دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

دانشکده ریاضی و علوم کامپیوتر
استاد درس: دکتر مهدی دهقان
بهار ۱۴۰۳

الگوریتم گالوپ کاهان

علیرضا پرورش
۹۹۱۲۰۱۳
جبر خطی عددی

فهرست مطالب

۱ مقدمه

روش گولوب_ کاهان یکی از الگوریتم‌های برجسته و کارآمد برای محاسبه تجزیه مقدار منفرد (SVD) است که توسط جین گولوب و ویلیام کاهان در سال ۱۹۶۵ معرفی شد. این روش به عنوان یک پیشرفت مهم در حوزه جبر خطی عددی شناخته می‌شود و به دلیل ویژگی‌های منحصر به فرد خود، به طور گسترده در علوم کامپیوتر، مهندسی و تحلیل داده‌ها مورد استفاده قرار می‌گیرد. الگوریتم گولوب_ کاهان با بهره‌گیری از بازتاب‌های هوسهولدر، ماتریس اصلی را به یک ماتریس بیدیگونال تبدیل می‌کند. این تبدیل، فرآیند محاسبه SVD را بهینه‌سازی کرده و پیچیدگی محاسباتی را به طور قابل توجهی کاهش می‌دهد.

یکی از مزایای برجسته این روش، پایداری عددی بالای آن است که باعث کاهش خطاهای گردشی و افزایش دقت نتایج می‌شود. بازتاب‌های هوسهولدر که در این الگوریتم به کار گرفته می‌شوند، امکان حفظ ساختار ماتریس‌های نادر (Sparse) را فراهم کرده و کارایی حافظه را بهبود می‌بخشند. این ویژگی‌ها، الگوریتم گولوب_ کاهان را به یک انتخاب مناسب برای تحلیل داده‌های بزرگ‌مقیاس و کاربردهای علمی و مهندسی تبدیل کرده‌اند.

علاوه بر این، روش گولوب_ کاهان با سایر روش‌های عددی، مانند روش‌های تکراری، سازگاری خوبی دارد و می‌تواند در ترکیب با آنها به نتایج بهتری منجر شود. در مقایسه با روش‌های مرسوم مانند الگوریتم ، گولوب_ کاهان نه تنها از لحاظ کارایی و دقت برتری دارد، بلکه از نظر پایداری عددی نیز مطمئن‌تر است. این ویژگی‌ها باعث شده‌اند که این روش در بسیاری از زمینه‌های کاربردی، از جمله پردازش تصویر، تحلیل داده‌های ژنومی و مدل‌سازی‌های علمی، مورد استفاده قرار گیرد.

در نتیجه، الگوریتم گولوب_ کاهان به عنوان یکی از ابزارهای اساسی در جبر خطی عددی شناخته می‌شود که توانسته است با بهبود کارایی محاسبات و حفظ دقت نتایج، سهم بسزایی در پیشرفت علوم محاسباتی و تحلیل داده‌ها ایفا کند. این روش همچنان به عنوان یک استاندارد در بسیاری از کاربردهای عملی و تحقیقاتی مورد توجه و استفاده قرار دارد.

۲ الگوریتم گولوب_ کاهان

الگوریتم گولوب_ کاهان به دو مرحله اصلی تقسیم می‌شود:

- تبدیل ماتریس به ماتریس بیدیگونال
- و محاسبه SVD از شکل بیدیگونال.

این مراحل در زیر با جزئیات توضیح داده شده‌اند.

۱.۲ تبدیل ماتریس به ماتریس بیدیگونال

۱.۱.۲ ماتریس بیدیگونال

ماتریس بیدیگونال (Bidiagonal Matrix) یک ماتریس مربعی است که تنها دارای عناصر غیر صفر بر روی قطر اصلی و یک باند زیر یا بالای قطر اصلی است. به عبارتی، در ماتریس بیدیگونال فقط عناصر روی قطر اصلی و عناصر یک ردیف یا یک ستون بالاتر یا پایین‌تر از قطر اصلی غیر صفر هستند.
در ماتریس‌ها، اصطلاحات "باند بالا" و "باند پایین" به عناصر غیرصفر ماتریس در نزدیکی قطر اصلی اشاره دارند. بیایید به تعریف و مثال‌های بیشتری بپردازیم تا مفهوم آن را روشن‌تر کنیم.

