

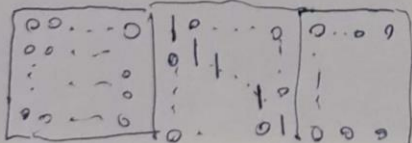
MIAP – HW3

Armin Navardi 99105129

بخش تئوری

.1

$$\begin{aligned}
 1. \quad \frac{\partial L}{\partial \hat{X}} = 0 &\Rightarrow 2\lambda(\hat{X} - Y) + 2 \sum_{i,j} -R_{ij}^T (b\alpha_{ij} - R_{ij}\hat{X}) = 0 \\
 &\Rightarrow (\lambda I + \sum_{i,j} R_{ij}^T R_{ij}) \hat{X} - \left[\sum_{i,j} R_{ij}^T (b\alpha_{ij}) + \lambda Y \right] = 0 \\
 &\Rightarrow \hat{X} = (\lambda I + \sum_{i,j} R_{ij}^T R_{ij})^{-1} (\lambda Y + \sum_{i,j} R_{ij}^T b\alpha_{ij})
 \end{aligned}$$

R : 
 R : $\begin{bmatrix} 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \\ 0 & 0 & 0 & 0 \end{bmatrix}$

.2

$$\begin{aligned}
 2. \quad \frac{\partial J}{\partial w_j} = 0 &\Rightarrow \sum_{i=1}^N 2u_{ij}^2 (x_i - w_j) = 0 \Rightarrow w_j = \frac{\sum_{i=1}^N u_{ij}^2 x_i}{\sum_{i=1}^N u_{ij}^2} \\
 \frac{\partial J}{\partial u_{ij}} = 0 &\Rightarrow 2u_{ij} \|x_i - w_j\|_2^2 + \lambda_i = 0 \Rightarrow u_{ij} = \frac{-\lambda_i}{2\|x_i - w_j\|_2^2} \\
 \frac{\partial J}{\partial \lambda_i} = 0 &\Rightarrow \sum_{j=1}^K u_{ij} = 1 \Rightarrow \sum_{j=1}^K \frac{-\lambda_i}{2\|x_i - w_j\|_2^2} = 1 \\
 \Rightarrow \lambda_i &= \frac{-2}{\sum_{j=1}^K \frac{1}{\|x_i - w_j\|_2^2}} \Rightarrow u_{ij} = \frac{1}{\sum_{l=1}^K \frac{1}{\|x_i - w_l\|_2^2}}
 \end{aligned}$$

.3

Unsupervised image noise modeling with self-consistent generator. J., 2019, W., Feng, V., Yang, H., Tan, Yan
arXiv preprint arXiv:1906.05762.

در این روش با استفاده از Generative Adversarial Network که شامل یک نتورک discriminator و generator است سعی میشود که توزیع احتمال نویز تخمین زده شود و برای آموزش به عکس های تمیز نیازی ندارد و صرفاً از عکس های نویزی استفاده میکند. بخش generator سعی در تولید عکس های دقیق دارد و بخش discriminator سعی در تشخیص عکس های generate شده از عکس های واقعی دارد.

A convolutional neural network for transform-domain collaborative filtering: Bm3d-net. J., 2017, D., Sun, Yang
55–59: (IEEE Signal Processing Letters 25 (1).

در این مقاله پایپ لاین BM3d در قالب یک شبکه CNN در آمده است که در حوزه تبدیل عمل میکند.

Attention mechanism enhanced kernel prediction networks for denoising of burst images. Z., 2020, Y., Xiong, Y., Huang, S., Xia, B., Jin, Zhang
ICASSP 2020-2020 IEEE. In: (Speech and Signal Processing (ICASSP, International Conference on Acoustics
.pp, IEEE. 2083–2087:

در این مقاله از مکانیزم attention برای تخمین کرنل مناسب برای حذف نویز استفاده شده است. مکانیزم attention کمک میکند تا در نواحی که دارای اهمیت و اطلاعات بیشتری هستند از بین نروند و اطلاعات بی اهمیت حذف نویز شوند.

