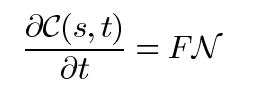
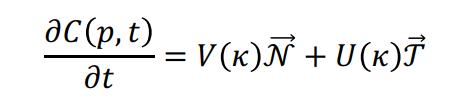
گزارش سری چهارم

# سوالات نوشتاری:

## سوال اول:

مبنای روش های مبتنی بر zero level set به تعریف یک تابع در یک بعد بالاتر و در درنهایت قطع دادن آن با سطح صفر است. معمولا در این روش در درون خم را منفی و خارج آن را مثبت فرض میکنیم. مهمترین مزیتی که دارد این است که نحوه گسسته سازی آن در رزولوشن خود تصویر تعریف میشود و معمولا به پاسخ هایی کمتر نویزی همگرا می‌شود. در خم های شکل پذیر هنگامی که گسسته سازی انجام می‌شود با بزرگ شدن خم، فاصله بین نقاط زیاد می‌شود و اگر کوچک شود ممکن است برخی نقاط روی هم بیفتند! بنابراین بهتر است که گسسته سازی در رزولوشن خود تصویر انجام گیرد که به این روش منجر می‌شود.

معادله‌ی تکامل خم به صورت زیر در مقاله تعریف شده است. البته در تعریف کامل تر باید یک ضریبی از بردار مماسی را نیز قرار می‌دادند که البته تاثیری نداشت. زیرا در آن راستا صرفا شیوه پارامتری شدن تغییر میکند و از روی خم جدا نمی‌شویم

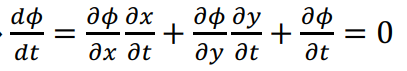
یا **

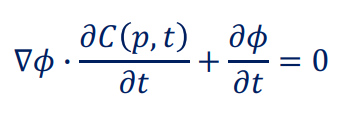
بردار مماسی که رو به داخل تعریف میکنیم به صورت زیر برحسب میتوان نوشت:

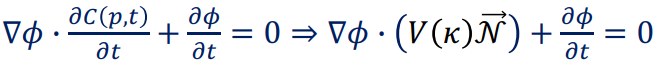
**

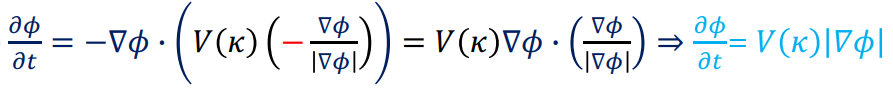
*بنابراین با جایگذاری آن در معادله بالا می‌توان به صورت یک معادله PDE نوشت:*

**

**

**

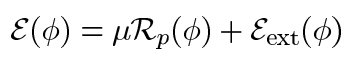
**



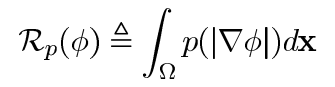
*که به یک معادله ی PDE منجر شد*

## سوال دوم:

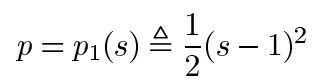
*در این روش انرژی فانکشن را به صورت زیر تعریف کردند:*

**

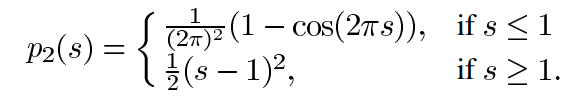
*که در آن رابطه،  انرژی خارجی و*  *که جمله رگیولاریزیشن است. و*  *یک ضریب ثابت است. و داریم:*



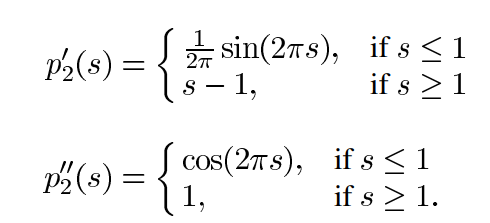
*که در آن p تابع پتانسل انرژی است. یک انتخاب برای p تابع سهمی است اما مینیمم آن در صفر رخ می‌دهد. ما دوست داریم در مجاورت خم اندازه گرادیان نزدیک یک بماند تا دقت بیشتری در تکامل خم حاصل شود. اگر صفر باشد به مرور خم نابود میشود.*



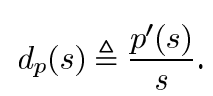
*همچنین این مقاله تابع دیگری را نیز برای p به شکل زیر نمایش داده است:*

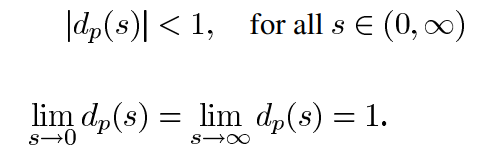
**

*اولا این تابع دوبار مشتق پذیر است:*

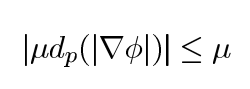
**

*ثانیا دوتا مینیمم در صفر و یک دارد.*

*سوما که با تعریف*  *می‌توان نشان داد برای این تابع به صورت زیر است.*



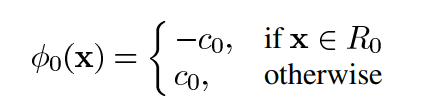
*بنابراین داریم:*



*که شرایط مرزی نرخ دیفیوژن را ارضا می‌کند.*

## سوال سوم:

*در این مقاله به ادعای نویسندگانش، به reinitialization نیازی ندارد و با رابطه زیر میتوان LSF را شروع کرد.*

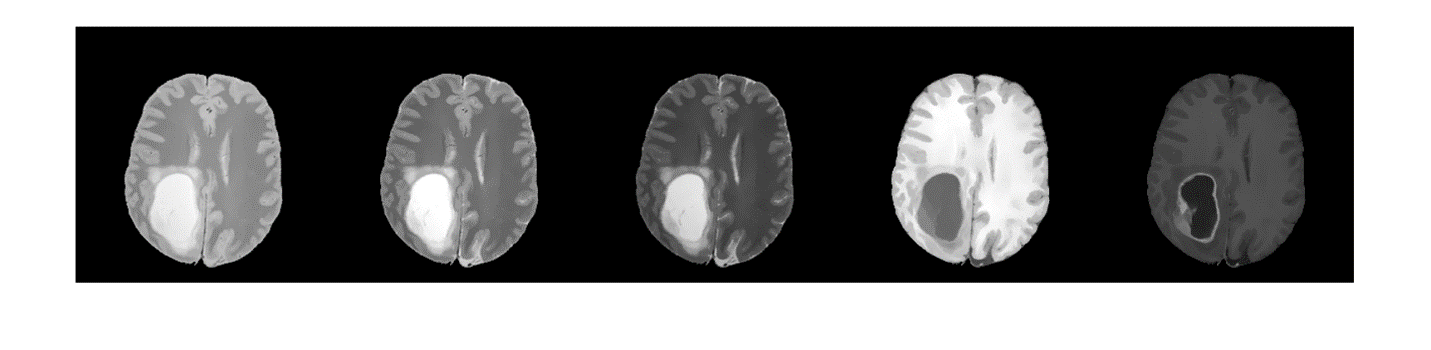
**

*البته ناحیه R0 می‌تواند خیلی ساده مثلا بایک ترشولد زدن تعیین شود و یا هر روش سگمنتیشن دیگری. اگر R0 تخمین خوبی باشد، با یک تعداد کم ایتریشن به جواب سریع همگرا می‌شود. همچنین بخاطر عرض SDB که تقریبا 2c0 است و چون تصویر گسسته شده است بنابراین پیشنهاد میشود که برای این روش c0>=1 انتخاب شود و پیشنهاد خود نویسندگان این مقاله برای شبیه سازی ها c0=2 است.*

# سوالات شبیه سازی:

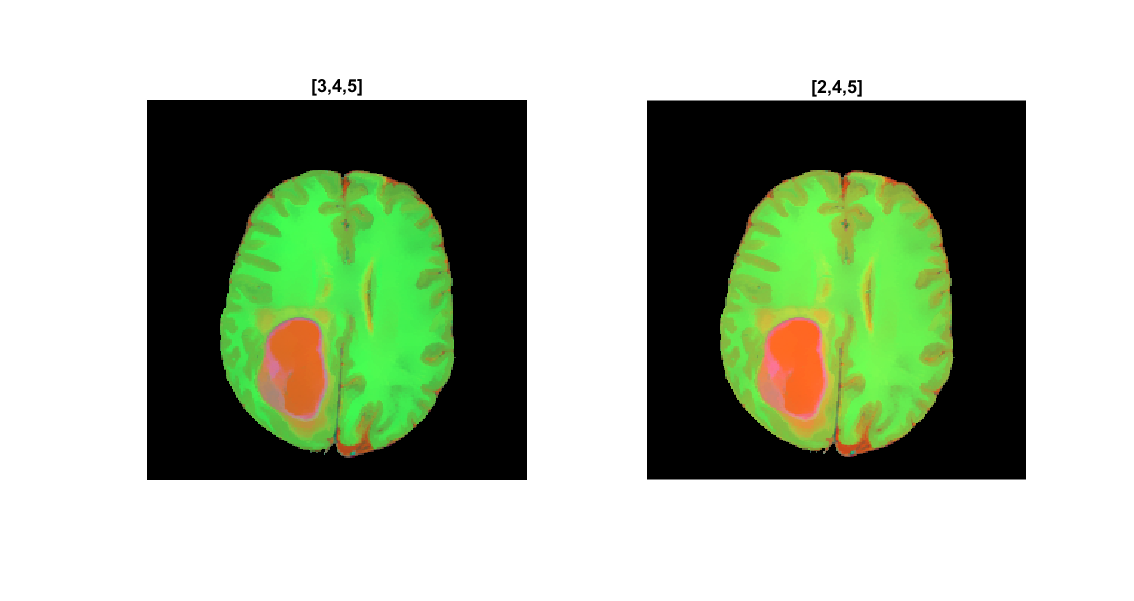
## تمرین اول:

در این سوال، تصاویر زیر مورد پردازش قرار گرفته است:



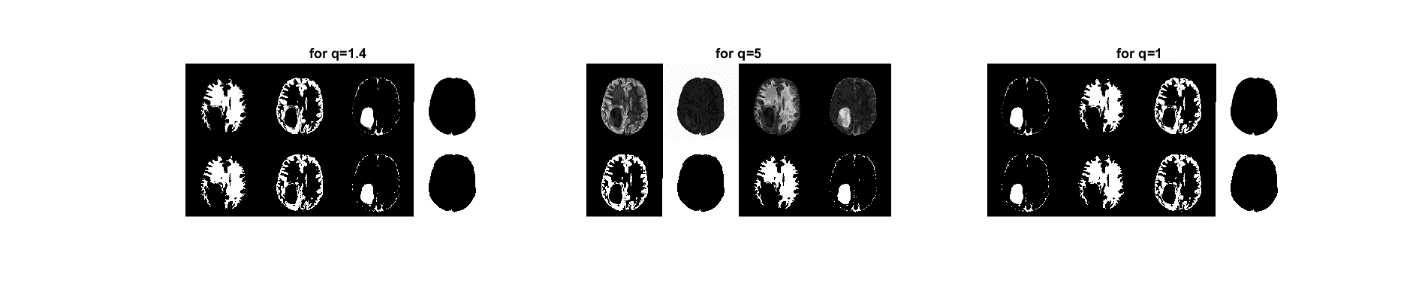
بنظر میرسد برای این تصاویر 4 سگمنت، تعداد معقولی بنظر برسد.

همچنین 3 تا از این تصاویر باهمدیگر و به صورت RGB نمایش داده شده است:

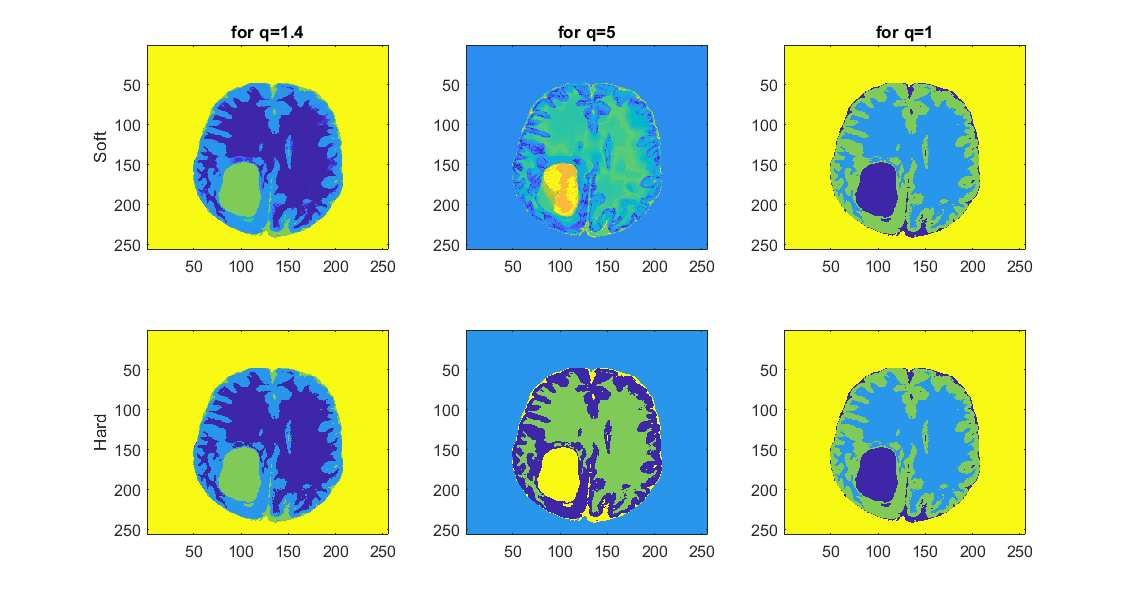


تصویر شماره 4 بخاطر ظاهر مناسب‌ترش برای پردازش های آتی مورد استفاده قرار گرفته است.