ماتریس بیدیگونال بالا یک ماتریس بیدیگونال بالا به شکل زیر است:

$$B_{\text{up}} = \begin{pmatrix} a_{11} & a_{12} & 0 & 0 \\ 0 & a_{22} & a_{23} & 0 \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{pmatrix}$$

که در آن $a_{11}, a_{12}, a_{22}, a_{23}, a_{33}, a_{34}, a_{44}$ می‌توانند هر مقدار عددی باشند.
مثال عددی بیدیگونال بالا:

$$B_{\text{up}} = \begin{pmatrix} 4 & 2 & 0 & 0 \\ 0 & 3 & 1 & 0 \\ 0 & 0 & 5 & 3 \\ 0 & 0 & 0 & 6 \end{pmatrix}$$

در این مثال، عناصر غیرصفر در باند بالای قطر اصلی عبارتند از $2, 1, 3$.

ماتریس بیدیگونال پایین یک ماتریس بیدیگونال پایین به شکل زیر است:

$$B_{\text{low}} = \begin{pmatrix} a_{11} & 0 & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 \\ 0 & a_{32} & a_{33} & 0 \\ 0 & 0 & a_{43} & a_{44} \end{pmatrix}$$

که در آن $a_{11}, a_{21}, a_{22}, a_{32}, a_{33}, a_{43}, a_{44}$ می‌توانند هر مقدار عددی باشند.
مثال عددی بیدیگونال پایین:

$$B_{\text{low}} = \begin{pmatrix} 4 & 0 & 0 & 0 \\ 2 & 3 & 0 & 0 \\ 0 & 1 & 5 & 0 \\ 0 & 0 & 3 & 6 \end{pmatrix}$$

در این مثال، عناصر غیرصفر در باند پایین قطر اصلی عبارتند از $2, 1, 3$.

در الگوریتم گولوپ ـ کاهان برای محاسبه SVD، معمولاً از ماتریس بیدیگونال بالا (upper bidiagonal matrix) استفاده می‌شود. این به این معناست که ماتریس اولیه به صورتی تبدیل می‌شود که عناصر غیرصفر تنها در قطر اصلی و یک ردیف بالاتر از قطر اصلی قرار دارند. این نوع ماتریس بیدیگونال بالا به دلیل ساختار ساده‌تری که دارد، برای محاسبات عددی و پیاده‌سازی الگوریتم‌های تجزیه مانند SVD مناسب‌تر است.

۲.۲ الگوریتم بیدیاگونالیزاسیون گولوب_ کاهان

ورودی و خروجی‌ها

- **ورودی**: ماتریس A با ابعاد $m \times n$
- **خروجی‌ها**:
  - ماتریس بیدیاگونال B با ابعاد $m \times n$
  - ماتریس ارتوگونال U با ابعاد $m \times m$
  - ماتریس ارتوگونال V با ابعاد $n \times n$

مقداردهی اولیه

$$
\begin{aligned}
U &= I_m \quad (\text{ماتریس واحد } m \times m)\\
V &= I_n \quad (\text{ماتریس واحد } n \times n)\\
B &= A
\end{aligned}
$$

حلقه اصلی

برای $i = 1$ تا $\min(m, n)$:
استفاده از 'min(m, n)' کمک می‌کند تا عملیات bidiagonalization به صورت بهینه‌تری انجام شود، زیرا ما تنها تا حداقل اندازه کوچکترین بُعد از ماتریس A نیاز داریم. این کمک می‌کند که الگوریتم به سرعت به نقاط کلیدی از فرایند برسد.

۱. **بازتاب هوسهولدر از چپ** بازتاب هوسهولدر از چپ به این صورت عمل می‌کند که با اعمال ماتریس هوسهولدر از سمت چپ به یک ماتریس، مولفه‌های زیر قطر اصلی در یک ستون خاص صفر می‌شوند. به طور مشخص، اگر بازتاب هوسهولدر به یک ماتریس A از سمت چپ اعمال شود، عناصر زیر قطر اصلی در یک ستون خاص از آن ماتریس صفر خواهند شد.

- بردار ستون x از ماتریس B تعریف می‌شود به صورت:

$$x = B[i : m, i]$$

این نشان می‌دهد که x برابر است با بردار ستونی از ماتریس B که از ردیف iام تا ردیف mام و در ستون iام قرار دارد.

