Example

Consider interpolating a given set of 3 data points with a straight line \( y = c_1 + c_2 x \), i.e.

\[
\begin{pmatrix}
1 & x_1 \\
1 & x_2 \\
1 & x_3 \\
\end{pmatrix}
\begin{pmatrix}
c_1 \\
c_2 \\
\end{pmatrix}
=
\begin{pmatrix}
y_1 \\
y_2 \\
y_3 \\
\end{pmatrix}
\]

- If all three points happen to lie on a line, there is a unique solution
- If the three points do not lie on a line, there is no solution, but least squares can be used to minimize the residual
- If all three points were exactly the same, e.g. (4,3), (4,3), and (4,3), then the second column of the Vandermonde matrix is a multiple of the first, the matrix is only rank 1, and there are infinite solutions
  - The same equation is repeated three times and there is only one nonzero singular value
- We call systems without full rank underdetermined
- Note: just because some degrees of freedom are undeterminable does not mean that one always has infinite solutions, because other variables may still be overdetermined (with contradictions), e.g. \( \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix} \) is undetermined for \( c_2 \) but overdetermined for \( c_1 \)

Solving Linear Systems

- \( A \mathbf{c} = \mathbf{b} \) or \( U \Sigma V^T \mathbf{c} = \mathbf{b} \) or \( V^T \mathbf{c} = U^T \mathbf{b} \) or \( \Sigma \hat{\mathbf{c}} = \hat{\mathbf{b}} \)
- For each \( \sigma_i \), one has an \( \sigma_i \hat{c}_i = \hat{b}_i \)
- A tall mxn matrix \( A \) will have some extra rows of the form \( 0 = \hat{b}_i \) for \( i > n \)
  - Least squares when any \( \hat{b}_i \neq 0 \) for \( i > n \) (otherwise a unique solution)
- A wide mxn matrix will have some extra \( \hat{c}_i \) for \( i > m \) that do not appear in any of the equations
  - The wide mxn \( \Sigma \) has columns of zeroes for columns \( > m \), which multiply the \( \hat{c}_i \) that do not show up in any of the equations
  - Also, when some \( \sigma_i = 0 \), \( \Sigma \) has corresponding columns of zeroes for columns \( \leq m \), which multiply \( \hat{c}_i \) that do not show up in any of the equations
Minimum norm solution
• \( \hat{c}_i \) that do not show up in any of the equations are extraneous variables that have no bearing on the problem as posed
• Since these \( \hat{c}_i \) may have any value and still satisfy the equations, one typically sets them to be identically zero (implying that they make no contribution)
• This makes more sense than using another of the possible infinite solutions for these variables
• The remaining equations are solved to obtain \( \hat{c}_i = \frac{\hat{b}_i}{\sigma_i} \) where \( c = \Sigma V_i \hat{c}_i \) with \( V_i \) the columns of \( V \)

Minimum norm solution \( c = \Sigma \sigma_{l\neq 0} v_i \frac{(u_i^T b)}{\sigma_i} \)

Pseudo Inverse \( A^+ = \Sigma \sigma_{l \neq 0} u_i v_i^T \), since minimum norm solution has \( c = A^+ b \)
• Each term is an outer product matrix \( u_i v_i^T \) divided by \( \sigma_i \)
• When \( A^{-1} \) exists, \( A^+ = A^{-1} \)

Sum of Rank One Matrices
• \( Ac = U\Sigma V^T c = U\Sigma \hat{c} = U\Sigma \left( \hat{c}_1^T \right) \) where the \( \hat{c}_1 \) multiply columns with nonzero singular values and the \( \hat{c}_2 \) multiply columns of zeroes (when present)
• That is, \( Ac = U \left( D \hat{c}_1 \right) \) where \( D \) is a diagonal matrix of the nonzero singular values, and \( \hat{0} \) is a column vector of zeroes corresponding to the rows of zeros in \( \Sigma \)
• So \( Ac = \Sigma_{l} u_i (D\hat{c}_1)_i \) where \( i \) ranges over the columns \( u_i \) of \( U \) with nonzero singular values
• For each nonzero \( \sigma_i \), we have \( (D\hat{c}_1)_l = \sigma_i v_i^T c \) recalling that \( \hat{c} = V^T c \)
• So \( Ac = \Sigma_{\sigma_{l \neq 0}} u_i \sigma_i v_i^T c \) and \( A = \Sigma_{\sigma_{l \neq 0}} \sigma_i u_i v_i^T \)
• Each term is an outer product matrix \( u_i v_i^T \) multiplied by \( \sigma_i \)
• Each matrix is the size of \( A \), but has rank 1, since every column is a multiple of \( u_i \)

Approximating a Matrix
• Dropping small \( \sigma_i \) from the sum gives \( A \approx \Sigma_{\sigma_{l > \sigma_{min}}} \sigma_i u_i v_i^T \), which can be a very good approximation when some \( \sigma_i \) are much smaller than others
• Keeping the \( r \) largest singular values in the sum is considered the best rank \( r \) approximation to \( A \)
• Dropping small singular values, also makes the pseudo-inverse better conditioned, i.e. using \( A^+ \approx \Sigma_{\sigma_{l > \sigma_{min}}} \left( \frac{1}{\sigma_i} \right) v_i u_i^T \) instead of \( A^+ = \Sigma_{\sigma_{l \neq 0}} \left( \frac{1}{\sigma_i} \right) v_i u_i^T \)

