به نام خدا

تمرین چهارم درس پردازش و تحلیل تصاویر پزشکی

نگین اسماعیل زاده ۹۷۱۰۴۰۳۴

تمرین های نوشتاری :

سوال ۱) این روش اولین بار برای ایجاد اشکال پویا معرفی شد و ایده اصلی آن به این صورت است که یک کانتور را به صورت مجموعه ی سطح صفر از یک تابع با درجه ی بالاتر در نظر میگیرند. به این صورت میتوان تغییر شکل کلی یک کانتور را به صور تغییر سطح صفر تابع با بعد بالاتر تعبیر کرد. به این تابع از بعد بالاتر LSF گفته میشود. ایده ی اولیه ای روش مدت ها قبل مطرح شد اما مدت ها بعد تازه شناخته شد و در حال حاضر کاربرد زیادی در مسائل ناحیه بندی در بینایی ماشین و پردازش تصویر دارد. مزیت آن این است که با این روش میتوان خم های پیچیده و ترکیب یا جدایی خم ها را با هزینه کم توصیف کرد.

معادله تکامل خم به صورت زیر است :

$$\frac{\partial \mathcal{C}(s,t)}{\partial t} = F\mathcal{N}$$

که در آن C کانتور مورد نظر است که با s پارامتری شده و با زمان نیز متغیر است (دینامیک) ، N بردار عمود به کانتور است و F در واقع تابع سرعت است که معادله ی بالا به خوبی نشان میدهد که F در واقع کنترل کننده سرعت تغییر شکل خم است.

حال اگر C را در سطح صفر LSF در نظر بگیریم و نقاط داخل کانتور را در سطح منفی و نقاط خارج کانتور را در سطح مثبت ، آنگاه میتوان N را به صورت تابعی از گرادیان LSF در نظر گرفت و به صورت دقیق تر میتوان PDEزیر را تشکیل داد :

$$\frac{\partial \phi}{\partial t} = F\,|\nabla \phi|$$

که در آن φ همان LSF میباشد.

سوال ۲) در روش level set این موضوع مهم است که تغییرات کانتور سبب تغییرات ناگهانی و غیر عادی LSF نشود به همین دلیل و برای جلوگیری از خطا در محاسبات عددی شرط دیگری به مسئله اضافه مشود که LSF به صورت یک تابع علامت دار از فاصله تشکیل شود. حال اگر LSF را یک سطح با این شرط تشکیل در نظر بگیریم، بردار عمود بر سطح آن با محور ها زاویه ۴۵ میسازد که نتیجه میدهد اندازه گرادیان LSF برابر واحد است.

برای ساخت این LSF تابع انرژی به صورت زیر تعریف میشود :

$$\mathcal{E}(\phi) = \mu \mathcal{R}_p(\phi) + \mathcal{E}_{ext}(\phi) \qquad \mathcal{R}_p(\phi) \triangleq \int_\Omega p(|\nabla \phi|) d\mathbf{x}$$

که در آن Rp ترم تنظیم کننده سطح ، P تابع پتانسیل و E انرژی خارجی است که میتواند بسته به مسئله متفاوت باشد.

ترم انرژی خارجی E به گونه ای طراحی شده است که وقتی مجموعه سطح صفر LSF در موقعیت دلخواه قرار گیرد ، به حداقل می رسد (مثلاً یک مرز شی برای هدف ناحیه بندی) . حال مسئله بهینه سازی فوق باید حل شود. برای تابع پتانسیل میتوان توابع مختلفی را در نظر گرفت اما یک انتخاب خوب و ساده تابع p = s-1^2 است. این تابع معیار های همواری را دارد ، علت شیفت s به این دلیل است که فاصله تابع تا صفحه را صفر نکنیم. با انتخاب این تابع s میمینم یکتا خواهد داشت.

$$p = p_1(s) \triangleq \frac{1}{2}(s-1)^2$$

رابطه ی بالا را میتوان به صورت زیر نوشت که تابعی از انحراف نسبت به فاصله علامتدار باشد.

$$\mathcal{P}(\phi) = \frac{1}{2}\int_\Omega (|\nabla \phi| - 1)^2 d\mathbf{x}$$

تابع دومی که برای تابع پتانسیل پیشهاد میشود این تابع است :

$$p_2(s) = \begin{cases} \frac{1}{(2\pi)^2}(1-\cos(2\pi s)), & \text{if } s \le 1 \\ \frac{1}{2}(s-1)^2, & \text{if } s \ge 1. \end{cases}$$

این تابع دو مینیم دارد و تا مکدو مرتبه مشتق پذیر است.

مزیت تابع p2 که یک پتانسیل دو چاه است نسبت به تابع p1 را میتوان به راحتی در حالتی که تابع اولیه یک step است مشاهده کرد . در این حالت در شرایط اولیه داریم : $|\nabla \varphi| = 0$ در نتیجه انتشار FAB در یک

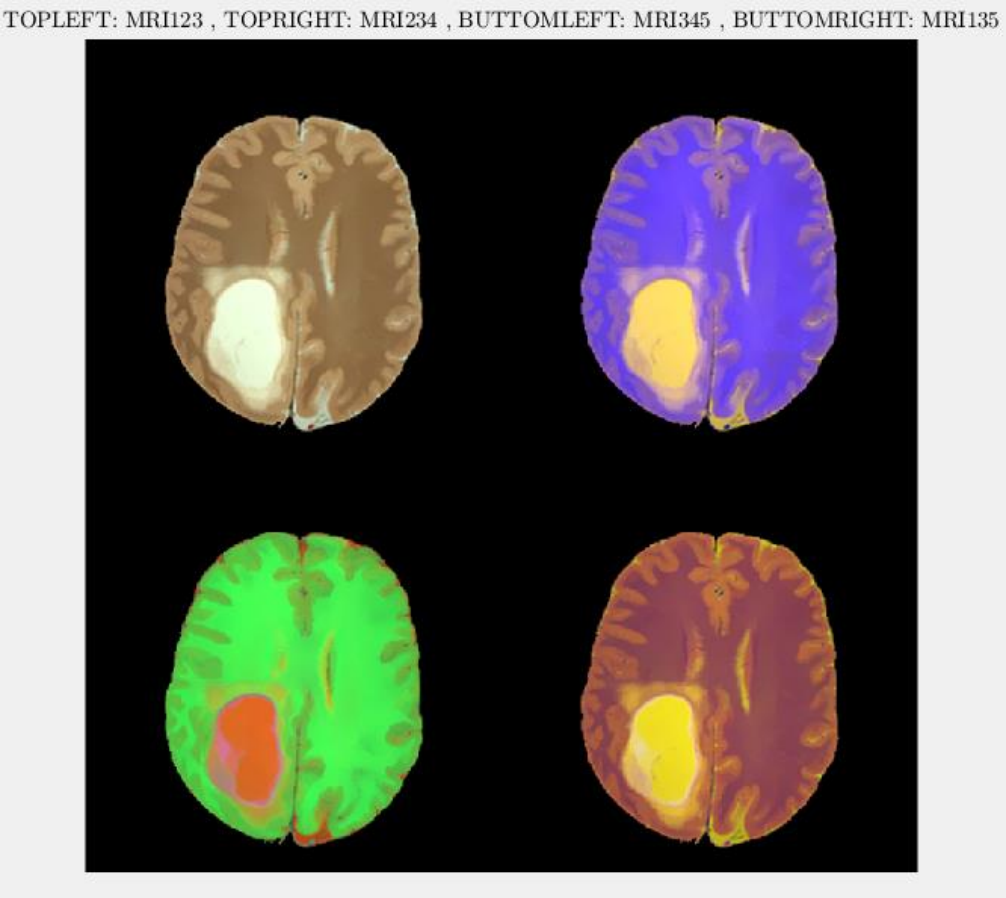
جهت بسیار بزرگ است که باعث میشود φ بسیار بزرگ شود و باعث ایجاد نوسان میشود که در نهایت به صورت قله و دره های دوره ای در LSF ضاهر میشود. اگرچه این قله و دره ها در یک فاصله مشخص از مجموعه سطح صفر ظاهر می شوند ، اما ممکن است کمی منحنی سطح صفر را تحریف کنند. با استفاده از پتانسیل دو چاه میتوان از این اثر جانبی نامطلوب جلوگیری کرد.

سوال ۳) در این روش برای حل عدی میتوان از گام های زمانی بزرگی در هر مرحله تکرار استفاده کرد تا به این ترتیب از تعداد تکرار های لازم کاسته شود. این روش دقت لازم را حتی با استفاده از گام های بزرگ تضمین میکند.

تمرین های شبیه سازی :

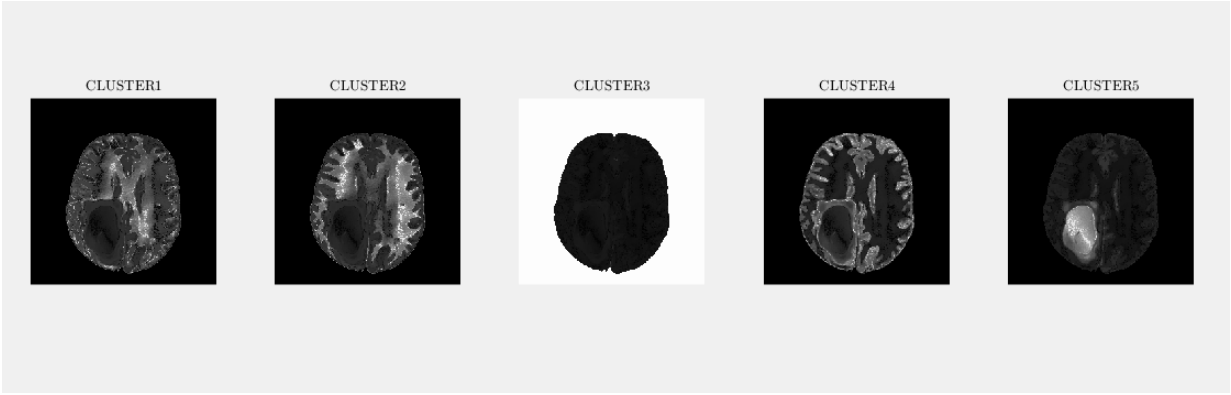
سوال ۱)

۱.۱) برای ۴ انتخاب متفاوت ۳ کانال رنگی از بین تصاویر ، تصویر RGB را نمایش می دهیم . همانطور که در شکل زیر مشخص است. با در نظر گرفتن بک گراند به عنوان یک ناحیه ی مجزا می توان گفت تصویر مجموعا از ۵ ناحیه تشکیل شده است (به صورت کلی و تقریبی شامل بک گراند ، حاشیه ی اطراف مغز ، قسمت میانی مغر ، تومور و حاشیه ی اطراف تومور می شود).