- بردار بازتاب هوسهولدر v برای x محاسبه می‌شود:

$$v = x + \text{sign}(x_1)\|x\|_2 e_1$$

که e_1 بردار واحد اول است و $\text{sign}(x_1)$ علامت اولین مولفه x است.

- ماتریس هوسهولدر H ساخته می‌شود:

$$H = I_{m-i} - 2\frac{vv^T}{v^T v}$$

در الگوریتم بازتاب هوسهولدر از سمت چپ، ماتریس H یک ماتریس مربعی است با ابعاد $(m-i) \times (m-i)$.

- ماتریس B به‌روزرسانی می‌شود:

$$B[i:m,i:n] = HB[i:m,i:n]$$

زیرماتریس $B[i:m,i:n]$ ابعاد $(m-i)\times(n-i)$ دارد.
زیرماتریس B:

- عبارت $B[i:m,i:n]$ به زیرماتریسی از B اشاره دارد که از ردیف i تا ردیف m و از ستون i تا ستون n قرار دارد. این زیرماتریس شامل المان‌هایی است که تحت تاثیر بازتاب هوسهولدر قرار می‌گیرند.

- ماتریس U به‌روزرسانی می‌شود:

$$U[:,i:m] = U[:,i:m]H$$

زیرماتریس U:

- عبارت $U[:,i:m]$ به زیرماتریسی از U اشاره دارد که شامل تمام ردیف‌ها و ستون‌های i تا m است. این زیرماتریس نشان‌دهنده بخشی از U است که باید تحت تاثیر ماتریس هوسهولدر H قرار گیرد.
ابعاد $(m)\times(m-i)$

۲. **بازتاب هوسهولدر از راست** زتاب هوسهولدر از راست باعث صفر شدن درایه‌های زیر قطر اصلی ستون‌های ماتریس می‌شود، در حالی که بازتاب هوسهولدر از سمت چپ باعث صفر شدن درایه‌های بالای قطر اصلی سطرهای ماتریس می‌شود.

- اگر $i<n$:
  - بردار ردیف x از ماتریس B تعریف می‌شود به صورت:

$$x = B[i,i+1:n]$$

، x یک بردار است که از سطر i ام ماتریس B استخراج شده است، از ستون $i+1$ تا ستون n.

  - بردار بازتاب هوسهولدر v برای x محاسبه می‌شود:

$$v = x + \text{sign}(x_1)\|x\|_2 e_1$$

  - ماتریس هوسهولدر H ساخته می‌شود:

$$H = I_{n-i-1} - 2\frac{vv^T}{v^Tv}$$

  - ماتریس B به‌روزرسانی می‌شود:

$$B[i:m,i+1:n] = B[i:m,i+1:n]H$$

زیرماتریس B:
این عبارت به معنی این است که بخشی از ماتریس B که از سطر i تا سطر $m-1$ و از ستون $i+1$ تا ستون $n-1$ است،

  - ماتریس V به‌روزرسانی می‌شود:

$$V[:,i+1:n] = V[:,i+1:n]H$$

۳.۲ تجزیه SVD به وسیله ماتریس بیدیگونال

ماتریس بیدیگونال B می‌تواند به صورت زیر تجزیه شود:

$$B = U_b \Sigma (V_b)^T$$

که در اینجا:

- U_b و V_b ماتریس‌های ارتوگونال برای بردارهای ویژه چپ و راست هستند.
- Σ بردار وکتور ویژه‌های ماتریس بیدیگونال B است.

۲. به‌روزرسانی ماتریس‌های ارتوگونال U و V:

ماتریس‌های ارتوگونال U و V بر اساس بردارهای ویژه به‌دست آمده از SVD به‌روزرسانی می‌شوند:

$$U = UU_b$$

$$V = VV_b^T$$

که این عملیات باعث به‌روزرسانی ماتریس‌های ارتوگونال U و V می‌شود.

در واقع ماتریس B که از ماتریس A بیدیگونال شده است پس از تجزیه SVD و به روز رسانی ماتریس های U_b و V_b و به دست اوردن ماتریس های U و V میتوان راطه زیر را نوشنت .