Principal Component Analysis (PCA)
• Uses a finite number of low rank approximations, starting with the largest \( \sigma_i \)
• Can iteratively add the one \( \sigma_i \) term at a time \( A^+ b = \left( \frac{1}{\sigma_1} \right) v_1 u_1^T b + \left( \frac{1}{\sigma_2} \right) v_2 u_2^T b + \cdots \)
• Often times, thousands/millions of terms can be thrown away keeping only on the order of 10 to 100
Finding Low Rank Approximations

- \( A^T A = V \Sigma^T \Sigma V^T \) so \( A^T A V = V \Sigma^T \Sigma \) and \( A A^T = U \Sigma \Sigma^T U^T \) so \( A A^T U = U \Sigma \Sigma^T \)
- Thus, if \( \sigma_i \neq 0 \), then \( \sigma_i^2 \) is an eigenvalue of both \( A^T A \) and \( A A^T \) and the associated columns of \( V \) and \( U \) are the eigenvectors, respectively
- Both \( A^T A \) and \( A A^T \) are symmetric, but it is more efficient to find the eigenvalues using the smaller of the two
  - Then a known eigenvalue can be used to find both associated eigenvectors

Computing Eigenvalues

- \( A v = \lambda v \) or \( (A - \lambda I) v = 0 \) has a nonzero \( v \) only if the matrix is singular with \( det (A - \lambda I) = 0 \), which gives an \( n \)-th degree characteristic polynomial in \( \lambda \)
  - Finding roots of general polynomials is difficult (recall quadratic/cubic), so usually don’t go this route for larger \( n \)
- Similarity transforms preserve the eigenvalue problem
  - \( (T^{-1} A T) v = \lambda v \) implies \( A (T v) = \lambda (T v) \), which means that \( T^{-1} A T \) and \( A \) have the same eigenvalues; though, with associated eigenvectors are \( T v \) and \( v \) respectively
  - If \( A \) has distinct eigenvalues, similarity transforms can be used to make \( T^{-1} A T \) a diagonal matrix of eigenvalues
  - If \( A \) is real and symmetric (or complex and Hermitian), and orthogonal (unitary) \( T \) exists to put \( A \) into diagonal form; and the eigenvalues are real in both cases
    - E.g. \( T = V \) for \( A^T A \) and \( T = U \) for \( A A^T \)
  - Any matrix can be put into upper triangular, Shur form, with a unitary \( T \); then, the eigenvalues can be read off the diagonal
  - Any matrix can be put into Jordan form where the eigenvalues are on the diagonal, and off diagonal elements only occur on the band above the diagonal and only for defective eigenvalues (defective eigenvalues are repeated eigenvalues that don’t possess a full set of eigenvectors)

Condition Number

- The condition number for finding an eigenvalue/eigenvector pair is \( \frac{1}{|w^T v|} \) where \( v \) and \( w \) are the normalized right and left eigenvectors, respectively
- Symmetric/Hermitian matrices have the same left and right eigenvectors, i.e. \( v = w \), and so \( w^T v = 1 \) and the condition number is 1

QR iteration

- Set \( A_0 = A \), and then iterate \( A_1, A_2 \), etc.
- For each \( k \), compute the QR factorization \( A_k = Q_k R_k \), and then define \( A_{k+1} = R_k Q_k = (Q_k^H Q_k) R_k Q_k = Q_k^H A_k Q_k \)
  - If the eigenvalues are all distinct, then the \( A_k \) converge to a triangular matrix.
  - If \( A \) is symmetric, the \( A_k \) converge to a diagonal matrix
Power Method

- The **Power Method** allows one to compute the largest eigenvalue and its associated eigenvector.
- Starting from a nonzero vector \( x_0 \), iterate with \( x_{k+1} = Ax_k \).
- Assume \( x_0 \) is a linear combination of eigenvectors \( x_0 = \sum_i \alpha_i u_i \).
- Then \( x_k = A^k x_0 = A^k \sum_i \alpha_i u_i = \sum_i \alpha_i A^k u_i = \sum_i \alpha_i \lambda_i^k u_i \).
- Let \( \lambda_i \) be the largest eigenvalue, then as \( k \to \infty \), the second term of
  \( x_k = \alpha_i \lambda_i^k u_i + \sum_{i=2}^\infty \alpha_i \lambda_i^k u_i = \lambda_i^k \left( \alpha_i u_i + \sum_{i=2}^\infty \alpha_i \left( \frac{\lambda_i}{\lambda_j} \right)^k u_i \right) \)
  vanishes since \( |\lambda_i / \lambda_j| < 1 \).
- Thus, as \( k \to \infty \), \( x_k \to \lambda_i^k \alpha_i u_i \) and \( \left( x_k \right)_j / \left( x_{k-1} \right)_j \to \lambda_i \) for any component \( j \) of \( x \).
- If \( x_0 = \sum_i \alpha_i u_i \) has \( \alpha_i = 0 \) for the largest eigenvalue, the method may fail (but roundoff).
- For a real valued matrix and real valued \( x_0 \), one can never get complex numbers.
- The largest eigenvalue may be repeated, in which case the final vector may be a linear combination of the multiple eigenvectors.
- After every iteration, \( x_k \) can be (should be) renormalized to stop \( x_k \) from growing too large.
- Inverse iteration can be used to find the smallest eigenvalue, since the largest eigenvalue of \( A^{-1} \) is the smallest eigenvalue of A.
- **Deflation** removes an eigenvalue from A by subtracting off the corresponding low rank update. Then the resulting matrix can be used to compute the next largest eigenvalue, etc…