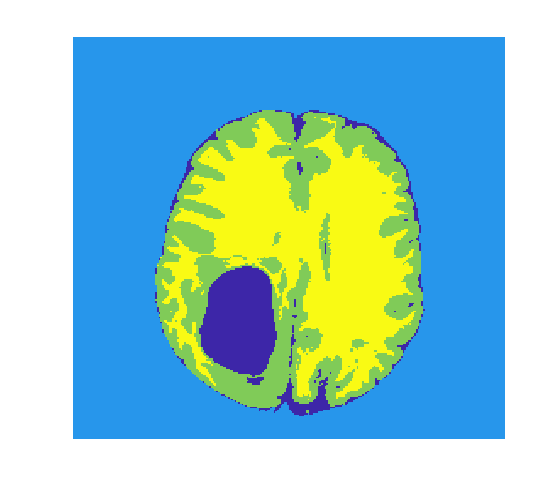
خروجی الگوریتم FCM با q های گفته شده به صورت زیر است:

البته چون استفاده از q=1 به خروجی NaN منجر میشد برای q = 1 از 1.04 استفاده شد و برای کمتر از آن ناپایدار میشد. تصاویر بالا همان خروجی های fcm هستند و تصاویر پایین همان خروجی ها همراه با یک hard thresholding است که صرفا به ماکسیمم یک داده است و باقی را صفر کرده است. همانطور که مشاهده میشود جای سگمنت ها بخاطر رندم بودن مدام تغییر می‌کند.

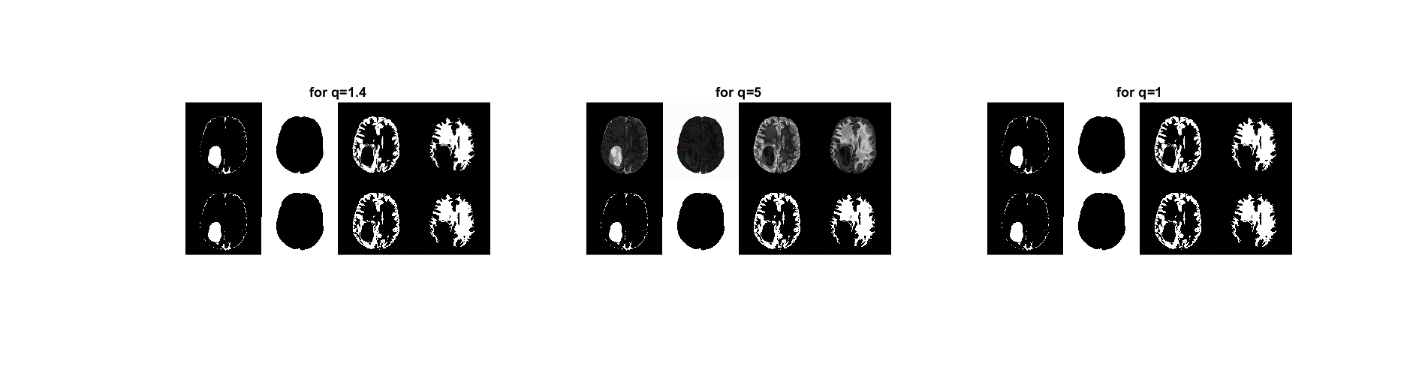
همچنین با یک تکنیک که در کدم قابل مشاهده است سعی شده است تا تمامی آن سگمنت ها را به یک تصویر تبدیل شود که خروجی تصاویر بالا را روی هم دیگر در زی مشاهده میکنیم:



با روش kmeans به شکل زیر میرسیم:



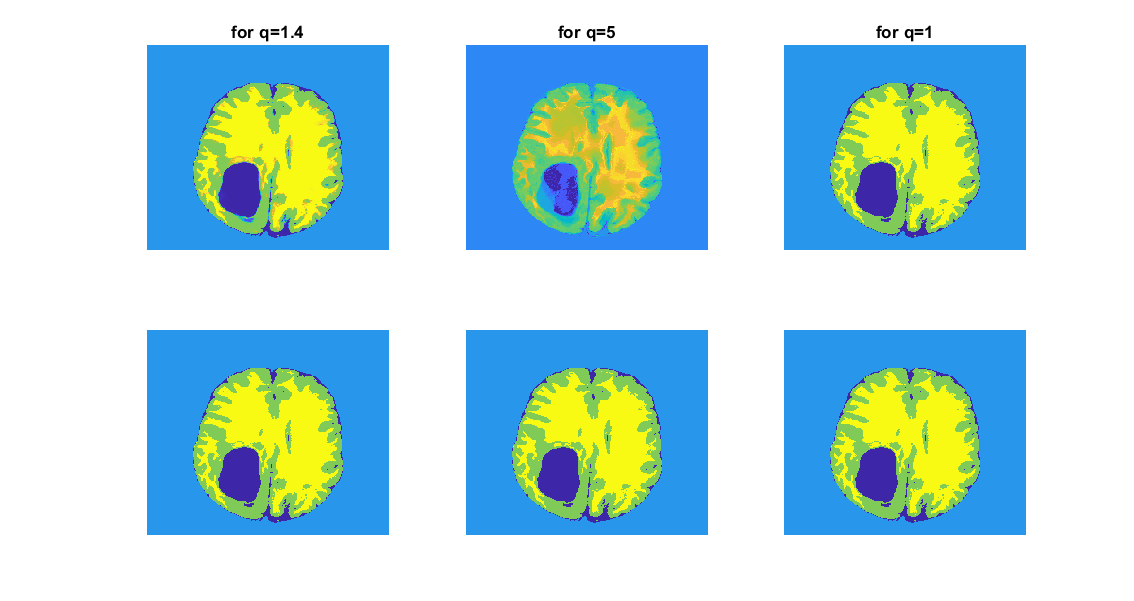
همچنین با تغییر fcm متلب یک فایل fcm2 ساخته شده است که شرایط اولیه را نیز می‌توان به آن داد و به عنوان شرایط اولیه همان خروجی kmeans داده شد. که به شرح زیر است.



همانطور که مشخص است با افزایش q از یک، مرزبندی ها نرم تر میشد و نقشه احتمالاتی از حالتی باینری فاصله میگیرد. البته وقتی از argmax آن ها استفاده میکنیم و خروجی آن هارد ترشولدینگ میکنیم، تصاویر همه آن ها شبیه به یکدیگر است.

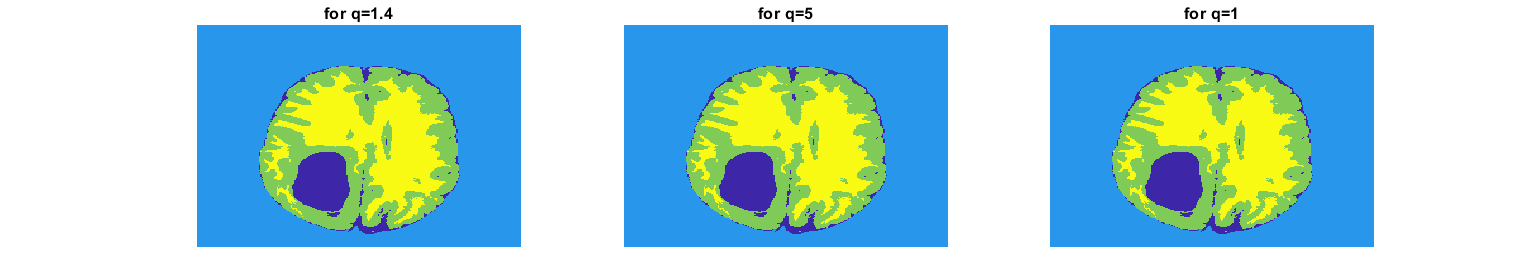
خوبی این روش این است که اگر الگوریتم kmeans به جواب خوبی (نه لوکال مینیمم) همگرا شود، به احتمال زیاد نیز fcm به جواب خوبی و سریع تر همگرا میشود. همچنین ترتیب سگمنت ها مانند ترتیب آن در kmeans است.

برای این قسمت نیز مانند قسمت قبل سعی شد تا تمامی سگمنت ها را باهم بتوانیم مشاهده کنیم:



البته یک نکته جالب این است خروجی hard thresholding برای تمامی روش ها تا حد خیلی زیادی به هم دیگر شبیه است و قابل تمایز نیستند.

با روش gmm که پیاده سازی آن برای من سخت تر از بقیه روش ها بود، با دستور cluster به شکل های زیر منجر شد:



برای ایجاد این تصاویر از خروجی fcm به عنوان شرایط اولیه استفاده شد. (هنگامی که نرمالیزیسیون یا شرایط اولیه ی خوب اعمال نمی‌شد الگوریتم gmm واگرا میشد!)

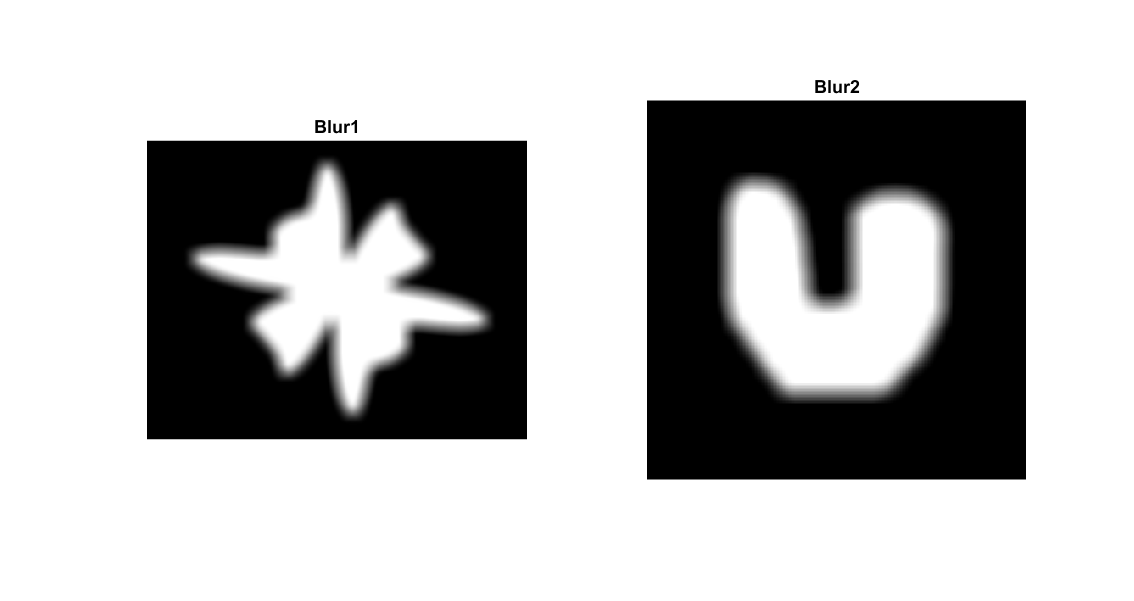
قسمت سخت آن استخراج نقشه های احتمالاتی بود که آن نیز به صورت زیر انجام شد.



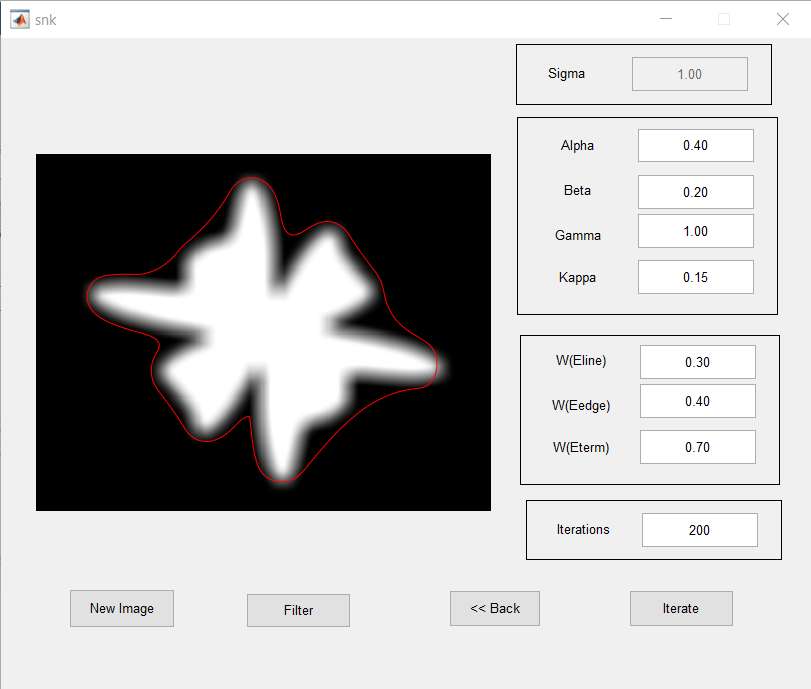
راستش قسمت پایانی را متوجه نشدم دقیقا چی میخواست اما حس میکنم فراتر از خواسته های آن در قسمت های قبل انجام پذیرفته است.