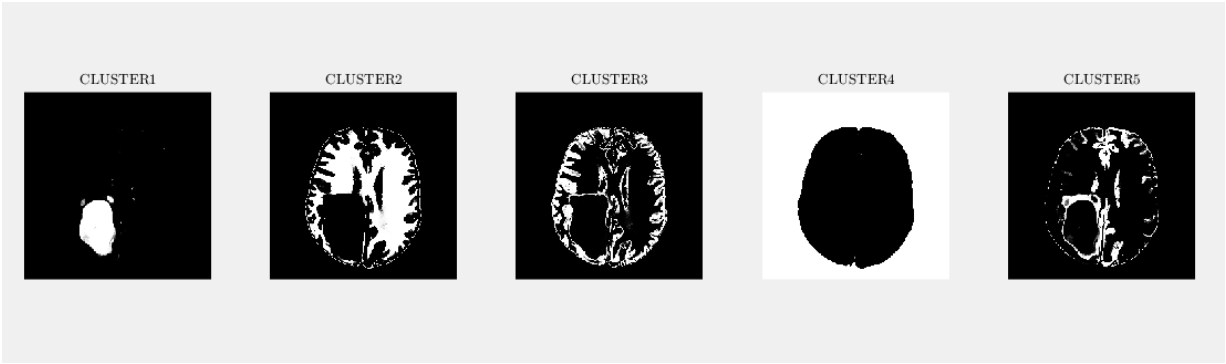


۱.۲) در کد تحویل داد شده روش FCM به صورت دستی پیاده سازی شده است اما میتوانستیم از تابع fcm متلب نیزاستفاده کنیم. بطور کلی چون روش FCM به صورت کلی برای ناحیه بندی نرم فرمولیزه شده است ضریب فازی یک را نمیتوان مستقیما روی مسئله اعمال کرد، این کار را با یک برسی نهایی روی نقشه های احتمال و در نظر گرفتن خوشه ی با بالاترین احتمال به عنوان احتمال ۱۰۰ درصد و باقی خوشه ها به عنوان۰ درصد برای هر پیکسل انجام می دهیم.

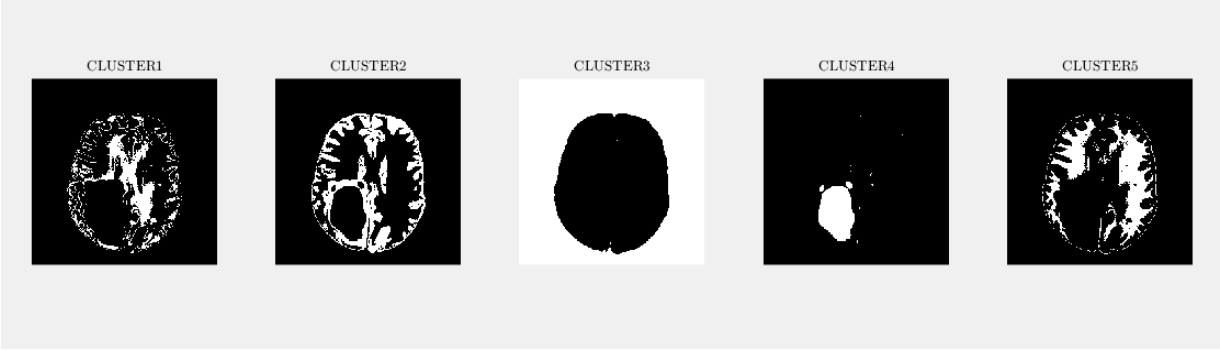
تصویر کلاستر ها به ازای ضریب فازی ۵ (ناحیه بندی نرم) :



تصویر کلاستر ها به ازای ضریب فازی ۱.۴ (ناحیه بندی نرم) :



تصویر کلاستر ها به ازای ضریب فازی واحد (ناحیه بندی سخت) :

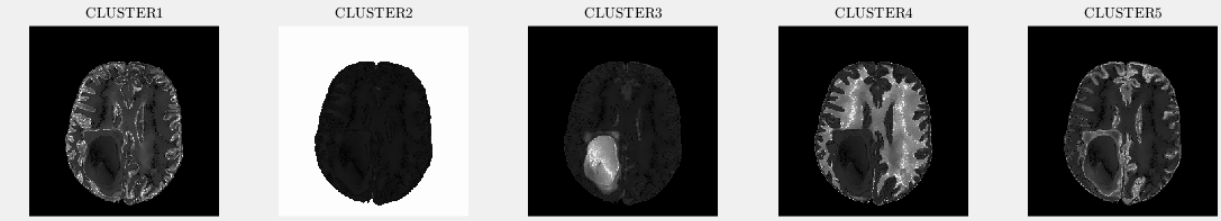


با دقت در تصاویر بالا میتوان فهمید که با افزایش ضریب فازی میتوان میزان نرم یا سخت بودن ناحیه بندی را کنترل کرد ، به این صورت که هر چه ضریب به ۱ نزدیک تر باشد نقشه احتمال هر ناحیه به صورت سیاه و سفید در میاید که نشان دهنده این است که یک پیکسل خاص در آن ناحیه بوده یا نبوده و هر چه ضریب فازی بزرگ تر شود نقشه های احتمال رنگی تر میشود به این معنا که در نقشه احتمال یک ناحیه یک پیشکل

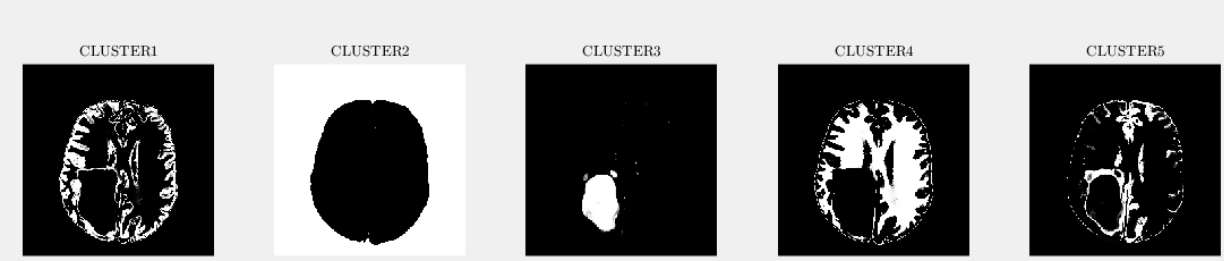
میتواند با احتمالی (نه لزوما ۱) حضور داشته باشد یا نداشته باشد و در واقع با بزرگ شدن ضریب فازی به این سمت میرویم که احتمال حضور یک پیکسل در نواحی مختلف به سمت برابری میرود.

۱.۳) با استفاده از دستور kmeans متلب خوشه بندی اولیه تعیین شد و سپس قسمت قبل تکرار شد. نتایج به صورت زیر اند :

تصویر کلاستر ها به ازای ضریب فازی ۵ (ناحیه بندی نرم) :



تصویر کلاستر ها به ازای ضریب فازی ۱.۴ (ناحیه بندی نرم) :



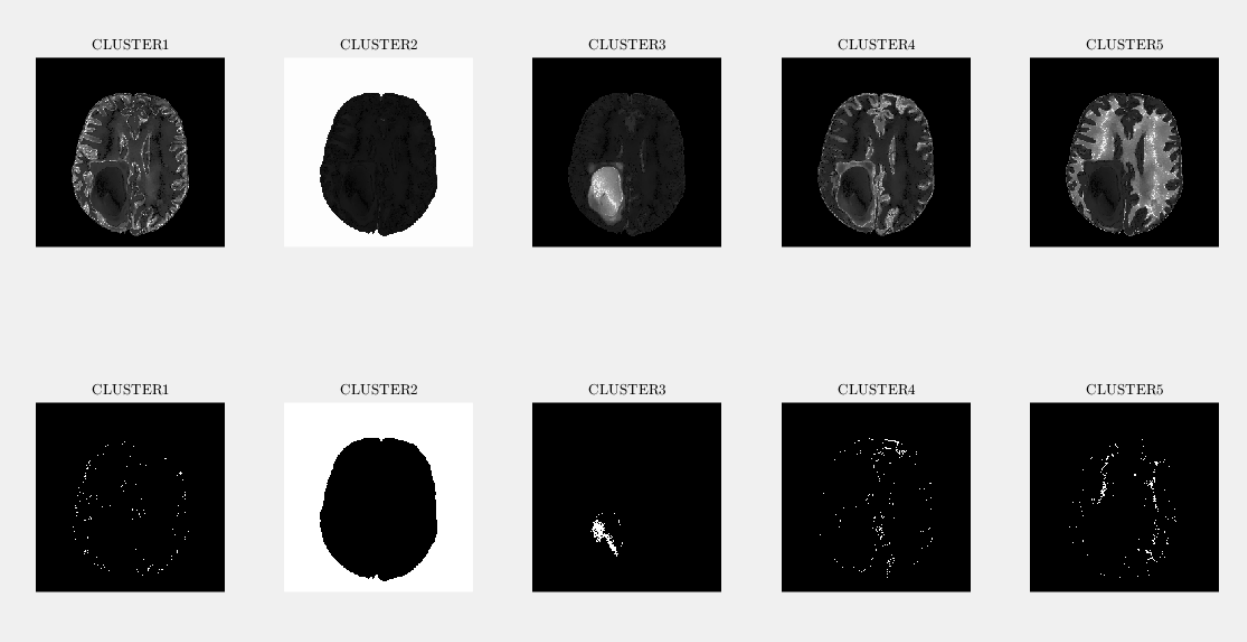
تصویر کلاستر ها به ازای ضریب فازی واحد (ناحیه بندی سخت) :

در این حالت چون به جای استفاده از اعداد رندم برای حالت اولیه مراکز خوشه ها از انتخاب هوشمندانه تری استفاده کردیم و مراکز را یک دور با یک روش ساده و سریع تر (اما با دقت کمتر) حساب کردیم این بار الگوریتم بسیار سریع تر و از حالت قبل همگرا میشود. همچنین اگر شرایط اولیه را ثابت نگه داریم در صورت اجرای مجدد الگورتیم دسته بندی ها جابجا نمیشوند، در صورتی که در حالت شرایط اولیه رندم این اتفاق بسیار محتمل بود. در واقع با این کار با فرض ثابت بودن تعداد تکرار نسبت به حالت قبل به دقت بیشتر ، و در صورت ثابت بودن دقت نسبت به حالت قبل به تعداد تکرار لازم کمتری میرسیم.

