Computational Methods

Eigenvalues and Singular Values
Eigenvalues and Singular Values

- Eigenvalues and singular values describe important aspects of transformations and of data relations
  - Eigenvalues determine the important the degree to which a linear transformation changes the length of transformed vectors
  - Eigenvectors indicate the directions in which the principal change happen
- Eigenvalues are important for many problems in computer science and engineering, including
  - Dimensionality reduction
  - Compression
Eigenvalues

- Eigenvalues $\lambda$ and eigenvectors $x$ characterize dimensions that are purely stretched by a given linear transformation

$$Ax = \lambda x$$

- The spectrum of $A$ is the set of its eigenvalues
- The spectral radius of $A$ is the magnitude of the largest of its eigenvalues
- Eigenvalues characterize the degree to which a linear transformation stretches input vectors
  - Also important for sensitivity analysis of linear problems
Eigenvalues

- A linear transformation has as many eigenvalues and eigenvectors as it has dimensions
  - Eigenvectors might be duplicates
  - Eigenvalues might be complex
- Any data point (vector) can be written as a linear combination of eigenvectors
  - Allows efficient decomposition of vectors
Power Iteration

- The eigenvalue equation is related to the fixed point equations (except with scaling)
  \[ Ax = \lambda x \]
  - Simplest solution method to find eigenvectors (and eigenvalues) is power iteration
  - Characterize dimensions that are purely stretched by a given linear transformation
- Power iteration converges to a scaled version of the eigenvector with the dominant eigenvalue
  \[ x_{t+1} = Ax_t \]
Power Iteration

- Power iteration converges except if
  - $x_0$ has no component of the dominant eigenvector
  - There are more than one eigenvector with the same eigenvalue

- Normalized power iteration renormalizes the result $x_{t+1}$ after each iteration

$$y_{k+1} = Ax_k \quad , \quad x_{k+1} = \frac{y_{k+1}}{\|y_{k+1}\|_\infty}$$

- Converges to dominant eigenvector and dominant eigenvalue

$$\|y_k\|_\infty \rightarrow \lambda_d \quad , \quad x_k \rightarrow \frac{1}{\|v_d\|_\infty} v_d$$
Inverse Iteration

- Inverse iteration is used to find the smallest eigenvalue
- Converges except if
- \( Ay_{k+1} = x_k \), \( x_{k+1} = \frac{y_{k+1}}{\|y_{k+1}\|\_\infty} \)
  - Inverse iteration corresponds to power iteration with the inverse matrix \( A^{-1} \)
  - Inverse iteration and power iteration can only find the smallest and the largest eigenvalues
    - Need to find a way to determine other eigenvalues and eigenvectors
Characteristic Polynomial

- The determination of eigenvectors and eigenvalues can be transformed into a root finding problem
  \[(A - \lambda I)x = 0\]

  - Has a nonzero solution for the eigenvector \(x\) if and only if \((A - \lambda I)\) is not singular
  - Eigenvalues of the nonsingular matrix are the roots of the characteristic polynomial
    \[\det(A - \lambda I) = 0\]
    - The characteristic polynomial is a polynomial of degree \(n\)
    - Complex eigenvalues occur in conjugate pairs

- Computation of the characteristic polynomial is complex
  - Can be accelerated by first performing LU factorization
Characteristic Polynomial

- Computing roots of a polynomial of degree larger than 4 cannot always be computed directly and require an iterative solution
- Computing eigenvalues using the characteristic polynomial is numerically not stable and highly complex
  - Computing coefficients of characteristic polynomial requires computation of the determinant
  - Root finding requires iterative solution process
  - Coefficients of characteristic are very sensitive
- Characteristic polynomial is a powerful theoretical tool but not a practical computational approach
Eigenvalue Problems

- Characteristics of eigenvalue problems influence the choice of algorithm
  - All or only some eigenvalues
  - Only eigenvalues or eigenvalues and eigenvectors
  - Dense or sparse matrix
  - Real of complex values
  - Other properties of matrix A
Problem Transformations

- A number of transformations either preserve or have a predictable effect on the eigenvalues
  - Shift: For any scalar $\sigma$
    \[ Ax = \lambda x \quad \rightarrow \quad (A - \sigma I)x = (\lambda - \sigma)x \]
  - Inversion:
    \[ Ax = \lambda x \quad \rightarrow \quad A^{-1}x = \frac{1}{\lambda}x \]
  - Powers:
    \[ Ax = \lambda x \quad \rightarrow \quad A^k x = \lambda^k x \]
  - Polynomial: for any polynomial $p(t)$
    \[ Ax = \lambda x \quad \rightarrow \quad p(A)x = p(\lambda)x \]
  - Similarity: for any similar matrix $B = T^{-1}AT$
    \[ Bx = \lambda x \quad \rightarrow \quad ATx = \lambda(Tx) \]
Problem Transformations

