دانشگاه صنعتی خواجه نصیرالدین طوسی

تمرین امتیازی ۱: ریاضیات و جبر خطی

علی باقری‌نژاد

۴۰۲۰۲۸۵۴

فهرست

سؤال ۱

اگر A مثبت معین باشد، آنگاه مقادیر ویژه آن مثبت هستند

طبق تعریف مقادیر ویژه (λ) و بردار ویژه (v)، داریم:

$$Av = \lambda v \qquad \text{I}$$

اگر A ماتریسی مثبت معین باشد و x نیز یک بردار غیرصفر:

$$x^T Ax > 0 \qquad \text{II}$$

حال اگر به‌جای x در عبارت II، بردار ویژه v را قرار دهیم، خواهیم داشت:

$$v^T Av > 0 \Rightarrow v^T(Av) > 0 \overset{I}{\Rightarrow} v^T(\lambda v) = \lambda(v^T v) > 0$$

ازآنجایی‌که عبارت $v^T v$ همواره مثبت می‌باشد، λ نیز باید همواره مثبت باشد تا عبارت بالا صدق کند.

اگر تمامی مقادیر ویژه ماتریس A مثبت باشند، ماتریس A مثبت معین است

با استفاده از eigen decomposition این بخش را حل می‌کنیم. فرض می‌کنیم ماتریس وارون‌پذیر P (به‌صورت $P^{-1} = P^T$) غیر صفر وجود دارد، به صورتی که :

$$P^T AP = D = diag(\lambda_1, \lambda_2, \dots) \Rightarrow A = PDP^T \qquad \text{I}$$

با درنظرگرفتن I، اگر عبارت $x^T Ax$ همواره مثبت باشد، آنگاه ماتریس A مثبت معین است.

$$x^T Ax = x^T(PDP^T)x = (x^T P)D(P^T x) \overset{P^T x = y}{=} y^T Dy = \lambda_1 y_1^2 + \lambda_2 y_2^2 + \cdots \qquad \text{II}$$

باتوجه‌به II، و با پیش‌فرض آنکه تمامی مقادیر ویژه مثبت هستند، عبارت همواره بزرگ‌تر از صفر است، پس داریم:

$$y^T Dy > 0 \Rightarrow x^T Ax > 0$$

سؤال ۲

مقادیر و بردارهای ویژه را برای بردار دوران دوبعدی به دست می‌آوریم. برای حل مسئله از دو روش تحلیلی و کد پایتون استفاده می‌کنیم.

روش تحلیلی

$$R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

$$Rv = \lambda v \Rightarrow (\lambda I - R)v = 0 \qquad \text{II}$$

برای برقراری معادله II، باید عبارت $(\lambda I - R) = 0$ برقرار باشد. معادله مشخصه ماتریس دوران به صورت زیر می باشد:

$$\lambda^2 - 2\cos(\theta)\,\lambda + 1 = 0 \Rightarrow \lambda_{1,2} = e^{-\theta j}, e^{\theta j} \Rightarrow \lambda_{1,2} = \cos(\theta) \mp jsin(\theta)$$

برای محاسبه بردارهای ویژه معادله $Rv = \lambda v$ را به ازای مقادیر مختلف λ حل می نماییم.(لازم به ذکر است که عناصر دوم هر بردار ویژه را برابر ۱ در نظر می‌گیریم؛ یعنی : $v^1 = \begin{bmatrix} v_1^1 \\ 1 \end{bmatrix}, v^2 \begin{bmatrix} v_1^2 \\ 1 \end{bmatrix}$)

$$\begin{bmatrix} \cos(\theta)\,v_1^1 - \sin(\theta) \\ \sin(\theta)\,v_1^1 + \cos(\theta) \end{bmatrix} = \begin{bmatrix} \cos(\theta)\,v_1^1 - jsin(\theta)v_1^1 \\ \cos(\theta) - jsin(\theta) \end{bmatrix} \Rightarrow v^1 = \frac{1}{\sqrt{2}} \begin{bmatrix} j \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} \cos(\theta)\,v_1^2 - \sin(\theta) \\ \sin(\theta)\,v_1^2 + \cos(\theta) \end{bmatrix} = \begin{bmatrix} \cos(\theta)\,v_1^2 + jsin(\theta)v_1^2 \\ \cos(\theta) + jsin(\theta) \end{bmatrix} \Rightarrow v^2 = \frac{1}{\sqrt{2}} \begin{bmatrix} -j \\ 1 \end{bmatrix}$$

ضریب $\frac{1}{\sqrt{2}}$ برای تبدیل کردن بردارهای ویژه به بردار های یکه در بردار ها ضرب شده است.