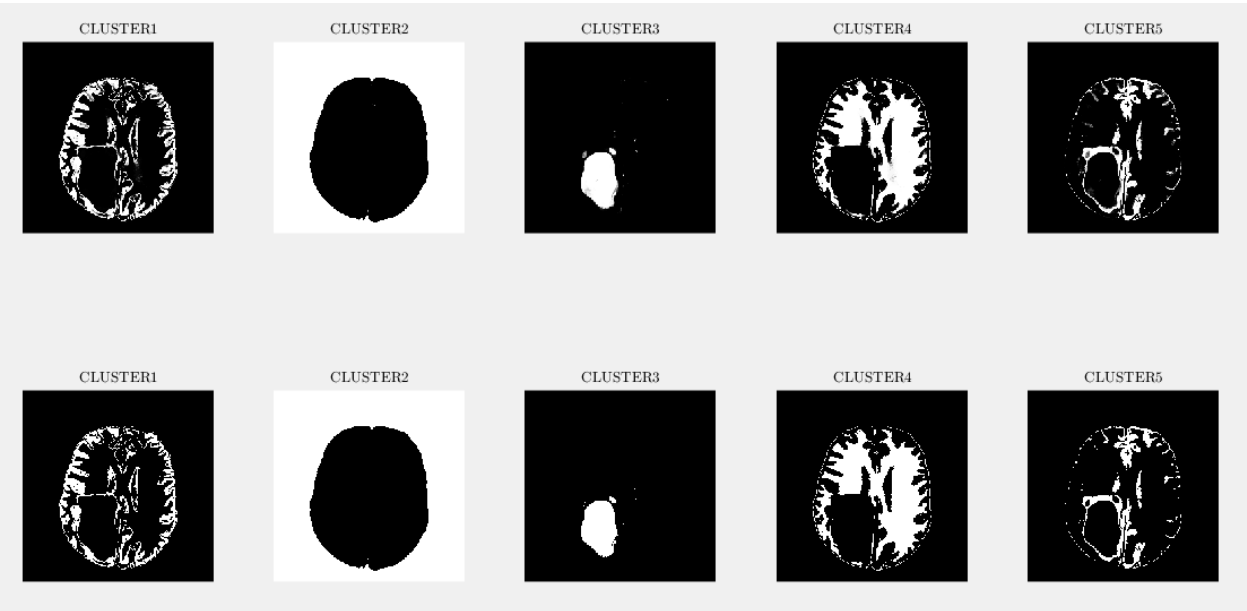
۱.۴) اینبار با استفاده از GMM ، خوشه بندی انجام شد . نتیجه به صورت زیر می باشد :

۱.۵) یک معیار مناسب برای تشخیص partial volume میتواند به این صورت باشد که بعد از رسم نقشه احتمال برای نواحی متفاوت از هر کدام از روش های قبل (طبیعتا روش های ناحیه بندی نرم) ، پیکسل ها را روی هر نقشه مشروط بر این که از یک حد معین بیشتر باشند کاملا تیره و در غیر این صورت کاملا روشن کنیم . با این کار اگر یک پیکسل به صورت غالب در یک کلاس مشخص باشد با سیاه و اگر یک پیکسل به صورت جزئی مربوط به کلاس های متفاوتی باشد و در یک کلاس خاص جای نگیرد با سفید نمایش داده میشود. این روش در ظاهر تقریبا مشابه روش ناحیه بندی سخت است اما تفات اصلی آن این است که برای مثال در ناحیه بندی سخت پیکسل (۰.۴۹ ، ۰.۵۱) به دسته ی اول تعلق پیدا میکند در صورتی که در این روش اگر حد تعلق به یک کلاس را ۰.۶۵ فرض کنیم این پیکسل مشخص کننده partial volume است و در هر دو کلاس با سفید نمایش داده میشود.

شکل زیر نتیجه اعمال این روش با انتخاب حد ۰.۶۵ است و نقشه اولیه از روش FCM با پارامتری فازی ۵ گرفته شده است (ردیف اول نقشه اولیه ، ردیف دوم مشخص کننده حجم های جزئی است) :

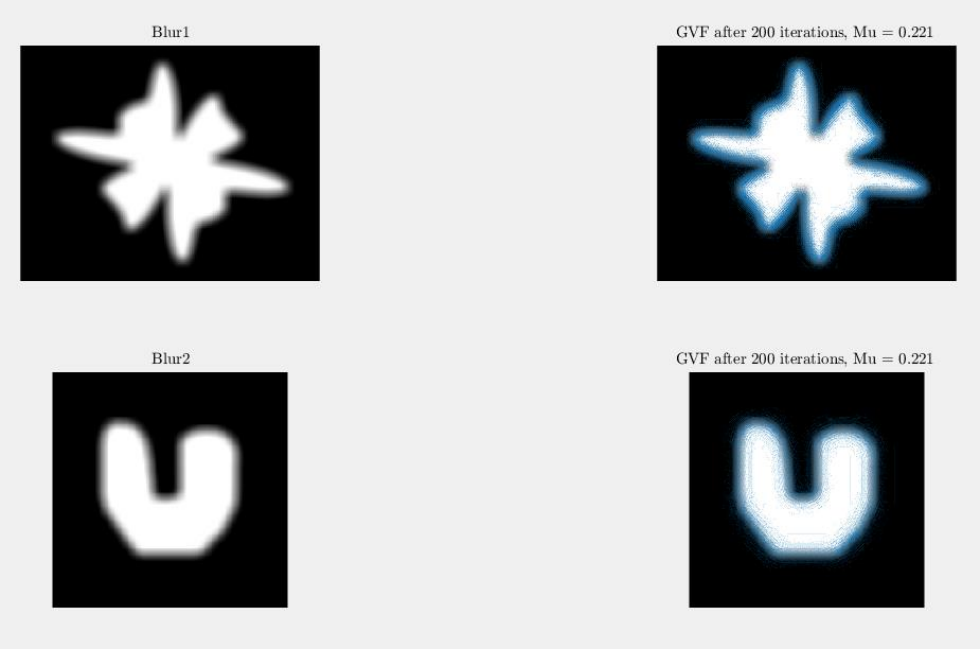


نتیجه اعمال این روش با انتخاب حد ۰.۶۵ و نقشه احتمال اولیه از روش FCM با پارامتر فازی ۱.۴ نیز در شکل زیر قابل مشاهده است:

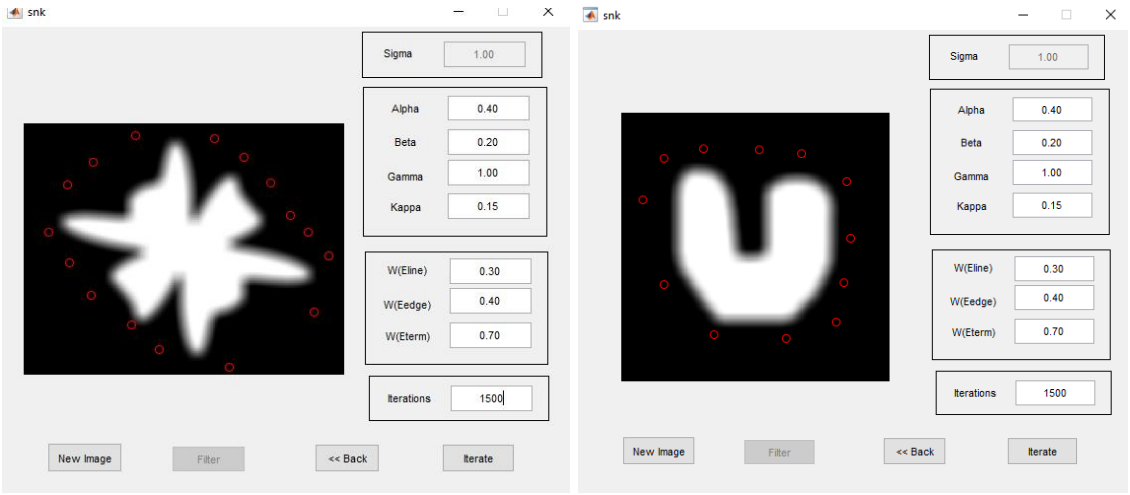


سوال ۲) " در این قسمت فقط برای روش GVF در فایل کد بخشی وجود دارد ، روش basic snake با استفاده از فایل دموی آماده اعمال شده و فقط نتایج آن در گزارش موجود است"

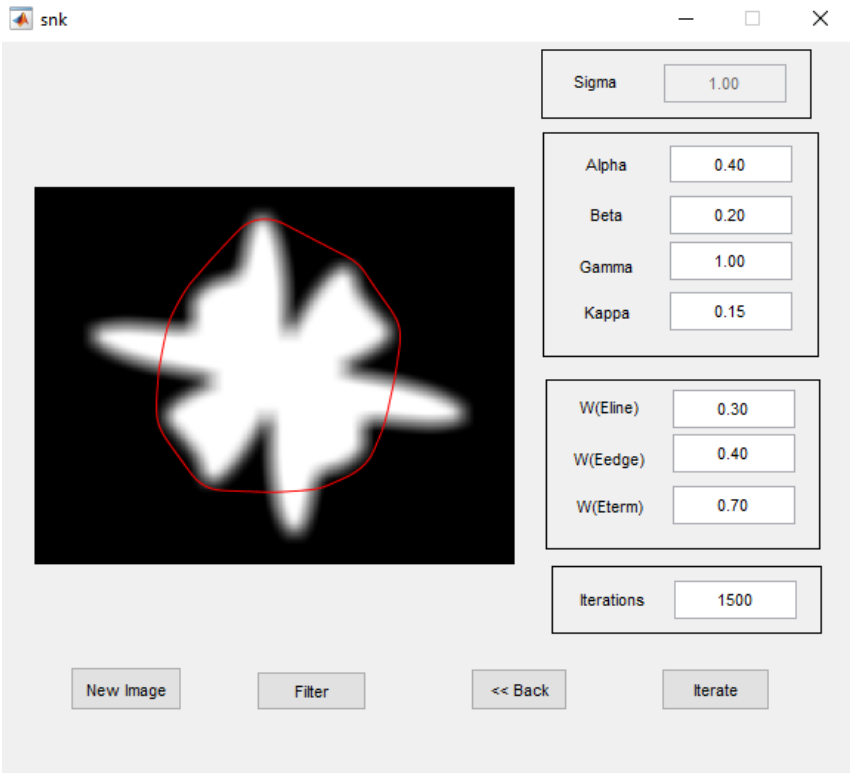
۲.۱) برای استفاده از روشGVF از فایل GVF.m استفاده شده تا وکتور اپتیموم بدست آید سپس با استفاده از تابع quiver مرز ناحیه بندی به صورت بردار های جربان نمایش شده است. شکل زیر حاصل اعمال این روش برای ناحیه بندی به ازای Mu = 0.221 و 200 مرتبه تکرار روش بر روی تصاویر Blur1 و Blur2 است.