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

۳ مزایای الگوریتم گولوب_ کاهان

الگوریتم گولوب_ کاهان برای محاسبه تجزیه مقدار منفرد (SVD) مزایای قابل توجهی دارد که آن را به یکی از روش‌های پرکاربرد در ریاضیات عددی و پردازش داده‌ها تبدیل کرده است. در ادامه به بررسی هر یک از این مزایا پرداخته می‌شود:

۱.۳ پایداری عددی

پایداری عددی به توانایی یک الگوریتم در کاهش خطاهای ناشی از محاسبات عددی اشاره دارد. این مسئله در محاسبات با اعداد بسیار بزرگ یا بسیار کوچک که ممکن است منجر به خطاهای گرد کردن یا ناپایداری‌های عددی شود، اهمیت ویژه‌ای دارد.
دلیل:

- *استفاده از بازتاب‌های هوسهولدر:* این بازتاب‌ها برای تبدیل ماتریس اصلی به ماتریس بیدیگونال استفاده می‌شوند و به حفظ ساختار عددی ماتریس کمک می‌کنند.
- *کاهش خطاهای تکراری:* تبدیل ماتریس به شکل بیدیگونال قبل از اعمال الگوریتم‌های تکراری، احتمال افزایش خطاهای عددی در طی مراحل تکرار را کاهش می‌دهد، زیرا این تبدیل ماتریس اصلی را به یک فرم ساده‌تر و عددی پایدارتر تبدیل می‌کند.

۲.۳ کارایی محاسباتی

الگوریتم گولوب_ کاهان به دلیل بهینه‌سازی مراحل محاسباتی و کاهش تعداد عملیات‌های مورد نیاز برای تجزیه ماتریس‌ها، کارایی بالایی دارد.
دلیل:

- *پیچیدگی محاسباتی کمتر:* تبدیل ماتریس به شکل بیدیگونال باعث کاهش تعداد عملیات‌های ضرب ماتریس و بردار می‌شود. این به این معناست که الگوریتم می‌تواند با پیچیدگی محاسباتی کمتری ماتریس‌ها را تجزیه کند.
- *تکرار سریع:* الگوریتم‌های تکراری مانند QR زمانی که بر روی ماتریس بیدیگونال اعمال می‌شوند، سریع‌تر عمل می‌کنند. این به دلیل ساختار ساده‌تر و کم‌تراکم‌تر ماتریس بیدیگونال نسبت به ماتریس اصلی است.

۳.۳ دقت نتایج

یکی از ویژگی‌های مهم الگوریتم گولوب_ کاهان، دقت بالای نتایج آن است که این مسئله در کاربردهای مختلف بسیار مهم است.
دلیل:

- *حفظ مقادیر منفرد:* این الگوریتم به صورت دقیق مقادیر منفرد ماتریس را محاسبه می‌کند. مقادیر منفرد نمایانگر ویژگی‌های مهمی از ماتریس اصلی هستند و دقت بالا در محاسبه آن‌ها منجر به تحلیل دقیق‌تر داده‌ها می‌شود.
- *تقریب بهینه:* با استفاده از این الگوریتم، تقریب بسیار دقیقی از مقادیر منفرد و بردارهای منفرد ماتریس به دست می‌آید. این امر به خصوص در کاربردهایی مانند فشرده‌سازی داده‌ها و کاهش ابعاد داده‌ها که نیاز به دقت بالا دارند، بسیار مهم است.

۴.۳ کاربرد در ماتریس‌های بزرگ و نادر

الگوریتم گولوب ـ کاهان به خوبی برای کار با ماتریس‌های بزرگ و نادر (Sparse) بهینه‌سازی شده است.
دلیل:

- *مدیریت حافظه*: این الگوریتم از بازتاب‌های هوسهولدر و ماتریس‌های بیدیگونال برای کاهش نیاز به حافظه استفاده می‌کند. با این کار، فضای حافظه مورد نیاز برای ذخیره و پردازش ماتریس‌ها بهینه‌سازی شده و امکان پردازش ماتریس‌های بسیار بزرگ فراهم می‌شود.

- *بهینه‌سازی برای ماتریس‌های نادر*: الگوریتم می‌تواند به صورت بهینه ماتریس‌های نادر را پردازش کند. این مسئله باعث می‌شود تا زمان و حافظه صرف شده برای پردازش ماتریس‌هایی که بخش زیادی از عناصرشان صفر است، کاهش یابد.