$$\left\{ e^{-\theta j}, \begin{bmatrix} \frac{j}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \right\}, \left\{ e^{\theta j}, \begin{bmatrix} -\frac{j}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \right\}$$

کد پایتون

در برنامه Q2.py، با استفاده از کتابخانه `sympy`، متغیرها لازم و ماتریس دوران را تعریف کرده. ازآنجایی‌که ماتریس دوران به‌عنوان یک شب `sympy.Matrix` ذخیره شده است، می‌توانیم از متدهای `eigenvals()` و `eigenvectors()` برای محاسبه مقادیر و بردارهای ویژه استفاده کنیم. در نتیجه خواهیم داشت:

```
EigenValues are:
        -sqrt(-sin(theta)**2) + cos(theta)
        sqrt(-sin(theta)**2) + cos(theta)


EigenVectors are:
        [Matrix([
[sin(theta)/sqrt(cos(theta)**2 - 1)],
[                                 1]])]
        [Matrix([
[-sin(theta)/sqrt(cos(theta)**2 - 1)],
[                                  1]])]
```

شکل ۱ خروجی Q2.py

در خروجی نمایش‌داده‌شده، عبارات منفی‌ای که زیر رادیکال قرار داشتند، ساده‌سازی نشده‌اند و به همین دلیل عبارت j در خروجی مشاهده نمی‌شود.

سؤال ۳

ماتریس A را، یک ماتریس متقارن در نظر می‌گیریم. فرض می‌کنیم ماتریس P ماتریس مودال ماتریس A است و بردارهای v_i، بردارها ویژه ماتریس A هستند. می‌دانیم که:

$$\left.\begin{array}{c} P = [v_1 \quad v_2 \quad \dots \quad v_n] \\ P \in \mathbb{R}^{n\times n} \\ < v_i . v_j > = 0; i \neq j, i, j = 1,2,\dots,n \quad \mathrm{II} \end{array}\right\} \Rightarrow P\ is\ \boldsymbol{Orthogonal} \qquad \mathrm{I}$$

$$Proof\ of\ \mathrm{II}: Av_1 = \lambda_1 v_1 \Rightarrow v_2^T A v_1 = v_2^T \lambda_1 v_1 \xrightarrow[v_2^T A v_1 = (v_1^T A^T v_2)\text{^}T]{A\ is\ symmetric} (v_1^T \lambda_2 v_2)^T = \lambda_1 v_2^T v_1$$

$$\Rightarrow \underbrace{(\lambda_2 - \lambda_1)}_{\neq 0} v_2^T v_1 = 0 \Rightarrow < v_1 . v_2 > = 0$$

در عبارت بالا اثبات شده است ازآنجایی‌که ستون‌های ماتریس مودال در واقع بردارهای ویژه هستند و بردارهای ویژه نیز به‌صورت خطی از هم مستقل هستند، می‌توان نتیجه گرفت که ماتریس مودال، یک ماتریس متعامد[۱] است.

اثبات شد که ماتریس مودال $n \times n$ یک ماتریس مربعی متقارن همواره متعامد است. حال داریم:

$$\xRightarrow{properties\ of\ orthogonal\ matrix} PP^T = I \Rightarrow \det|P| \times \det|P^T| = \det|I| \xRightarrow{|P|=|P^T|} |P|^2 = 1$$
$$\Rightarrow |P| = \pm 1$$

طبق راه‌حل ارائه شده، دترمینان ماتریس مودال برابر ۱± می‌باشد.

سؤال ۴

در این بخش، روش حل دستگاه معادلاتی بیش تعیین‌شده[۲] با استفاده از تجزیه مقادیر منفرد[۳] به دست می‌آوریم.

$$V^T V = I, U^T U = I \qquad a$$
$$\left.\begin{matrix} A = U\Sigma V^T \\ Ax = b \end{matrix}\right\} \Rightarrow A^T A x = A^T b \Rightarrow V\Sigma \underbrace{U^T U}_{I} (\Sigma V^T x) = V\Sigma U^T b \overset{a}{\Rightarrow} \Sigma^2 V^T x$$
$$= \Sigma U^T b \xRightarrow{\times(\Sigma^{-1})^2} V^T x = \Sigma^{-1} U^T b \overset{\times V}{\Rightarrow} x = V\Sigma^{-1} U^T b$$

می‌دانیم که $\Sigma \in \mathbb{R}^{n\times m}$ و اینچنین ماتریسی وارون ندارد. $\Sigma^{-1} \in \mathbb{R}^{m\times n}$را بدین صورت تعریف می‌کنیم که مقادیر عناصر آن، معکوس مقادیر ماتریس Σ هستند و هنگامی که این دو ماتریس در هم ضرب شوند، یک ماتریس همانی را بوجود می‌آورند. پس می‌توان به این روش تاثیر ماتریس Σ را در یک طرف معادله از بین برد.