- Eigenvalues and eigenvectors of diagonal matrices are easy to determine
  - Eigenvalues are the values on the diagonal
  - Eigenvectors are the columns of the identity matrix
- Not all matrices are diagonalizable using similarity transformations
- Eigenvalues of triangular matrices can also be determined easily
  - Eigenvalues are diagonal entries of the matrix
  - Eigenvectors can be computed from \((A - \lambda I)x = 0\)
Convergence of Iterations

- Speed of convergence of power iteration and inverse iteration depends on the ratio of two eigenvalues
  - For power iteration, convergence is faster the larger the ratio of the largest and the second largest eigenvalue is
  - For inverse iteration, convergence is faster the smaller the ratio of the smallest and the second smallest eigenvector is
- Shift transformation allows to change the ratio of eigenvalues
  \[
  \frac{\lambda_1}{\lambda_2} \rightarrow \frac{\lambda_1 - \sigma}{\lambda_2 - \sigma}
  \]
  - Knowledge of eigenvalue of sought after eigenvector would allow to lower this ratio to 0
    - Allows to increase the convergence rate of inverse iteration
Rayleigh Quotient Iteration

- Rayleigh quotient iteration uses the Rayleigh quotient as a shift parameter \( \sigma = \frac{x^T Ax}{x^T x} \), \((A - \sigma I)\)

- This allows to make the ratio of eigenvalues close to 0 and thus accelerates the convergence of inverse iteration

\[
(A - \sigma_k I)y_{k+1} = x_k
\]

\[
x_{k+1} = \frac{y_{k+1}}{\|y_{k+1}\|_\infty}
\]

- This algorithm is usually called Rayleigh quotient iteration

- Rayleigh quotient iteration converges usually very fast
  - Each iteration requires a new matrix factorization and is therefore \(O(n^3)\) F
Computing All Eigenvalues

- Power iteration and inverse iteration allow to compute only the largest and the smallest eigenvalues and eigenvectors.

  - To compute the other eigenvalues we need to either
    - Remove the already found eigenvector (and eigenvalue) from the matrix to be able to reapply power or inverse iteration
    - Find a way to find all the eigenvectors simultaneously
  - Removing eigenvectors from the space spanned by a transformation $A$ is called deflation
Deflation

- To remove an eigenvalue (and corresponding eigenvector) we have to find a set of transformations that preserves all other eigenvalues
  - Householder transforms can be used to derive such a transformation \( H \) with
    \[
    Hx_1 = \alpha e_1
    \]
  - The similarity transform described by \( H \) yields a matrix
    \[
    HAH^{-1} = \begin{pmatrix}
    \lambda & b^T \\
    0 & B
    \end{pmatrix}
    \]
    - Since similarity transforms were used this matrix has the same eigenvalues
    - \( B \) has all the eigenvalues of \( A \) with the exception of \( \lambda_1 \)
  - Power iteration can be applied to this new matrix \( B \)
Deflation

- Power iteration with deflation can compute all eigenvalues but requires determining the eigenvector in each iteration
  - Eigenvector in B can be used to compute eigenvector in A
    \[
    x_3 = H^{-1} \begin{pmatrix} \frac{b^T y_2}{\lambda_2 - \lambda_1} \\ y_2 \end{pmatrix}
    \]
  - Alternatively, the eigenvalue could be used directly in A to determine the eigenvector
    - More computationally complex
Simultaneous Iteration

- Simultaneous iteration attempts to simultaneously iterate multiple vectors

\[ X_{k+1} = AX_k \]

- \( X \) converges to the space spanned by the \( p \) dominant eigenvectors
  - Subspace iteration

- But \( X \) becomes ill-conditioned since all columns in \( X \) ultimately converge to the dominant eigenvector
  - Need normalization that keeps vectors well conditioned and non-equal
    - Orthogonal iteration using QR factorization
QR Iteration

- As for least squares (and equation solving) QR factorization allows a factorization of the matrix into components that stay well conditioned

\[ Q_{k+1}R_{k+1} = X_k \]
\[ X_{k+1} = AQ_{k+1} \]

- By using Q (a similarity transform) for the iteration, the eigenvalues are preserved and it converges to block triangular form
  - Triangular form if all eigenvalues are real values and distinct
QR Iteration

- To find eigenvalues, QR iteration can be applied directly to $A$
  \[ A_k = Q_k^H A_{k-1} Q_k \]
  - Converges to triangular or block triangular matrix containing all eigenvalues as diagonal elements of diagonal blocks
    - Can be computed without explicitly performing the product
    \[ Q_{k+1} R_{k+1} = A_k \]
  \[ A_{k+1} = R_{k+1} Q_{k+1} (= Q_{k+1}^H A_k Q_{k+1}) \]
- Can be accelerated using shift transformation
Singular Values

