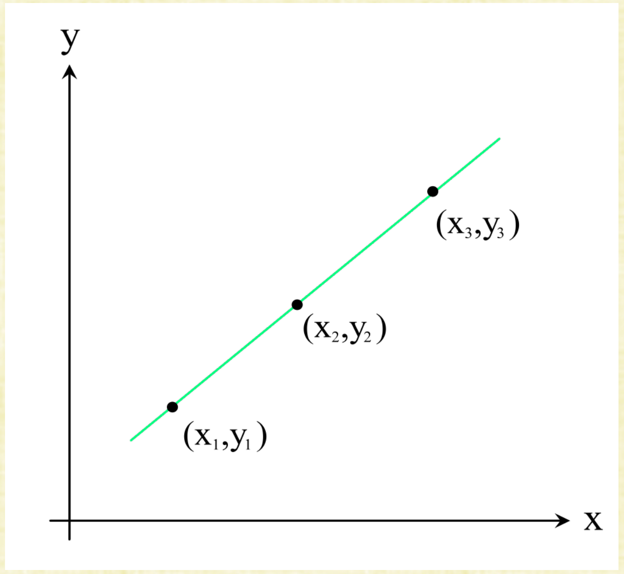
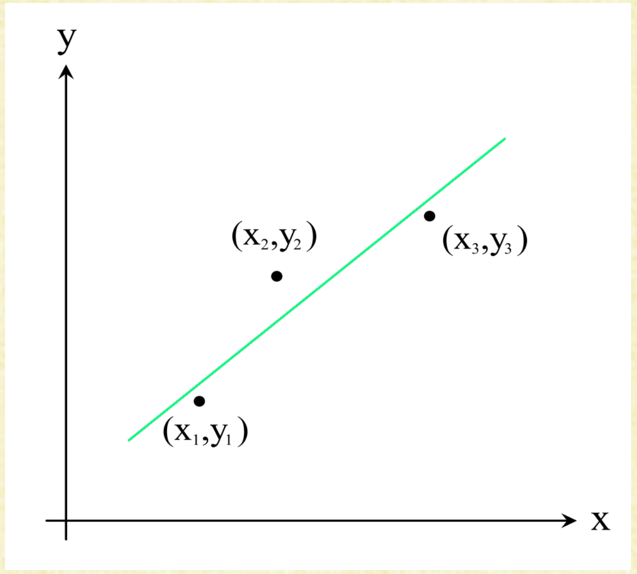
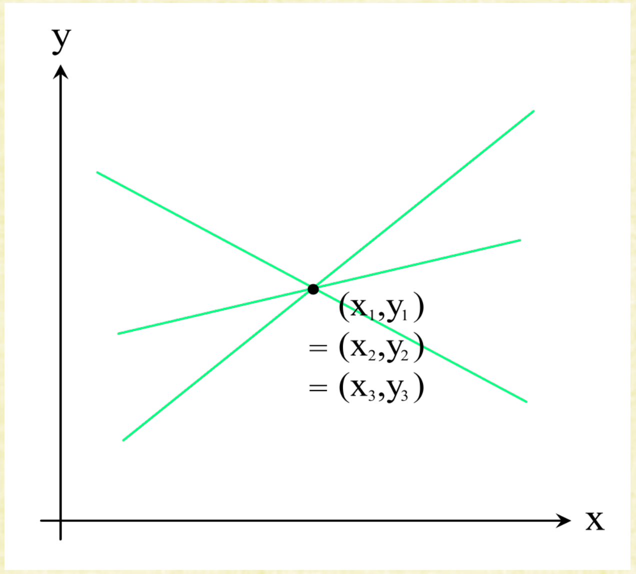
Zero Singular Values

Underdetermined Systems

- Consider drawing a line $y = c_1 + c_2 x$ through 3 data points
- When the points are colinear, there is a unique solution
- When the points are not colinear, there is a least squares solution
- When the points are co-located (i.e. identical), there are infinite solutions







Underdetermined Systems

- The Vandermonde matrix equation is $\begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$
- Let $x_1 = x_2 = x_3$, so that the columns are multiples of each other (and the matrix is rank 1)
- If $y_1 = y_2 = y_3$, the right hand side is in the range of the rank 1 columns implying infinite solutions
- Otherwise, the right hand side is not in the range of the columns implying no solutions (toss away the second column and c_2, then do least squares on c_1)