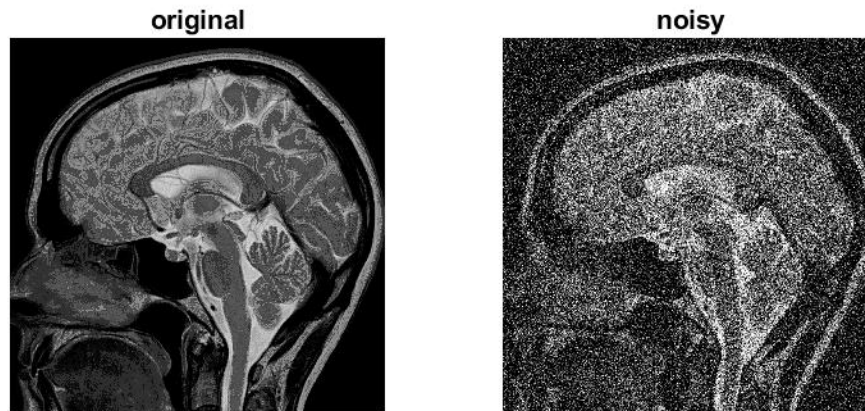
در روش DNCNN از یک شبکه CNN چند لایه شامل batch normalization و فعالسازی ReLU استفاده میشود تا نویز تخمین زده شود تصویر تمیز از کم کردن خروجی از ورودی بدست می آید. در FFDNet تصویر تمیز تخمین زده میشود به این صورت که ابتدا 4 تصویر downsample شده از تصویر اصلی بدست آورده و این 4 تصویر را به همراه noise level map که نشان دهنده سطح نویز در نقاط مختلف تصویر است به ورودی شبکه CNN میدهیم و تصویر تمیز را در خروجی میگیریم.

تفاوت آن ها در این است که اولی نویز را تخمین میزند و دومی تصویر تمیز را. همچنین دومی از تصاویر noise level map , downsample در ورودی استفاده میکند.

بخش عملی

1.

نویز گاوسی اضافه میکنیم.



با شیفیت دادن تصویر به جهت های بالا، پایین، چپ و راست و کم کردن آن از تصویر اصلی ماتریس های گرادیان در هر جهت را محاسبه میکنیم

دو روش زیر برای محاسبه conduction coefficients وجود دارد که در آن I تصویر اصلی و X راستای مدنظر است.

$$cX = \exp\left(-\left(\frac{1}{\kappa} \cdot \frac{\partial I}{\partial X}\right)^2\right)$$

$$cX = \frac{1}{1 + \left(\frac{1}{\kappa} \cdot \frac{\partial I}{\partial X}\right)^2}$$

به ازای پارامترهای مختلف جستجوی حریصانه انجام میدهم برای معیارهای ssim و niqe بهترین تصویر را ذخیره میکنیم.

NIQE: [niter, k, l, option] = 20.0000 0.5000 0.0500 1.0000

SSIM: [niter, k, l, option] = 30.0000 0.5000 0.0500 2.0000

NIQE:



SSIM:



2.

(الف)

نحوه عملکرد تابع isodiff به این صورت است که ابتدا با تفاضل گیری نقاط شیفته یافته به اندازه 1 واحد، گرادیان عکس را در 4 جهت محاسبه میکند و برای محاسبه عکس در لحظه بعدی، عکس در لحظه فعلی را با dt ضرب در عددی ثابت در گرادیان های هر جهت جمع میکند.