- Singular values are related to Eigenvalues and characterize important aspects of the space described by the transformation
  - Nullspace
  - Span
- Singular Value Decomposition divides a transformation $A$ into a sequence of 3 transformations where the second is pure rescaling
  - Scaling parameters are the singular values
  - Columns of the other two transformations are the left and right singular vectors, respectively
Singular Values

- Singular values exist for all transformations A, independent of A being square or not
  - Right singular vectors represent the input vectors that span the orthogonal basis that is being scaled
  - Left singular vectors represent the vectors that the scaled internal basis vectors are transformed into for the output
- Singular values are directly related to the eigenvalues
  - Singular values are the nonnegative square roots of the eigenvalues of $AA^T$ or $A^TA$
  - Left singular vectors are eigenvectors of $AA^T$
  - Right singular vectors are eigenvectors of $A^TA$
Singular Value Decomposition

- Singular value decomposition (SVD) factorizes $A$
  \[
  A = U \Sigma V^T
  \]
  - $U$ is an $m \times m$ orthogonal matrix of left singular vectors
  - $V$ is an $n \times n$ orthogonal matrix of right singular vectors
  - $\Sigma$ is an $m \times n$ diagonal matrix of singular values
    - Usually $\Sigma$ is arranged such that the singular values are ordered by magnitude
  - Left and right singular vectors are related through the singular values
    \[
    Av_{i} = \sigma_{i}u_{i}
    \]
    \[
    A^T u_{i} = \sigma_{i}v_{i}
    \]
Singular Value Decomposition

- Singular value decomposition (SVD) can be computed in different ways
  - Using eigenvalue computation on $AA^T$
    - Compute eigenvalues of $AA^T$
    - Determine left singular vectors as eigenvectors for $AA^T$
    - Determine right singular vectors as eigenvectors for $A^TA$
    - Leads to some conditioning issues due to the need for matrix multiplication
  - Directly from $A$ by performing Householder transformations and givens rotations until a diagonal matrix is reached
    - Perform QR factorization to achieve triangular matrix
    - Use Householder transforms to achieve bidiagonal shape
    - Use Givens rotations to achieve diagonal form
    - This is usually better conditioned
Singular Value Decomposition

- Singular value decomposition (SVD) can be used for a range of applications
  - Compute least squares solution \( Ax \equiv b \rightarrow x = \sum_{\sigma_i \neq 0} \frac{u_i^T b}{\sigma_i} v_i \)
  - Compute pseudoinverse \( A^+ = V \Sigma^+ U^T \)
  - Euclidean matrix norm: \( \|A\|_2 = \sigma_{\text{max}} \)
  - Condition number of a matrix: \( \text{cond}(A) = \sigma_{\text{max}} / \sigma_{\text{min}} \)
  - Matrix rank is equal to the number of non-zero singular values
  - Nullspace of the matrix is spanned by the set of right singular vectors corresponding to singular values of 0
  - Span of a matrix is spanned by the left singular vectors corresponding to non-zero singular values
Singular Value Decomposition

- Singular value decomposition (SVD) is useful in a number of applications

  - Data compression
    - Right singular values transform data into a basis in which it is only scaled
    - Data dimensions with 0 or very small scaling factors are not important for the overall data
  
  - Wide range of applications:
    - Image compression
    - Dimensionality reduction for data
    - Dimensionality reduction for matrix operations

- Filtering and noise reduction
  - Most of the time, data has only few important dimensions and noise is most apparent in additional dimensions (with smaller singular values)
  - Ignoring dimensions with small singular values can lead to less noisy data
Compression Example

- Image compression is an area where SVD has been used relatively early on
  - Given an image, can we reduce the amount of data that has to be transmitted without losing too much information
    - Use SVD to find a lower rank approximation of the image that has only limited loss.
Compression Example

- In SVD, the magnitude of the singular values often decreases rapidly after the first few singular values.

- To compress the image, only keep the $k$ largest singular values (and thus singular vectors) to reconstruct the image

$$A \approx U_p \Sigma_p V_p^T$$
Compression Example

- Different compression levels have different loss
Eigenvalues and Singular Values

- Eigenvalues and Eigenvectors capture important properties about linear transformations $A$.

- Eigenvalues and Singular values indicate the importance of particular dimensions of the space.
  - Can be used for compression.

- Singular values can capture noise characteristics.
  - Can be used for filtering of data.
  - Can be used to remove noise from data before transformations are applied.

- Singular values are also important to analyze problems such as conditioning and sensitivity.