <= high_in)  
            resultado(i,j)=((pixel-low_in)/(high_in-low_in))^0.5;  
        else if pixel>high_in  
            resultado(i,j)=65535;  
        else  
            resultado(i,j)=0;  
        end  
    end  
end  
end  
end
```

Todos los datos a obtenidos del **dicominfo**:

Demás datos que no vi de carácter obligatorio mostrar porque ocupan mucho espacio y parece ser información repetitiva

```

FileModDate: '05-oct.-2023 11:34:29'
FileSize: 138766
Format: 'DICOM'
FormatVersion: 3
Width: 256
Height: 256
BitDepth: 12
ColorType: 'grayscale'
FileMetaInformationGroupLength: 194
FileMetaInformationVersion: [2×1 uint8]
MediaStorageSOPClassUID: '1.2.840.10008.5.1.4.1.1.4'
MediaStorageSOPInstanceUID:
'1.3.46.670589.11.0.0.11.4.2.0.825.5.2060.2008072320484526991'
TransferSyntaxUID: '1.2.840.10008.1.2.1'
ImplementationClassUID: '1.3.46.670589.5.2.23'
ImplementationVersionName: 'ViewForum R7.2'
IdentifyingGroupLength: 1446
SpecificCharacterSet: 'ISO_IR 100'
ImageType: 'ORIGINAL\PRIMARY\M_SE\M\SE'
InstanceCreationDate: '20080723'
InstanceCreationTime: '214210'
InstanceCreatorUID: '1.3.46.670589.11.825.5'
SOPClassUID: '1.2.840.10008.5.1.4.1.1.4'
SOPInstanceUID:
'1.3.46.670589.11.0.0.11.4.2.0.825.5.2060.2008072320484526991'
StudyDate: '20080723'
SeriesDate: '20080723'
AcquisitionDate: '20080723'
ContentDate: '20080723'
StudyTime: '203923'
SeriesTime: '204457.10000'
AcquisitionTime: '204457.10'
ContentTime: '204457.10'
AccessionNumber: '1.943486.1.1'
Modality: 'MR'
ModalitiesInStudy: 'MR'
Manufacturer: 'Philips Medical Systems'
InstitutionName: 'DADISA-CADIZ'
ReferringPhysicianName: [1×1 struct]
StationName: 'Intera'
StudyDescription: 'CRANEO Y CAIS S/C/C'
ProcedureCodeSequence: [1×1 struct]
SeriesDescription: 'T1/SE/TRA'
InstitutionalDepartmentName: 'RESONANCIA'
OperatorsName: [1×1 struct]
AdmittingDiagnosesDescription: "
ManufacturerModelName: 'Intera'

```

ReferencedStudySequence: [1×1 struct]
ReferencedPerformedProcedureStepSequence: [1×1 struct]
ReferencedImageSequence: [1×1 struct]
PatientGroupLength: 86
PatientName: [1×1 struct]
PatientID: 'CAIS'
IssuerOfPatientID: "
PatientBirthDate: '19780119'
PatientSex: 'F'
PatientWeight: 58
PregnancyStatus: 4
AcquisitionGroupLength: 468
ScanningSequence: 'SE'
SequenceVariant: 'SS'
ScanOptions: "
MRAcquisitionType: "
SequenceName: "
SliceThickness: 5
RepetitionTime: 596.1186
EchoTime: 15
NumberOfAverages: 3
ImagingFrequency: 21.2956
ImagedNucleus: '1H'
EchoNumbers: 1
MagneticFieldStrength: 0.5000
SpacingBetweenSlices: 6
NumberOfPhaseEncodingSteps: 153
EchoTrainLength: 0
PercentSampling: 60
PercentPhaseFieldOfView: 79.6875
PixelBandwidth: 0
DeviceSerialNumber: '00825'