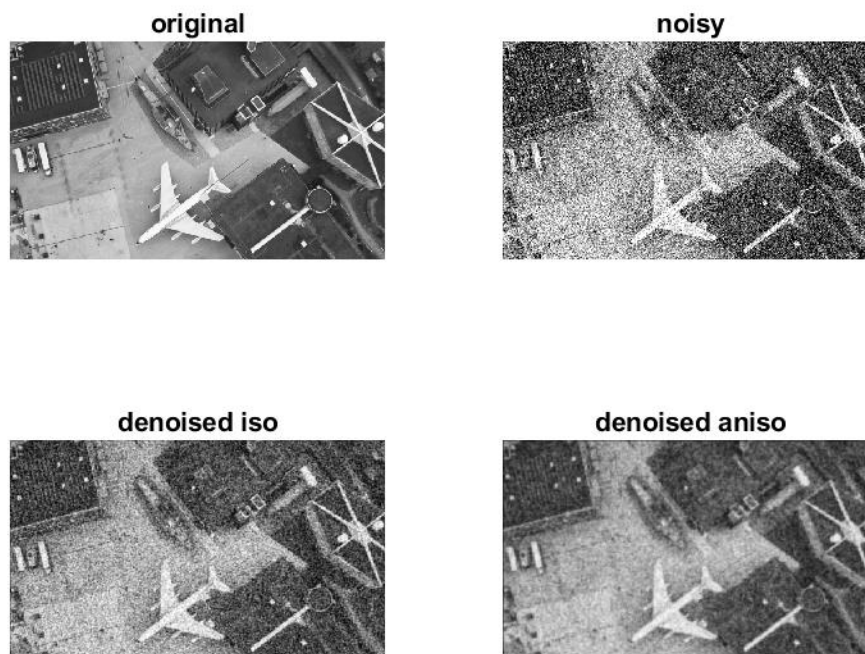
نحوه عملکرد anisodiff نیز مشابه isodiff است با این تفاوت که برای محاسبه عکس در لحظه بعدی به گرادیان در هر جهت ضریب متفاوتی میدهد. این ضریب ها تابعی از گرادیان هر جهت بوده و دو روش برای محاسبه آن ها وجود دارد.

$$cX = \exp\left(-\left(\frac{1}{\kappa} \cdot \frac{\partial I}{\partial X}\right)^2\right)$$

$$cX = \frac{1}{1 + \left(\frac{1}{\kappa} \cdot \frac{\partial I}{\partial X}\right)^2}$$

پارامترها شامل:

Im: عکس ورودی



(ب)

معیار nique است که برای محاسبه به عکس مرجع نیازی ندارد و براساس میزان تفاوت ویژگی های آماری آن با ویژگی های آماری عکس های طبیعی محاسبه میشود. مقدار 0 بهترین مقدار و مقادیر بیشتر نشان دهنده نویز بیشتر هستند.

معیار SSIM نیز به گونه ای محاسبه میشود که با ادراک بینایی انسان مطابقت داشته باشد و از فرمول زیر بدست می آید. بهترین مقدار آن برابر 1 است و هرچه بیشتر باشد بهتر است.

$$SSIM(x(i,j), \hat{x}(i,j)) = \frac{(2\mu_{x(i,j)}\mu_{\hat{x}(i,j)} + C_1)(2\sigma_{x(i,j)}\sigma_{\hat{x}(i,j)} + C_2)}{(\mu_{x(i,j)}^2 + \mu_{\hat{x}(i,j)}^2 + C_1)(\sigma_{x(i,j)}^2 + \sigma_{\hat{x}(i,j)}^2 + C_2)}$$

1.555515e+01 :niqe iso

6.460670e+00 :niqe aniso

3.474005e-01 :ssim iso

5.260259e-01 :ssim aniso

با توجه به اعداد بدست آمده، مقدار niqe برای aniso کمتر و مقدار ssim نیز برای aniso بیشتر است که نشان دهنده عملکرد بهتر فیلتر aniso است.