(Careful) Variable Classification

- Consider $\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 0 \end{pmatrix}$
- The first two rows, $c_1 = 1$ and $c_1 = 2$, overdetermine c_1
- The third row, $c_2 = 3$, uniquely determines c_2
- The last row, $0c_3 = 0$, leaves c_3 underdetermined with infinite possibilities

- It's often misleading to classify an entire system (as either having a unique solution, no solution, or infinite solutions)
- Rather, one should do the best they can with what has been given
  - E.g. Shouldn't skip dinner because of uncertainties about what time the sun will go down

Understanding Underdetermined Systems

- Transform $Ac = b$ into $\Sigma\hat{c} = \hat{b}$ (as usual)
- For each $\sigma_k \neq 0$, compute $\hat{c}_k = \frac{\hat{b}_k}{\sigma_k}$ (as usual)
- When $\sigma_k = 0$, $\hat{c}_k$ is undefined (moreover, division by a small σ_k is dubious)
- Tall matrices have extra rows with $0 = \hat{b}_k$ ($\sigma_k = 0$ rows contribute to this too), and nonzero $\hat{b}_k$ imply a nonzero residual
- Wide matrices have extra columns of zeros, leaving some $\hat{c}_k$ undetermined (just like $\sigma_k = 0$ columns)

Understanding Underdetermined Systems

- Can write $U(\hat{\Sigma} \quad 0)V^T$ for wide matrices, similar to $A = U\begin{pmatrix}\hat{\Sigma}\\0\end{pmatrix}V^T$ for tall matrices
  - In general, $\hat{\Sigma}$ may contain zeros on the diagonal (for tall matrices too, if not full rank)
- For any matrix, can write $A = U\begin{pmatrix}\hat{\Sigma} & 0\\0 & 0\end{pmatrix}V^T$ with $\hat{\Sigma}$ diagonal and full rank
- Then, $\Sigma\hat{c} = \hat{b}$ has the form $\begin{pmatrix}\hat{\Sigma} & 0\\0 & 0\end{pmatrix}\begin{pmatrix}\hat{c}_r\\\hat{c}_z\end{pmatrix} = \begin{pmatrix}\hat{b}_r\\\hat{b}_z\end{pmatrix}$
- $\|r\|_2 = \|U^T(b - Ac)\|_2 = \left\|\begin{pmatrix}\hat{b}_r\\\hat{b}_z\end{pmatrix} - \begin{pmatrix}\hat{\Sigma} & 0\\0 & 0\end{pmatrix}\begin{pmatrix}\hat{c}_r\\\hat{c}_z\end{pmatrix}\right\|_2 = \left\|\begin{pmatrix}\hat{b}_r\\\hat{b}_z\end{pmatrix} - \begin{pmatrix}\hat{\Sigma}\hat{c}_r\\0\end{pmatrix}\right\|_2$
- Thus, solving $\hat{\Sigma}\hat{c}_r = \hat{b}_r$ for $\hat{c}_r$ minimizes the residual to $\|r\|_2 = \|\hat{b}_z\|_2$
- Meanwhile, any values are acceptable for the non-determined $\hat{c}_z$

Minimum Norm Solution

- Setting $\hat{c}_z = 0$ stresses that these parameters have no bearing on the solution
- This is more sensical than setting $\hat{c}_z$ to some nonzero value as if those values mattered
- Example:
  - Consider a variable related to how a hat is worn while driving, which could matter when the hat blocks the sun or keeps longer hair away from the eyes
  - Someone with short hair driving at night would likely have no driving dependence on a hat; in this case, reporting information about hats is misleading
- So, $c = V\hat{c} = V\begin{pmatrix}\hat{c}_r \\ \hat{c}_z\end{pmatrix} = V\begin{pmatrix}\hat{\Sigma}^{-1}\hat{b}_r \\ 0\end{pmatrix} = \sum_{\sigma_k \neq 0} v_k \frac{\hat{b}_k}{\sigma_k} = \sum_{\sigma_k \neq 0} v_k \frac{u_k^T b}{\sigma_k}$