SoftwareVersions: '11.1.4\11.1.4.2\Gyrosan PMS/DICOM 2.0 MR \$Id:
datadefs,v 5.27 2004/10/18 06:50:'
ProtocolName: 'T1/SE/TRA'
LowRRValue: 0
HighRRValue: 0
IntervalsAcquired: 0
IntervalsRejected: 0
HeartRate: 0
ReconstructionDiameter: 230
ReceiveCoilName: 'Head'
TransmitCoilName: 'B'
AcquisitionMatrix: [4×1 uint16]
InPlanePhaseEncodingDirection: 'ROW'
FlipAngle: 60
PatientPosition: 'HFS'
RelationshipGroupLength: 492
StudyInstanceUID:
'748048.649048.265055.448055.168053.867.20080723.202619.916'

SeriesInstanceUID:
'1.3.46.670589.11.0.0.11.4.2.0.825.5.2060.2008072320445712975'
StudyID: '270074363'
SeriesNumber: 301
AcquisitionNumber: 3
InstanceNumber: 5
ImagePositionPatient: [3×1 double]
ImageOrientationPatient: [6×1 double]
FrameOfReferenceUID:
'1.3.46.670589.11.0.0.11.4.2.0.825.5.9560.2008072320383625000'
Laterality: "
TemporalPositionIdentifier: 1
NumberOfTemporalPositions: 1
PositionReferenceIndicator: "
SliceLocation: 24.0000
NumberOfStudyRelatedInstances: 260
ImagePresentationGroupLength: 204
SamplesPerPixel: 1
PhotometricInterpretation: 'MONOCHROME2'
Rows: 256
Columns: 256
PixelSpacing: [2×1 double]
BitsAllocated: 16
BitsStored: 12
HighBit: 11
PixelRepresentation: 0
WindowCenter: 1048
WindowWidth: 1822
RescaleIntercept: 0
RescaleSlope: 3.8398
RescaleType: 'normalized'
LossyImageCompression: '00'
StudyGroupLength: 14
RequestedProcedureDescription: 'Craneo'
Unknown_0040_0000: 254
PerformedStationAETitle: 'MR_STORE_CADIZ'
PerformedProcedureStepStartDate: '20080723'
PerformedProcedureStepStartTime: '203923'
PerformedProcedureStepEndDate: '20080723'
PerformedProcedureStepEndTime: '203923'
PerformedProcedureStepID: '270074363'
PerformedProcedureStepDescription: 'CRANEO Y CAIS S/C/C'
PerformedProtocolCodeSequence: [1×1 struct]
RequestAttributesSequence: [1×1 struct]
FilmConsumptionSequence: [1×1 struct]
RequestedProcedureID: '124886'
Private_07a1_GroupLength: 192
Private_07a1_a0xx_Creator: 'ELSCINT1'
Private_07a1_a002: 22
Private_07a1_a02a: 'UNREAD'
Private_07a1_a043: 1

Private_07a1_a050: 0
Private_07a1_a058: 'A'
Private_07a1_a05f: 'N'
Private_07a1_a070: 'AN0128461646'
Private_07a1_a071: 'CDZ-383918.1'
Private_07a1_a085: 3
Private_07a1_a088: 'N'
Private_07a1_a098: 'N'
Private_07a1_a0c0: '1.43'
Private_07a1_a0c1: '1.412'
Private_07a1_a0c2: '260'
Private_07a3_GroupLength: 204
Private_07a3_a0xx_Creator: 'ELSCINT1'
Private_07a3_a001: '11.4.0.0'
Private_07a3_a003: 'N'
Private_07a3_a014: '001011'
Private_07a3_a017: 'SC'
Private_07a3_a01f: 'MONTERO DOMINGUEZ^SILVIA'
Private_07a3_a022: '1.16323'
Private_07a3_a023: '1.782.1'
Private_07a3_a024: 'Y'
Private_07a3_a034: '11140'
Private_07a3_a055: 'N'
Private_07a3_a099: 'N'
Private_07a3_a09c: 'N'
Private_07a3_a0b9: 'N'