۵.۳ مزایای کلی الگوریتم گولوب ـ کاهان

با توجه به مزایای فوق، الگوریتم گولوب ـ کاهان یکی از ابزارهای قدرتمند در ریاضیات عددی و پردازش داده‌ها است. این الگوریتم در بسیاری از کاربردهای عملی مانند تحلیل داده‌های بزرگ، فشرده‌سازی داده‌ها، و تجزیه و تحلیل تصاویر استفاده می‌شود. پایداری عددی، کارایی محاسباتی، دقت نتایج و توانایی کار با ماتریس‌های بزرگ و نادر، از جمله ویژگی‌های برجسته‌ای هستند که این الگوریتم را از دیگر روش‌های مشابه متمایز می‌کنند. این ویژگی‌ها سبب شده‌اند که الگوریتم گولوب ـ کاهان به عنوان یکی از بهترین روش‌ها برای محاسبه SVD در مسائل عملی و تحقیقاتی شناخته شود.

۴ مزایای الگوریتم گولوب-کاهان

الگوریتم گولوب-کاهان برای محاسبه تجزیه مقدار منفرد (SVD) مزایای قابل توجهی دارد که آن را به یکی از روش‌های پرکاربرد در ریاضیات عددی و پردازش داده‌ها تبدیل کرده است. در ادامه به بررسی هر یک از این مزایا پرداخته می‌شود:

۱.۴ پایداری عددی

پایداری عددی به توانایی یک الگوریتم در کاهش خطاهای ناشی از محاسبات عددی اشاره دارد. این مسئله در محاسبات با اعداد بسیار بزرگ یا بسیار کوچک که ممکن است منجر به خطاهای گرد کردن یا ناپایداری‌های عددی شود، اهمیت ویژه‌ای دارد.
دلیل:

- *استفاده از بازتاب‌های هوسهولدر*: این بازتاب‌ها برای تبدیل ماتریس اصلی به ماتریس بیدیگونال استفاده می‌شوند و به حفظ ساختار عددی ماتریس کمک می‌کنند. بازتاب‌های هوسهولدر به دلیل ماهیت ارتوگونال‌شان، از تجمع خطاهای گرد کردن جلوگیری می‌کنند.
- *کاهش خطاهای تکراری*: تبدیل ماتریس به شکل بیدیگونال قبل از اعمال الگوریتم‌های تکراری، احتمال افزایش خطاهای عددی در طی مراحل تکرار را کاهش می‌دهد، زیرا این تبدیل ماتریس اصلی را به یک فرم ساده‌تر و عددی پایدارتر تبدیل می‌کند.

۲.۴ کارایی محاسباتی

الگوریتم گولوب-کاهان به دلیل بهینه‌سازی مراحل محاسباتی و کاهش تعداد عملیات‌های مورد نیاز برای تجزیه ماتریس‌ها، کارایی بالایی دارد.
دلیل:

- *پیچیدگی محاسباتی کمتر*: تبدیل ماتریس به شکل بیدیگونال باعث کاهش تعداد عملیات‌های ضرب ماتریس و بردار می‌شود. این به این معناست که الگوریتم می‌تواند با پیچیدگی محاسباتی کمتری ماتریس‌ها را تجزیه کند.
- *تکرار سریع*: الگوریتم‌های تکراری مانند QR زمانی که بر روی ماتریس بیدیگونال اعمال می‌شوند، سریع‌تر عمل می‌کنند. این به دلیل ساختار ساده‌تر و کم‌تراکم‌تر ماتریس بیدیگونال نسبت به ماتریس اصلی است.

۳.۴ دقت نتایج

یکی از ویژگی‌های مهم الگوریتم گولوب-کاهان، دقت بالای نتایج آن است که این مسئله در کاربردهای مختلف بسیار مهم است.
دلیل:

- *حفظ مقادیر منفرد*: این الگوریتم به صورت دقیق مقادیر منفرد ماتریس را محاسبه می‌کند. مقادیر منفرد نمایانگر ویژگی‌های مهمی از ماتریس اصلی هستند و دقت بالا در محاسبه آن‌ها منجر به تحلیل دقیق‌تر داده‌ها می‌شود.

- *تقریب بهینه*: با استفاده از این الگوریتم، تقریب بسیار دقیقی از مقادیر منفرد و بردارهای منفرد ماتریس به دست می‌آید. این امر به خصوص در کاربردهایی مانند فشرده‌سازی داده‌ها و کاهش ابعاد داده‌ها که نیاز به دقت بالا دارند، بسیار مهم است.