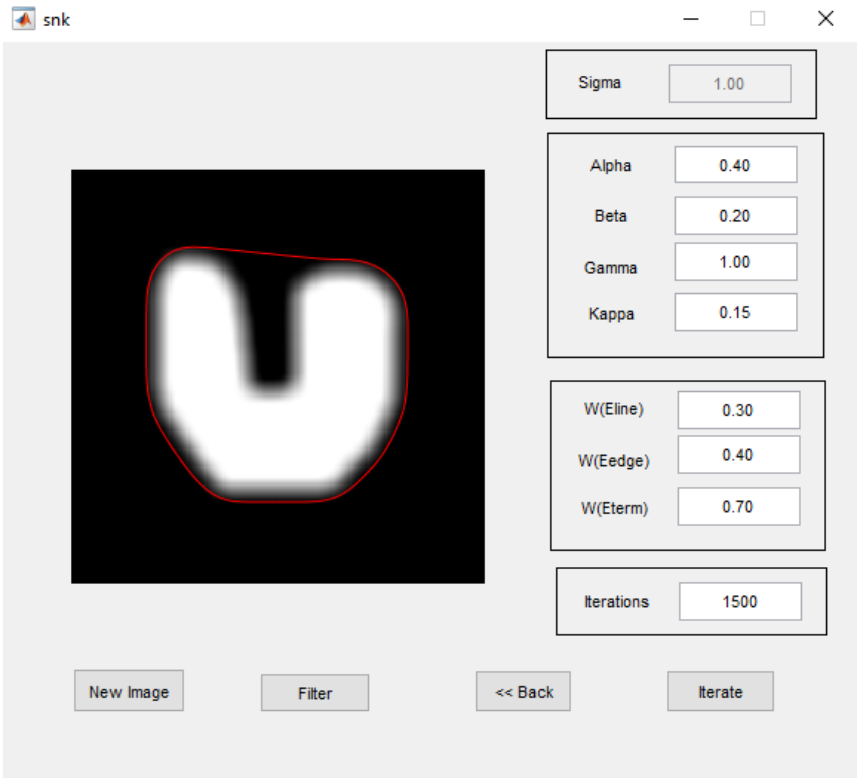


حال همین کار را با استفاده از روش basic snake انجام میدهیم . برای این روش از فایل snk.m استفاده شده است. این روش به مشخص کردن یک کانتور اولیه نیاز دارد ، در شکل های زیر به ترتیب کانتور اولیه رسم شده برای تصاویر Blur1 و Blur2 قابل مشاهده هست.



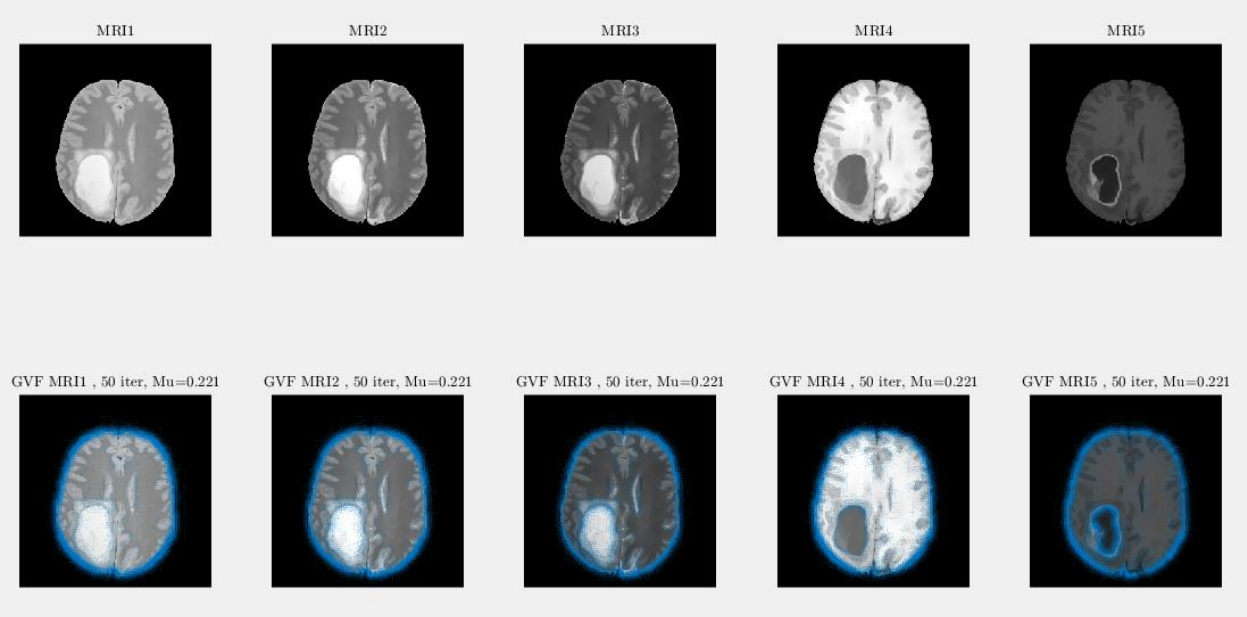
پس از انتخاب کانتور اولیه و پارامتر ها به گونه ای که در شکل مشخص شده است ، روش با ۱۵۰۰ تکرار اجرا شده است. شکل های زیر کانتور نهایی این روش را به ترتیب برای تصاویر Blur1 و Blur2 نشان میدهد :





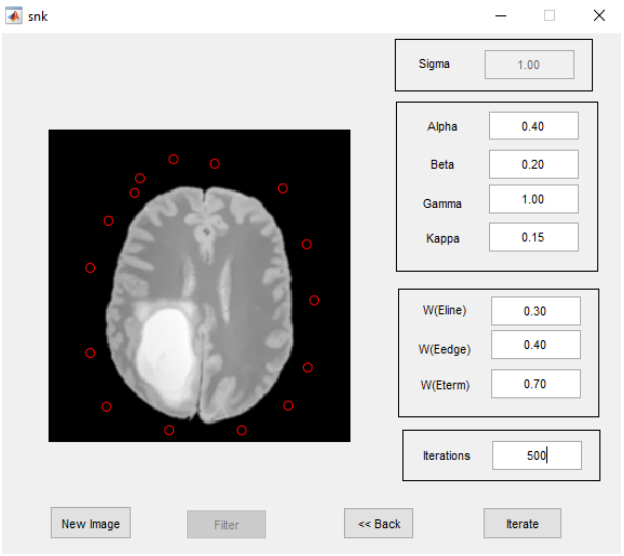
مقایسه : همانطور که در شکل های بالا مشخص است روش basic snake در شناسایی دره ها (فرورفتگی های تصویر) بسیار ضعیف عمل میکند. علت این امر هم این است که در این روش نیرو های وارد شده به خم از طرفین در بیرون از فرورفتگی ها با هم خنثی میشود و نیروی کشش به داخل نیز بسیار کم است به طوری که نمیتواند به ترم مینیم کردن طول خم غلبه کند و نتیجتا خمنمیتواند به نقاط دره ی شکل نفوذ کند. اما این موضوع به خوبی در روش GVF اصلاح شده و همانطور که میبینیم این روش توانسته است به خوبی نقاط مرز اشکال را تشخیص دهد.مورد بعدی هم انتخاب کانتور اولیه است ، روش basic sanke نیاز به انتخاب کانتور اولیه دارد که این کانتور بسیار در سرعت همگرایی وحتی درست بودن تشخیص اثر گذار است که این امر ینی این روش به اطلاعات پیشین از تصویر نیاز دارد که نامطلوب است و این موضوع در روش GVF مطرح نیست.

۲.۲) روش GVF مطابق آنچه در قسمت قبل گفته شد اینبار روی هر ۵ تصویر MRI اعمال شده است و نتیجه حال در شکل زیر قابل مشاهده است.

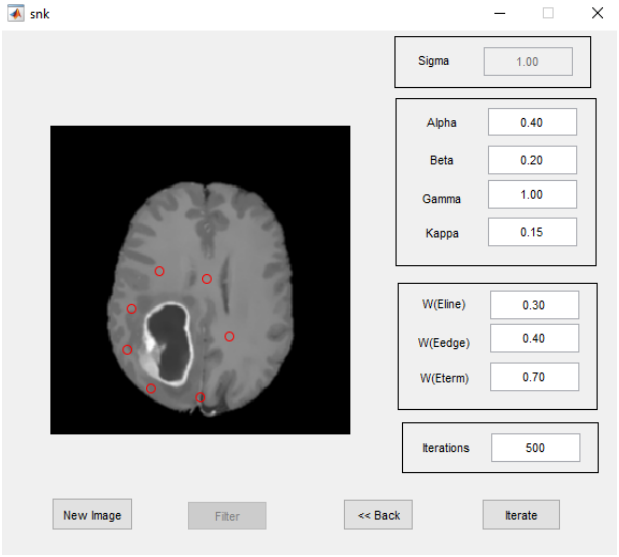


روش basic snake نیز به عنوان نمونه برای دو انتخاب متفاوت کانتور اولیه روی دو تصویر MRI1 و MRI5 مطابق قسمت قبل اعمال شده است. کانتور های اولیه به صورت زیر فرض شده اند.