Private_07a3_a0bb: 'N'
Private_07a5_GroupLength: 26
Private_07a5_a0xx_Creator: 'ELSCINT1'
Private_07a5_a056: 'N'
Private_2001_GroupLength: 1048
Private_2001_10xx_Creator: 'Philips Imaging DD 001'
Private_2001_1001: 0
Private_2001_1002: 0
Private_2001_1003: 0
Private_2001_1006: 'N'
Private_2001_1007: 'U'
Private_2001_1008: 1
Private_2001_1009: 0
Private_2001_100a: 5
Private_2001_100b: 'TRANSVERSAL'
Private_2001_100c: 'N'
Private_2001_100e: 'N'
Private_2001_100f: 0
Private_2001_1010: 'NO'
Private_2001_1011: 0
Private_2001_1012: 'N'
Private_2001_1013: 1
Private_2001_1014: 1
Private_2001_1015: 1
Private_2001_1016: 0

Private_2001_1017: 1
Private_2001_1018: 22
Private_2001_1019: 'N'
Private_2001_101a: [3×1 single]
Private_2001_101b: 0
Private_2001_101c: 'NO'
Private_2001_101d: 1
Private_2001_101f: 'NO'
Private_2001_1020: 'SE'
Private_2001_1021: 'N'
Private_2001_1022: 0.7340
Private_2001_1023: 60
Private_2001_1024: 'N'
Private_2001_1025: '15'
Private_2001_105f: [1×1 struct]
Private_2001_1060: 1
Private_2001_1061: 'N'
Private_2001_1062: 'N'
Private_2001_1063: 'ELSEWHERE'
Private_2001_107b: 3
Private_2001_1081: 1
Private_2001_1082: 0
Private_2001_1083: 21.2956
Private_2001_1084: 0
Private_2001_1085: 0.5000
Private_2001_1086: 0
Private_2001_1087: '1H'
Private_2001_1088: 3
Private_2001_1089: 0
Private_2001_108a: 0
Private_2001_108b: 'B'
Private_2001_10c8: 'ISQUEMIA, DEMENCIA, TUMOR OPERADO O NO'
Private_2005_GroupLength: 2654
Private_2005_10xx_Creator: 'Philips MR Imaging DD 001'
Private_2005_11xx_Creator: 'Philips MR Imaging DD 002'
Private_2005_12xx_Creator: 'Philips MR Imaging DD 003'
Private_2005_13xx_Creator: 'Philips MR Imaging DD 004'
Private_2005_14xx_Creator: 'Philips MR Imaging DD 005'
Private_2005_1000: -4.1577
Private_2005_1001: 3.2952
Private_2005_1002: -0.2860
Private_2005_1004: 'NONE'
Private_2005_1008: 2.1701
Private_2005_1009: -21.1863
Private_2005_100a: 4.1194
Private_2005_100b: 3.6270e+04
Private_2005_100c: 2.5049
Private_2005_100d: 0
Private_2005_100e: 0.0148
Private_2005_100f: 1048
Private_2005_1010: 1822

Private_2005_1011: 'M'
Private_2005_1012: 'N'
Private_2005_1013: 'NO'
Private_2005_1014: 'N'
Private_2005_1015: 'N'
Private_2005_1016: 'N'
Private_2005_1017: 'N'
Private_2005_1019: 'N'
Private_2005_101a: 0
Private_2005_101b: 'N'
Private_2005_101c: 'N'
Private_2005_101d: 256
Private_2005_101e: 'compose'
Private_2005_101f: 'compose'
Private_2005_1020: 0
Private_2005_1021: 1
Private_2005_1022: 0
Private_2005_1023: 0
Private_2005_1025: 0
Private_2005_1026: 'N'
Private_2005_1027: 'MINIMUM'
Private_2005_1028: 'N'
Private_2005_1029: 'N'
Private_2005_102a: 277936726
Private_2005_102b: 100
Private_2005_102c: 'N'
Private_2005_102d: 0
