



دانشکده‌ی مهندسی برق

مدرس: دکتر یاسایی

بهینه‌سازی محدب ۱

تمرین کامپیوتری سری اول

شماره دانشجویی: ۹۸۱۰۰۲۱۵

نام و نام خانوادگی: خشايار غفارى

پرسش ۱ مقادیر ویژه و بردارهای ویژه

۱. الگوریتم QR برای پیدا کردن تقریبی مقادیر ویژه:

در این الگوریتم هدف محاسبه‌ی تقریبی مقادیر ویژه‌ی ماتریس A می‌باشد.

ابتدا ماتریس $A^{(1)}$ را برابر A تعریف می‌کنیم و سپس برای هر $i > i$, ماتریس $A^{(i)}$ را برابر ماتریس $R^{(i-1)}Q^{(i-1)}$ تعریف می‌کنیم که در آن $(i-1)$ ماتریس متعامد یک‌های است که در تجزیه QR ماتریس $A^{(i-1)}$ ظاهر می‌شود و $R^{(i-1)}$ ماتریس بالامثلی همان تجزیه می‌باشد.

در این فرآیند به راحتی می‌توان دید که مقادیر ویژه‌ی $A^{(i)}$ ‌ها ثابت می‌ماند زیرا به دلیل متعامد یکه بودن ماتریس‌های Q , ماتریس‌های ساخته شده با یکدیگر متشابه هستند و طبق قضایای جبرخطی مقدماتی، می‌دانیم که مقادیر ویژه‌ی ماتریس‌های متشابه با هم برابرند.

حال اگر شرایط مناسبی که در قسمت ۳ به آن اشاره می‌کنیم برقرار باشند، با انجام تعداد زیاد این عملیات، این دنباله‌ی ماتریسی به یک ماتریس بالامثلی میل می‌کند و می‌دانیم مقادیر ویژه‌ی ماتریس‌های بالامثلی همان درآیه‌های روی قطر اصلی می‌باشد. پس درآیه‌های روی قطر اصلی این ماتریس همان مقادیر ویژه‌ی ماتریس A می‌باشند.

اکنون که مقادیر ویژه را محاسبه کردیم برای به دست آوردن بردارهای ویژه از این نکته استفاده می‌کنیم که بردارهای ویژه‌ی متناظر با مقدار ویژه‌ی λ_i , پایه‌ی فضای پوچ ماتریس $A - \lambda_i I_n$ می‌باشند. این پایه نیز همان SVD ماتریس V که از تجزیه‌ی $null(A - \lambda_i I_n)$ ستون آخر ماتریس A به دست می‌آیند، می‌باشند.

۲. برای پیاده‌سازی این الگوریتم ابتدا تابع triangulation را تعریف کردیم که با ورودی گرفتن یک ماتریس و تعداد تکرارهای فرآیند، الگوریتم مطرح شده را به آن تعداد انجام می‌دهد تا به یک ماتریس بالا مثالی برسد. سپس ماتریسی که تابع قبل خروجی می‌دهد را درآیه‌های روی قطر اصلی اش را مرتب می‌کنیم و به عنوان مقادیر ویژه خروجی می‌دهیم.

برای بردار ویژه‌ها نیز همانطور که در بخش قبل بیان کردیم عمل می‌کنیم و همچنین با توجه به توضیحاتی که در بخش ۳ دادیم در اینجا فرض می‌کنیم که مقادیر ویژه همگی متمایز هستند و لذا پوچی ماتریسی که به دنبال فضای پوچش می‌گردیم همیشه ۱ است و همیشه ستون آخر ماتریس مذکور را به عنوان بردار ویژه در نظر می‌گیریم.

۳. محدودیت‌های الگوریتم QR :

- اگر ماتریس A دارای دو مقدار ویژه مانند λ_i و λ_j باشد به طوری که $|\lambda_j| = |\lambda_i|$ ، آن‌گاه الگوریتم درست کار نمی‌کند. زیرا ادعایی که در رابطه با صفر شدن درآیه‌های زیر قطر اصلی ماتریس داشتیم در این شرایط درست نیست.
درواقع سرعت صفر شدن هر درآیه زیر قطر برابر $|\lambda_i/\lambda_j|$ است که در آن، مقادیر ویژه به ترتیب نزولی اندازه‌هایشان مرتب شده‌اند.

