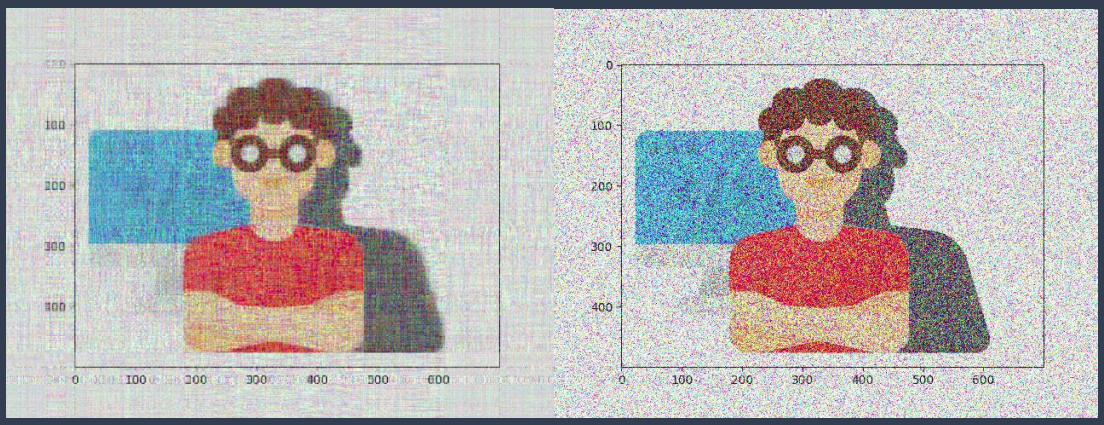
مقدمه و توضیح الگوریتم

در این مینی پروژه قصد داریم با توجه به دانشی که از جبر خطی کسب کرده‌ایم، به کاهش نویز از داده بپردازیم.



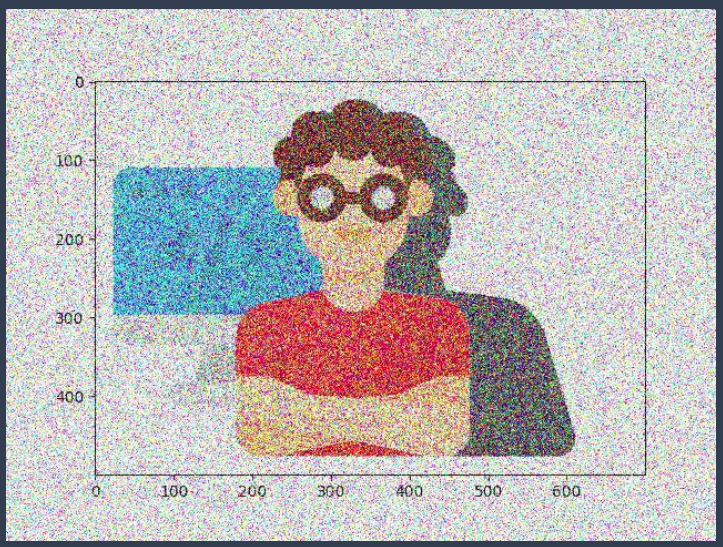
تصویر با نویز زیاد تصویر با نویز کمتر

یکی از مباحث پرکاربردی که با آن آشنا شدیم مبحث **SVD** است .

$$A = U \Sigma V^T$$

یکی از این کاربردهای **SVD** حذف کردن نویز از تصویر (سیگنال)های دریافتی است .

فرض کنید ما عکس زیر را دریافت کرده‌ایم .



در بسیاری از موارد (همان‌طور که شاید در درس مدار الکتریکی مشاهده کرده باشید) خروجی مورد انتظار از یک مدار دچار نویز یا خطا می‌شود. عکس بالا نیز یک عکس حاوی نویز می‌باشد . ما می‌خواهیم به کمک آنچه در جبر خطی یاد گرفته‌ایم به کم کردن این نویزها بپردازیم.

فرض کنید ماتریس عکس ورودی برابر **Noisy_matrix** باشد و از طرفی ماتریسی اصلی مورد نظر ما **Main_matrix** باشد .

می توان چنین تصور کرد که **Noisy_matrix = Main_matrix + Noises** است که **Noises** یک ماتریس با ابعاد ماتریس اصلی است که نویزهای تصویر (همانند **Gaussian Noise**) را نشان می دهد.

قرار است با داشتن **Noisy_matrix** ماتریس **CLEANED** را بدست آوریم که تا جایی ممکن نزدیک به ماتریس اصلی (**Main_matrix**) ما باشد .

