**University:** Sharif University of Technology

**Department:** Electrical Engineering

**Course Name:** Medical Signal and Image Processing Lab

# Lab 8 Report

**Student Name:** Ali Shahbazi, Zahra Kavian, MohammadReza Safavi

**Student ID:** 98101866, 98102121, 98106701

**Instructor:** Dr. Sepideh Hajipour

Academic Semester: 2023 Spring

# فهرست مطالب



# فهرست تصاویر

## ۱ بخش تئوری

*اثبات.* مسئله‌ی بهینه‌سازی اصلی اگر کانولوشن را به صورت $Df$ نشان دهیم به صورت زیر است:

$$\underset{f}{\operatorname{argmin}} \, ||g - Df||^2$$

باید بردار گرادیان را برای این تابع حساب کنیم. اگر $g$ و $f$ بردار و $D$ ماتریس باشد داریم:

$$\begin{aligned} ||g - Df||^2 &= (g - Df)^T(g - Df) \\ &= (g^T - f^T D^T)(g - Df) \\ &= g^T g - g^T Df - f^T D^T g + f^T D^T Df \\ &= f^T(D^T D)f - f^T(D^T g) - (g^T D)f + g^T g \qquad (۱) \end{aligned}$$

از مشتق‌گیری برای بردارها و ماتریس‌ها می‌دانیم که:

$$\frac{\partial(a^T x)}{\partial x} = \frac{\partial(x^T a)}{\partial x} = a \qquad (۲)$$

$$\frac{\partial(x^T Ax)}{\partial x} = (A + A^T)x \qquad (۳)$$

در این حالت گرادیان همان مشتق ماتریسی می‌شود. بنابراین:

$$\begin{aligned} \frac{\partial}{\partial f} ||g - Df||^2 &\overset{(۱)}{=} \frac{\partial}{\partial f} \left(f^T(D^T D)f - f^T(D^T g) - (g^T D)f + g^T g\right) \\ &\overset{(۲),(۳)}{=} 2(D^T Df) - 2(D^T g) \\ &= 2D^T(Df - g) \end{aligned}$$

از آنجایی که روش ما gradient descent است، باید در خلاف جهت گرادیان حرکت کنیم و همچنین در هر iteration با ضریبی مانند $\eta$ حدس قبلی را آپدیت کنیم. در نهایت:

$$\begin{aligned} f_{k+1} &= f_k - \eta \frac{\partial}{\partial f} ||g - Df||^2 \\ &= f_k - \eta(2D^T(Df - g)) \\ &= f_k + \beta D^T(g - Df) \end{aligned}$$

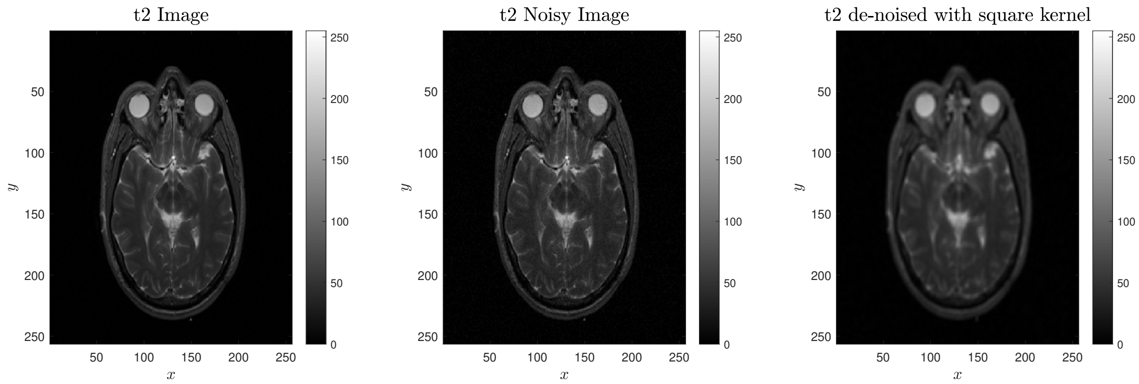
□

## ۲ بخش شبیه‌سازی

۱) برای ساخت نویز گوسی با واریانس ۱۵، از دستور زیر استفاده می‌کنیم:

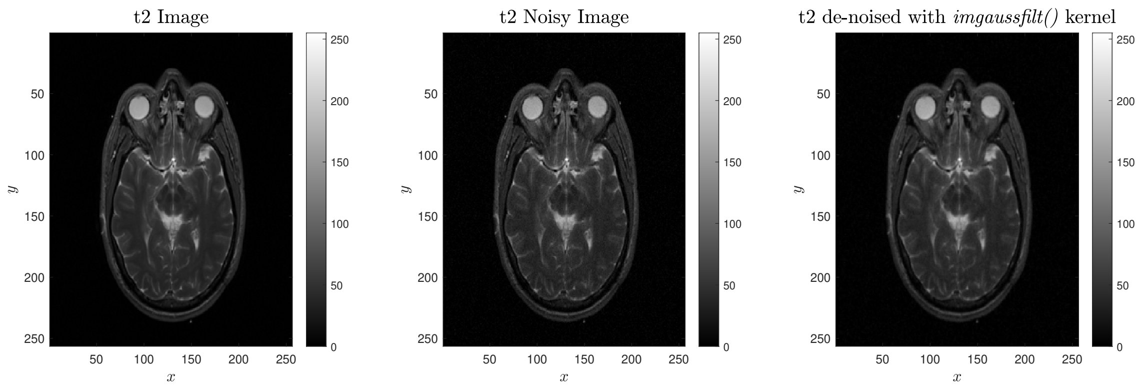
```
t2_noisy = t2 + randn(size(t2))*sqrt(15);
```

که از توزیع نرمال عدد تولید کرده و واریانس مورد نظر را می‌سازد. پس از ساخت کرنل مورد نظر، تصویر اصلی، تصویر نویزی و تصویر حاصل از ضرب تصویر اصلی و کرنل در حوزه فوریه را در شکل ۱ نشان داده‌ایم.



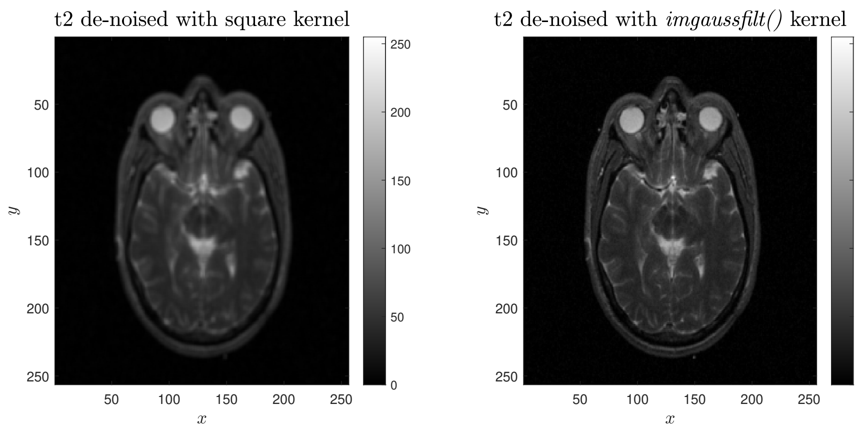

شکل ۱: مقایسه تصویر حاصل از کرنل مربعی و تصویرهای اصلی

به کمک دستور *imgaussfilt()* تصویر نویزی را با کرنل گوسی با واریانس ۱ کانولوشن می‌کنیم و نتیجه در شکل ۲ نشان داده شده است.