الگوریتم‌های دیگر:

- نسبت ریلی: در این الگوریتم ابتدا بردار دلخواه مانند b و یک عدد دلخواه مانند μ انتخاب می‌کنیم و بردارها و اعداد بعدی را بدین شکل تعریف می‌کنیم:

$$b_{k+1} = \frac{(A - \mu_k I_n)^{-1} b_k}{\|(A - \mu_k I_n)^{-1} b_k\|} \quad \mu_{k+1} = \frac{b_{k+1}^H A b_{k+1}}{\|b_{k+1}\|^2}$$

با تکرار کمی از این فرآیند می‌توانیم با تقریب خوبی یک مقدار ویژه و بردار متناظر را محاسبه کنیم. توجه کنید که ماتریسی که از وارون آن استفاده کردیم فقط زمانی وارون پذیر نیست که μ_k مقدار ویژه‌ی ماتریس A باشد.
محدودیت‌های الگوریتم نسبت ریلی:

- این الگوریتم فقط برای ماتریس‌های هرمیتی کار می‌کند.
- همچنین محاسبه‌ی وارون ماتریس مذکور در الگوریتم می‌تواند زمان زیادی ببرد.
- پاور متدهای در الگوریتم پاور متدهای در کلاس درس نیز مطرح شد. با انتخاب یک بردار و تکرار یک عملیات روی آن به بردار ویژه‌ی متناظر با مقدار ویژه‌ی اول میل می‌کنیم.
محدودیت‌های الگوریتم پاور متدهای:
- در صورتی که دو مقدار ویژه‌ی بزرگ ماتریس به هم نزدیک باشند این الگوریتم نیازمند تعداد تکرار خیلی بیشتری می‌باشد.

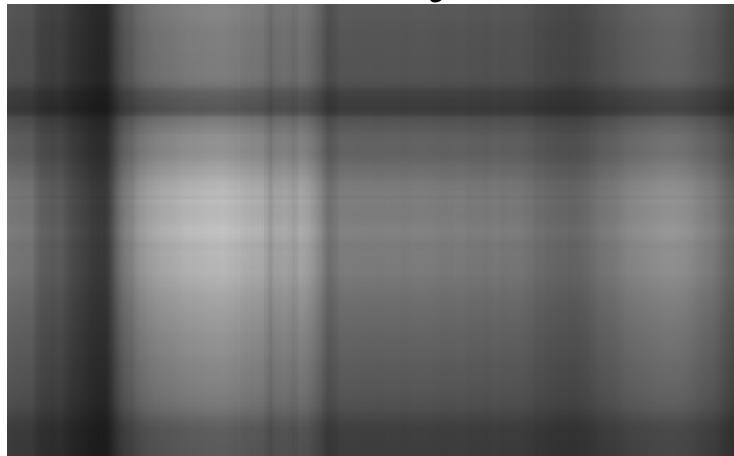
پرسش ۲ SVD و پردازش تصویر

۱. در این بخش ابتدا به بررسی چند خروجی این الگوریتم می‌پردازیم.