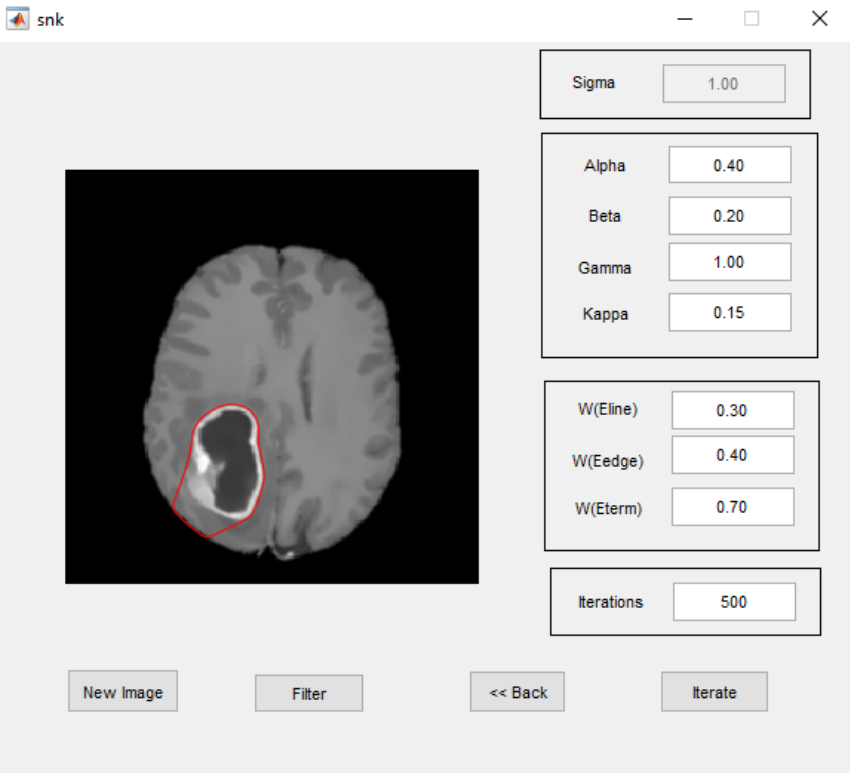
کانتور اولیه اعمال شده روی تصویر MRI1



کانتور اولیه اعمال شده روی تصویر MRI5



پس از انتخاب کانتور اولیه و پارامتر ها به گونه ای که در شکل مشخص شده است ، روش با ۵۰۰ تکرار اجرا شده است. شکل های زیر کانتور نهایی این روش را به ترتیب برای تصاویر MRI1 و MRI5 نشان میدهد :



نتیجه : همچنان در مورد مقایسه روش basic snake و GVF مواردی که در قسمت قبل گفته شد وجود دارد به علاوه این موضوع که روش GVF میتواند در یک مرحله تمام دسته بندی ها را انجام دهد اما basic snake برای هر ناحیه بندی کانتور اولیه مناسبی از طرف کاربر نیاز دارد. اما هر دوی این روش ها به نسبت روش FCM یک تفاوت بزرگ دارند که آن هم این است که نمتوانند پیکسل های متعلق به یک کلاس اما جدا از هم را تشخیص دهند در واقع این روش ها شرط همبند بودن نواحی را دارند. برای مثال همانطور که میدانیم روش GVF بر اساس وجود لبه کانتور ها را رسم میکند پس فقط تشخیص میدهد که در یک منطقه لبه وجود دارد یا نه و اگر وجود داشت به عنوان یک سگمنت جدید مشخص میشود. همچنین دقت این روش ها همچنان حداقل در مقایسه این تصاویر تمیز نسبت به روش FCM کمتر است.

سوال ۳)

۳.۱) سر آغاز این روش از نقطه ای بود که به نظر رسید روش FCM نمیتواند در برابر وجود نویز درست عمل کند . بنابراین یک ترم ناشی از اثر همسایه های هر پیکسل به تابع هدف اضافه شد که ابتدا این ترم ماند رابطه

۱ حاصل عبور تصویر از فیلتر میانگین گیر بود اما به دلیل هزینه ی محساباتی بالا این رابطه را به شکل رابطه ۲ تغییر دادند تا بجای فیلتر میانگین گیر از فیلتر میانه گیر استفاده شود و بار محاسباتی کم شود.

$$J_{\mathrm{FCM\_S}} = \sum_{i=1}^{N}\sum_{j=1}^{c} u_{ij}^{m}\left\|x_i - v_j\right\|^2 + \frac{\alpha}{N_R}\sum_{i=1}^{N}\sum_{j=1}^{c} u_{ij}^{m}\left(\sum_{r\in N_i}\left\|x_r - v_j\right\|^2\right),$$

رابطه ۱

$$J_{\mathrm{FCM\_S1,2}} = \sum_{i=1}^{N}\sum_{j=1}^{c} u_{ij}^{m}\left\|x_i - v_j\right\|^2 + \alpha\sum_{i=1}^{N}\sum_{j=1}^{c} u_{ij}^{m}\left\|\overline{x}_i - v_j\right\|^2.$$

رابطه ۲

در این روش ها ضریب تاثیر پیکسل های همسایه که همان آلفا است توسط کاربر تعیین میشود به همین دلیل نیاز به داشتن اطلاعات قبلی است بنابراین این روش بهبود داده شد و به شکل زیر در آمد تا پارامتر تاثیر به صورت خودکار و تطبیقی در هر مرحله خودش محاسبه شود ، در این حالت پارامتر G شامل تلفیق اثر اطلاعات مکانی و اطلاعات شدت روشنایی است.

$$J_{\mathrm{FLICM}} = \sum_{i=1}^{N}\sum_{j=1}^{c}\left[u_{ij}^{m}\left\|x_i - v_j\right\|^2 + G_{ij}\right], \qquad \acute{G}_{ij} = \sum_{k\in N_i, i\neq k}\omega_{ik}\left(1-u_{ij}\right)^m\left(1-K\left(x_i, v_j\right)\right),$$

اما همچنان اراداتی وجود داشت بنابراین تابع هدف در نهایت به شکل زیر تغییر پیدا کرد :

$$\mathrm{LVC}_i = \frac{\sum_{k\in N_i}\left(x_k - \overline{x}_i\right)^2}{N_R * \left(\overline{x}_i\right)^2},$$

$$\zeta_i = \exp\left(\sum_{k\in N_i, i\neq k}\mathrm{LVC}_k\right), \qquad \varphi_i = \begin{cases} 2+\omega_i, & \overline{x}_i < x_i \\ 2-\omega_i, & \overline{x}_i > x_i \\ 0, & \overline{x}_i = x_i. \end{cases}$$

$$\omega_i = \frac{\zeta_i}{\sum_{k\in N_i}\zeta_k}.$$

$$J_{\mathrm{ARKFCM}} = 2\left[\sum_{i=1}^{N}\sum_{j=1}^{c} u_{ij}^{m}\left(1 - K\left(x_i, v_j\right)\right)\right.$$

$$\left. + \sum_{i=1}^{N}\sum_{j=1}^{c} \varphi_i u_{ij}^{m}\left(1 - K\left(\bar{x}_i, v_j\right)\right)\right].$$

به صورت کلی این روش بیان میکند که اگر در یک پیکسل میانگین پیکسل های همسایه برابر با پیکسل مرکزی بود فورمولاسیون و تابع هدف مانند FCM باشد و جمله ای اضافه نشود ($\varphi = 0$) . اما اگر اینگونه نبود پارامتر φ تعیین میکند چه مقدار (بر اساس اختلاف و روابط بالا) تاثیر پیکسل های اطراف را اضافه کند. نکته بسیار خوب این روش این است که φ میتواند قبل از ناحیه بندی محاسبه شود به همین دلیل بار محاسباتی بالایی ندارد. همچنین از روش بهتری برای تاثیر دادن پیکسل های اطراف در φ استفاده شده است و به علاوه در همه ی حالت (مخصوصا در حالت نویزی نبودن) لزوما پیکسل های اطراف دخالت داده نمیشوند بلکه این کار هوشمندانه صورت میگیرد.

۳.۲) توابع استفاده شده را به اختصار توضیح میدهیم :

Demo : این تابع تابع اصلی است که کل فرایند را برای تصاویر مختلف اجرا میکند.

Noise : این تابع به تصاویر MRI انوع و درصد های متفاوت نویز را اضافه میکند.

ARKFCM : این تابع روش ARKFCM را روی تصویر اعمال میکند ، ورودی تابع تصویر ، ω_i ها ، تعداد خوشه ها ، نوع فیلتر اعمالی ، سایز یک پنجره (که همان پنجره شامل پیکسل های همسایه هر پیکسل است) ، حد نهایی و ضریب فازی m است و خروجی آن مرکز خوشه ها ، خوشه ی مربوط به هر پکسل ، مقدار تابع هدف و تعداد مراحل تکرار روش است.

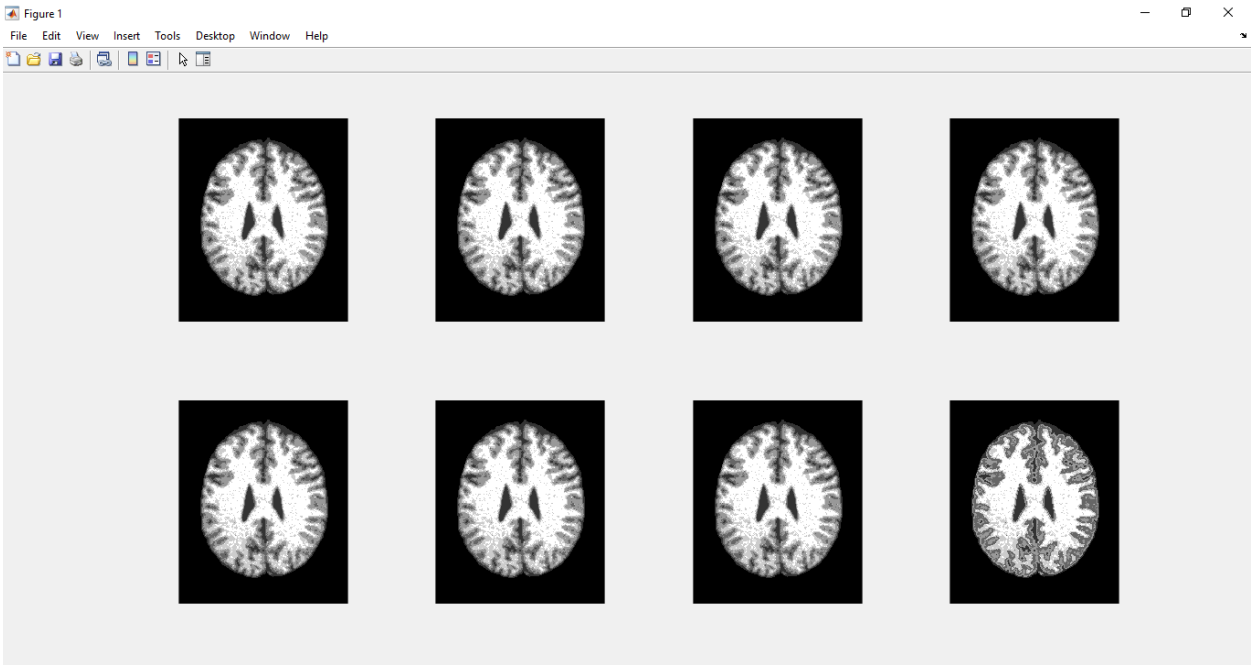
distARKFCM : این تابع در هر مرحله فاصله ی بین مراکز خوشه ها و اعضای خوشه ها را از طریق صدا زدن تابع gaussKernel محاسبه میکند.