---

۱ Orthogonal

۲ Over-determined

۳ Singular Value Decomposition (SVD)

سؤال ۵

دترمینان ماتریس‌های بالا مثلثی برابر است با ضرب تمامی عناصر روی قطر اصلی در هم. حال اگر در این میان حداقل یکی از درایه‌های قطر اصلی ۰ باشد، آنگاه ماتریس منفرد خواهد شد.

اثبات

$$A = [a] \Rightarrow |A| = a$$

$$\left.\begin{array}{l} \bar{\bar{A}}_{n\times n} = \begin{bmatrix} \bar{\bar{B}}_{n-1\times n-1} & \bar{b} \\ \bar{0} & a \end{bmatrix} \Rightarrow |A| = a|B| \\ \bar{\bar{B}}_{n-1\times n-1} = \begin{bmatrix} \bar{\bar{C}}_{n-2\times n-2} & \bar{c} \\ \bar{0} & b \end{bmatrix} \Rightarrow |B| = b|C| \end{array}\right\} \Rightarrow \mathrm{II}$$

اگر روند II را ادامه دهیم، عبارت ادعا شده در قسمت قبل اثبات می‌شود که یعنی دترمینان ماتریس بالا مثلثی برابر با حاصل‌ضرب درایه‌های قطر اصلی ماتریس است.

تأیید استدلال به کمک پایتون

با استفاده از زبان برنامه‌نویسی پایتون برنامه‌ای نوشته شد که ابتدا یک ماتریس بالا مثلثی ایجاد کند و سپس ضرب مقادیر قطر اصلی و دترمینان آن را محاسبه نماید. در نهایت نیز مقادیر محاسبه شده را نمایش می‌دهد. اما قبل از محاسبه مقادیر موردنیاز، برنامه به‌صورت رندوم یک عدد را تعیین می‌کند، اگر عدد زوج بود، در آن درایه از قطر اصلی عدد ۰ را جایگزین می‌کند و در غیر این صورت هیچ صفری در ماتریس جایگذاری نمی‌شود. برنامه را چندین بار اجرا کرده و خروجی‌ها را نمایش می‌دهیم.

```
In [47]: runfile('D:/Data/Esmaeel/HW1/Q5.py', wdir='D:/Data/Esmaeel/HW1')
the matrix craeted is:
[[0.29619455 0.70006764 0.66469668 0.24652763]
 [0.         0.21251747 0.8769691  0.20810095]
 [0.         0.         0.17353118 0.85538685]
 [0.         0.         0.         0.57993904]]

No element on the main diagonal is set to zero

The determinant of the martix and the product of the main diameter are the same
Determinant = 0.006334780316901336
Product = 0.006334780316901339
The difference is 0.00000
```

شکل ۲ اولین خروجی برنامه

```
In [48]: runfile('D:/Data/Esmaeel/HW1/Q5.py', wdir='D:/Data/Esmaeel/HW1')
the matrix craeted is:
[[0.91167766 0.08734685 0.06201327 0.09995522]
 [0.         0.85552955 0.77328461 0.30448686]
 [0.         0.         0.32060679 0.12109468]
 [0.         0.         0.         0.92200626]]

The [0]th element is set to zero

The new matrix is:
[[0.         0.08734685 0.06201327 0.09995522]
 [0.         0.85552955 0.77328461 0.30448686]
 [0.         0.         0.32060679 0.12109468]
 [0.         0.         0.         0.92200626]]
The determinant of the martix and the product of the main diameter are the same
Determinant = 0.0
Product = 0.0
The difference is 0.00000
```