شکل ۱: عکس اصلی



شکل ۱:۲ $rank = 1 : 2$



$rank = 6 : 3$ شکل



$rank = 15 : 4$ شکل



$rank = 40 : 5$ شکل



تمرین کامپیوتری سری اول - ۴

شكل ٦ : $rank = 70$



شكل ٧ : $rank = 150$



شكل ٨ : $rank = 500$



تمرين كامپيوترى سرى اول-٥

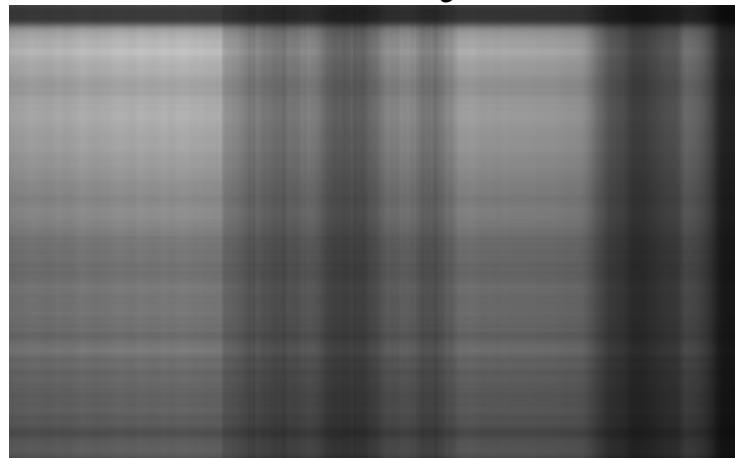
در این سوال از الگوریتم *low rank approximation* استفاده کردیم که در سوال ۷ تمرین سری اول مطرح شده بود.

در پیاده‌سازی این الگوریتم در پایتون ابتدا تابعی به نام *image-to-array* تعریف شده که اسم فایل عکس را به عنوان ورودی می‌گیرد و عکس را به صورت سیاه و سفید در یک آرایه خروجی می‌دهد.
تابع low-rank هم به عنوان ورودی اجزای تجزیه‌ی *SVD* ماتریس مربوط یک عکس دریافت می‌کند و ماتریس با رتبه‌ی کمتر که در ورودی تابع دریافت کرده را خروجی می‌دهد.
بررسی خروجی‌های مختلف: همانطور که در عکس‌ها می‌بینیم با رتبه‌ی ۴۰ توانستیم تا حد خوبی ذخیره کنیم و با رتبه‌ی ۷۰ به جزئیات بسیار بیشتری هم رسیدیم و به نظر می‌آید از رتبه‌ی ۷۰ به بعد تغییر ملموسی را در عکس‌ها نمی‌بینیم.
خروجی‌های یک عکس دیگر:

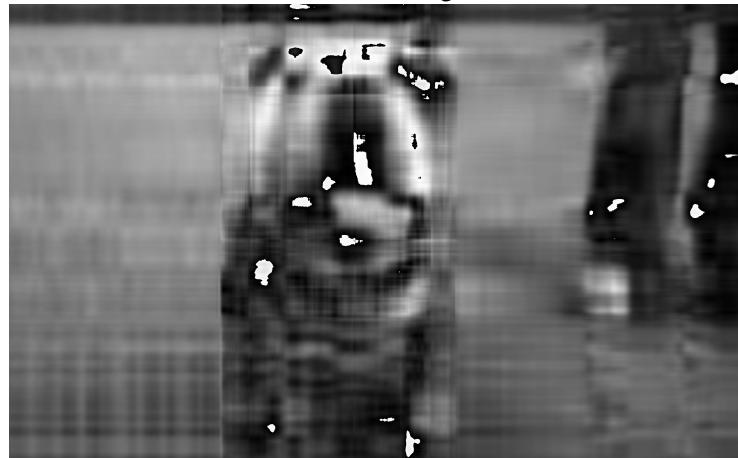
شکل ۹: عکس اصلی



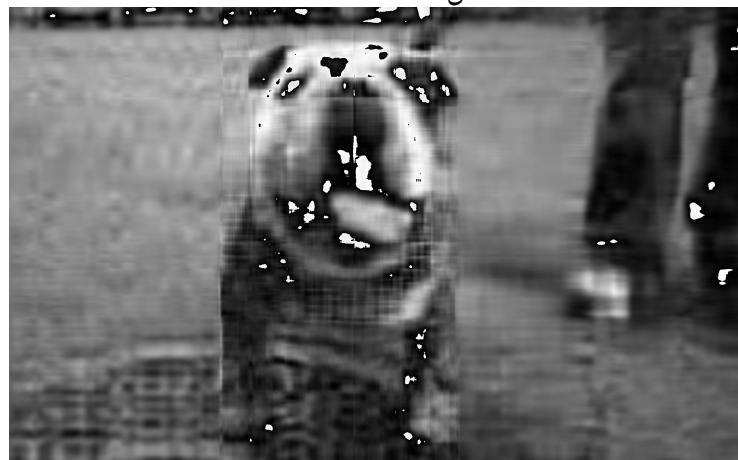
شکل ۱۰: $rank = 1 : 10$



شكل ۱۱ : $rank = 8$



شكل ۱۲ : $rank = 15$



شكل ۱۳ : $rank = 30$



rank = ٦٠ : ١٤



شكل ١٥ : ١٠٠



شكل ١٦ : ١٧٠



تمرين كامپيوترى سرى اول-٨

rank = ٢٦٠ : شكل



rank = ٣٠٠ : شكل

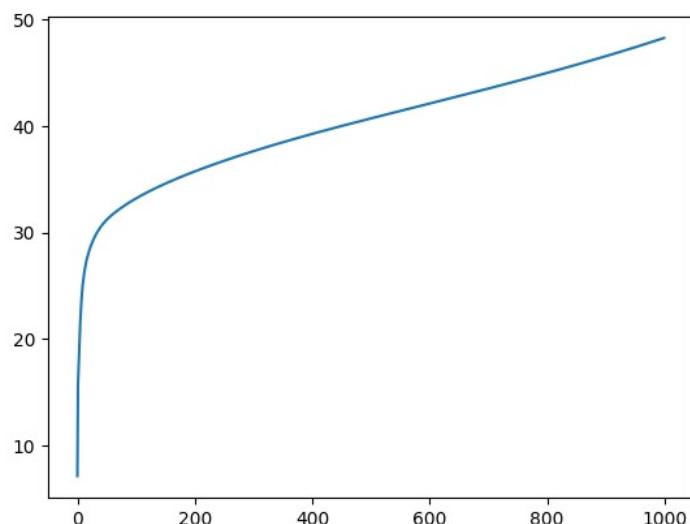


rank = ٤٠٠ : شكل

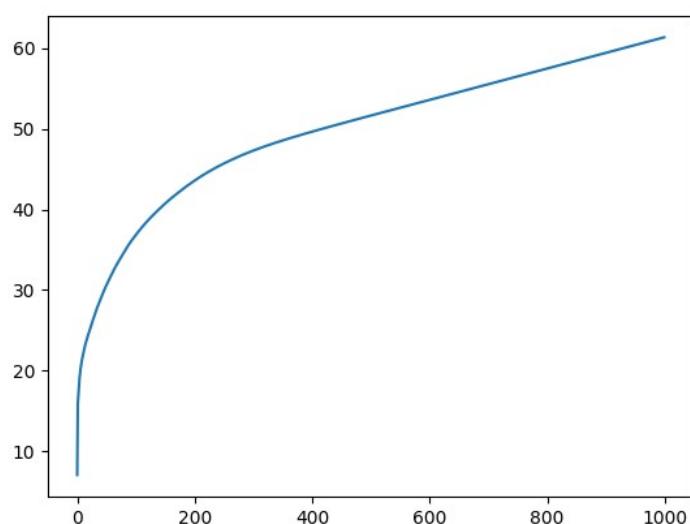


در بررسی این عکس می‌بینیم که تا رتبه‌ی ۶۰ تا حد خوبی به تصویر اصلی نزدیک شدیم اما در صورت سگ نویزهای زیادی داریم. با ادامه‌ی این فرآیند می‌بینیم که نویز داخل دهان سگ تا رتبه‌ی ۱۷۰ هم نسبتاً زیاد است اما بعد از آن بهتر شده و در رتبه‌ی ۴۰۰ دیگر به سختی می‌توان نویز دید.

شکل ۲۰: نمودار میزان PSNR بر حسب زیادشدن رتبه‌ی ماتریس عکس پرنده



شکل ۲۱: نمودار میزان PSNR بر حسب زیادشدن رتبه‌ی ماتریس عکس سگ



با توجه به نمودار می‌توان دید که عکس‌های خروجی نیز منطبق با انتظارمان از نمودار هستند چرا که در نمودار عکس اول می‌بینیم که بعد از حدود رتبه‌ی ۷۰ شب نمودار خیلی کمتر شده پس انتخاب عدد ۷۰ برای ما مناسب است اما برای عکس دوم این اتفاق در حدود رتبه‌ی ۱۸۰ افتاده که با عکس‌هایی که بررسی کردیم هم مطابقت دارد.