Pseudo-Inverse

- The minimum norm solution is $c = \left(\sum_{\sigma_k \neq 0} \frac{v_k u_k^T}{\sigma_k}\right) b = A^+ b$ where the <u>pseudo-inverse</u> is $A^+ = \sum_{\sigma_k \neq 0} \frac{v_k u_k^T}{\sigma_k}$

- When A is square and full rank $A^+ = A^{-1}$

- Each term is an outer product between corresponding columns of U and V, weighted by one over their corresponding singular value
- Each term is a size nxm matrix, so this a sum of matrices

Sum of Rank One Matrices

- $Ac = U\begin{pmatrix}\hat{\Sigma} & 0\\ 0 & 0\end{pmatrix}V^T c = U\begin{pmatrix}\hat{\Sigma} & 0\\ 0 & 0\end{pmatrix}\begin{pmatrix}\hat{c}_r\\ \hat{c}_z\end{pmatrix} = U\begin{pmatrix}\hat{\Sigma}\hat{c}_r\\ 0\end{pmatrix} = \sum_{\sigma_k\neq 0} u_k\sigma_k\hat{c}_k = \sum_{\sigma_k\neq 0} u_k\sigma_k v_k^T c = \left(\sum_{\sigma_k\neq 0}\sigma_k u_k v_k^T\right)c$

- Thus, $A = \sum_{\sigma_k\neq 0}\sigma_k u_k v_k^T$

- Each term is an outer product between corresponding columns of U and V, weighted by their corresponding singular value
- Each term is a size mxn matrix (the same size as A)
- Each term is rank 1, since every column in the term is a multiple of u_k

Recall: Understanding Ac (unit 3)

$$Ac = \begin{pmatrix} .141 & .825 & -.420 & -.351 \\ .344 & .426 & .298 & .782 \\ .547 & .028 & .644 & -.509 \\ .750 & -.371 & -.542 & .079 \end{pmatrix} \begin{pmatrix} 25.5 & 0 & 0 \\ 0 & 1.29 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} .504 & .574 & .644 \\ -.761 & -.057 & .646 \\ .408 & -.816 & .408 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix}$$

$$= \begin{pmatrix} .141 & .825 & -.420 & -.351 \\ .344 & .426 & .298 & .782 \\ .547 & .028 & .644 & -.509 \\ .750 & -.371 & -.542 & .079 \end{pmatrix} \begin{pmatrix} 25.5 & 0 & 0 \\ 0 & 1.29 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} v_1^T c \\ v_2^T c \\ v_3^T c \end{pmatrix}$$

$$= \begin{pmatrix} .141 & .825 & -.420 & -.351 \\ .344 & .426 & .298 & .782 \\ .547 & .028 & .644 & -.509 \\ .750 & -.371 & -.542 & .079 \end{pmatrix} \begin{pmatrix} \sigma_1 v_1^T c \\ \sigma_2 v_2^T c \\ \sigma_3 v_3^T c \\ 0 \end{pmatrix}$$

$$= u_1 \sigma_1 v_1^T c + u_2 \sigma_2 v_2^T c + u_3 \sigma_3 v_3^T c + u_4 0$$

- Ac projects c onto the basis vectors in V, scales by the associated singular values, and uses those results as weights on the basis vectors in U

Matrix Approximation

- Use the p largest singular values: $A \approx \sum_{k=1}^{p} \sigma_k u_k v_k^T$
- The pseudo-inverse is approximated similarly: $A^+ \approx \sum_{k=1}^{p} \frac{1}{\sigma_k} v_k u_k^T$
- This is the best rank p approximation to A, and the main idea behind principle component analysis (PCA)
  - Often, thousands/millions of terms can be thrown away keeping only 10 to 100 terms

- Can also drop small singular values: $A \approx \sum_{\sigma_k > \epsilon} \sigma_k u_k v_k^T$
- This makes the pseudo-inverse better conditioned: $A^+ \approx \sum_{\sigma_k > \epsilon} \frac{1}{\sigma_k} v_k u_k^T$
  - This relies on a good choice of $\epsilon > 0$

Recall: Approximating A (unit 3)