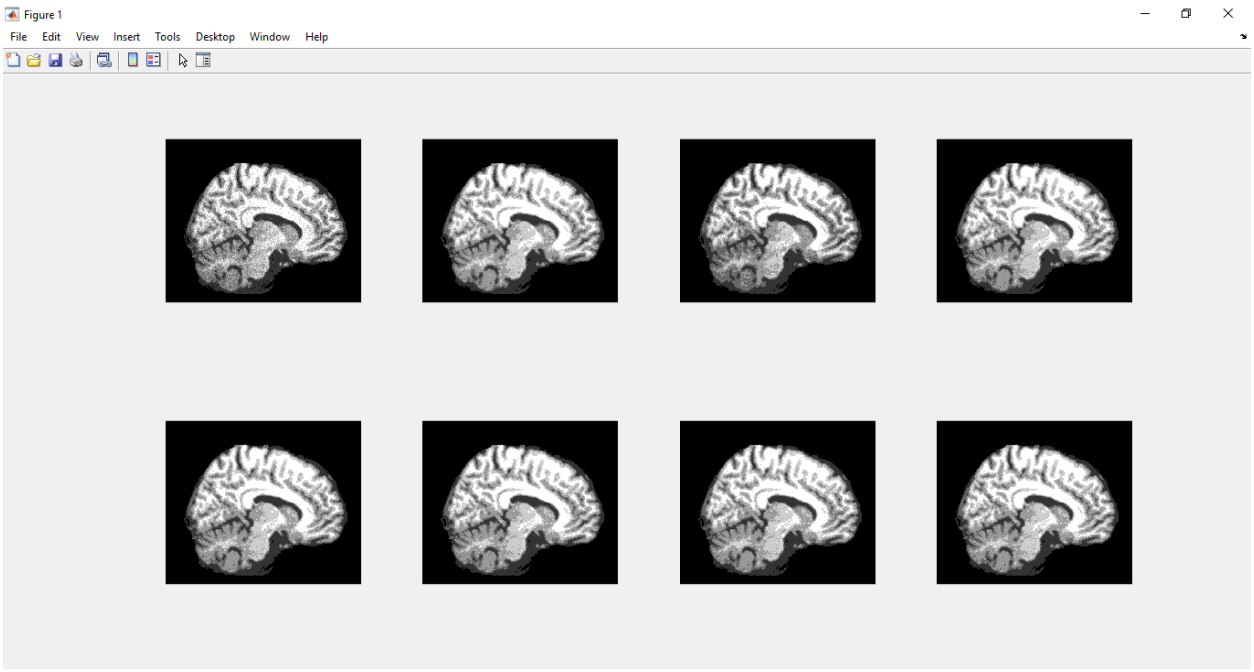
gaussKernel : این تابع یک کرنل گاوسی را روی ناحیه ورودی پیاده سازی میکند(درواقع جمع مجذور فاصله ها را پیدا میکند).

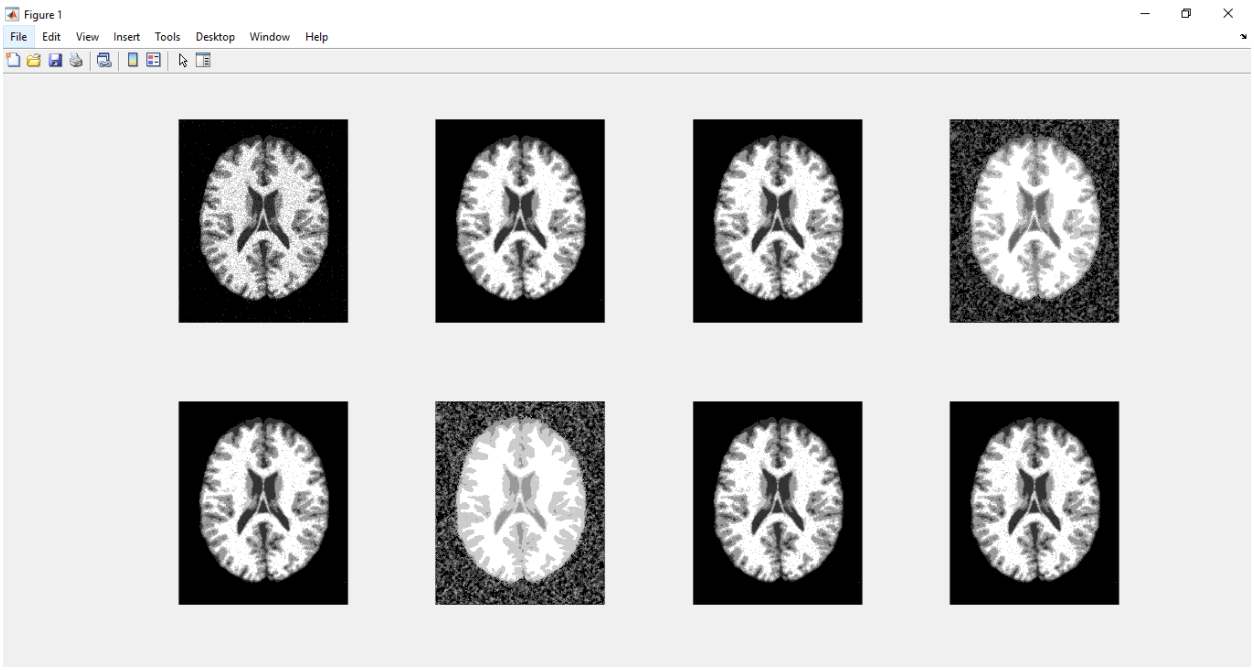
PixWgt : این تابع ω_i ها را محاسبه میکند . ورودی تابع تصویر و سایز پنجره همسایگی (بالاتر توضیح داده شده) استو خروجی تابع ω_i ها هستند.

۳.۳) نتیجه اعمال روش ARCFCM روی تصاویر اول دوم و سوم به ترتیب در سه شکل زیر قابل مشاهده است.

در هر کدام از این تصاویر تصویر ورودی با ۸ سایز متفاوت برای اثر همسایگی به الگورتیم داده شده و خروجی نمایش داده میشود ، از مقایسه عکس اول (برش axial با ۱۰ درصد نویز) و عکس اخر (برش sagital با ۷ درصد نویز) مشخص میشود که با کمتر بودن نویز به مقدار کمی ناحیه بندی بهتر انجام شده است.

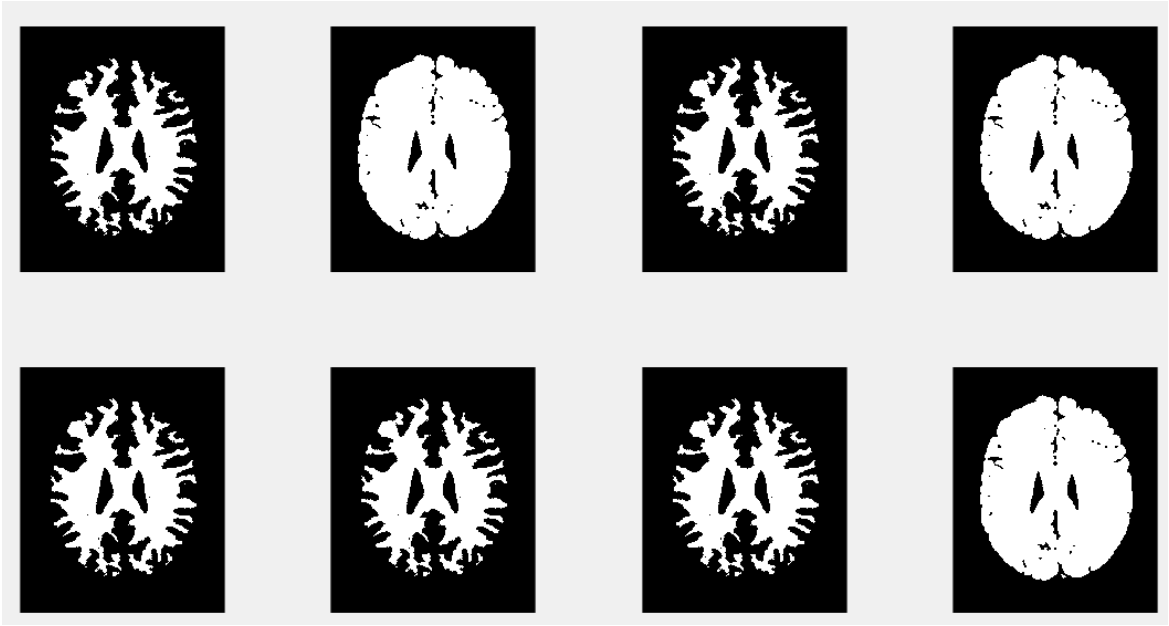




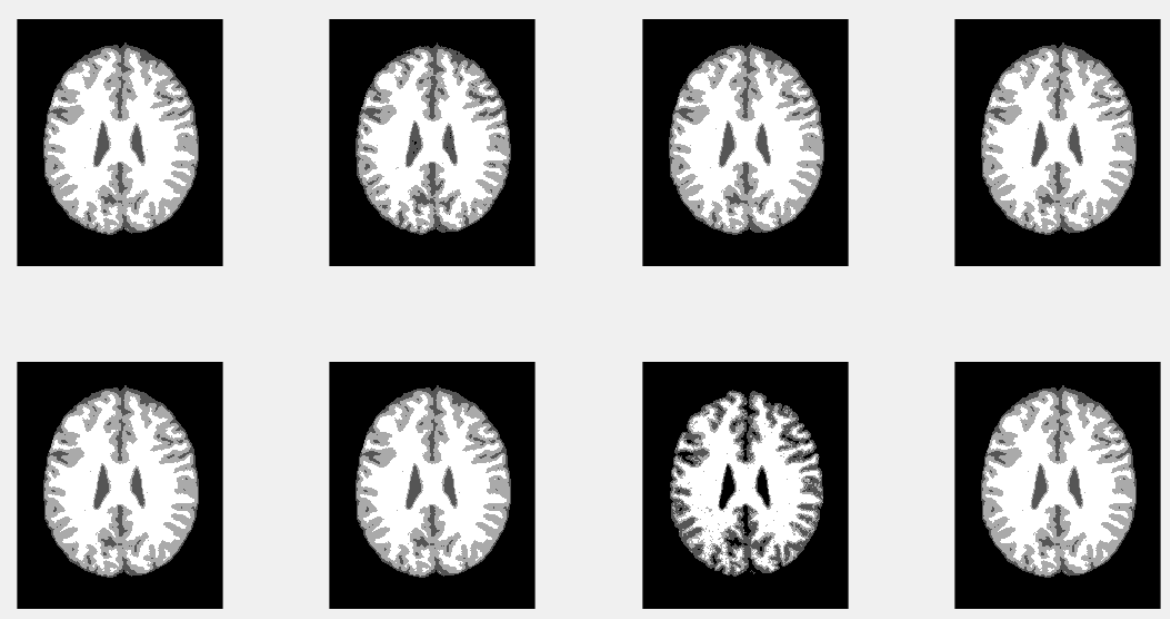


۳.۴) فایل برای تصویر اول و به ازای تعداد کلاستر های ۲،۴،۶،۸،۱۰ اجرا شده و نتایج به شرح زیر است:

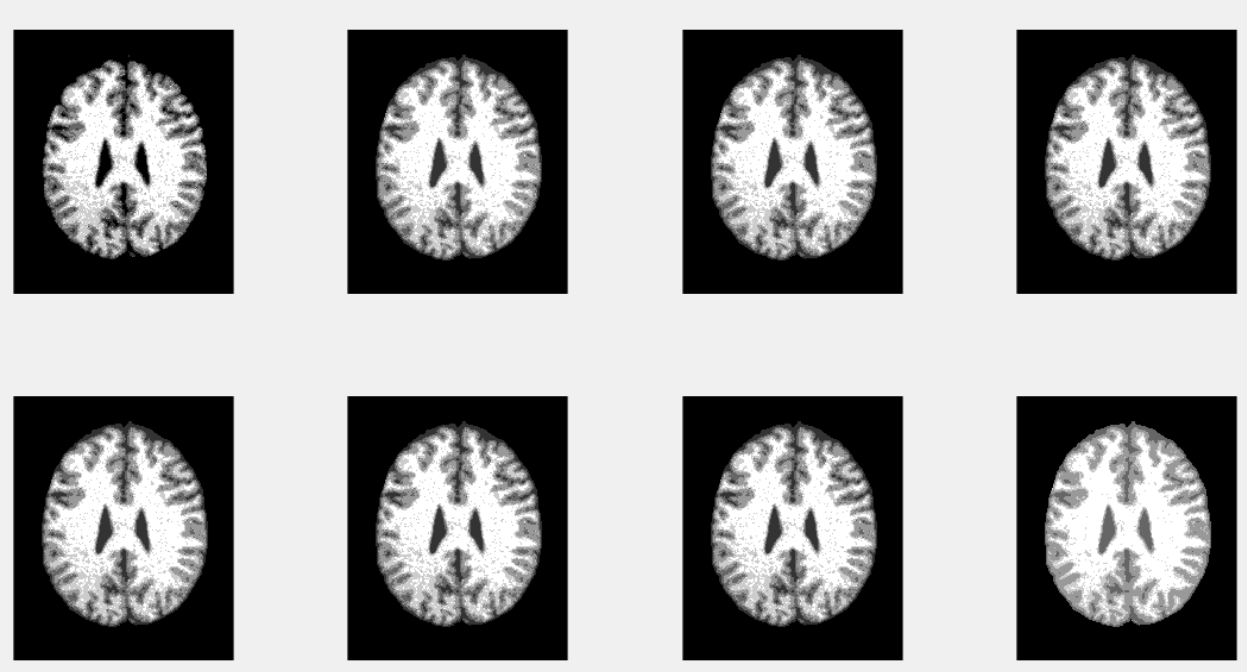
Nc = 2



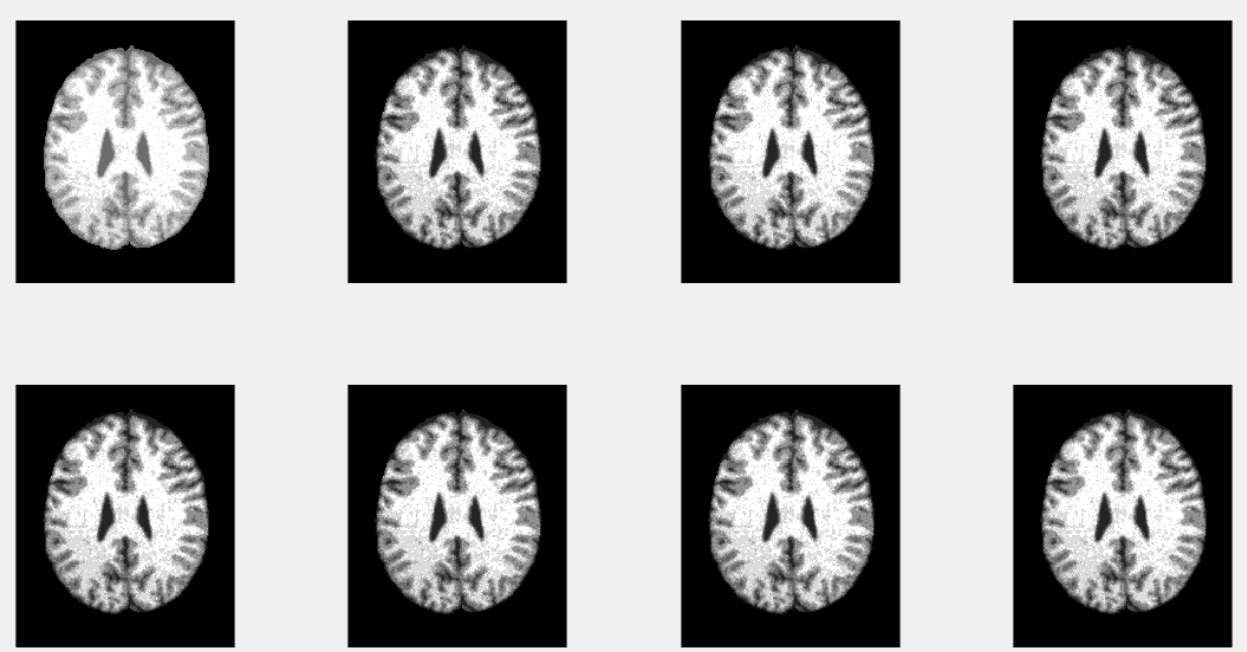
Nc = 4



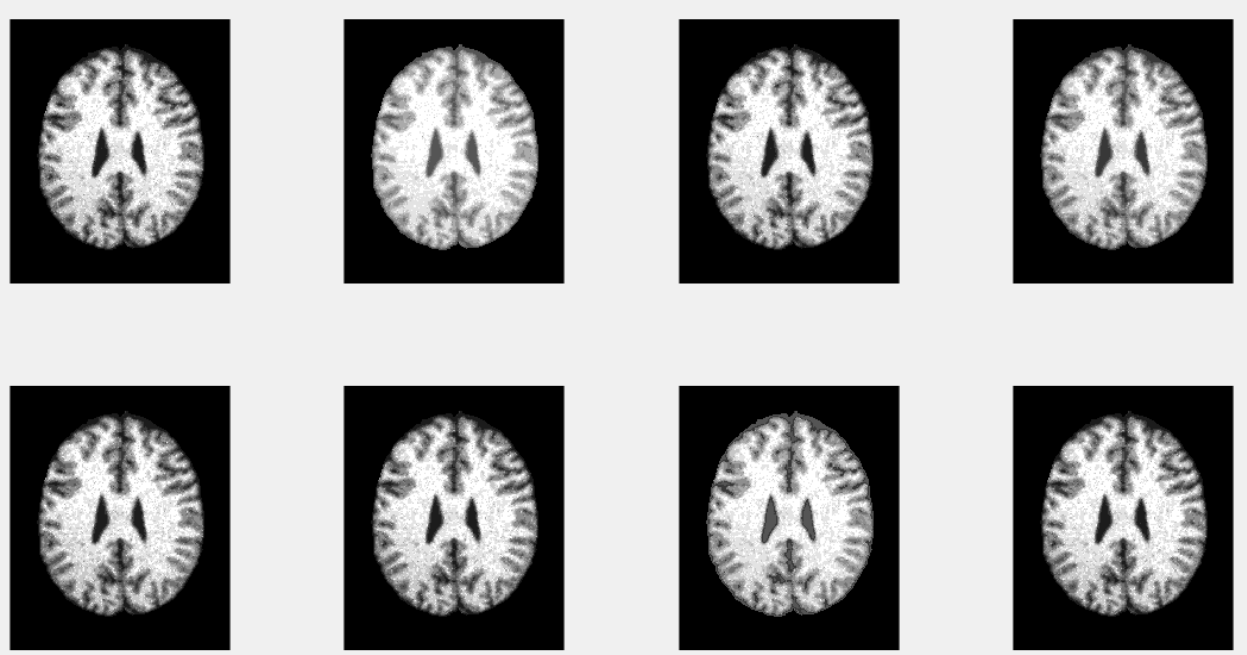
Nc = 6



Nc = 8



Nc = 10



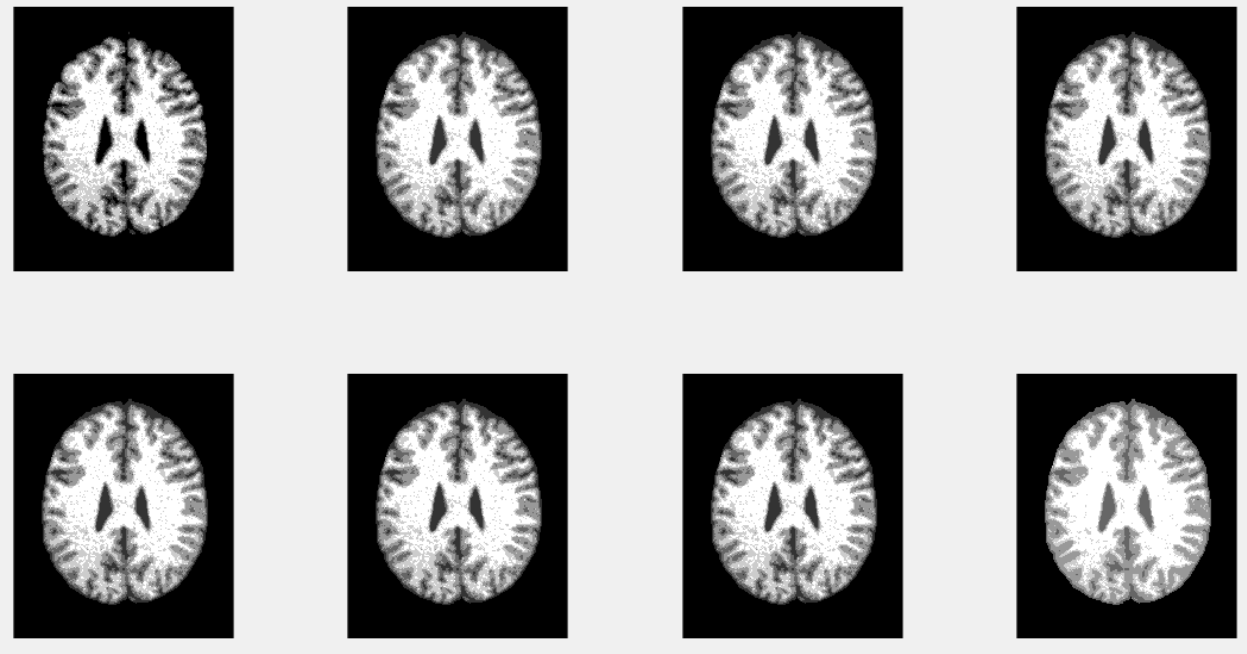
با توجه به اشکال بالا متوجه میشویم کم بودن تعداد خوشه ها تقریبا باعث از بین رفتن نویز میشود اما میدانیم که کم بودن خوشه ها باعث میشود برخی اطلاعات مهم تصویر نیز شناسایی نشوند، از طرفی زیاد شدن تعداد

خوشه ها هم باعث میوشد پیکسل های نویزی به صورت دسته ها جداگانه دسته بندی شوند و در کل نویز زیادی در تصاویر باقی بماند. نتیجتا با یک بده بستون بین نویز نهایی و خوشه بندی درست که از یک جهت نیز میتوان هر دو را به یکسان دید مقابلیم . در تصاویر زیر بعد از برسی ۵ مورد برای تعدا خوشه ها به نظر میرسد حالت بهینه در تعداد ۴ خوشه رخ داده است .

