Power method

- Power iteration:

  Given an initial vector $u_0$,
  
  $i = 0$

  repeat

  $t_{i+1} = Au_i$

  $u_{i+1} = t_{i+1}/\|t_{i+1}\|_2$  (approximate eigenvector)

  $\theta_{i+1} = u_{i+1}^HAu_{i+1}$  (approximate eigenvalue)

  $i = i + 1$

  until convergence

- Simple stopping criterion: $|\theta_{i+1} - \theta_i| \leq tol \cdot |\theta_i|$. 
Power method

Example: Let

\[
A = \begin{bmatrix}
-261 & 209 & -49 \\
-530 & 422 & -98 \\
-800 & 631 & -144
\end{bmatrix}.
\]

and \( \lambda(A) = \{10, 4, 3\} \). Let \( u_0 = (1, 0, 0)^T \), then

<table>
<thead>
<tr>
<th>( i )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>( \cdots )</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \theta_i )</td>
<td>994.49</td>
<td>13.0606</td>
<td>10.07191</td>
<td>( \cdots )</td>
<td>10.0002</td>
</tr>
</tbody>
</table>
Power method

Convergence analysis: Assume that $A$ is diagonalizable, i.e.,

$$A = X \Lambda X^{-1}$$

with $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ and $|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_n|$. Then, we can show that

- $u_i = \frac{A^i u_0}{\|A^i u_0\|} \to x_1/\|x_1\|$, where $x_1 = X e_1$ as $i \to \infty$.
- $\theta_i \to \lambda_1$ as $i \to \infty$.
- The convergence rate depends on $\frac{|\lambda_2|}{|\lambda_1|}$.

Therefore, if $\frac{|\lambda_2|}{|\lambda_1|}$ is close to 1, then the power method could be very slow convergent or doesn’t converge at all.
Inverse iteration

Purposes:

▶ Overcome the drawbacks of the power method (slow convergence)
▶ find an eigenvalue closest to a particular given number (called shift): $\sigma$

Observation: if $\lambda$ is an eigenvalue of $A$, then

▶ $\lambda - \sigma$ is an eigenvalue of $A - \sigma I$,
▶ $\frac{1}{\lambda - \sigma}$ is an eigenvalue of $(A - \sigma I)^{-1}$.
Inverse iteration

Given an initial vector $u_0$ and a shift $\sigma$

$i = 0$

repeat

solve $(A - \sigma I)t_{i+1} = u_i$ for $t_{i+1}$

$u_{i+1} = t_{i+1}/\|t_{i+1}\|_2$ (approximate eigenvector)

$\theta_{i+1} = u_{i+1}^H A u_{i+1}$ (approximate eigenvalue)

$i = i + 1$

until convergence
Inverse iteration

Convergence analysis: Assume $A = X\Lambda X^{-1}$ with $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ and $\lambda_k$ is the eigenvalue closest to the shift $\sigma$. It can be shown that

- $u_i \to x_k/\|x_k\|$ as $i \to \infty$, where $x_k = Xe_k$
- $\theta_i$ converges to $\lambda_k$ as $i \to \infty$.
- Convergence rate depends on $\max_{j \neq k} \frac{|\lambda_k - \sigma|}{|\lambda_j - \sigma|}$. 
Inverse iteration

1. Advantages:
   - the ability to converge to any desired eigenvalue nearest to the shift $\sigma$;
   - typically converges very quickly, and is particularly effective when we have a good approximation to an eigenvalue and want only its corresponding eigenvector.

2. Drawbacks:
   - expensive in general: solving linear systems $(A - \sigma I)t_{i+1} = u_i$ for $u_{i+1}$.
     One LU factorization of $A - \sigma I$ is required, which could be very expensive for large matrices
   - Only compute one eigenpair.
Orthogonal (subspace/simultaneous) iteration

- Purpose: compute $p$ eigenvalues (and eigenvectors) at a time.