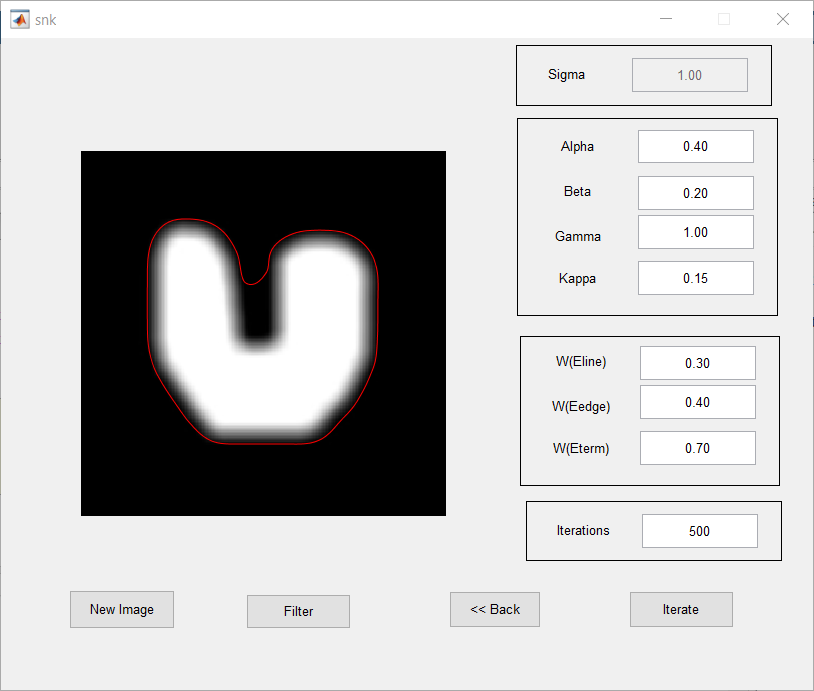
## تمرین دوم:

قسمت اول: دو تصویر Blur1 و Blur2 را در زیر مشاهده می‌کنید.



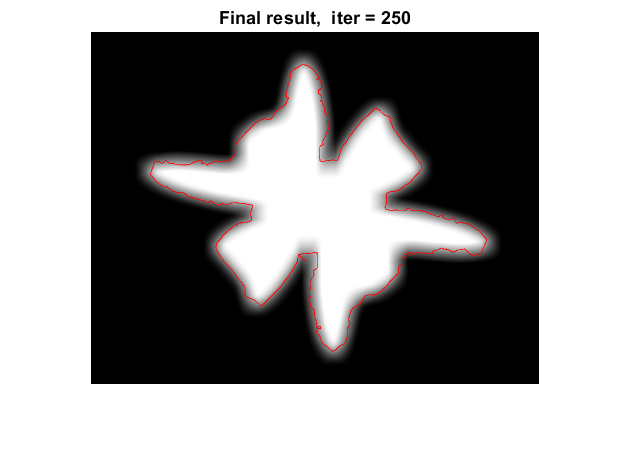
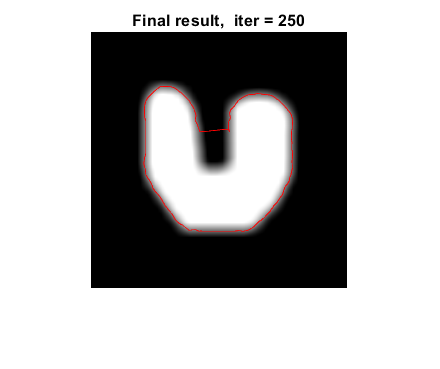
ابتدا با استفاده از تابع گرافیکی snk که در سوال داده شد تصاویر را یکی یکی لود و پردازش می‌کنیم. در هر مرحله به صورت گرافیکی نقاطی را حول شکل (به طور اغراق آمیز) انتخاب شد و در نهایت خمی به شکل های زیر روی آن فیت شد.



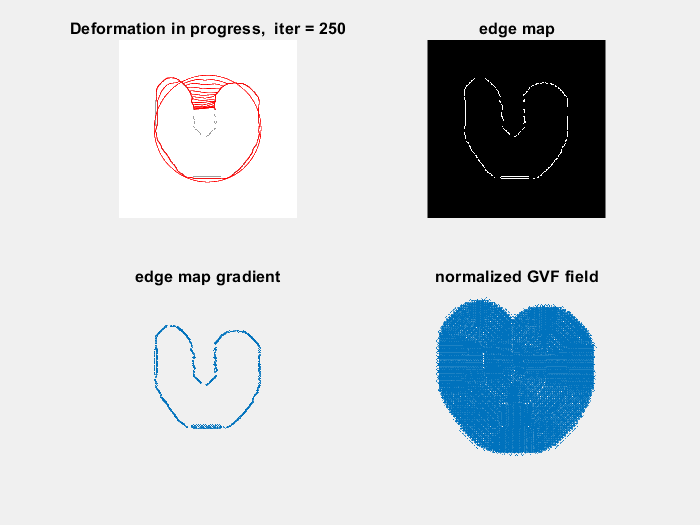
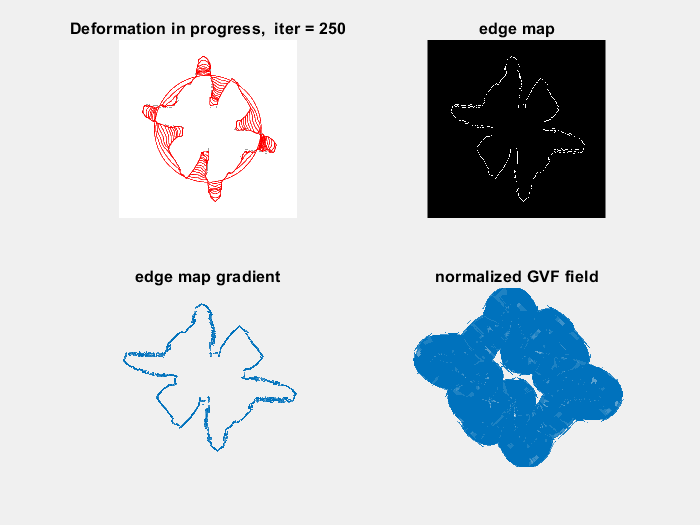


که در این حالت PVE زیاد خوب هندل نشده است و همچنین در نواحی دره ها نتوانسته است به خوبی نفوذ کند.

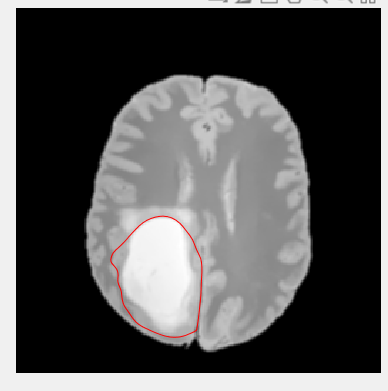
پاسخ نهایی روش GVF که با استفاده از تابع خودم به نام use\_gvf و با کمک پکیج های سوال نوشته شده است به شرح زیر است:

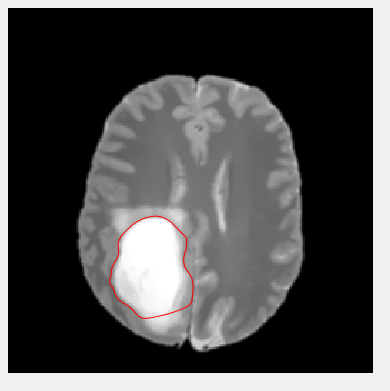


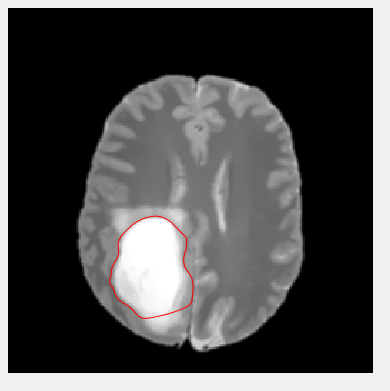
البته مراحل همگرایی آن به شکل زیر نیز نشان داده می‌شود.



بخاطر سادگی تست snk این روش صرفا روی تصاویر mri و برای سگمنت کردن قسمت توموری به شکل زیر رفتار میکند.



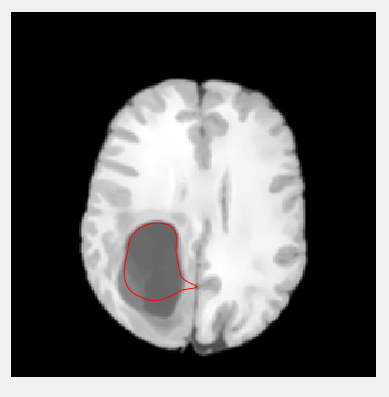






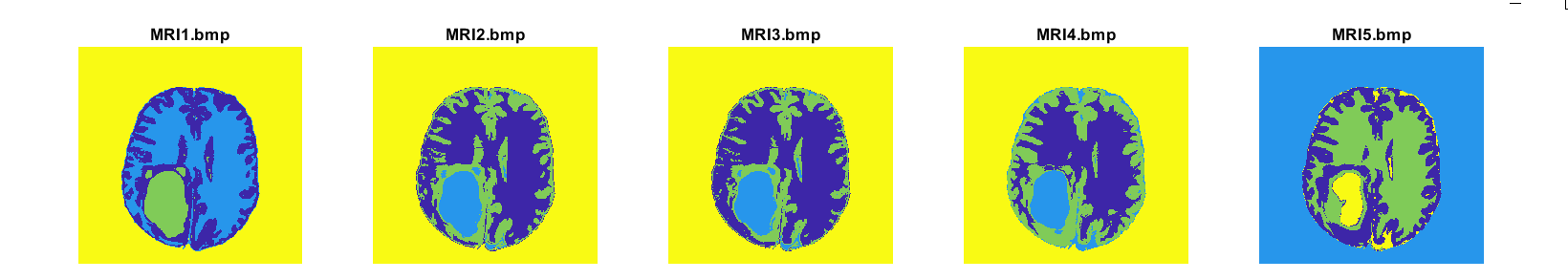


البته نتایج بالا به شرایط اولیه خیلی وابسته است مثلا اگر شرایط اولیه را کمی راحت تر برای تصویر 4 ام میگرفتیم ممکن بود به نتیجه زیر برسد.



و ضمنا خروجی آن حتما یک خم بسته است و ترجیح دارد که تا حد ممکن نیز هموار باشد. بنابراین نمی‌تواند اعضای جدا از هم اما متعلق به یک کلاس را استخراج نماید در حالی که در روش های fcm و ... این مشکل وجود نداشت. البته این مشکل به نوعی یک ویژگی مثبت این روش ها نیز به شمار می‌رود. همچنین این روش ها در تصاویری که لبه ی تندی نداشتند مانند تصویر دوم و سوم، زیاد جالب عمل نکرد.

برای q=2 در fcm نتیجه سگمنتیشن برای fcm مشخص شده است.

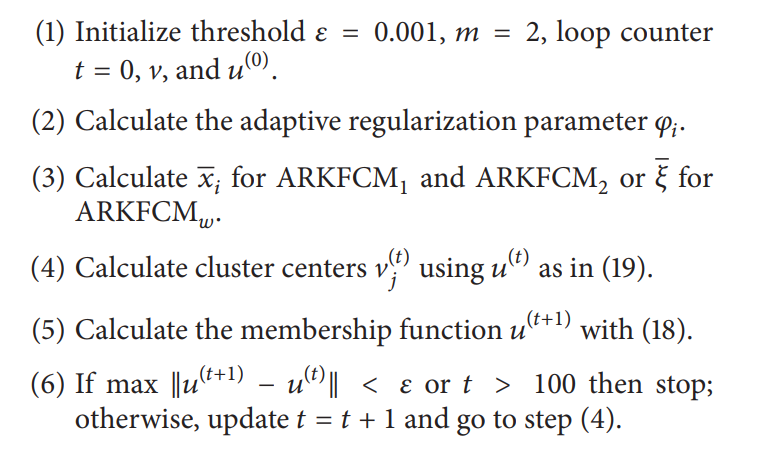


مشاهده میشود برای تصاویری با گرادیان کم این روش از روش خم ما بهتر بوده است!