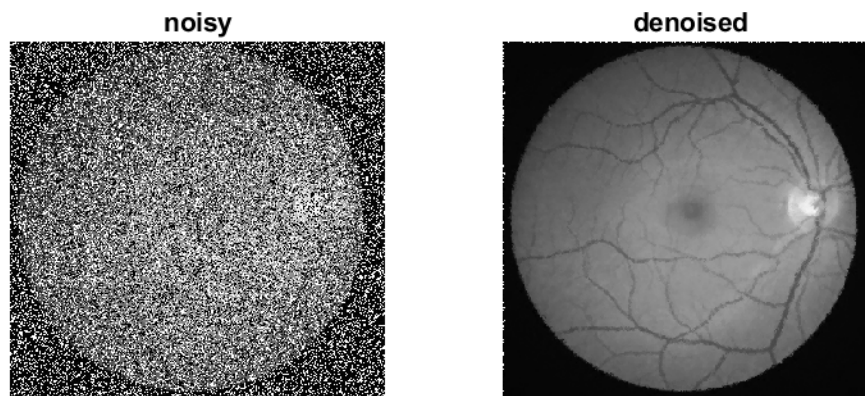
(ج)

تفاوت این فیلتر با سایر فیلتر ها این است که میتواند در هر راستا به میزان دلخواه smoothing داشته باشد که کمک میکند لبه ها را حفظ کند. همچنین این فیلتر به صورت local عمل میکند و تابعی از مشتقات در هر نقطه است و غیرخطی است.

3.

(الف)

تصویر را به نویز نمک و فلفل آلوده کرده و الگوریتم را پیاده سازی و بر روی تصویر نویزی اعمال میکنیم. واضح است که تصویر به مقدار خوبی بازسازی شده است.



(ب)

با توجه به اینکه مقدار میانه برابر 78 و مقدار کمینه برابر 66 است $S_{min} = y_i, j$ پس مقدار خروجی برابر میانه میشود که 78 است.

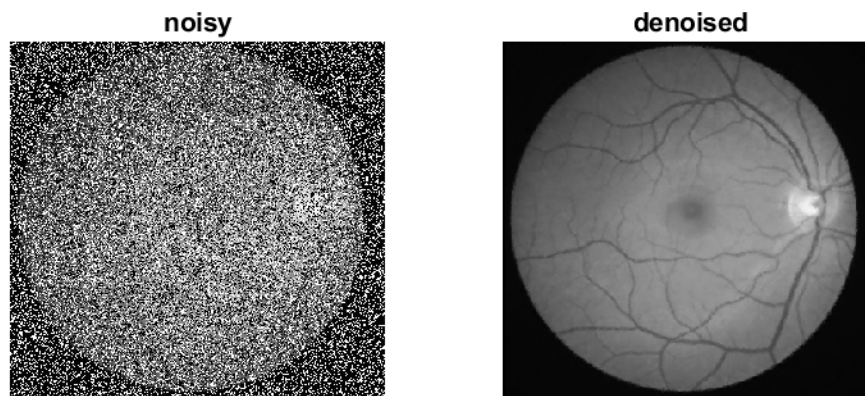
در مواقعی که پیکسل مرکزی مقداری برابر \max یا \min دارد ولی مربوط به نویز نمیباشد این الگوریتم دچار مشکل میشود.

(ج)

الگوریتم جدید فقط در صورتی پیکسل مرکزی را نویز در نظر میگیرد که هم در پنجره w مقدار \max یا \min داشته باشد و هم در پنجره با سایز $w+h$ مقدار \max یا \min داشته باشد که احتمال تشخیص اشتباه را کم میکند و در صورت نویز بودن پیکسل مرکزی آن را با میانگین تمام پیکسل های غیرنویزی جایگزین میکند.

(د)

با توجه به اینکه مقدار $psnr$ در الگوریتم از amf بیشتر و مقدار mse کمتر است، یعنی عملکرد الگوریتم بهتر بوده است.



