به نام حضرت دوست

تمرینات سری پنجم – فصل هفتم و نهم

لطفا تمرینات خود را خوانا و در قالب HW?_name_stdnumber.pdf بنویسید و تا قبل از موعد تحویل بارگذاری نمایید.
(نمونه HW5_Ross Geller_9631057.pdf)

دقت کنید که سوال پیاده سازی امتیازی می باشد و میتوانید با پایتون یا متلب آن را نوشته و به همراه فایل PDF در قالب فایل زیپ با فرمت HW?_name_stdnumber.zip بفرستید. زمان تحویل تمرین ها تا ساعت 24 روز جمعه 13 تیر می باشد.

درصورت داشتن هرگونه ابهام در سوال ، به ایمیل linalgebra.spring2020@gmail.com پیام دهید.

1- ماتریس زیر را قطری سازی عمودی(orthogonally diagonalize) نمایید.

$$A = \begin{bmatrix} 3 & 1 & 1 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \end{bmatrix}.$$

2- فرض کنید ماتریس متقارن B دارای مقدار ویژه های λ_1 و λ_2 باشد.(که $\lambda_1 \neq \lambda_2$) و بردار ویژه های متناظر با این دو مقدار ویژه نیز، v_1 و v_2 باشد.
ثابت کنید که این دو بردار ویژه بر هم عمودند؛ یعنی:

$$v_1 . v_2 = 0$$

3- فرم درجه دوم(Quadratic Form) موارد زیر را بیابید.

$$10x_1^2 - 6x_1x_2 - 3x_2^2$$

$$20x_1^2 + 15x_1x_2 - 10x_2^2$$

$$5x_1^2 + 3x_1x_2$$

4- ماکزیمم مقدار $Q(\mathbf{x}) = -3x_1^2 + 5x_2^2 - 2x_1x_2,$ را با فرض $x_1^2 + x_2^2 = 1.$ پیدا کنید.

5- تجزیه SVD را برای ماتریس زیر محاسبه کنید.

$$\begin{bmatrix} 4 & -2 \\ 2 & -1 \\ 0 & 0 \end{bmatrix}$$

6- عبارات زیر را درصورت امکان نسبت به شرط های داده شده بهینه کنید.

الف) ماکزیمم: $80x_1 + 65x_2$

نسبت به: $2x_1 + x_2 \leq 32,\ x_1 + x_2 \leq 18,\ x_1 + 3x_2 \leq 24,$
$x_1 \geq 0\ , x_2 \geq 0$

ب) ماکزیمم: $2x_1 + 7x_2$

نسبت به: $-2x_1 + x_2 \leq 4,\ x_1 - 2x_2 \leq 4, x_1 \geq 0\ , x_2 \geq 0$

7- دوگان مسائل برنامه نویسی خطی زیر را بنویسید.

$$\begin{aligned} \text{Maximize} \quad & 21x_1 + 25x_2 + 15x_3 \\ \text{subject to} \quad & 2x_1 + 7x_2 + 10x_3 \leq 20 \\ & 3x_1 + 4x_2 + 18x_3 \leq 25 \end{aligned}$$

and $x_1 \geq 0, x_2 \geq 0, x_3 \geq 0.$

پیاده‌سازی (امتیازی): فشرده سازی عکس با تجزیه SVD

یکی از کاربردهای تجزیه‌ی SVD، کاهش حجم داده‌ها یا فشرده‌سازی اطلاعات است. در پیاده‌سازی این تمرین قصد داریم از تجزیه‌ی SVD برای فشرده‌سازی عکس استفاده کنیم. روش‌های مختلفی برای فشرده‌سازی عکس‌ها وجود دارد که فرمت‌هایی نظیر JPG،PNG و... از آن‌ها بهره می‌برند. در این پیاده‌سازی می‌خواهیم با کمک تجزیه‌ی SVD، فایل‌های خام عکس را فشرده‌سازی کنیم.

فرمت PPM فرمت‌ای برای فایل‌های عکس است که در آن هیچ‌گونه فشرده‌سازی صورت نمی‌گیرد و داده‌ها به صورت کاملا خام ذخیره می‌شود. فایل‌های PPM مربوط به تصاویر RGB، تشکیل شده است از ۳ آرایه‌ی دوبعدی که هر کدام از آرایه ها، مربوط به یک کانالِ رنگی است و در آن مقادیر عددی مربوط به هر پیکسل (به صورت integer یک‌بایتی، از ۰ تا ۲۵۵)، نوشته شده است.

مراحل کار:

۱- ابتدا از آدرس PPM Images یکی از فایل‌های عکس را دریافت نمایید و آن را در کد خود باز کرده و در یک آرایه‌ی ۳ بعدی ذخیره کنید. در پایتون به شکل زیر:

```
import matplotlib.pyplot as plt
img = plt.imread('image.ppm')
```

۲- عکس را پلات کنید:

```
plt.imshow(img)
plt.show()
```

(نیاز است کتابخانه‌ی Pillow را نیز از کامند لاین دریافت کنید: pip install Pillow)

۳- کانال‌های rgb را از هم تفکیک کنید و هر یک را در ماتریسی جداگانه ذخیره کنید:

```
r = img[:, :, 0]
g = img[:, :, 1]
b = img[:, :, 2]
```

۴- حال باید برای ۳ ماتریس بالا، به طور جداگانه تجزیه‌ی SVD را بدست آورید. برای این‌کار از توابع کتابخانه‌ای استفاده کنید (در پایتون تابع numpy.linalg.svd).