Private_2005_102e: 'N'
Private_2005_102f: 'N'
Private_2005_1030: 596.1186
Private_2005_1031: 'N'
Private_2005_1033: 219.9678
Private_2005_1034: 'N'
Private_2005_1035: 'PIXEL'
Private_2005_1036: 'N'
Private_2005_1037: 'N'
Private_2005_1038: 'N'
Private_2005_1039: 'Y'
Private_2005_103b: 'N'
Private_2005_103c: 'N'
Private_2005_103d: 102
Private_2005_103e: [102×1 int32]
Private_2005_105f: 'UNKNOWN'
Private_2005_1060: -1
Private_2005_1061: 'NO'
Private_2005_1063: 0
Private_2005_106e: 'SE'
Private_2005_106f: 'MS'
Private_2005_1080: [1×1 struct]
Private_2005_1083: [1×1 struct]
Private_2005_1084: [1×1 struct]

Private_2005_1085: [1×1 struct]
Private_2005_1086: 0
Private_2005_109e: [1×1 struct]
Private_2005_10a0: 0
Private_2005_10a2: 'N'
Private_2005_10a8: 0
Private_2005_10a9: 'NA'
Private_2005_10b0: 0
Private_2005_10b1: 0
Private_2005_10b2: 0
Private_2005_10c0: 'SE'
Private_2005_1199: 1
Private_2005_1200: 1
Private_2005_1201: 0
Private_2005_1213: 0
Private_2005_1245: 2
Private_2005_1249: 0
Private_2005_1251: 0
Private_2005_1252: 1
Private_2005_1253: 1
Private_2005_1325: 'N'
Private_2005_1326: 0
Private_2005_1327: 'REAL'
Private_2005_1328: 'ORIGINAL'
Private_2005_1329: 50
Private_2005_1330:
'PLUS_A_PLUS_B\PLUS_A_PLUS_B\PLUS_A_PLUS_B\PLUS_A_PLUS_B'
Private_2005_1331: 0
Private_2005_1333: [3×1 single]
Private_2005_1334: 'UNKNOWN'
Private_2005_1335: 'UNKNOWN'
Private_2005_1336: 0
Private_2005_1337: 0
Private_2005_1338: 0
Private_2005_1339: [2×1 int16]
Private_2005_1340: 'PRE_FT'
Private_2005_1341: 'UNKNOWN'
Private_2005_1342: 'FID'
Private_2005_1343: 'Y'
Private_2005_1344: [40×1 int16]
Private_2005_1345: 'NO'
Private_2005_1346: 'HERTZ'
Private_2005_1347: 0
Private_2005_1348: 'OFF'
Private_2005_1349: 0
Private_2005_1350: [2×1 single]
Private_2005_1351: 0
Private_2005_1352: 0
Private_2005_1355: [30×1 single]
Private_2005_1356: 'NO'
Private_2005_1357: 0

Private_2005_1359: 1
Private_2005_1360: 0
Private_2005_1361: [2×1 single]
Private_2005_1362: 0
Private_2005_1363: 0
Private_2005_1364: 'NO'
Private_2005_1370: 0
Private_2005_1371: [1×1 struct]
Private_2005_1381: 3
Private_2005_1382: 0
Private_2005_1391: [1×1 struct]
Private_2005_1392: 0
Private_2005_1393: -1
Private_2005_1396: 'NO'
Private_2005_1398: 'NO'
Private_2005_1399: 'NO'
Private_2005_1400: 'YES'
Private_2005_1401: 1
Private_2005_1403: 1
Private_2005_1406: 0
Private_2005_1409: 0
Private_2005_140a: 3.8398
Private_2005_140b: 'normalized'
Private_2005_140f: [1×1 struct]
Private_2005_1412: 1
Private_2005_1413: 1
Private_2005_1414: 1
Private_2005_1415: 1
Private_2005_1428: 0
Private_2005_1435: 'N'
Private_2005_143a: 'UNKNOWN'
Unknown_2050_0000: 16
PresentationLUTShape: 'IDENTITY'
PixelDataGroupLength: 131084