۴.۴ کاربرد در ماتریس‌های بزرگ و نادر

الگوریتم گولوب ـ کاهان به خوبی برای کار با ماتریس‌های بزرگ و نادر (Sparse) بهینه‌سازی شده است.
دلیل:

- *مدیریت حافظه*: این الگوریتم از بازتاب‌های هوسهولدر و ماتریس‌های بیدیگونال برای کاهش نیاز به حافظه استفاده می‌کند. با این کار، فضای حافظه مورد نیاز برای ذخیره و پردازش ماتریس‌ها بهینه‌سازی شده و امکان پردازش ماتریس‌های بسیار بزرگ فراهم می‌شود.

- *بهینه‌سازی برای ماتریس‌های نادر*: الگوریتم می‌تواند به صورت بهینه ماتریس‌های نادر را پردازش کند. این مسئله باعث می‌شود تا زمان و حافظه صرف شده برای پردازش ماتریس‌هایی که بخش زیادی از عناصرشان صفر است، کاهش یابد.

۵.۴ مزایای کلی الگوریتم گولوب ـ کاهان

با توجه به مزایای فوق، الگوریتم گولوب ـ کاهان یکی از ابزارهای قدرتمند در ریاضیات عددی و پردازش داده‌ها است. این الگوریتم در بسیاری از کاربردهای عملی مانند تحلیل داده‌های بزرگ، فشرده‌سازی داده‌ها، و تجزیه و تحلیل تصاویر استفاده می‌شود. پایداری عددی، کارایی محاسباتی، دقت نتایج و توانایی کار با ماتریس‌های بزرگ و نادر، از جمله ویژگی‌های برجسته‌ای هستند که این الگوریتم را از دیگر روش‌های مشابه متمایز می‌کنند. این ویژگی‌ها سبب شده‌اند که الگوریتم گولوب ـ کاهان به عنوان یکی از بهترین روش‌ها برای محاسبه SVD در مسائل عملی و تحقیقاتی شناخته شود.

۵ مثال عددی از الگوریتم بیدیاگونالیزاسیون گولوب_کاهان

فرض کنید ماتریس A به صورت زیر باشد:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

گام اول: بازتاب هوسهولدر از چپ $(\mathbf{i = 1})$

$$B = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

$$x = \begin{bmatrix} 1 \\ 4 \\ 7 \end{bmatrix}$$

$$v = \begin{bmatrix} 0.75 \\ 0.33 \\ 0.57 \end{bmatrix}$$

$$H = I - 2\frac{vv^T}{v^Tv} = \begin{bmatrix} -0.12 & -0.49 & -0.86 \\ -0.49 & 0.78 & -0.38 \\ -0.86 & -0.38 & 0.34 \end{bmatrix}$$

$$\text{ضرب} H \cdot B = \begin{bmatrix} -0.12 & -0.49 & -0.86 \\ -0.49 & 0.78 & -0.38 \\ -0.86 & -0.38 & 0.34 \end{bmatrix} \cdot \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} = \begin{bmatrix} -8.12 & -9.6 & -11.08 \\ 0 & -0.09 & -0.17 \\ 0 & -0.9 & -1.8 \end{bmatrix}$$

$$\text{بروزرسانی} U = I \cdot H = \begin{bmatrix} -0.12 & -0.49 & -0.86 \\ -0.49 & 0.78 & -0.38 \\ -0.86 & -0.38 & 0.34 \end{bmatrix}$$

گام دوم: بازتاب هوسهولدر از راست $(\mathbf{i} = \mathbf{0})$

$$B = \begin{bmatrix} -8.12 & -9.6 & -11.08 \\ 0 & -0.09 & -0.17 \\ 0 & -0.9 & -1.8 \end{bmatrix}$$

$$x = \begin{bmatrix} -9.6 \\ -11.08 \end{bmatrix}$$

$$v = \begin{bmatrix} -0.91 \\ -0.42 \end{bmatrix}$$

$$H = I - 2\frac{vv^T}{v^Tv} = \begin{bmatrix} -0.65 & -0.76 \\ -0.76 & 0.65 \end{bmatrix}$$

$$\text{ضرب} B \cdot H = \begin{bmatrix} -9.6011363 & -11.07823419 \\ -0.08596557 & -0.17193114 \\ -0.90043975 & -1.8008795 \end{bmatrix} \cdot \begin{bmatrix} -0.65 & -0.76 \\ -0.76 & 0.65 \end{bmatrix} = \begin{bmatrix} -8.12 & 14.66 & -0 \\ 0 & 0.19 & -0.05 \\ 0 & 1.95 & -0.5 \end{bmatrix}$$

$$\text{بروزرسانی} V = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -0.65 & -0.76 \\ 0 & -0.76 & 0.65 \end{bmatrix}$$