2.497996e+01 :psnr amf

3.016171e+01 :psnr alg

3.176900e-03 :mse amf

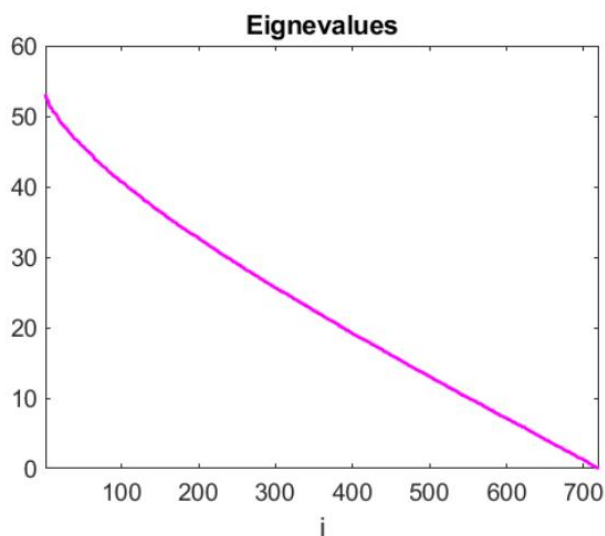
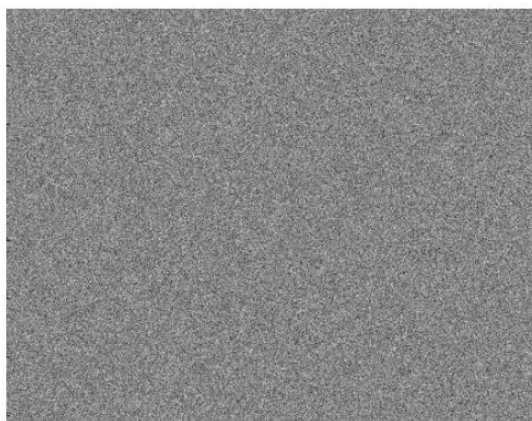
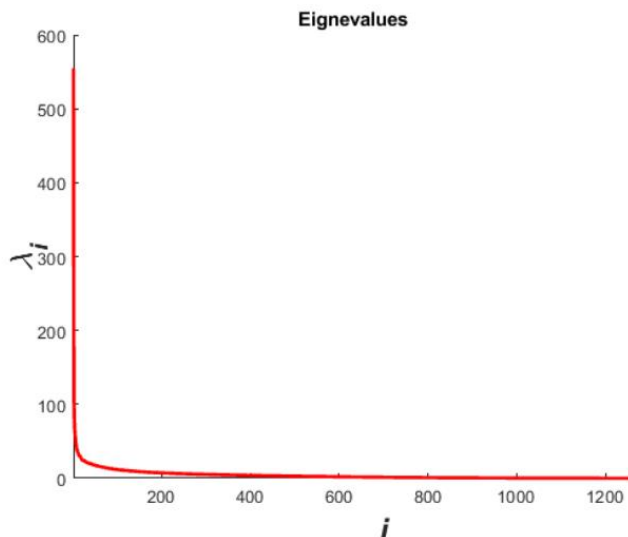
9.634496e-04 :mse alg

.4

-1

در روش low rank تصویر را به تعدادی patch تقسیم میکنیم و برای هر patch به دنبال patch های مشابه در تصویر میگردیم. سپس این پچ های مشابه را vectorize کرده و در یک ماتریس قرار میدهیم. اکنون تجزیه SVD ماتریس را محاسبه کرده و بهترین تقریب رnk k ماتریس را با نگه داشتن k مقدار تکین بزرگ بدست می آوریم. فرض بر این است که مقادیر تکین بزرگتر مربوط به ویژگی های تصویر و سایر مقادیر تکین مربوط به نویز هستند. میتوانیم مقادیر تکین واقعی را تخمین بزنیم. پس از بدست آوردن ماتریس جدید patch ها را به جای خود برمیگردانیم و در نهایت patch های همپوشان را aggregate میکنیم.

در تصاویر طبیعی تعداد مقادیر ویژه ای که مقدار بزرگی دارند نسبتاً کم است و بقیه تقریباً صفر هستند و اگر به ترتیب نزولی آن ها را رسم کنیم بعد از یک مقداری تقریباً صفر میشوند و سریع به صفر میل میکنند. اما در نویز همه مقادیر ویژه مقدار بزرگی دارند و به صفر میل نمیکنند و نمیتوان از آنها صرف نظر کرد.



-2

دو شیوه اصلی آستانه گذاری **soft thresholding** و **hard thresholding** هستند.