پس به همین شکل می‌توان برای هر عکسی بررسی کرد که جه عددی برای رتبه‌ی ماتریس مناسب می‌باشد.

۲. در این قسمت ابتدا به هر کدام از عکس‌ها نویز گاوی و نیز نویز نمک-فلفل اضافه می‌کنیم.
برای اضافه کردن نویز گاوی یک ماتریس همانند از عکس می‌سازیم که مولفه‌های آن از یک توزیع نرمال با میانگین صفر و واریانس ۳۲ به دست آمده‌اند و آن را با ماتریس عکس جمع می‌کنیم.
برای اضافه کردن نویز نمک-فلفل ابتدا دو عدد تصادفی در یک بازه‌ی مشخص (نسبت به تعداد کل پیکسل‌ها) انتخاب می‌کنیم که نشانگر تعداد پیکسل‌هایی هستند که می‌خواهیم سیاه و سفید بکنیم. سپس به همین تعداد انتخاب شده، مختصات انتخاب می‌کنیم و رنگ آن پیکسل‌ها را تغییر می‌دهیم.
حال که عکس‌های نویزی داریم، این عکس‌ها را با استفاده از الگوریتم بخش قبل ذخیره می‌کنیم تا نویز عکس کم شود و کلیات عکس اصلی حفظ شود.
در ادامه خروجی‌های مختلف را بررسی می‌کنیم.