## تمرین سوم:

قسمت اول: در این مقاله پس از معرفی روش کلاسیک fcm روش دیگری را به عنوان FLICM پایه بررسی خود قرار داده است. در این مقاله آن روش که نسبت به نویز مقاوم است از لحاظ میزان محاسبات نقد نیز شده است که یک پارامتر 𝐺𝑖𝑗 باید در هر ایتریشن آن حساب شود و روش خود را که نسبت به نویز نیز مقاوم است را مطرح میکند که محاسبات کمتری دارد.

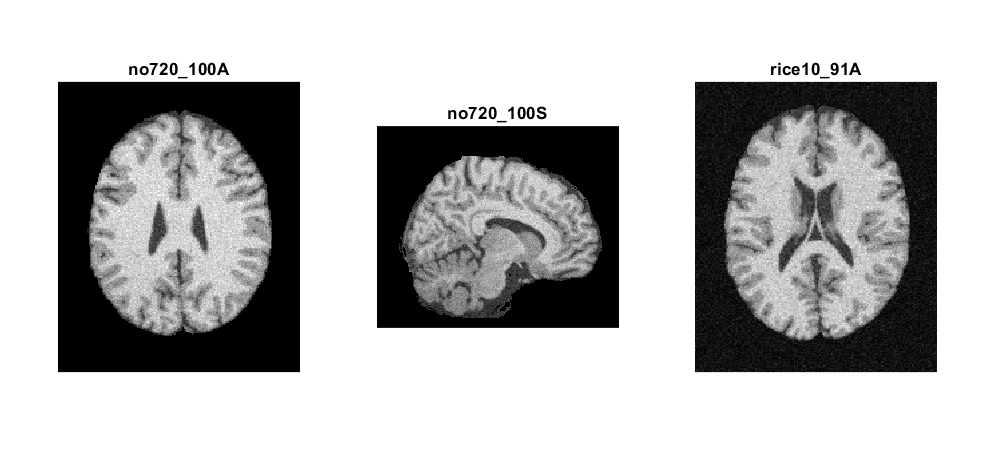
شالوده این الگوریتم در زیر آمده است و روابط 9 تا 19 روابط اصلی این مقاله به شمار می‌آیند.



قسمت دوم: راستش در پاسخ این قسمت در صورت سوال که گفته بود هر یک از توابع را به اختصار توضیح دهید، ابهامی وجود دارد که منظور از توضیح دادن، نحوه کار کردنشان است و یا وظیفه کلی شان. یعنی همان توضیحی که در ابتدای هر فایل داده است و همچنین مجددا در فایل Readme نیز توضیح داده شده است:

فایل demo یک اسکریپت برای تست و نحوه استفاده از باقی فانکشن ها را نشان میدهد. ابتدا با استفاده از PixWgt، برای هر پیکسل وزن مخصوصش را محاسبه میکنیم و سپس آن وزن ها را به ARKFCM که تابع اصلی و الگوریتم اصلی این مقاله است داده میشود و با آن سگمنت ها استخراج میشود. چند تابع کمکی دیگر نیز در این قسمت وجود دارد. تابع distARKFCM که فاصله بین مراکز توابع را مشخص میکند که سیگمای آن با استفاده از kerWidth محاسبه میشود. تابع gaussKernel نیز که مشخص است فقط نکته مهم آن این است که تابع گوسی با مقدار ماکس خود نرمالیز شده است.(نه مساحتش)

قسمت سوم: برای تصاویر زیر الگوریتم ران شده است:

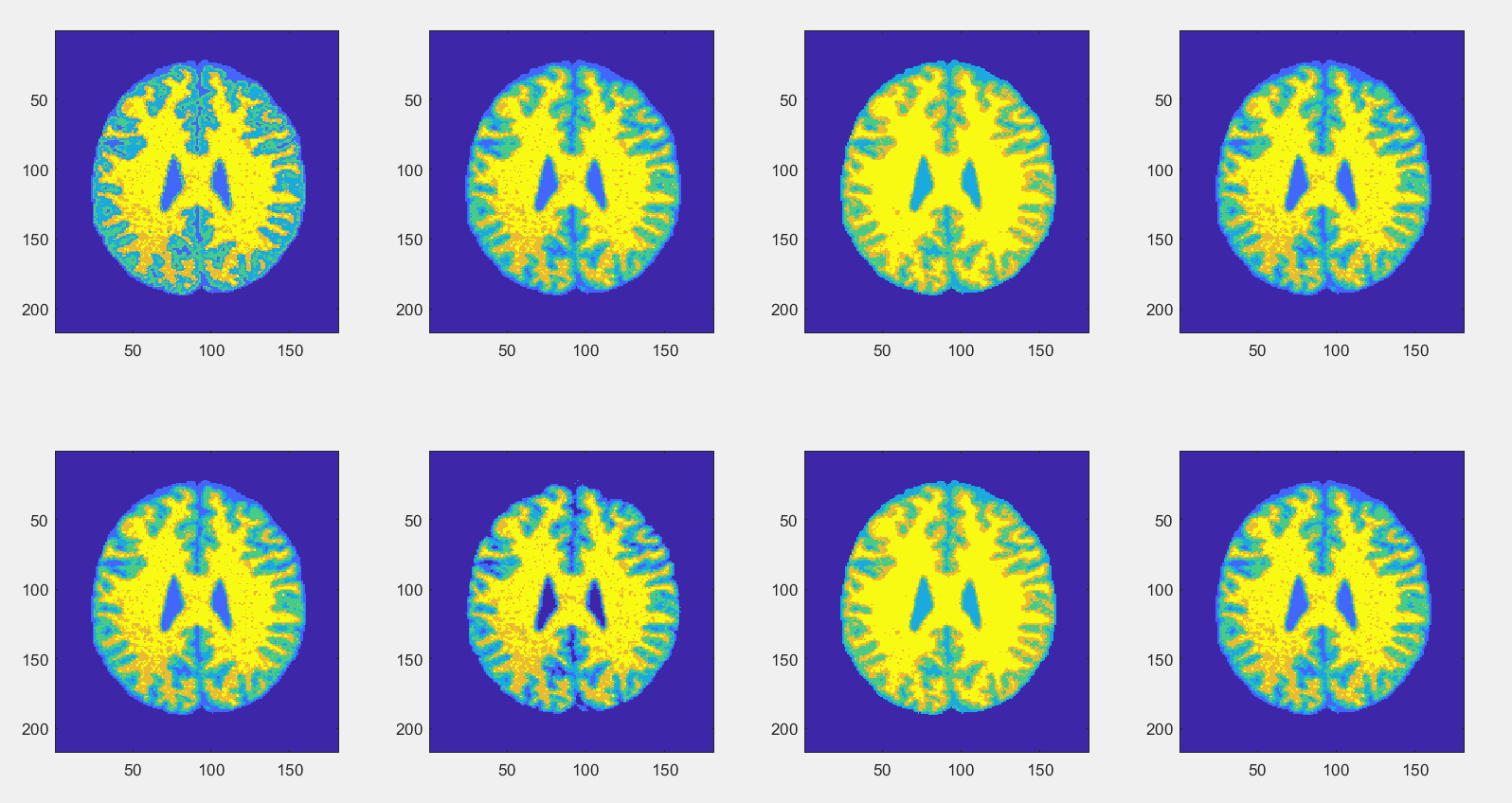


البته نحوه نمایش آن را به imagesc تغییر دادم تا بهتر بتوان سگمنت ها را مشاهده نمود.

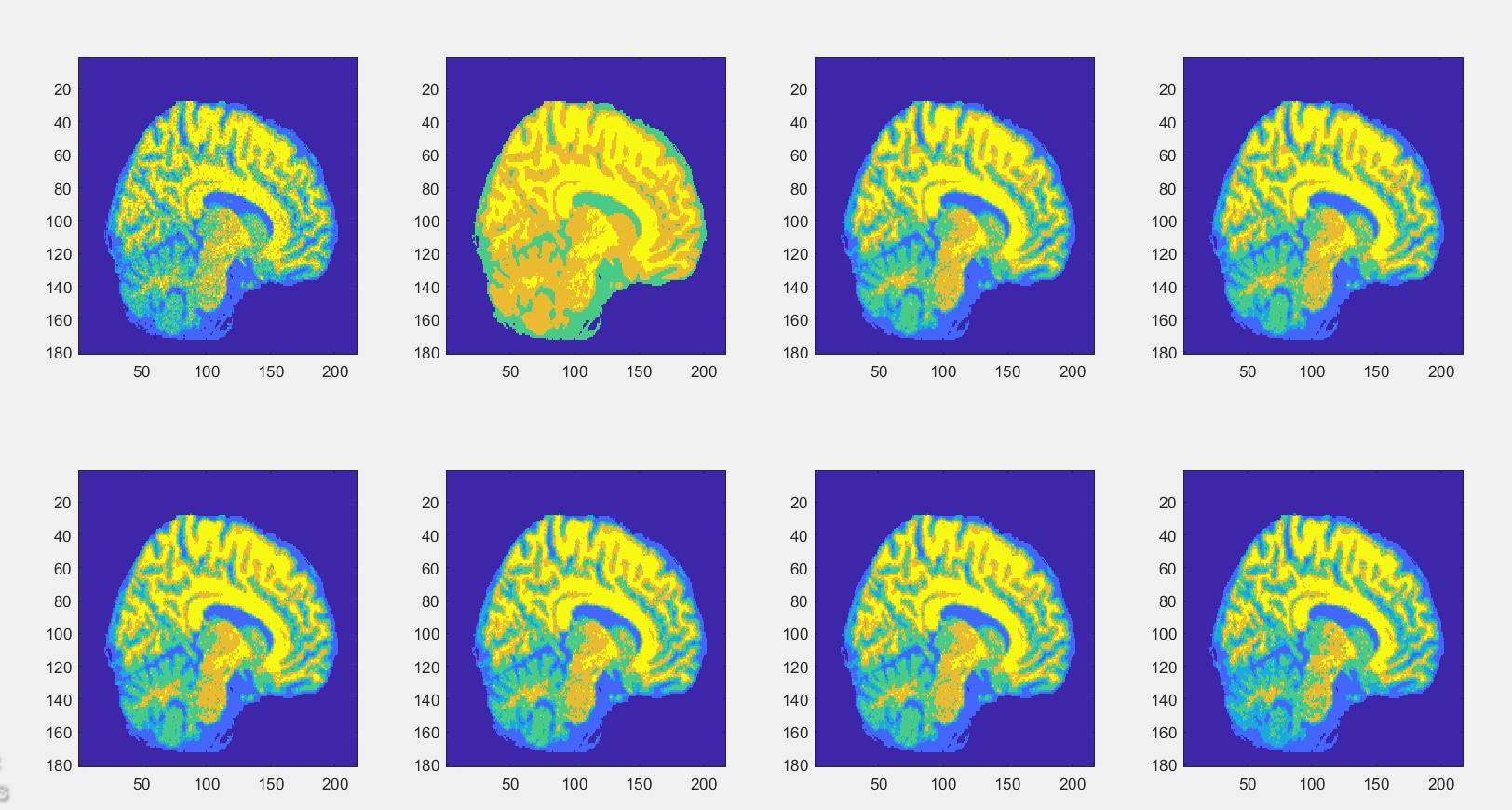
توجه : تصاویر زیر به ازای windowSize های مختلف زیر رسم شده است:

winSizeo=[1 3 5 7 9 11 13 15];