شکل ۳دومین خرجی برنامه

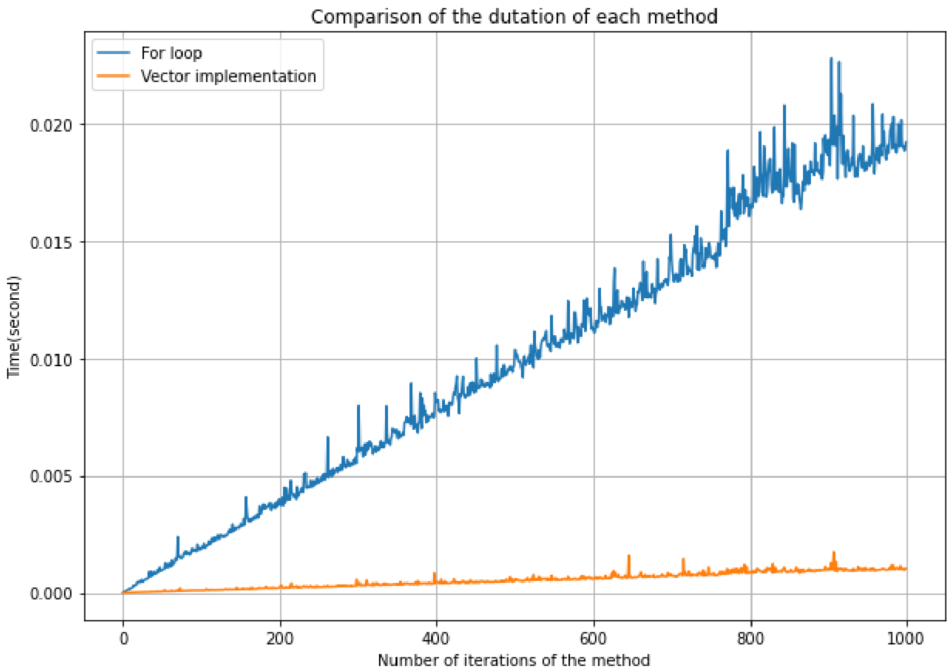
در شکل ۲ که اولین خروجی برنامه می‌باشد، هیچ تغییری در ماتریس ایجاد نشده است و محاسبات برای خود ماتریس انجام شده است. در بخش انتهایی خروجی برنامه نیز ضرب مقادیر قطر اصلی و دترمینان ماتریس با هم مقایسه شده اند و مشخص است که هیچ تفاوتی با هم ندارد.

اما در شکل ۳، دومین خروجی برنامه به نمایش در آمده است که در آن به صورت رندوم یکی از عناصر قطر اصلی صفر شده است که می‌توان در بخش انتهایی خروجی مقادیر محاسبه شده را دید که همه آن‌ها صفر شده اند.

در نتیجه، یک ماتریس بال‌مثلثی هنگامی منفرد می‌شود که حداقل یکی از مقادیر دترمینان قطر اصلی آن برابر صفر باشد.

سوال ۵

ضرب ماتریس را از دو روش حلقه و ضرب برداری انجام می‌دهیم. البته لازم به ذکر است که برای مقایسه بهتر عملکرد دو روش خواسته شده، به هنگام استفاده از تابع `timeit.timeit()`، مقدار ویژگی[۱] `number` را اعداد مختلفی در نظر گرفتیم تا سرعت عملکرد روش‌ها را به‌ازای تکرارهای مختلفی که انجام می‌شود را مقایسه کنیم.



شکل ۴ زمان طی شده توسط هر الگوریتم برای تکرار های مختلف

همان‌طور که در شکل ۴ مشاهده می‌شود، به ازای تکرارهای پایین، سرعت عملکرد هر دو روش یکسان است ولی هنگامی که تعداد تکرارها و محاسبات لازم برای انجام کار خواسته شده بالاتر می‌رود، سرعت روش حلقه بسیار بیشتر از روش ضرب برداری کاهش پیدا می‌کند.

اما به‌عنوان‌مثال یکی از خروجی‌ها را با هم مقایسه کردیم تا از یکسان بودن جواب هر دو روش اطمینان حال کنیم. (با استفاده از تابع `np.allclose()`)

---

[۱] Attribute

```
In [78]: runfile('D:/Data/Esmaeel/HW1/untitled1.py', wdir='D:/Data/Esmaeel/HW1')
outcome of loop method: [2.13312249 2.18736354 2.64879856 1.47226938 2.50499782 2.77563286
 2.10321581 1.63321757 2.4492479  2.70815862 3.00416897 3.01106084
 2.94281761 3.22968592 3.09909033 2.78078148 3.1628988  3.00439318
 1.8904498  1.76113328]

outcome of vectro implementation method: [2.13312249 2.18736354 2.64879856 1.47226938 2.50499782 2.77563286
 2.10321581 1.63321757 2.4492479  2.70815862 3.00416897 3.01106084
 2.94281761 3.22968592 3.09909033 2.78078148 3.1628988  3.00439318
 1.8904498  1.76113328]

The outcome of both methods is the same vector
```

شکل ۵ مقایسه بردار محاسبه شده برای هر دو روش

در شکل ۵، بردارهای بدست آمده از هر دو روش با استفاده از تابع `np.allclose()` با هم مقایسه شده اند و مشخص است که خروجی هر دو روش یکی است.