در **soft thresholding** جزئیات بیشتر حفظ میشوند و نسبت به نویز غیرایستا مقاوم تر است و نرم تر عمل میکند. در **hard thresholding** نویز به مقدار بیشتری حذف میشود، پیاده سازی راحت تری دارد و تنک تر است.

-3

در روش **WNNM** پس از استخراج **patch** های مشابه و کنار هم قرار دادن آن ها، تابع هزینه زیر را کمینه میکنیم. این تابع هزینه به گونه ای نوشته شده است که تا حد امکان خطای بازسازی را کم کند و تا حد امکان مقادیر تکین کوچک را از تصویر حذف کند. تفاوت اصلی **WNNM** با **low rank** این است که به مقادیر تکین وزن های متفاوتی در تابع هزینه

می‌دهیم و قصد داریم مقادیر تکین کوچک که مربوط به نویز هستند وزن بیشتری داشته باشند تا بیشتر حذف شوند و مقادیر تکین بزرگ که مربوط به ویژگی‌های اصلی تصویر هستند وزن کوچکی در تابع هزینه داشته باشند که حذف نشوند.

$$\widehat{\mathbf{X}}_j = \min_{\mathbf{X}_j} \left\{ \frac{1}{\sigma_n^2} \|\mathbf{X}_j - \mathbf{Y}_j\|_F^2 + \|\mathbf{X}_j\|_{w_*} \right\}$$

میتوان مقادیر تکین و وزن‌ها را تخمین زد و جواب بهینه تابع بالا را به صورت صریح بدست آورد و چندین بار روش را تکرار کرد.

$$\hat{\sigma}_i(\mathbf{X}_j) = \sqrt{\max(\sigma_i^2(\mathbf{Y}_j) - n\sigma_n^2, 0)}$$

$$w_i = \frac{c\sqrt{n}}{\sigma_i(\mathbf{X}_j) + \varepsilon}$$

Algorithm 1 Image Denoising by WNNM

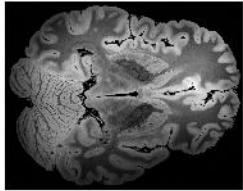
Input: Noisy image \mathbf{y}

- 1: Initialize $\hat{\mathbf{x}}^{(0)} = \mathbf{y}, \mathbf{y}^{(0)} = \mathbf{y}$
 - 2: **for** $k=1:K$ **do**
 - 3: Iterative regularization $\mathbf{y}^{(k)} = \hat{\mathbf{x}}^{(k-1)} + \delta(\mathbf{y} - \hat{\mathbf{y}}^{(k-1)})$
 - 4: **for** each patch \mathbf{y}_j in $\mathbf{y}^{(k)}$ **do**
 - 5: Find similar patch group \mathbf{Y}_j
 - 6: Estimate weight vector \mathbf{w}
 - 7: Singular value decomposition $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \text{SVD}(\mathbf{Y}_j)$
 - 8: Get the estimation: $\hat{\mathbf{X}}_j = \mathbf{U}\mathbf{S}_{\mathbf{w}}(\mathbf{\Sigma})\mathbf{V}^T$
 - 9: **end for**
 - 10: Aggregate $\hat{\mathbf{X}}_j$ to form the clean image $\hat{\mathbf{x}}^{(k)}$
 - 11: **end for**
- Output:** Clean image $\hat{\mathbf{x}}^{(K)}$
-

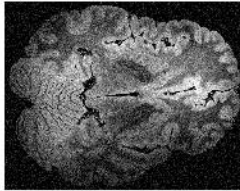
-4

شبه کد بالا را پیاده سازی میکنیم و برای تخمین مقادیر تکین و وزن ها از رابطه های بالا استفاده میکنیم. برای پیدا کردن پچ های مشابه، یک پنجره لغزان با ابعاد پچ در نظر میگیریم و در تمام تصویر فاصله اقلیدسی را محاسبه میکنیم و 10 پچ با کمترین فاصله را به عنوان تصاویر مشابه در نظر میگیریم.

original



noisy



denoised