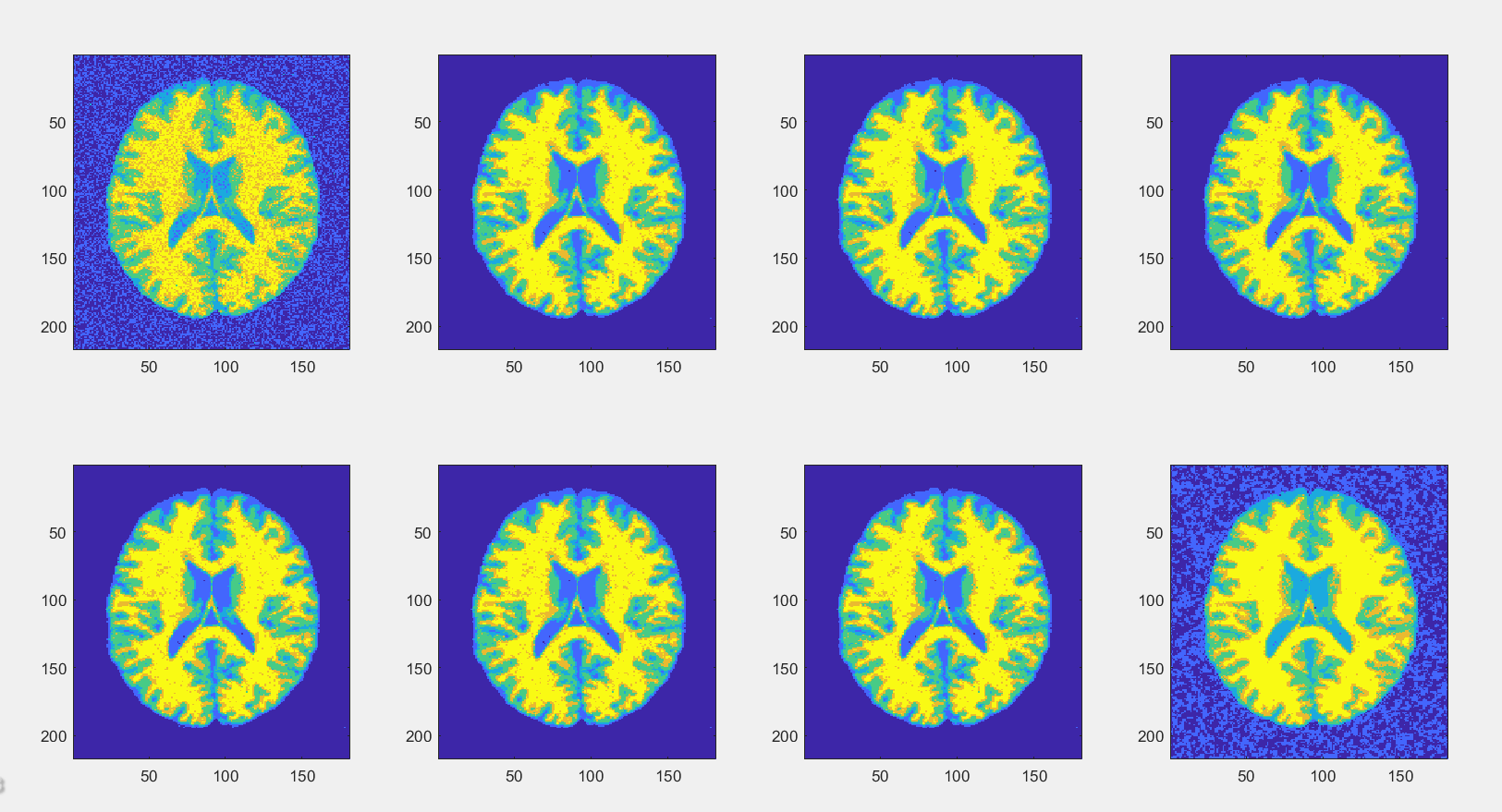
برای تصویر no720\_100A



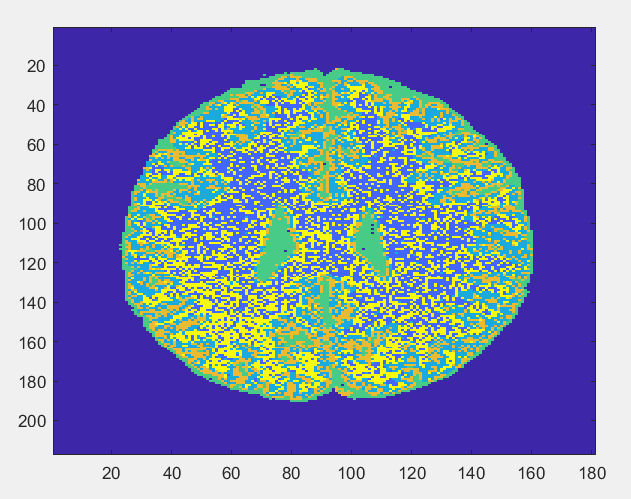
برای تصویر no720\_100S



برای تصویر rice10\_91A



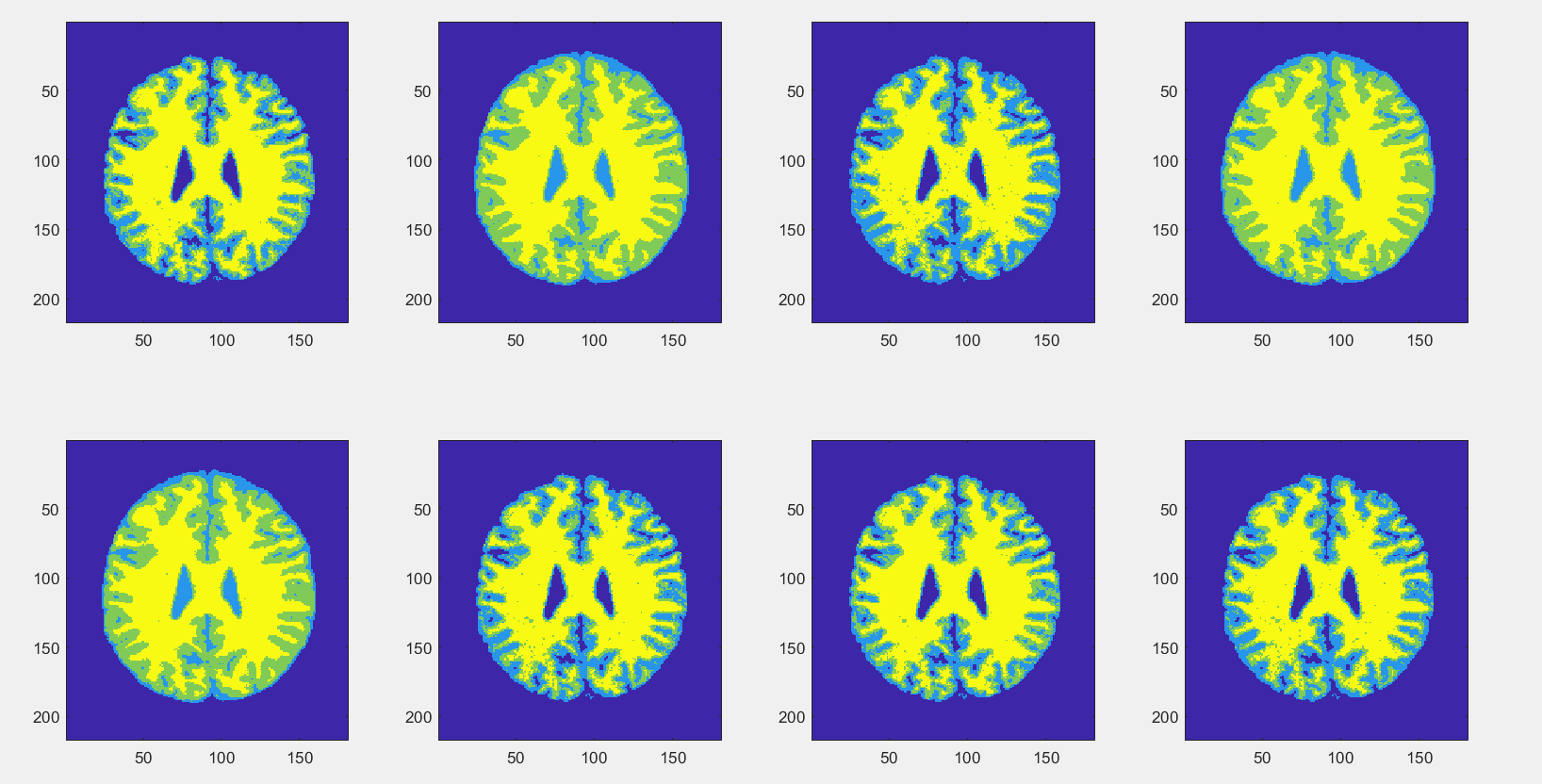
همچنین یک نمونه از آن تصاویر را با هدف مقایسه با این روش با استفاده از kmeans زده شده است. مشاهده میشود که در روش مقاله تاثیر نویز به اندازه قابل توجهی کاسته شده است:



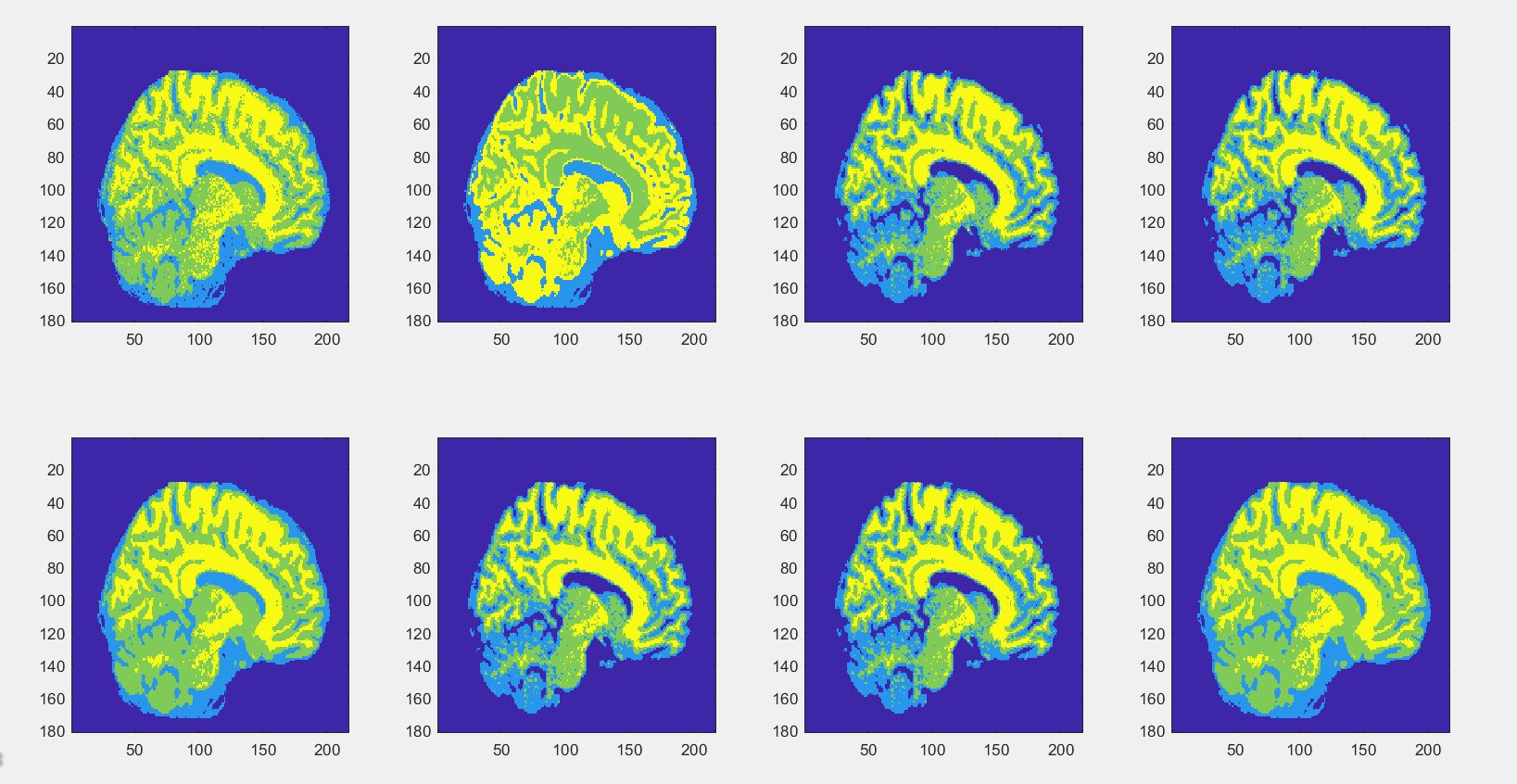
قسمت چهارم:

با تغییر cNum = 4 داریم:

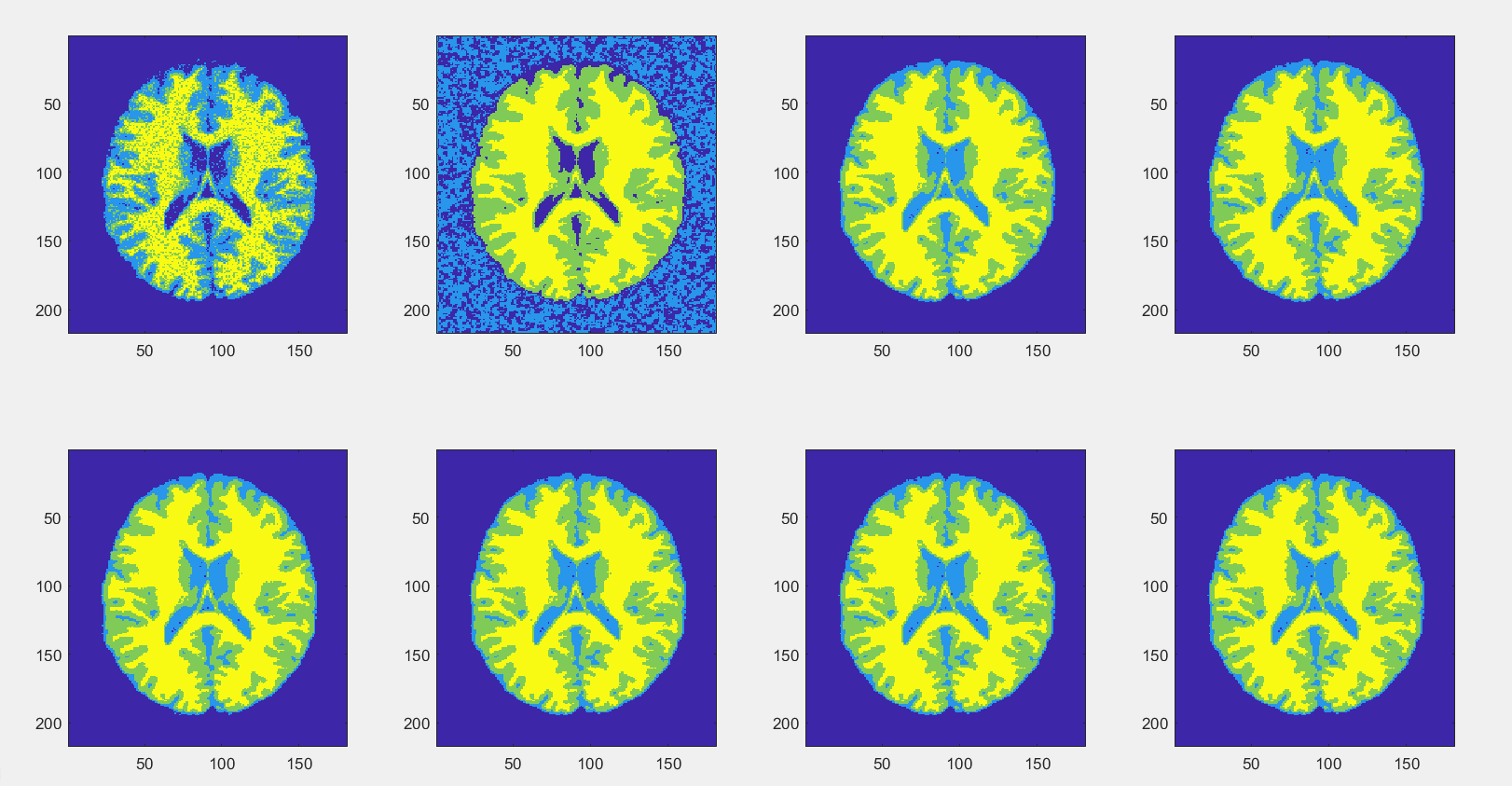
برای تصویر no720\_100A



برای تصویر no720\_100S

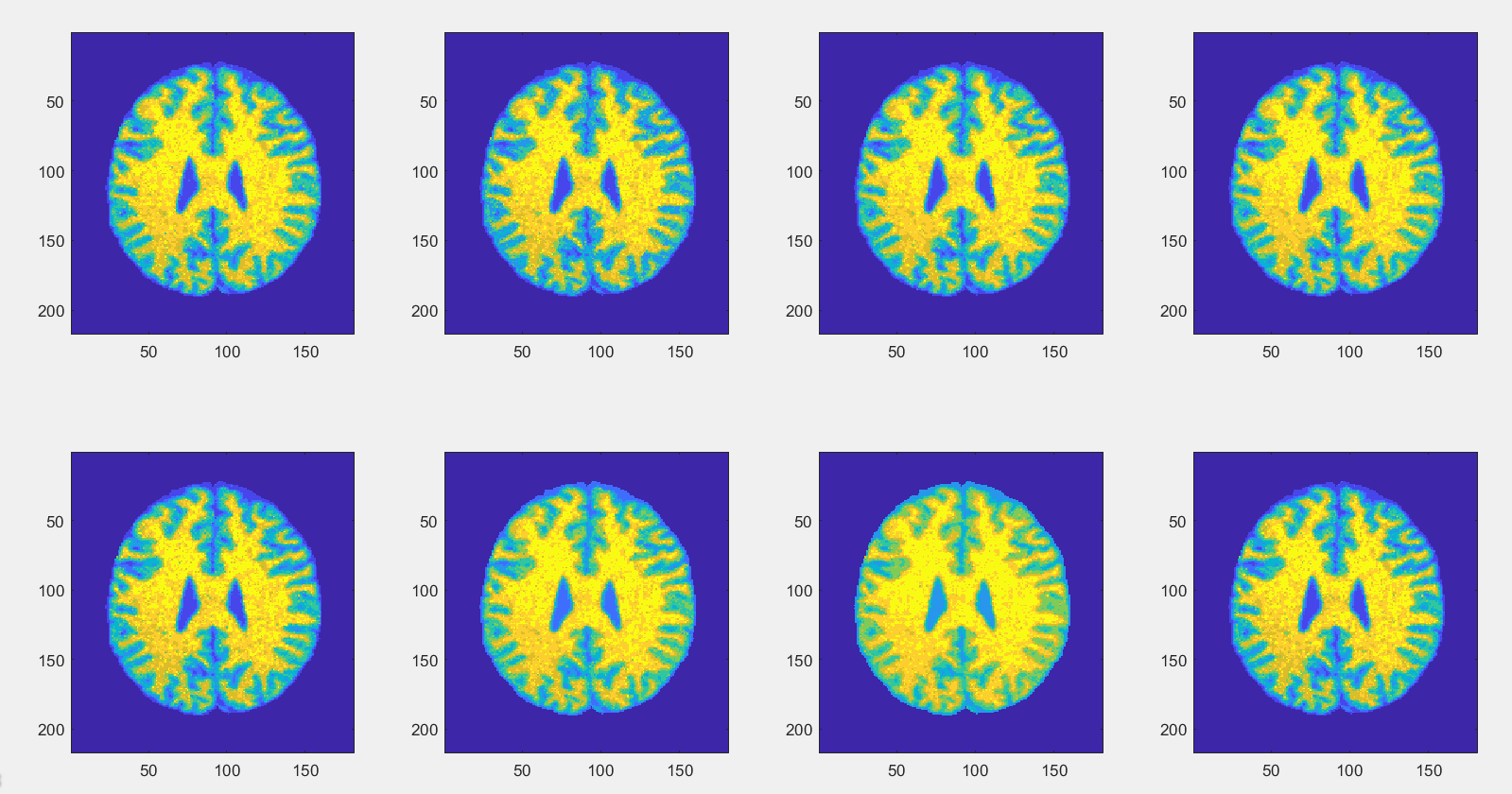


برای تصویر rice10\_91A

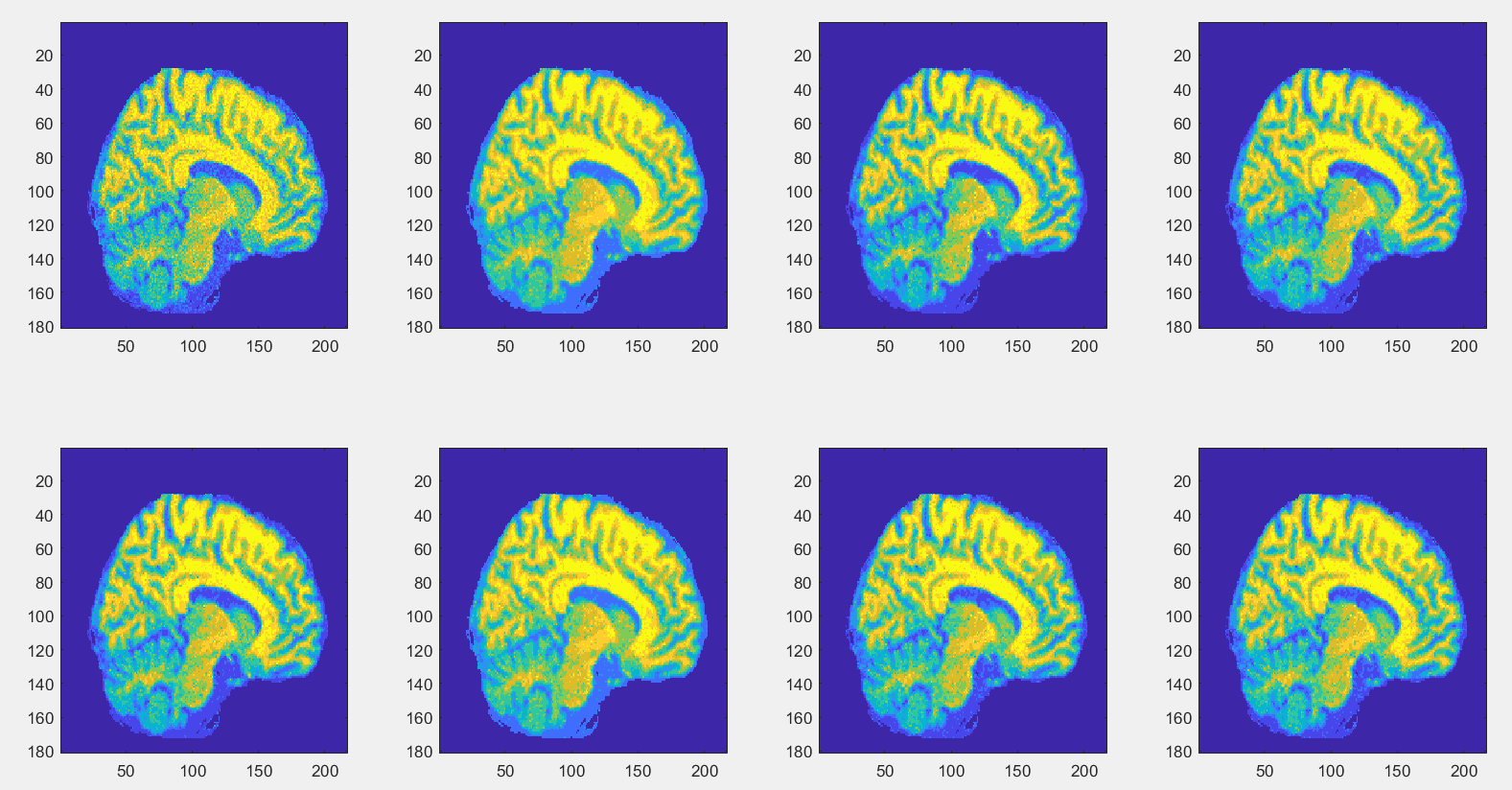


همچنین به ازای مقادیر cNum=10 نیز به شکل زیر است:

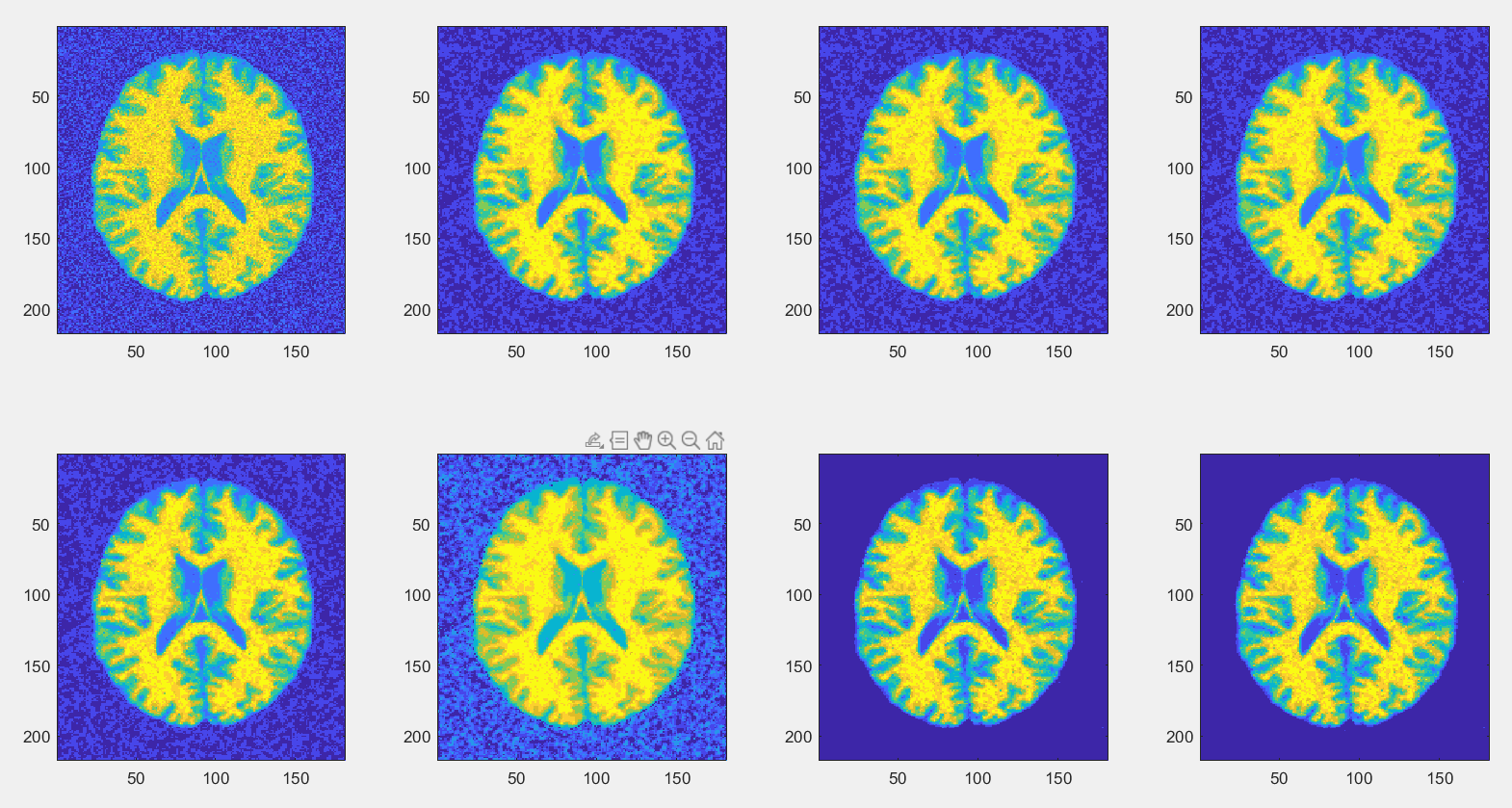
برای تصویر no720\_100A



برای تصویر no720\_100S



برای تصویر rice10\_91A



همانگونه که مشاهده میشود اگر تعداد کلاستر ها را از تعداد واقعی کلاستر ها خیلی بیشتر در نظر بگیریم تصویر خروجی نویز زیاد تصویر را نیز کلاستر میکند و برای همین تصویر خروجی نویزی میشود و اگر همان تعداد واقعی باشد کیفیت مطلوب خواهد بود. زمانی که نویز تصویر از حدی بیشتر نیست، با این حالی که تصویر را با تعداد کلاس های بالاتری سگمنتیشن انجام دادیم اما نتایج آن باز هم قابل قبول است و در حد مناسبی قرار دارد.