شکل ۲۲: عکس با نویز گاوی



شكل ۲۳: کاهش نویز گاوی با $rank = 20$



شكل ۲۴: کاهش نویز گاوی با $rank = 35$



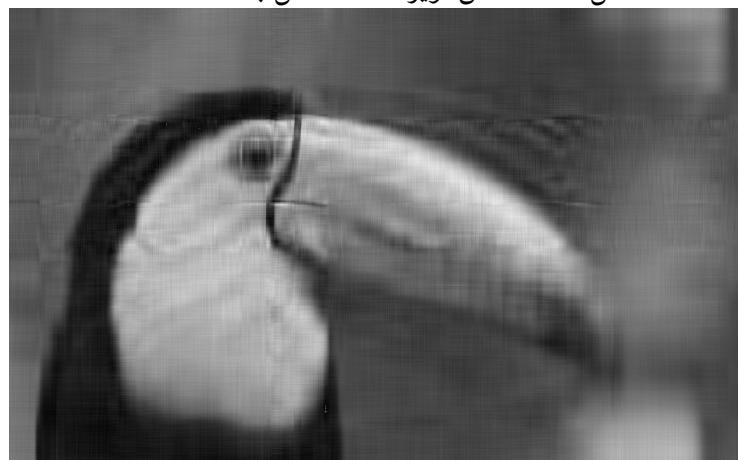
شكل ۲۵: کاهش نویز گاوی با $rank = 60$



شكل ۲۶: عکس با نویز نمک-فلفل



شكل ۲۷: کاهش نویز نمک-فلفل با $15 rank$



شكل ۲۸: کاهش نویز نمک-فلفل با $25 rank$



شكل ۲۹: کاهش نویز نمک_فلفل با $rank = ۶۵$



شكل ۳۰: عکس با نویز گاوی



شكل ۳۱: کاهش نویز گاوی با $rank = ۲۰$



شكل ٣٢: کاهش نویز گاوی با $rank = 60$



شكل ٣٣: کاهش نویز گاوی با $rank = 90$



شكل ٣٤: عکس با نویز نمک-فلفل



شكل ٣٥: کاهش نویز نمک_فلفل با 20 $rank$



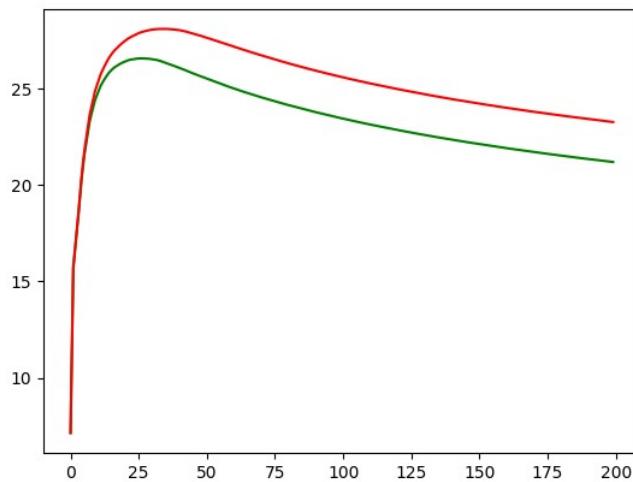
شكل ٣٦: کاهش نویز نمک_فلفل با 50 $rank$



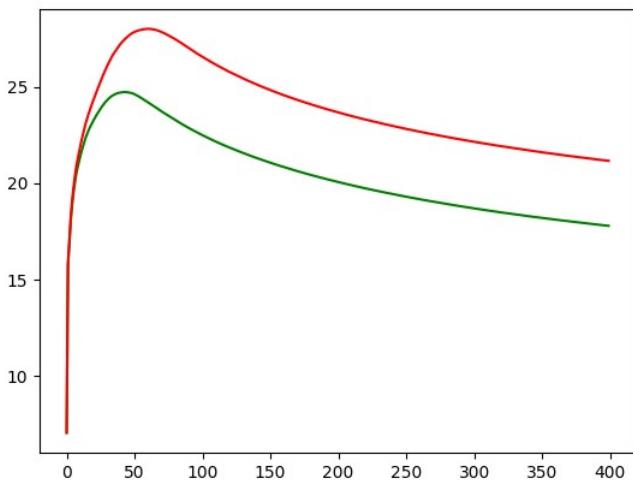
شكل ٣٧: کاهش نویز نمک_فلفل با 90 $rank$



شکل ۳۸: نمودار میزان PSNR بر حسب زیادشدن رتبه‌ی ماتریس عکس پرنده



شکل ۳۹: نمودار میزان PSNR بر حسب زیادشدن رتبه‌ی ماتریس عکس سگ



در این نمودارها خط قرمز مربوط به نویز گاوسی است. مطابق نمودار برای عکس پرنده، بهترین رتبه‌ی ماتریس برای حذف نویز، رتبه‌ی حدود ۳۵ است و برای نویز نمک-فلفل در حدود ۲۵ اما برای عکس سگ این اعداد بیشتر بوده و به ترتیب ۶۰ و ۵۰ می‌باشند. همچنین با توجه به خروجی‌هایی که نشان دادیم عکس‌ها مطابق انتظار ما از نمودار هستند. هم با توجه به نمودار و هم عکس‌هایی که دیدیم به وضوح این روش حذف نویز برای نویز گاوسی بهتر است و نتایج خوبی را به ما می‌دهد.

پرسش ۳ کاهش بعد داده‌ها با روش PCA

۱. ماتریس مجموعه‌ی داده‌ها همان ماتریس X می‌باشد پس ماتریس کوواریانس مجموعه‌ی داده‌ها نیز برابر ماتریس کوواریانس ماتریس X می‌باشد.

$$\text{cov}(X) = (X - \bar{X})(X - \bar{X})^T = \tilde{X}\tilde{X}^T$$

پس ماتریس کوواریانس مجموعه‌ی داده‌ها برابر $\tilde{X}\tilde{X}^T$ است. پس داریم:

$$\tilde{X}\tilde{X}^T = U\Sigma V^T$$

۲. فرض کنید $W'^T \in \mathbb{R}^{n \times l}$ ماتریسی باشد که ستون‌هایش متعامد یکه هستند و $y \in \mathbb{R}^l$ باشد.

$$\|x - W'^T y\|^2 = \|x\|^2 + y^T W' W'^T y - 2y^T W' x = \|x\|^2 + \|y\|^2 - 2y^T (W' x)$$

اگر از رابطه‌ی بالا مشتق بگیریم، خواهیم داشت:

$$W'^T W' = \arg \min \|x - \hat{x}\|^2$$

که در آن \hat{x} متغیر بین تمام اعضای \mathbb{R}^n است که با ضرب یک ماتریس در یک عضو \mathbb{R}^l ساخته شدند. پس ماتریس W که می‌خواهیم بسازیم باید دارای ویژگی بالا باشد. همچنین تساوی زیر را هم داریم:

$$\|x - W'^T W' x\|^2 = \|x\|^2 - \text{Tr}(W' x x^T W'^T)$$

پس مساله‌ی ما معادل بیشینه کردن $\text{Tr}(W' \sum x_i x_i^T W'^T)$ با ویژگی بالا می‌شود.
جواب این مساله نیز $U_l^T W$ می‌باشد.