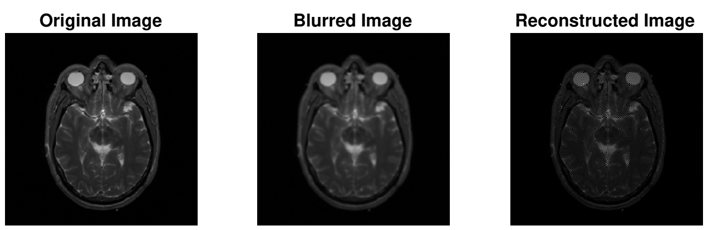

شکل ۲: مقایسه تصویر حاصل از کرنل گوسی و تصویرهای اصلی

برای مقایسه‌ی بهتر این دو کرنل، آن‌ها را کنار همدیگر در شکل ۳ رسم کرده‌ایم. همانطور که مشاهده می‌شود، نتیجه حاصل از کرنل مربعی کمی blur شده است در حالی که نتیجه کرنل گوسی رزولوشن مناسب را حفظ کرده است. عملیاتی که در این دو کرنل انجام شد مشابه است، یعنی در هر دو یک کرنل با تصویر نویزی کانولوشن شد تا اثر نویز را کم کند اما در یکی مربعی و در دیگری گوسی. دلیل بیشتر blur شدن تصویر حاصل از کرنل مربعی این است که این کرنل در واقع برای بدست آوردن روشنایی هر پیکسل، میانگین آن نقطه و ۱۵ نقطه اطرافش را با وزن یکسان در نظر می‌گیرد. اما در کرنل گوسی هر چه همسایه‌ی آن پیکسل دورتر باشد، اثرش نیز کمتر خواهد بود.




شکل ۳: مقایسه کرنل‌های اعمال شده

۲) برای این بخش ابتدا تصویر را از فایل‌ها خوانده، اسلایس اول آن را در نظر گرفته و پلات می‌کنیم. سپس با استفاده از تابع داده شده یک کرنل گوسی می‌سازیم و آن را به تصویر اعمال می‌کنیم و نتیجه را نمایش می‌دهیم. حال تبدیل فوریه تصویر محو شده و کرنل گوسی که ساختیم را محاسبه می‌کنیم و تبدیل فوریه وارون حاصل تقسیم آن‌ها را به دست می‌آوریم و به عنوان تصویر بازسازی‌شده نمایش می‌دهیم. با انجام این مراحل به خروجی شکل ۴ می‌رسیم.




شکل ۴: تصویر اصلی و محو شده و بازسازی‌شده در کنار هم

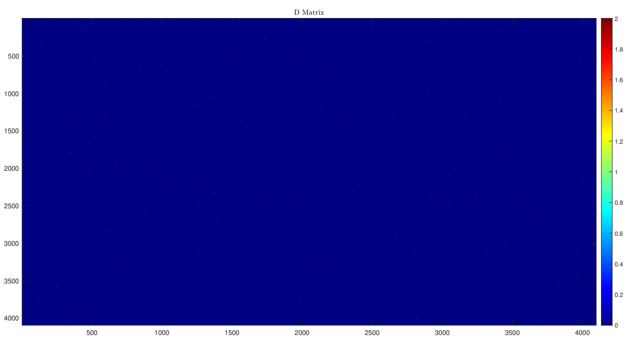
بار دیگر مراحل بالا را تکرار می‌کنیم اما این بار به تصویر محوشده اندکی نویز نیز اضافه می‌کنیم. این بار به خروجی شکل ۵ می‌رسیم.


شکل ۵: تصویر اصلی و محوشده به همراه نویز و بازسازی‌شده

مشاهده می‌شود با این که شده نویز خیلی کم است و عملا تصویر محوشده با و بدون نویز تفاوت زیادی ندارند اما نتیجه بازسازی از این روش نسبت به حالت بدون نویز، خطای زیادی دارد که نشان‌دهنده عدم مقاومت این الگوریتم نسبت به نویز و تغییرات کوچک تصویر است که ناشی از ذات الگوریتم است که در خود تقسیمی دارد که ممکن است در برخی نقاط مخرج صفر شود و ناپایداری داشته باشیم.