اگر تجزیه **SVD** را برای ماتریس **Noisy_matrix , Main_matrix** انجام دهیم :

$$\mathbf{Noisy\_matrix = USV^T}$$

$$S = \begin{bmatrix} S1 & & & & & & \\ & S2 & & & & & \\ & & S3 & & & & \\ & & & \ldots & & & \\ & & & & Sk & & \\ & & & & & Sk+1 & \\ & & & & & & \ddots \end{bmatrix}$$

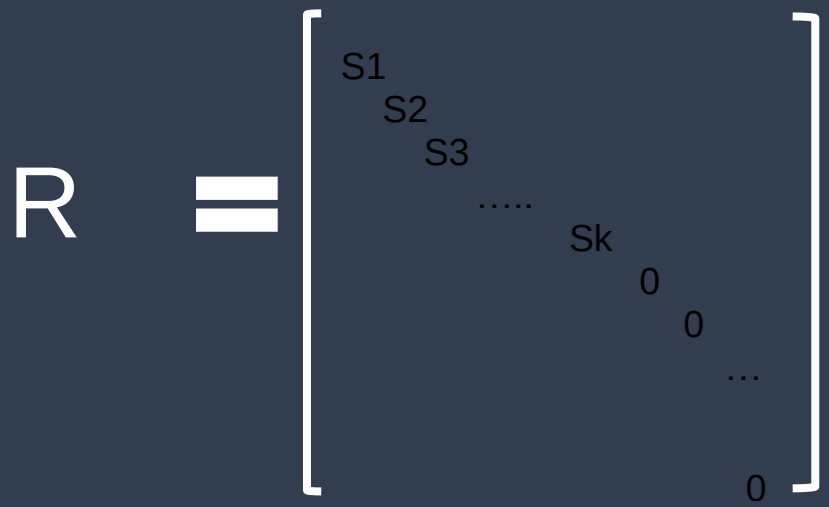
(دقت کنید که در ماتریس **S** داریم **s1>s2>s3>...>sr**)

متوجه می‌شویم که از $\mathbf{t = S_K}$ به بعد مقدار تکین های موجود در ماتریس قطری **S** تاثیر زیادی بر روی ایجاد نویز دارند و مقدار تکین های بزرگتر از **t** بیشتر بر روی **main_matrix** موثر هستند پس با در نظر نگرفتن مقدار تکین های **k** به بعد می‌توان تصویری با نویز کمتر و شباهت‌های بسیار زیاد با تصویر اصلی ایجاد کرد.

همچنین با توجه به اینکه اعداد آن‌ها بسیار کوچکتر از بقیه اعداد است پس اگر به جای آنها صفر بگذاریم عکس آنقدر تغییر نمیکند اما داده هایی که نویز روی آن تاثیر بسیاری گذاشته است حذف می‌شوند . پس اگر یک کات اف مانند **t** در نظر بگیریم و ماتریس **R** اینگونه تعریف کنیم که

```
for all i, j:
    if i == j:
        if Si > t: # t is our threshold
            Rii = Si
        else:
            Rii = 0
```

آنگاه ماتریس تمیز شده ما برابر است با **U *R * V^T = cleared**



- حال داریم :

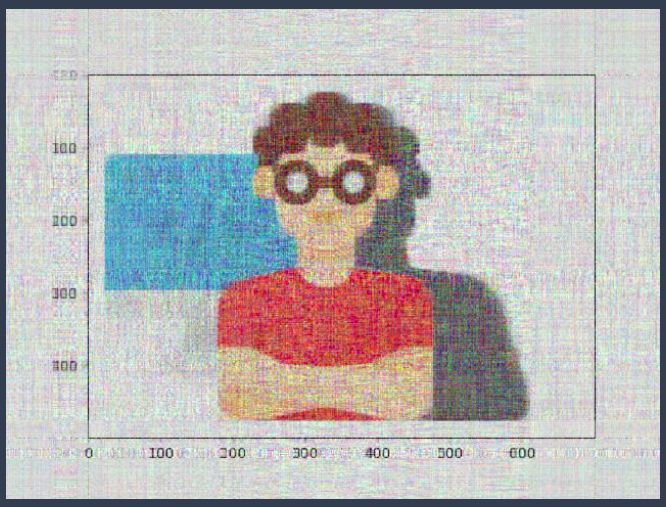
Noisy_matrix = u0s0v0 + u1s1v2 +...+ ukskvk +...+ ursrvr

cleared_matrix = u0s0v0 + u2s2v2 + .. + ukskvk

for all i < k : si >t

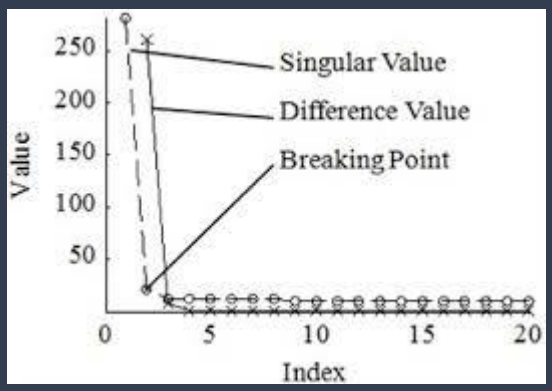
نتیجه کار برابر است با:

(هنوز با عکس اصلی تفاوت زیادی موجود است اما نتیجه حذف کردن نویزها به وضوح دیده می‌شود)



دقت کنید که پیدا کردن **k** همیشه کار آسانی نیست و بعضا نیازمند محاسبات ریاضی به خصوصی است که از درس جبر خطی خارج است . ما در این پروژه برای انتخاب **k** به صورت حدس و آزمایش عمل می‌کنیم و مقادیر مختلف و نتایج آن را بررسی می‌کنیم. باید به این مسئله توجه کرد که انتخاب **K** بسیار مهم است زیرا اگر **K** بزرگتر از حد استاندارد باشد از نویز تصویر به طور ملموسی کاسته نمی‌شود و اگر **K** کوچک تر از حد استاندارد باشد آنگاه ما بر روی عکس‌های خود فشرده سازی انجام دادیم که باعث کاهش کیفیت عکس ما می‌شود و ممکن است عکس بدست آمده کارایی مد نظر ما را نداشته باشد.

در اکثر توابعی که باعث ایجاد نویز روی عکس ما می‌شود در یک نقطه مشخص (**breaking point**) اختلاف زیادی بین مقدار تکین ها ایجاد می‌شود.



پیاده سازی

پیاده سازی قسمت اول :

```
import matplotlib.pyplot as plt

img = plt.imread(img.jpg')  #read picture from path
```

تصویر موجود در فایل **ZIP** پیام‌های کلاس **quera** را بخوانید . دقت کنید **img** سه ماتریس **r ,g, b** را شامل می‌شود .

سپس عملیات بالا را برای سه ماتریس انجام دهید و بهترین **k** (یا **threshold**) را پیدا کنید.

برای تجزیه **svd** میتوانید از تابع زیر استفاده کنید .

```
U, S, V = np.linalg.svd(your_matrix)
```

سپس بر اساس **K** انتخاب شده یک ماتریس جدید از روی ماتریس تصویر اولیه(با نویز زیاد) بسازید و در آخر نتیجه را در فایل **jpeg** ذخیره کنید.

```
plt.imsave('pa th.jpeg', cleaned_matrix)
```

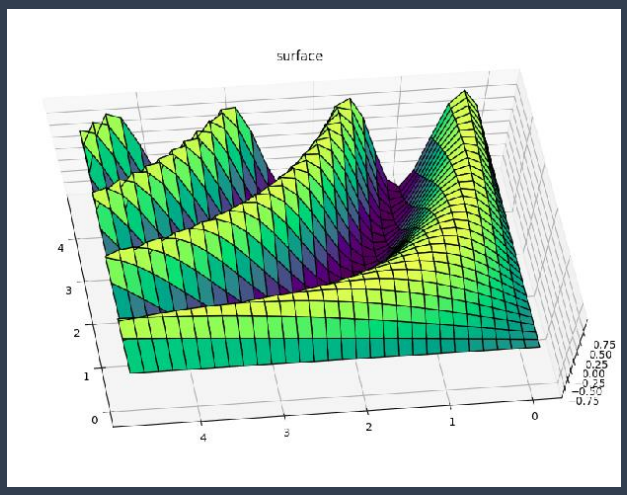
دقت شود کافیست مقداری از نویز کاسته شود و لازم نیست همه نویز موجود در تصویر گرفته شود

قسمت دوم(امتیازی) :

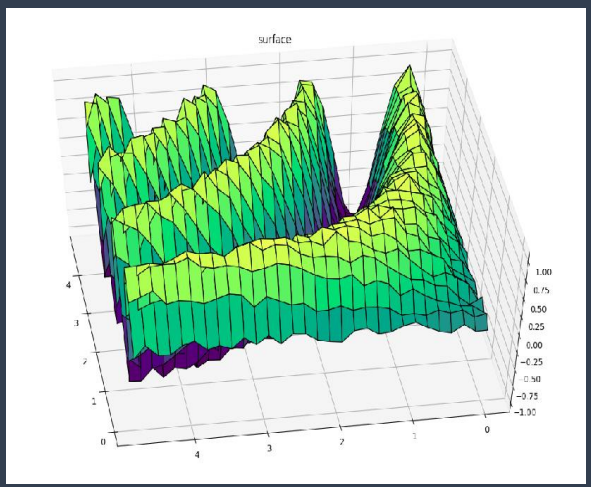
برای شهود بهتر شما در کنار فایل متن یه فایل **utility.py** قرار داده شده است .