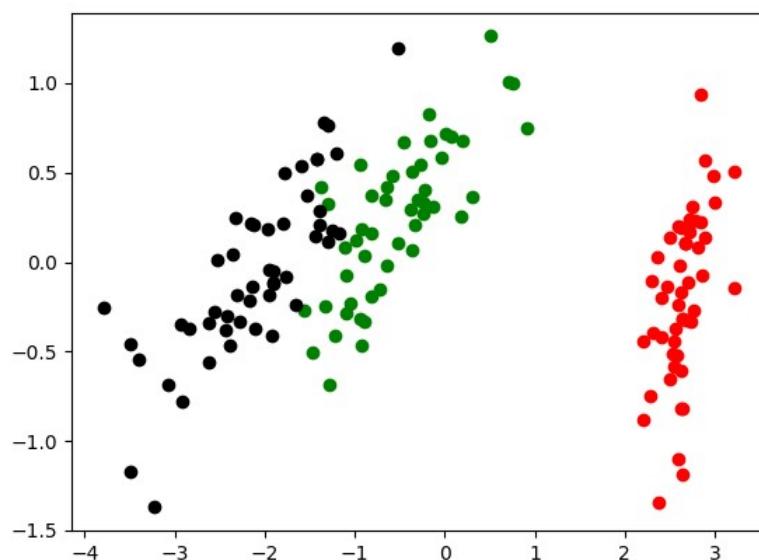
ماتریس‌های U و V برابر هستند و درواقع همان ماتریس بردارهای ویژه‌ی ماتریس کوواریانس هستند زیرا ماتریس کوواریانس یک ماتریس حقیقی متقارن می‌باشد که قطری شدنی است. و ماتریس Σ نیز همان ماتریس قطری شامل مقادیر ویژه‌ی مرتب شده با ترتیب نزولی است.

۳. ابتدا با l ستون اول ماتریس U ، ماتریس $U_l \in \mathbb{R}^{n \times l}$ را تشکیل می‌دهیم.
ماتریس $U_l^T X \in \mathbb{R}^{l \times m}$ همان ماتریس داده‌های جدیدمان می‌شود. یعنی هر ستون آن که به وضوح دارای l مولفه است درواقع متناظر با داده‌ی همین ستون در ماتریس X است که بعد آن از n به l کاهش یافته است.

۴. در حالاتی که بعد داده‌ها به میزان خوبی از تعداد داده‌ها بیشتر باشد الگوریتمی که مطرح کردیم پیچیدگی زیادی خواهد داشت و بدین منظور برای این حالات الگوریتم دیگری مطرح می‌کنیم.
در این حالت به جای اینکه بردار ویژه‌های ماتریس XX^T را محاسبه کنیم کافی است بردار ویژه‌های ماتریس $X^T X$ را محاسبه کنیم.
اگر u بردار ویژه‌ی $X^T X$ باشد، آن‌گاه Xu نیز بردار ویژه‌ی $XX^T X$ خواهد بود و به این ترتیب با پیچیدگی زمانی کمتر توانستیم همان جواب الگوریتم را به دست آوریم.

۵. نمودار داده‌های گل به شکل زیر می‌شوند: همانطور که در نمودار می‌بینیم سه دسته گل مختلف به خوبی از هم جدا شدند و اگر نمی‌دانستیم هر داده

شکل ۴۰: نمودار PCA ۲-بعدی داده‌های گل



مربوط به کدام نوع گل است هم تا حد خوبی می‌توانستیم درست تشخیص دهیم. پس نمودار با انتظارمان مطابقت دارد.