- Orthogonal iteration
  
  Given an initial $n \times p$ orthogonal matrix $Z_0$
  
  $i = 0$
  
  repeat
  
  $Y_{i+1} = AZ_i$
  $Y_{i+1} = Z_{i+1}R_{i+1}$ (QR decomposition)
  
  $i = i + 1$
  
  until convergence

- The use of QR decomposition keeps the vectors spanning $\text{span}\{A^iZ_0\}$ of full rank.
Example: Let $Z_0 = [e_1, e_2, e_3]$ and

$$A = \begin{bmatrix}
-0.4326 & 1.1892 & -0.5883 & -0.0956 & -0.6918 & -0.3999 \\
-1.6656 & -0.0376 & 2.1832 & -0.8323 & 0.8580 & 0.6900 \\
0.1253 & 0.3273 & -0.1364 & 0.2944 & 1.2540 & 0.8156 \\
0.2877 & 0.1746 & 0.1139 & -1.3362 & -1.5937 & 0.7119 \\
-1.1465 & -0.1867 & 1.0668 & 0.7143 & -1.4410 & 1.2902 \\
1.1909 & 0.7258 & 0.0593 & 1.6236 & 0.5711 & 0.6686
\end{bmatrix}$$

Eigvals of $A$ = -2.1659+-0.5560i, 2.1493, 0.2111+-1.9014i, -0.9548

i=10: Eigvals of $Z'_10*A*Z_10$: -1.4383+-0.3479i, 2.1500

i=30: Eigvals of $Z'_30*A*Z_30$: -2.1592+-0.5494i, 2.1118

i=70: Eigvals of $Z'_70*A*Z_70$: -2.1659+-0.5560i, 2.1493
Orthogonal (subspace/simultaneous) iteration

Convergence results:

- under mild conditions, $Z_i$ converges to a subspace spanned by the first $p$ eigenvectors corresponding to the $p$ dominant eigenvalues, where

$$|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_p| > |\lambda_{p+1}| \geq \cdots \geq |\lambda_n|.$$ 

- If we let $B_i = Z_i^T A Z_i$, then

$$\|A Z_i - Z_i B_i\| \to 0 \text{ as } i \to \infty$$

and eigenvalues of $B_i$ approximate the dominant eigenvalues of $A$.

- Convergence rate depends on $|\lambda_{p+1}|/|\lambda_p|$. 


QR iteration

- Goal: reorganize orthogonal iteration to incorporate shifting and inverting as in the inverse iteration. This will make it more efficient and eliminate the assumption that eigenvalues differ in magnitude.

- QR iteration

\[ A_0 = A \]
\[ i = 0 \]

repeat

\[ A_i = Q_i R_i \quad \text{(QR decomposition)} \]
\[ A_{i+1} = R_i Q_i \]
\[ i = i + 1 \]

until convergence
QR iteration

Properties:

- Observe that $A_{i+1} = R_i Q_i = Q_i^T Q_i R_i Q_i = Q_i^T A_i Q_i$. Therefore it performs an orthogonal similarity transformation at each iteration.

- $A_{i+1}$ is orthogonally similar to $A_0 = A$. Therefore $A_{i+1}$ and $A$ have same eigenvalues:

$$A_{i+1} = (Q_0 Q_1 \cdots Q_{i-1} Q_i)^T A (Q_0 Q_1 \cdots Q_{i-1} Q_i).$$

Note that $Q_0 \cdots Q_{i-1} Q_i$ is an orthogonal matrix since all $Q_j$ are.
Example. The same test matrix, numerical results of QR iteration

$$A_{10} =$$

\[
\begin{bmatrix}
-1.6994 & 0.2201 & -0.8787 & -1.4292 & -0.3847 & 0.0112 \\
-0.0007 & 1.1325 & 1.2186 & 1.2245 & -0.0867 & 0.0648 \\
0.2637 & 1.9636 & -0.1598 & -2.3959 & 0.8136 & -0.4311 \\
0.0364 & -0.2346 & -0.5527 & -0.4393 & -1.9263 & 1.2496 \\
0.4290 & 1.3482 & -1.1484 & 0.6121 & -0.5937 & 0.2416 \\
0.0003 & 0.0013 & -0.0003 & -0.0011 & 0.0014 & -0.9554
\end{bmatrix}
\]

$$A_{30} =$$

\[
\begin{bmatrix}
-2.4055 & -1.0586 & 1.3420 & -0.0991 & 1.1210 & 0.1720 \\
0.0517 & 0.9645 & 1.6519 & -0.8512 & -0.7215 & 0.7654 \\
-0.2248 & 1.9947 & -0.7656 & -1.1876 & -0.2736 & -0.1552 \\
-0.0029 & -0.0263 & -0.0682 & 0.1381 & -2.3094 & 0.6765 \\
-0.0147 & 0.0808 & -0.0569 & 1.5462 & 0.3082 & -0.8476 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & -0.9548
\end{bmatrix}
\]