در این فایل به کمک **matplot** تابع **Z=X*Y/2** برای برخی نقاط رسم شده است .

با فراخوانی تابع **show_main**() تابع اصلی نمایش داده می‌شود .



سپس تابع **show_noisy**() را فراخوانی کنید. این بار تابع برای همان نقاط رسم شده است ولی با این تفاوت که برای هر نقطه **Z=X*Y/2 + noise** است که **noise** مقدار خطایی وارد شده است.



با فراخوانی تابع

z=get_function()

یک ماتریس برگردانده می‌شود که نتایج این تابع را بر حسب نقاط

Y,X=k*pi*3/60 for all int k in 0 to 30 :[0 , pi*3/60 , pi*3*2/60 , pi *3*3/60 , ….,pi*3/60]

در مجموع ۳۰ نقطه **X** و ۳۰ نقطه **Y**

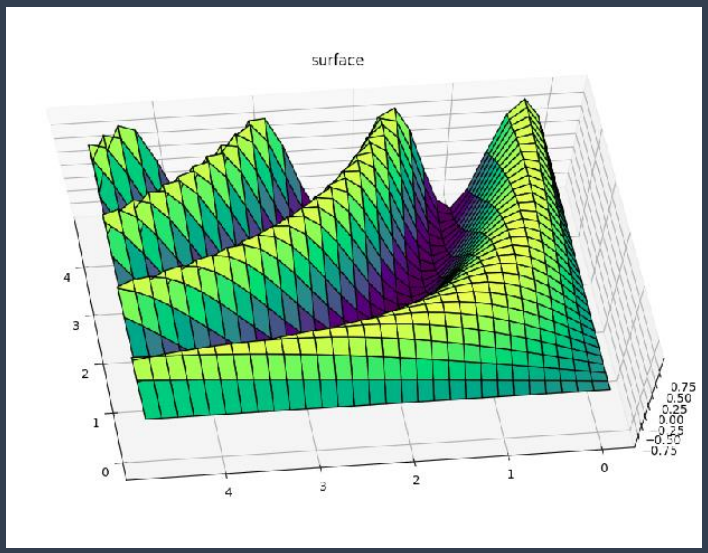
حساب شده است .

با استفاده از کدی که برای قسمت اول زده‌اید و با کمی تغییر تلاش کنید ماتریس **z** را **denoise** کنید و سپس نتیجه را برای تابع

show_my_matrix(cleaned_matrix)

بفرستید تا ماتریس **denoised** شده را به نمایش بگذارد. بهترین **k** را برای این کار بدست آورید و تفاوت در سه ماتریس را مقایسه کنید.

نتیجه **denoised** کردن ماتریس آخر باید نزدیک به تصویر زیر باشد



* دقت کنید که نیازی به نوشتن تابع محاسبه گر svd یا ضرب ماتریس ها ندارید و میتوانید از توابع آماده numpy که در بالا اشاره شده استفاده کنید.

برای علاقه مندان :

۱. تابع get_main_function در فایل utility.py اصل تابع z=f(x,y) را محاسبه می‌کند . این تابع را تغییر داده و برای توابع مختلف نیز مراحل را طی کنید .

۲.مراحل طی شده متناظر با مراحل طی کردن کاهش حجم تصویر می‌باشد . اما چرا این مراحل یکسان هستند؟ در ماتریسی که دچار خطا شده می‌توان اینطور در نظر گرفت که یک دیتا اضافی که همان ماتریس noise می‌باشد به ماتریس اصلی اضافه شده است و با طی کردن این مراحل به نوعی داریم داده اضافی را از ماتریس اصلی کم می‌کنیم .

قوانین

1. مینی پروژه به صورت انفرادی می باشد. تقلب ها توسط سامانه کوئرا چک می شود اما تصحیح توسط تیم تدریسیاری صورت می‌گیرد.
2. برای باز کردن عکس‌ها و خروجی دادن آن ها میتوانید از هر کتابخانه ای نظیر **matplotlib** یا **pillow** استفاده کنید . پیشنهاد می‌شود طبق دستور عمل کنید.
3. در **quera** برای هر قسمت(بخش اجباری و امتیازی) یک تمرین جدا ایجاد شده است فایل مربوطه را در قسمت اختصاص یافته آپلود کنید.
4. برای هر دو فایل، فرمت فایل ارسالی باید به صورت **miniproject2_STUDENTNUMBER_Name** باشد. به طور مثال: **miniproject2_9628099_SoroushMehraban**

موفق باشید
تیم تدریس یاری جبر خطی کاربردی
دی 99