**۳)** ابتدا ماتریس تمام صفر k به اندازه تصویر ساخته و فیلتر $h$ بالا سمت چپ ماتریس قرار داده و با شیفت چرخشی آن، ماتریس $D$ ساخته می شود (طبق مراحل گفته شده در صورت سوال). ماتریس $D$ به ابعاد $N^2 \times N^2$ است، یعنی تعداد سطر های آن به تعداد پیکسل های تصویر است. زمانیکه تصویرمان (که به بردار $N^2 \times 1$ تبدیل می شود) در ماتریس $D$ ضرب می شود، جمع وزن دار پیکسل هایی که مقدار متناظر فیلتر برای شان صفر نیست برابر مقدار پیکسل $r$ و $c$ در تصویر فیلتر شده مان است. اینکار معادل با لغزاندن فیلتر $h$ روی کل تصویر و ضرب داخلی فیلتر و پیکسل های پوشانده شده و قرار دادن مقدار محاسبه شده در پیکسل میانی است.

شکل ۶ نمایی از ماتریس $D$ ساخته شده نشان می دهد.

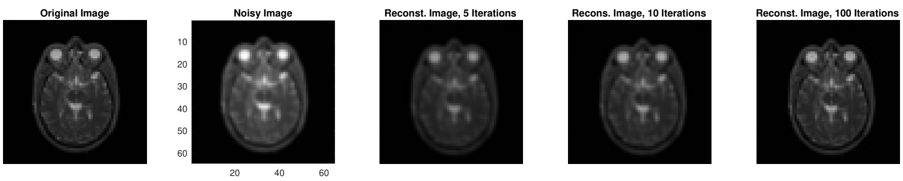



شکل ۶: ماتریس D ساخته شده

شکل ۷ تصویر اولیه، فیلتر شده و بازیابی شده را نمایش می دهد. تصویر بازیابی شده به طور قابل توجهی مشابه تصویر اصلی است و بازسازی تصویر نویزی به خوبی انجام شده است.


شکل ۷: مقایسه تصویر اصلی، نویزی و بازسازی شده

**۴)** با استفاده از روابط ریاضی داده شده برای روش گرادیان کاهشی و ماتریس $D$ ساخته‌شده در قسمت قبل، الگوریتم گرادیان کاهشی را اجرا می‌کنیم. برای بررسی همگرایی، یک بار ۵ گام و بار دیگر ۱۰ گام سپس ۵۰ گام و در نهایت ۱۰۰ گام این الگوریتم را اجرا و نتایج را در شکل ۸ مشاهده می‌کنیم.




شکل ۸: مقایسه تصویر اصلی، نویزی و بازسازی شده در تکرارهای مختلف

اولا همگرایی الگوریتم کاملا مشخص است و با اجرای گام‌های بیشتر، خروجی شفاف‌تر شده است و همچنین در گام ۱۰۰ام، عملا خروجی بسیار شبیه به ورودی است. همچنین مشاهده می‌شود عملکرد این روش بهتر از بخش قبل

است و با تکرار بیشتر گام‌ها به نتایج حتی بهتری هم خواهیم رسید. مزیت بسیار مهم این روش، عدم نیاز به محاسبه شبه‌وارون ماتریس D است زیرا محاسبه شبه‌وارون، بار محاسباتی بسیار زیادی دارد و موجب کند شدن الگوریتم می‌شود. همچنین مشاهده می‌شود کد این بخش از بخش قبل سریع‌تر نیز اجرا می‌شود که البته بخش قابل توجهی از این تفاوت سرعت مرهون همان عدم نیاز به محاسبه وارون ماتریس است. مزیت دیگر این روش آن است بسته به شرایط مسئله، ما می‌توانیم تعداد گام اجرای الگوریتم را به دلخواه کم یا زیاد کنیم و سرعت و دقت را با آن کنترل کنیم. موردی که در قسمت قبل بر آن کنترلی نداریم و عملا آزادی عمل بیشتری در الگوریتم گرادیان کاهشی داریم.