گام سوم: بازتاب هوسهولدر از چپ $(\mathbf{i} = \mathbf{1})$

$$B = \begin{bmatrix} -8.12 & 14.66 & -0 \\ 0 & 0.19 & -0.05 \\ 0 & 1.95 & -0.5 \end{bmatrix}$$

$$x = \begin{bmatrix} 0.19 \\ 1.95 \end{bmatrix}$$

$$v = \begin{bmatrix} 0.74 \\ 0.67 \end{bmatrix}$$

$$H = I - 2\frac{vv^T}{v^Tv} = \begin{bmatrix} -0.1 & -1 \\ -1 & 0.1 \end{bmatrix}$$

$$\text{ضرب} H \cdot B = \begin{bmatrix} -0.1 & -1 \\ -1 & 0.1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0.19 & -0.05 \\ 0 & 1.95 & -0.5 \end{bmatrix} = \begin{bmatrix} -8.12 & 14.66 & -0 \\ -0 & -1.96 & 0.5 \\ -0 & -0 & 0 \end{bmatrix}$$

$$\text{بروزرسانی} U = \begin{bmatrix} -0.49 & -0.86 \\ 0.78 & -0.38 \\ -0.38 & 0.34 \end{bmatrix} \cdot \begin{bmatrix} -0.1 & -1 \\ -1 & 0.1 \end{bmatrix} = \begin{bmatrix} -0.12 & 0.9 & 0.41 \\ -0.49 & 0.3 & -0.82 \\ -0.86 & -0.3 & 0.41 \end{bmatrix}$$

گام چهارم: بازتاب هوسهولدر از راست $(\mathbf{i = 1})$

$$B = \begin{bmatrix} -8.12 & 14.66 & -0 \\ -0 & -1.96 & 0.5 \\ -0 & -0 & 0 \end{bmatrix}$$
$$x = \begin{bmatrix} 0.5 \end{bmatrix}$$
$$v = \begin{bmatrix} 1 \end{bmatrix}$$
$$H = I - 2\frac{vv^T}{v^Tv} = \begin{bmatrix} -1 \end{bmatrix}$$
$$\text{ضرب} B \cdot H = \begin{bmatrix} -8.12 & 14.66 & -0 \\ -0 & -1.96 & 0.5 \\ -0 & -0 & 0 \end{bmatrix} \cdot \begin{bmatrix} -1 \end{bmatrix} = \begin{bmatrix} -8.12 & 14.66 & 0 \\ -0 & -1.96 & -0.5 \\ -0 & -0 & -0 \end{bmatrix}$$
$$\text{بروزرسانی} V = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -0.65 & 0.76 \\ 0 & -0.76 & -0.65 \end{bmatrix}$$

گام پنجم: بازتاب هوسهولدر از چپ $(\mathbf{i = 2})$

$$B = \begin{bmatrix} -8.12 & 14.66 & 0 \\ -0 & -1.96 & -0.5 \\ -0 & -0 & -0 \end{bmatrix}$$
$$x = \begin{bmatrix} -0 \end{bmatrix}$$
$$v = \begin{bmatrix} -1 \end{bmatrix}$$
$$H = I - 2\frac{vv^T}{v^Tv} = \begin{bmatrix} -1 \end{bmatrix}$$
$$\text{ضرب} H \cdot B = \begin{bmatrix} -1 \end{bmatrix} \cdot \begin{bmatrix} -0 & -0 & -0 \end{bmatrix} = \begin{bmatrix} -8.12 & 14.66 & 0 \\ -0 & -1.96 & -0.5 \\ 0 & 0 & 0 \end{bmatrix}$$
$$\text{بروزرسانی} U = \begin{bmatrix} 0.41 \\ -0.82 \\ 0.41 \end{bmatrix} \cdot \begin{bmatrix} -1 \end{bmatrix} = \begin{bmatrix} -0.12 & 0.9 & -0.41 \\ -0.49 & 0.3 & 0.82 \\ -0.86 & -0.3 & -0.41 \end{bmatrix}$$