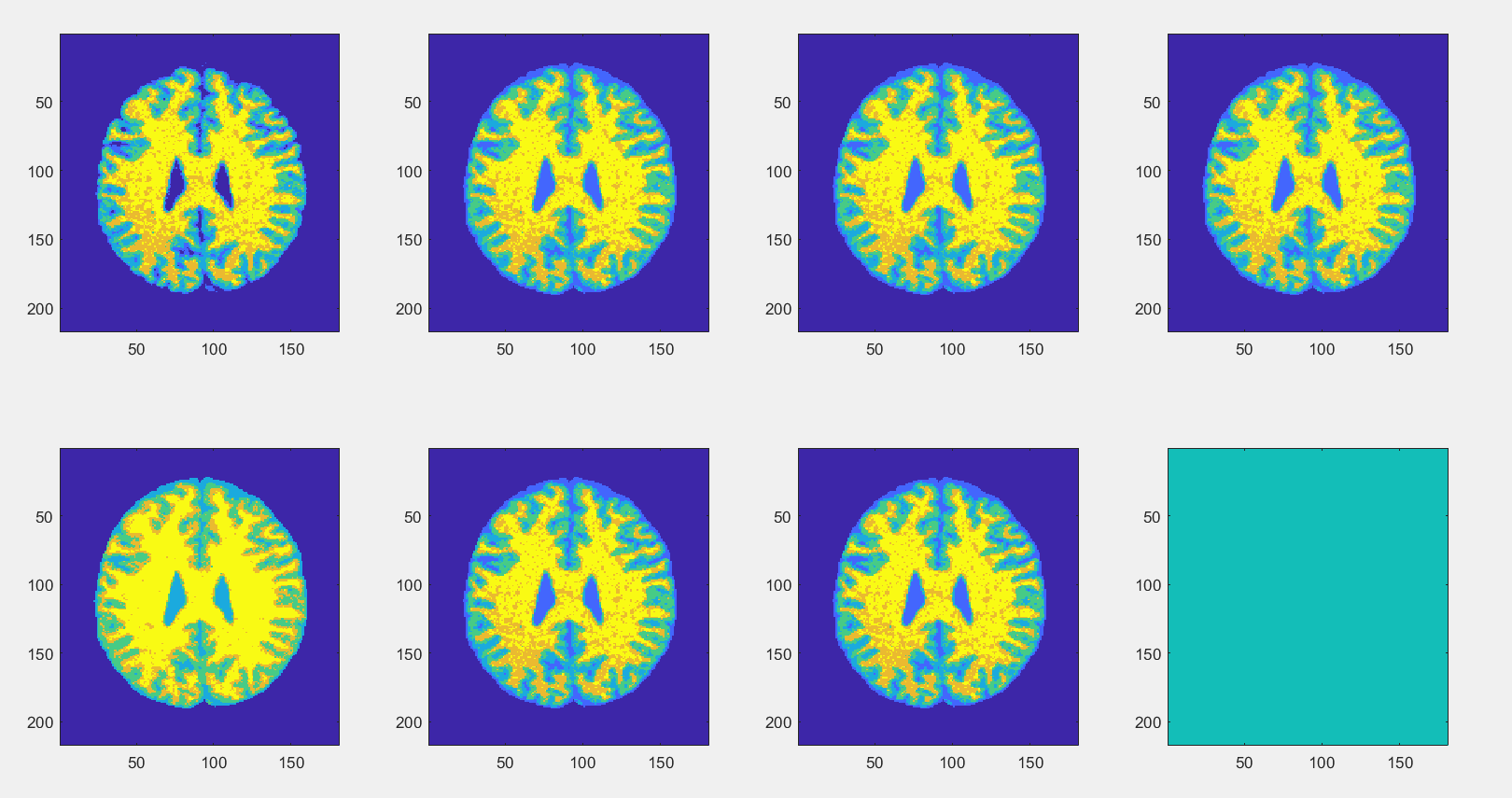
قسمت پنجم:

در این قسمت cNum=6 در نظر گرفته شده است و همچنین:

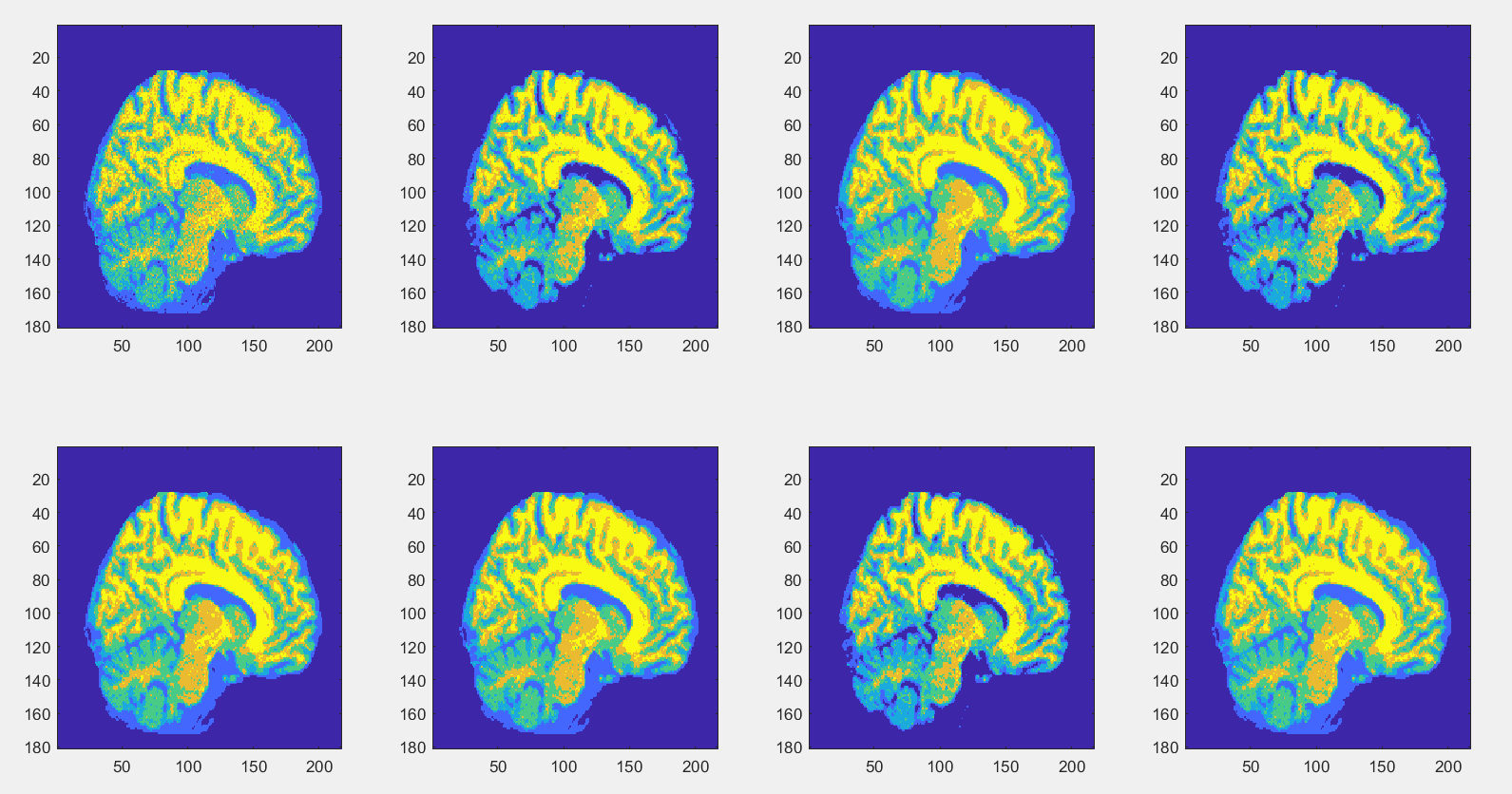
winSizeo=[1 3 5 7 9 11 15 29];

با تغییر پارامتر opt=’median’ داریم:

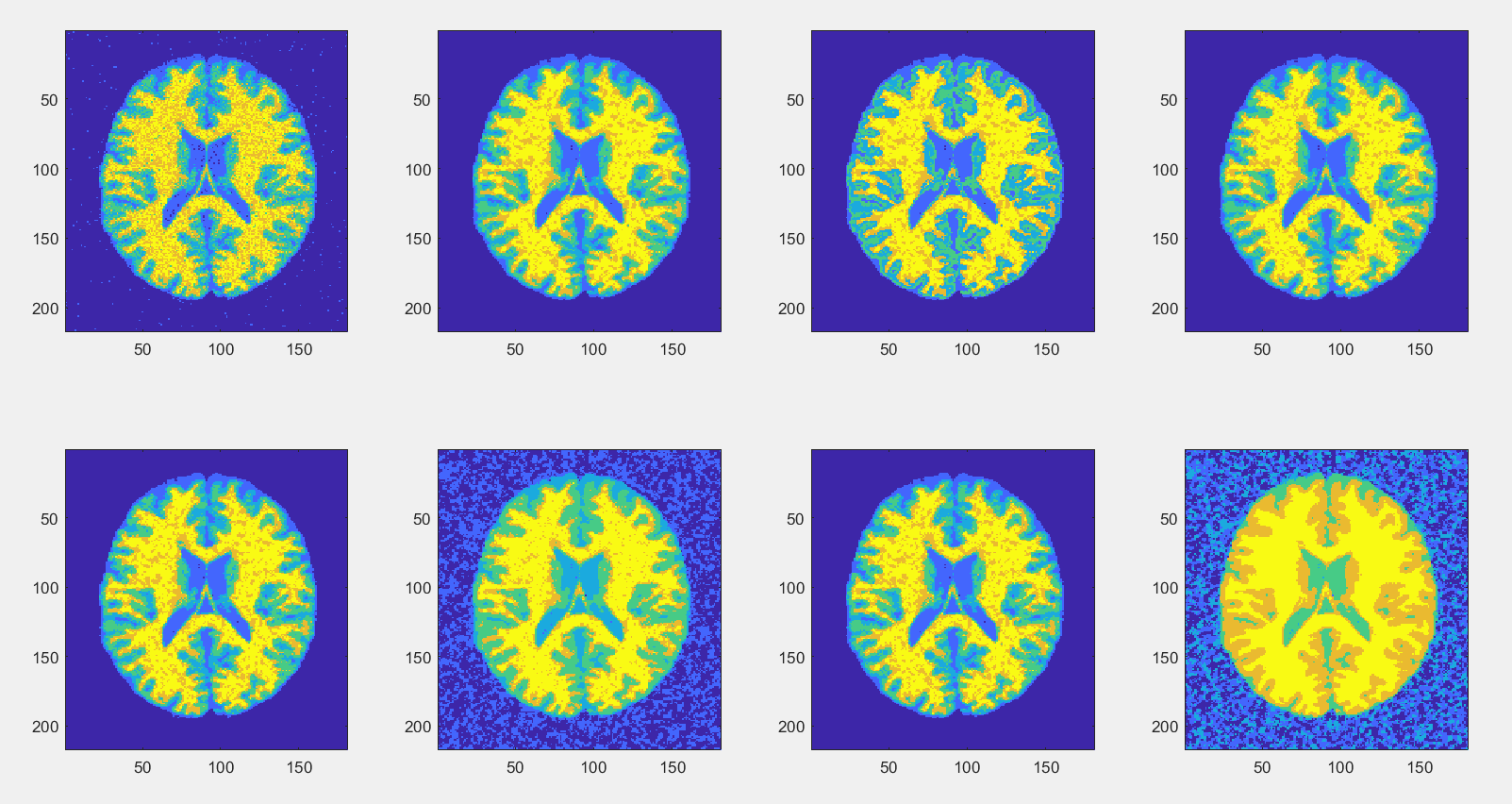
برای تصویر no720\_100A



برای تصویر no720\_100S

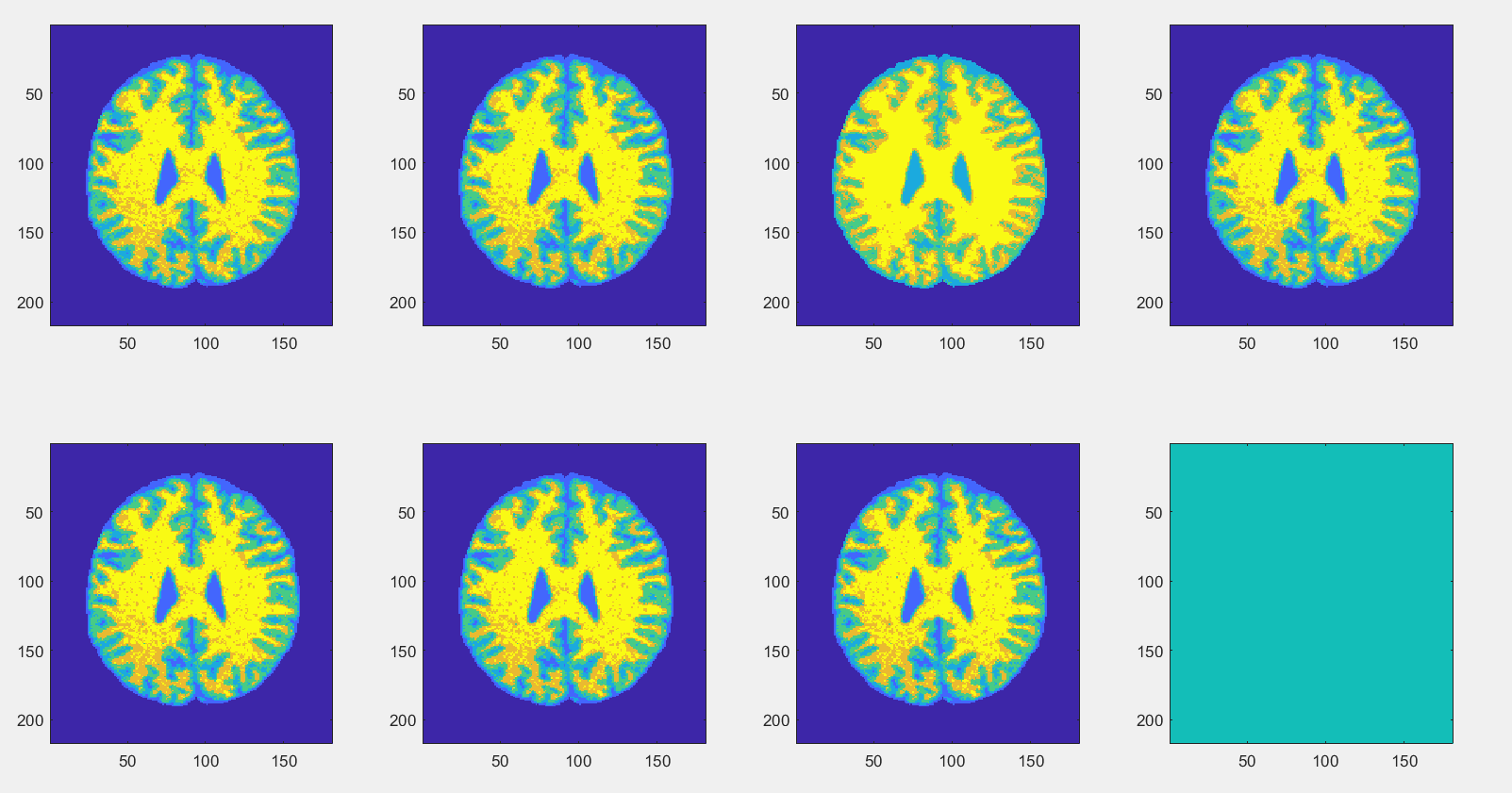


برای تصویر rice10\_91A

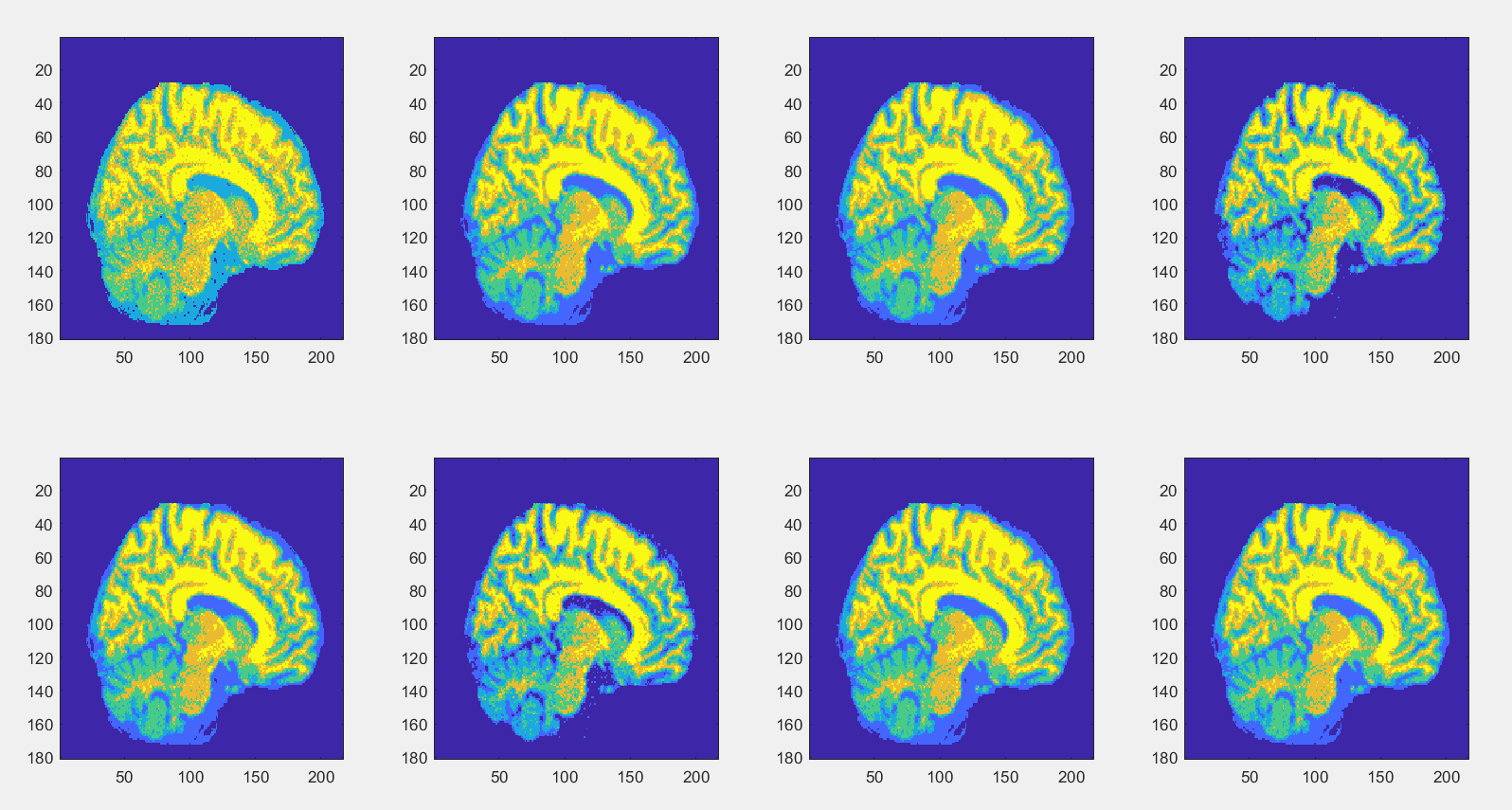


با تغییر پارامتر opt=’weighted’ داریم:

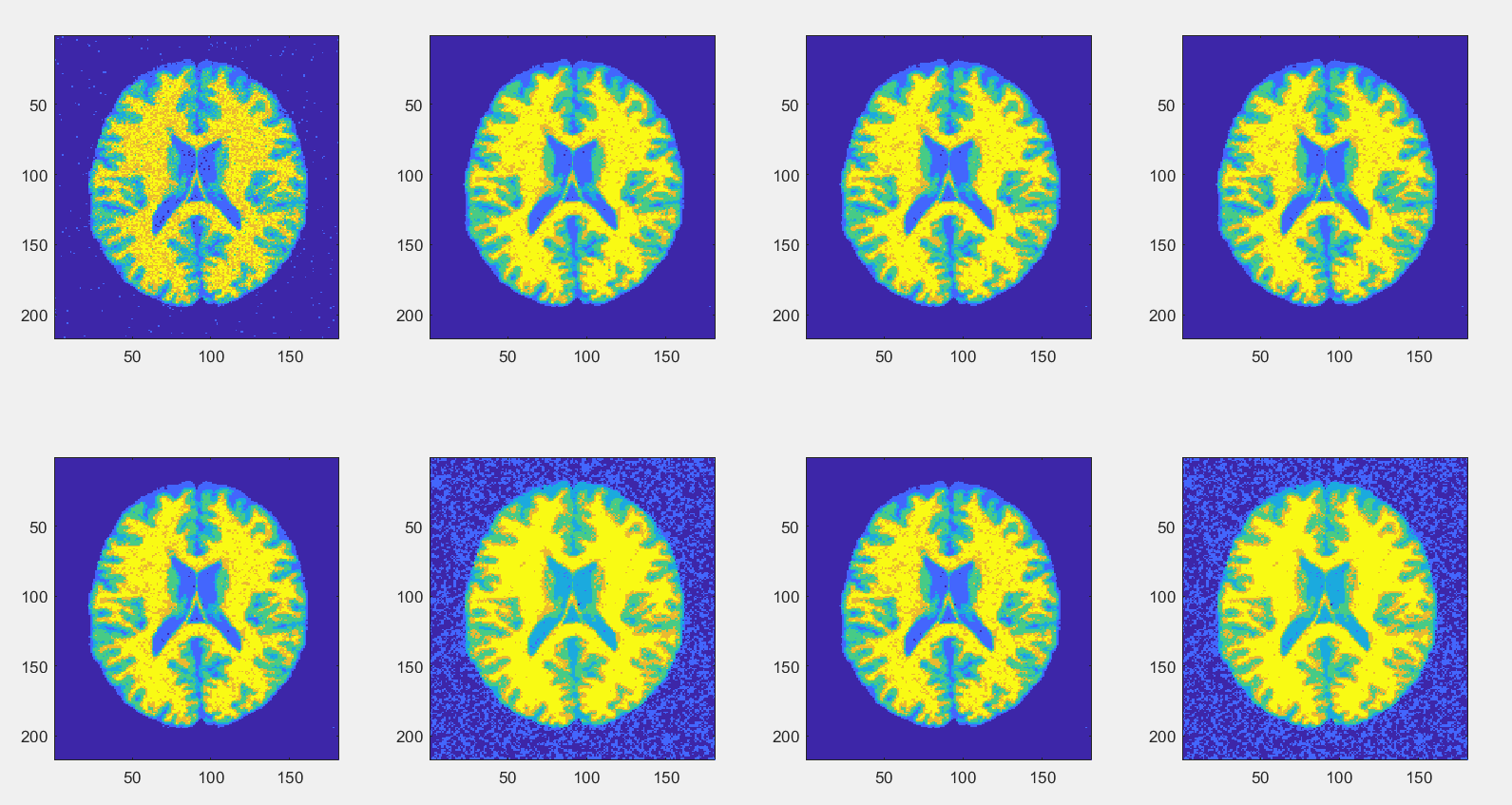
برای تصویر no720\_100A



برای تصویر no720\_100S



برای تصویر rice10\_91A



به ازای پنجره ی 29 در تصویری با روش های median و weighedمتاسفانه همگرا نشد. همچنین بنظر میرسد روش average نسبت به median بهتر عمل کرده است هم از نظر کیفیت جواب و هم از نظر سرعت اجرا. روش weighted از لحاظ کیفیت از دو روش دیگر کیفیت مطلوب تری را دارا می‌باشد به گونه ای که انتخاب پیشفرض بنظرم باید این روش باشد.

قسمت ششم: در این تمرین برای تمامی بخش ها به ازای پنجره های مختلف تست شده است. چیزی که از این اجرا ها به ذهن میرسد این است که این روش بسته به نویز و تصویر به ازای یک حداقل طول و حداکثر طول جواب های مطلوب تری را میدهند. کمتر از آن مقدار که باشد پاسخ ها کمی نویزی است و بیشتر از آن نیز باشد نیز علاوه برا این اتفاق ممکن است حتی ناپایدار نیز شود. اما به ازای اعدادی مانند 9 و یا 11 پاسخ برای تقریبا تمامی حالت ها خوب بوده است. روش weighted از این لحاظ وابستگی کمتری به طول پنجره داشته است و پاسخ های های مناسبی به ازای طول های کم مانند طول 3 داشته است.