۵) Anisotropic Diffusion Filtering: این روش اولین بار توسط Perona & Malik در سال ۱۹۹۰ ارائه شد. در واقع این روش یک linear diffusion است که به شکل زیر تعریف می شود:

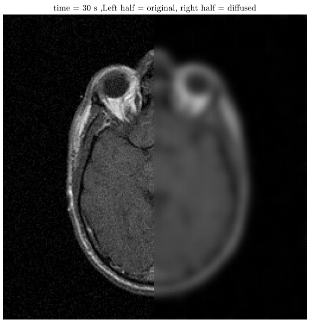
$$\begin{aligned} \frac{\partial I_t(x,y)}{\partial t} &= div[c_t.\nabla I_t(x,y)] \\ c_t^i(x,y) &= g(\nabla I_t^i(x,y)) \\ g(\nabla I) &= \frac{1}{1+(|\nabla I|/K)^2} \end{aligned} \tag{۴}$$

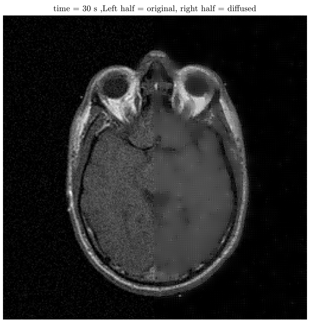
که $I_t(x,y)$ تصویر در زمان $t$ ، div همان دیورژانس، $\nabla I_t(x,y)$ گرادیان تصویر، $c_t$ ضریب دیفیوژن است. اگر $c_i$ ثابت باشد ،به معادله isotropic diffusion تبدل می شود. این روش معادل کانوالو کردن تصویر در فیلتر گاوسی است. ایده اصلی معادله anisotropic diffusion تغییر مقدار $C_i$ در هر مرحله به گونه ای که هر ناحیه از تصویر تا حد امکان صاف شود و لبه های بین ناحیه ها نگه داشته شود. در واقع $C_i$ یک تابع همواره مثبت و نزولی از گرادیان تصویر است. در نهایت ناحیه های نویزی و یا سایه شده (تغییرات شدت تصویر و مقدار گرادیان کم) تا حد ممکن از بین می رود و نواحی مثل لبه ها (شدت تغییرات زیاد) حفظ می شود.[۱]

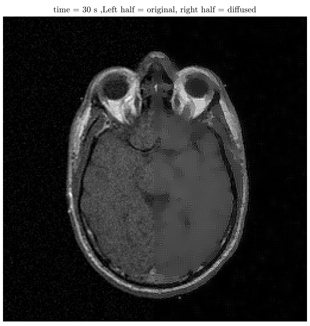
کد S2_Q5_Anisotropic_Diffusion.m داده شده، ابتدا به تصویر مورد نظر نویز گوسی اضافه شده و برای حذف نویز از سه مدل روش AD استفاده می شود. در مدل ساده (method=0 ) روابط ۴ پیاده سازی شده است که قسمت های نویزی تصویر کاهش یافته و لبه ها حفظ می شود. در مدل sophisticated علاوه بر لبه ها، گوشه ها نیز حفظ شده، در مدل ساده تنها قسمت های نویزی محو می شود.

مقایسه تصویر اولیه و حذف نویز شده بعد از ۳۰۰ بار اجرا روش (استفاده از سه مدل):

[۱] chao۲۰۱۰anisotropic.




شکل ۹: مقایسه تصویر اصلی و بازسازی شده با سه مدل روش lrAD - به ترتیب از چپ به راست: مدل بلور شده، ساده و پیچیده