در پایان، ماتریس B بیدیاگونال به صورت زیر است:

$$B = \begin{bmatrix} -8.12 & 14.66 & 0 \\ -0 & -1.96 & -0.5 \\ 0 & 0 & 0 \end{bmatrix}$$

۶ پیوست

```python
import numpy as np
```

Implementation

```python
def householder_reflection(a):
    """
    Create a Householder reflection matrix for a vector a.

    Parameters:
    a (numpy.ndarray): Input vector.

    Returns:
    v (numpy.ndarray): Householder reflection vector.
    """

    v = a.copy()
    v[0] += np.sign(a[0]) * np.linalg.norm(a)
    v = v / np.linalg.norm(v)
    return v


def golub_kahan_bidiagonalization(A):

    """
    Perform Golub-Kahan bidiagonalization on matrix A.

    Parameters:
    A (numpy.ndarray): Input matrix with shape (m, n).

    Returns:
    B (numpy.ndarray): Bidiagonal matrix.
    U (numpy.ndarray): Orthogonal matrix U.
    V (numpy.ndarray): Orthogonal matrix V.
    """

    m, n = A.shape
    U = np.eye(m)
    V = np.eye(n)
    B = A.copy()

    for i in range(min(m, n)):
        # Hosholder's reflection from the left
        x = B[i:, i]
        v = householder_reflection(x)
        H = np.eye(m - i) - 2 * np.outer(v, v)
        B[i:, :] = np.dot(H, B[i:, :])
        U[:, i:] = np.dot(U[:, i:], H)
```

```python
        if i < n - 1:
            # Hosholder's reflection from the right
            x = B[i, i+1:]
            v = householder_reflection(x)
            H = np.eye(n - (i+1)) - 2 * np.outer(v, v)
            B[:, i+1:] = np.dot(B[:, i+1:], H)
            V[:, i+1:] = np.dot(V[:, i+1:], H)



    return B, U, V

def compute_svd_from_bidiagonal(B, U, V):
    m, n = B.shape
    U_b, Sigma, Vt_b = np.linalg.svd(B)
    U = np.dot(U, U_b)
    V = np.dot(V, Vt_b.T)
    return U, Sigma, V.T



def reconstruct_matrix(U, Sigma, Vt):
    Sigma_mat = np.zeros((U.shape[0], Vt.shape[1]))
    np.fill_diagonal(Sigma_mat, Sigma)
    A_reconstructed = np.dot(U, np.dot(Sigma_mat, Vt))
    return A_reconstructed
```

Example

```python
A = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]], dtype=float)
A
```

```
array([[1., 2., 3.],
       [4., 5., 6.],
       [7., 8., 9.]])
```

```python
B, U, V = golub_kahan_bidiagonalization(A)
U_svd, Sigma, Vt_svd = compute_svd_from_bidiagonal(B, U, V)
A_reconstructed = reconstruct_matrix(U_svd, Sigma, Vt_svd)
A_reconstructed
```

```
array([[1., 2., 3.],
       [4., 5., 6.],
       [7., 8., 9.]])
```

```python
np.round(Sigma, 3)
```

```
array([16.848,  1.068,  0.   ])
```

 مراجع

مراجع

• G. Golub and W. Kahan, Calculating the Singular Values and Pseudo-Inverse of a Matrix, SIAM. J. Numer. Anal., Vol. 2, 1965, pp. 202-224 • J. Wilkinson, Calculation of the eigenvalues of a symmetric tridiagonal matrix by the method of bisection, Numerische Mathematik, Vol. 4, 1962, pp. 362-367. • G. Golub and P. Businger, Least Squares, Singular Values and Matrix Approximations, Stanford Technical Report No. CS73, 1967 • G. H. Golub and C. Reinsch, Singular Value Decomposition and Least Squares Solutions, Numer. Math., Vol. 14, 1970, pp. 403-420. • J. Wilkinson and Chr. Reinsch, Linear Algebra, Springer, 1971. • Raymond H. Chan, Chen Greif and Dianne P. O'Leary eds., Milestones in Matrix Computation: Selected Works of Gene H. Golub, with Commentaries, Oxford University Press, 2007.