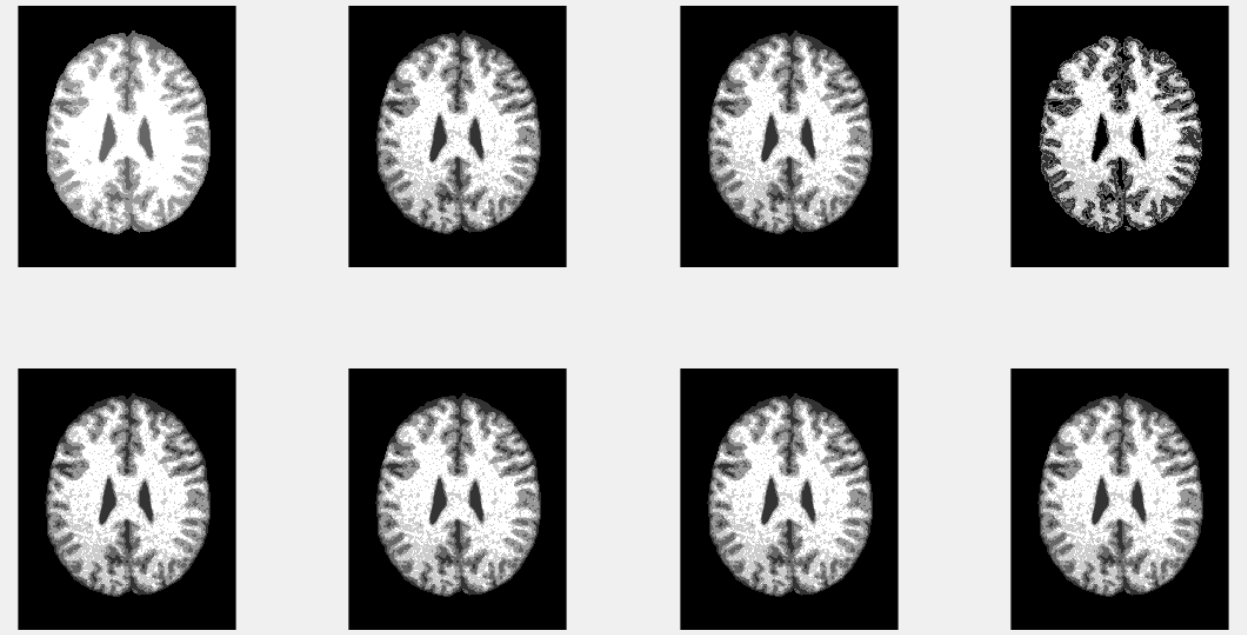
۳.۵) فایل برای تصویر اول و با سه حالت فیلتر اجرا شده و نتایج به شرح زیر است:

از مقایسه تصاویر ابتدا متوجه میشویم که هر نوع فیلتر بسته به سایز پنجره همسایگی که در نظر گرفته میشود عملکرد متفاوتی را دارد . بنابراین نمیتوان به قطع و در همه حالات مقایسه را انجام داد اما به طور کلی به نظر میرسد فیلتر میانگین گیر حداقل در این نوع تصویر آغشته به نویز در بیشتر حالات بهتر توانسته است نویز را حذف نادیده بگیرد در عین حال که اجزای تصویر را به خوبی حفظ کند.

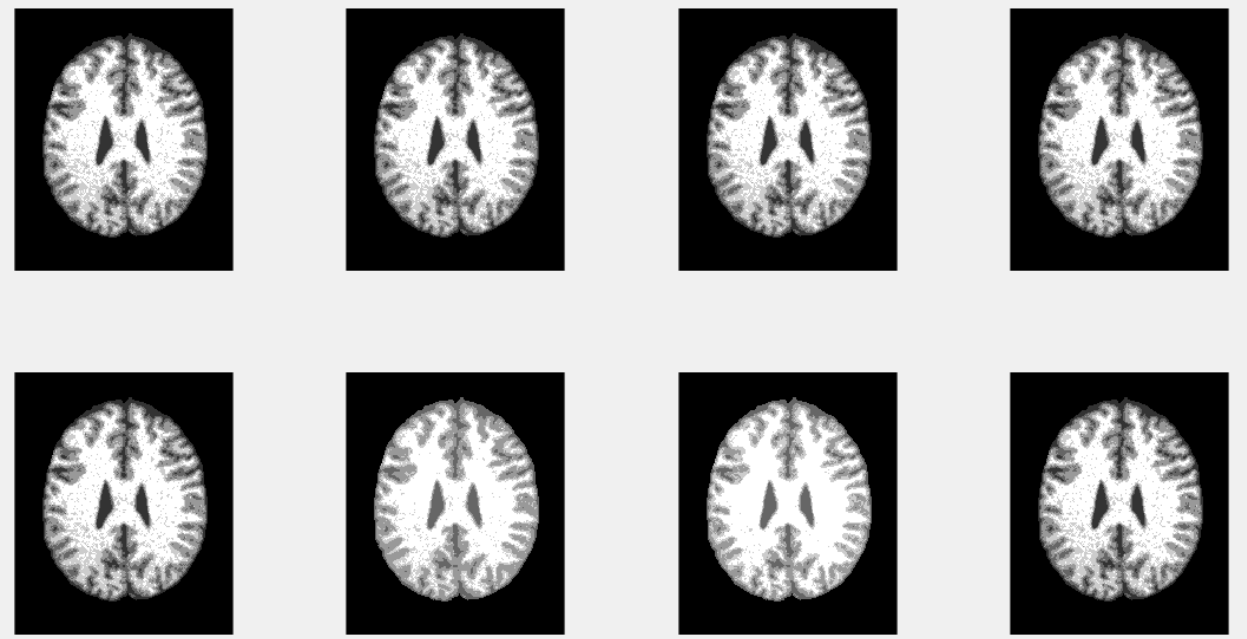
فیلتر average :



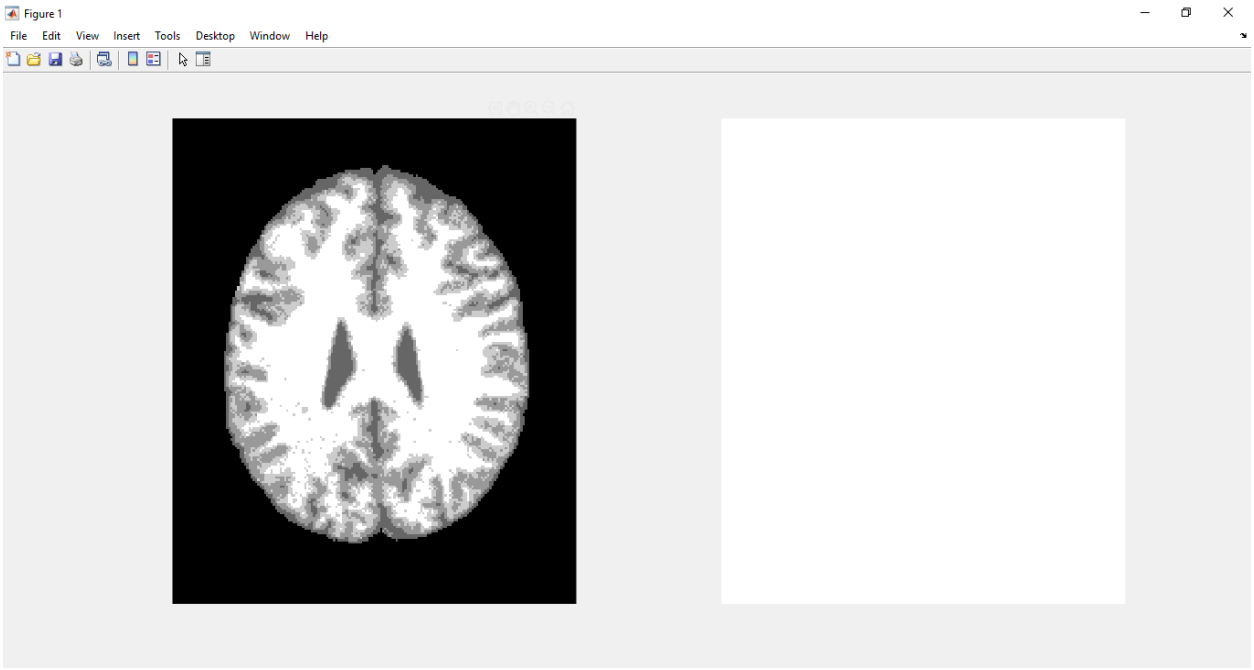
فیلتر median :



فیلتر weighted :



۳.۶) در تصویر زیر سمت چپ نتیجه اعمال پنجره با سایز ۱ و سمت راست نتیجه اعمال پنجره با سایز ۲۹ را نشان می دهد (هر دو روی تصویر اول اعمال شده) :



با برسی افزایش سایز پنجره متوجه میشویم ماکزیمم سایزی که به تصویر اسیب نمیزند سایز ۱۵ است ، همانطور که میبینیم در سایز ۲۹ تصویر کاملا از بین رفته این موضوع قابل توجیه است به این علت که هنگامی که سایز پنجره همسایگی زیاد باشد فیلتر روی سایز بزرگی عمل میکنند و برای مثال فیلتر میانگین گیر تمام توصر را میانگین میگیرد و همین اسیب زیاد اولیه به توصیر باعث میشود نتیجه نهایی هم عملا اطلاعات زیادی از تصویر نداشته باشد. از طرفی اگر سایز پنجره خیلی کوچک باشد هم بسته به مقدار نویز مثلا در فیلتر میانگین گیر باعث میشود پیکسل نویزی به خوبی اصلاح نشود و نویز در تمام مراحل خوشه بندی باقی بماند.

" در این بخش از تمرین فقط از فایل اماده ی demo.m استفاده شده و بنابراین در فایل کد اصلی بخشی برای این قسمت وجود ندارد"