حاصل تجزیه SVD به شکل زیر است:

$$\mathbf{A} = \begin{bmatrix} \mathbf{u_1} & \mathbf{u_2} & \dots & \mathbf{u_m} \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & \sigma_r & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v_1}^T \\ \mathbf{v_2}^T \\ \vdots \\ \mathbf{v_n}^T \end{bmatrix}$$

می‌دانیم که عبارت بالا را می‌توان به شکل زیر بازنویسی کرد:

$$\mathbf{A} = \begin{bmatrix} \mathbf{u_1} & \mathbf{u_2} & \dots & \mathbf{u_m} \end{bmatrix} \begin{bmatrix} \sigma_1 \mathbf{v_1}^T \\ \sigma_2 \mathbf{v_2}^T \\ \vdots \\ \sigma_r \mathbf{v_r}^T \\ 0 \\ \vdots \\ 0 \end{bmatrix} \} \text{ m rows}$$

$$= \sigma_1 \mathbf{u_1} \mathbf{v_1}^T + \sigma_2 \mathbf{u_2} \mathbf{v_2}^T + \dots + \sigma_r \mathbf{u_r} \mathbf{v_r}^T$$

با توجه به اینکه مقادیر تکین، به صورت نزولی در قطر ماتریسِ Sigma مرتب شده اند، پس در عبارت آخر، تاثیر جملات ابتدایی بیشتر از جملات بعدی است. در نتیجه می‌توانیم تنها با در نظر گرفتن k جمله اول، تخمین بسیار خوبی از ماتریس‌مان داشته باشیم؛ یعنی:

$$\mathbf{A_k} = \begin{bmatrix} \mathbf{u_1} & \mathbf{u_2} & \dots & \mathbf{u_k} \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & \sigma_k \end{bmatrix} \begin{bmatrix} \mathbf{v_1}^T \\ \mathbf{v_2}^T \\ \vdots \\ \mathbf{v_k}^T \end{bmatrix}$$

$$= \sigma_1 \mathbf{u_1} \mathbf{v_1}^T + \sigma_2 \mathbf{u_2} \mathbf{v_2}^T + \dots + \sigma_k \mathbf{u_k} \mathbf{v_k}^T$$

۵- با توجه به نکته‌ی بالا، k = 50 جمله‌ی اول عبارت بالا را محاسبه کنید و جمع بزنید. حاصلِ این مجموع، تخمینی است برای ماتریس اصلی.

۶- سپس ۳ ماتریس بدست آمده در هر کانالِ رنگی را، با هم ادغام کنید تا ماتریس مربوط به عکسِ کامل تشکیل شود.

۷- حال، تصویر بدست آمده در مرحله‌ی قبل را پلات کنید و وضوح آن را با تصویر اصلی مقایسه کنید.

۸- فرآیند بالا را برای k = 100 و k = 150 نیز طی کنید. مشاهده خواهید کرد که با افزایش مقدار k، وضوح تصویر بهتر خواهد شد.

نکته بسیار مهم: اگر دقت کنید، ماتریس تصویر حاصل از فرآیندی که طی کردید، هم‌اندازه با ماتریس تصویر اولیه است. پس به نظر می‌رسد که اصلا چیزی به نام فشرده‌سازی صورت نگرفته است و صرفا کیفیت عکس را کاهش داده‌ایم. نکته اینجاست که ما نیازی به ذخیره‌ی تصویر نهایی نداریم، بلکه کافیست تنها ستون‌هایی از U، مقادیر تکین و سطرهایی از ماتریس V^T که مربوط به k جمله ابتدایی بسط SVD است را در فایل ذخیره کنیم و صرفا در هنگام باز کردن عکس، از طریق بسط SVD، عکس را بازتولید (reconstruct) کنیم. این سطر و ستون‌ها و مقادیر تکین‌ای که نیاز است ذخیره کنیم، به مراتب کم‌حجم‌تر از داده‌های عکس است؛ اما قادر است عکس اصلی را با وضوح بسیار خوبی بازتولید کند!

مثلا اگر عکس ورودی 1920 در 1080 باشد، در مجموع 6,220,800 درایه‌ی ماتریس در این عکس داریم. حال اگر تا جمله‌ی ۱۰۰ ام بسط SVD را در نظر بگیریم، باید ۱۰۰ ستون اول ماتریس U، و ۱۰۰ مقدار تکین و همچنین ۱۰۰ سطر اول ماتریس V^T مربوط به هر کانالِ رنگی را ذخیره کنیم، که می‌شود:

3 * (100 * 1920 + 100 + 100 * 1080) = 900,300

که حدود 14% درایه‌های تصویر اصلی است! اما همین میزان داده، قادر است عکس‌مان را با وضوح بسیار خوبی بازتولید کند.

پی‌نوشت :نیازی به آپلود کردن عکس‌ها نیست و فقط کد خود را آپلود نمایید.