From the last two of $A_{30}$, we can conclude -0.9548 is an eigenvalue of $A$. The subsequent QR iterations are performed on the leading $4 \times 4$ submatrices to find the rest of eigenvalues.
1. Purpose: accelerate the convergence of QR iteration by using shifts

2. QR Iteration with shifts

\[
A_0 = A; \ i = 0
\]

repeat

Choose a shift \( \sigma_i \)

\[
A_i - \sigma_i I = Q_i R_i \quad \text{(QR decomposition)}
\]

\[
A_{i+1} = R_i Q_i + \sigma_i I
\]

\[
i = i + 1
\]

until convergence

3. Property: \( A_i \) and \( A_{i+1} \) are orthogonally similar: 

\[
A_{i+1} = Q_i^T A_i Q_i.
\]

Therefore, \( A_{i+1} \) and \( A \) are orthogonally similar, and \( A_{i+1} \) and \( A \) have the same eigenvalues.
QR iteration with shifts ⇒ QR Algorithm

How to choose the shifts $\sigma_i$?

- If $\sigma_i$ is an exact eigenvalue of $A$, then it can be shown that

$$A_{i+1} = R_i Q_i + \sigma_i I = \begin{bmatrix} A' & a \\ 0 & \sigma_i \end{bmatrix}.$$ 

This means that the algorithm converges in one iteration. If more eigenvalues are wanted, we can apply the algorithm again to the $n - 1$ by $n - 1$ matrix $A'$.

- In practice, a common choice of the $\sigma_i$ is

$$\sigma_i = A_i(n,n).$$

A motivation of this choice is by observing that the convergence of the QR iteration (without a shift), the $(n, n)$ entry of $A_i$ usually converges to an eigenvalue of $A$ first.
Example. The same test matrix as before. The following is the numerical result of QR iteration with a shift. With the shift $\sigma_0$ as an exact eigenvalue 2.1493, i.e.,

$$\sigma_0 = 2.1493,$$

then

$$A_{-1} =
\begin{bmatrix}
-1.4127 & 1.4420 & 1.0845 & -0.6866 & -0.1013 & -0.2042 \\
-1.2949 & -0.2334 & 1.4047 & -1.3695 & 1.5274 & -0.7062 \\
0.5473 & 0.1343 & -0.7991 & -0.6716 & 1.1585 & 0.0736 \\
-0.2630 & 0.0284 & 0.5440 & -1.4616 & -1.5892 & 0.9205 \\
-1.6063 & -0.3898 & 0.3410 & 0.1623 & -0.9576 & -0.5795 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 2.1493
\end{bmatrix}$$

We observe that by the QR iteration converged in one-iteration.
QR iteration with shifts $\Rightarrow$ QR Algorithm

With the shifts $\sigma_i = A_i(n, n)$.

$$A_{-7} =\
\begin{bmatrix}
-2.4302 & 2.0264 & -0.2799 & -0.2384 & 0.3210 & -0.0526 \\
-0.1865 & -1.4295 & -1.3515 & 0.0812 & 0.8577 & -0.0388 \\
-0.1087 & -0.8991 & 0.4491 & 0.4890 & -1.8463 & -1.2034 \\
-0.0008 & 0.0511 & -0.5997 & -0.7839 & -0.8088 & -0.5188 \\
-0.0916 & -0.8273 & 1.6940 & 0.0645 & -0.6698 & -0.0854 \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 2.1493
\end{bmatrix}$$

We observe that by 7th iteration, from the last rown, we have found an eigenvalue 2.1493 of $A$. The subsequent QR iterations with shift are performed on the leading $4 \times 4$ submatrices to find the rest of eigenvalues.
QR iteration with shifts ⇒ QR Algorithm

Remarks:

1. The QR decomposition takes $\mathcal{O}(n^3)$ flops. Even if the QR iteration took $n$ iterations to converge, the overall cost will be $\mathcal{O}(n^4)$. This is too expensive.\(^1\)

2. However, if the matrix is initially reduced to upper Hessenberg form, then the QR decomposition of a Hessenberg form costs $\mathcal{O}(n^2)$ flops. As a result, the overall cost of the algorithm is reduced to $\mathcal{O}(n^3)$. This is referred to as the Hessenberg QR algorithm.

3. The Hessenberg QR algorithm is the method of choice for dense eigenvalue problem today, and is considered as one of the top 10 algorithms invented in the 20th century.

\(^1\)The complexity of algorithms for all standard matrix computation problems is at $\mathcal{O}(n^3)$. 