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{pmatrix} \approx$$

$$\begin{pmatrix} .141 & .8\text{[redacted]}5 & -.4\text{[redacted]}0 & -.\text{[redacted]}51 \\ .344 & .4\text{[redacted]}6 & .2\text{[redacted]}8 & .78\text{[redacted]} \\ .547 & .0\text{[redacted]}3 & .64\text{[redacted]} & -.5\text{[redacted]}9 \\ .750 & -.3\text{[redacted]}1 & -.\text{[redacted]}42 & .0\text{[redacted]}9 \end{pmatrix} \begin{pmatrix} 25.5 & \text{[redacted]} & \text{[redacted]} \\ \text{[redacted]} & \text{[redacted]} & \text{[redacted]} \\ \text{[redacted]} & \text{[redacted]} & \text{[redacted]} \\ \text{[redacted]} & \text{[redacted]} & \text{[redacted]} \end{pmatrix} \begin{pmatrix} .504 & .574 & .644 \\ -\text{[redacted]} & & \\ \text{[redacted]} & & \end{pmatrix}$$

- The first singular value is much bigger than the second, and so represents the vast majority of what A does (note, the vectors in U and V are unit length)
- Thus, one could approximate A quite well by only using the terms associated with the largest singular value
- This is not a valid factorization, but an approximation (and the idea behind PCA)

Rank One Updates

- For real time applications (real time decision making, etc.), iteratively add one term at a time (slowly improving the estimate)
- $c = A^+ b \approx \frac{u_1^T b}{\sigma_1} v_1 + \frac{u_2^T b}{\sigma_2} v_2 + \frac{u_3^T b}{\sigma_3} v_3 + \cdots$

- Note the efficient ordering of the operations:
  - $u_k^T b$ is m multiplies, and the result times v_k is n multiplies (for a total of $m + n$ multiplies)
  - Don't form the size nxm matrix!
  - Multiplying the size mxn matrix $v_k u_k^T$ times b is $m \cdot n$ multiplies

Computing the SVD

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

- If $\sigma_k \neq 0$, then σ_k^2 is an eigenvalue of both $A^T A$ and $A A^T$ (with eigenvectors v_k and u_k respectively)

- Work with the smaller of $A^T A$ and $A A^T$ (which are both SP(S)D) to find the eigenvalues σ_k^2
- Then, σ_k^2 can be used in both $A^T A$ and $A A^T$ to find the corresponding eigenvectors

Finding Eigenvectors from Eigenvalues

- Given an eigenvalue λ, form the matrix $\hat{A} - \lambda I$

- If $\hat{A}$ is symmetric, then $\hat{A} - \lambda I$ is symmetric
- $\hat{A} - \lambda I$ has (at least) a rank 1 null space (from the definition of eigenvalues)

- Solve the linear system $\left(\hat{A} - \lambda I\right)v = 0$ to find the eigenvector v

Condition Number of Eigenproblems

- The condition number for finding an eigenvalue is different than the condition number for solving a linear system

- The condition number for finding an eigenvalue/eigenvector pair is $\frac{1}{v_L^T v_R}$ where v_L and v_R are the normalized left and right eigenvectors

- Symmetric (Hermitian) matrices have identical left and right eigenvectors; so, $v_L^T v_R = 1$ and the condition number is 1

~~Characteristic Polynomial~~

- ~~The eigenvalue problem is typically written as $\hat{A}v = \lambda v$~~
- ~~Alternatively, $(\hat{A} - \lambda I)v = 0$ implying that $\hat{A} - \lambda I$ is singular~~
- ~~Setting $\det(\hat{A} - \lambda I) = 0$ leads to a degree n characteristic polynomial equation in λ (for a size nxn matrix $\hat{A}$)~~

- ~~Finding the roots of this polynomial equation can be quite difficult~~
  - ~~Recall how difficult it was to find roots for a mere cubic equation~~
- ~~Finding roots for degree $n > 3$ polynomals is undesirable!~~

Similarity Transforms

- Similarity transforms, which look like $T^{-1}\hat{A}T$, preserve the eigenstructure
  - $T^{-1}\hat{A}Tv = \lambda v$ or $\hat{A}(Tv) = \lambda(Tv)$ still has eigenvalue λ with a modified eigenvector Tv
- When $\hat{A}$ is <u>real and symmetric</u> (complex and Hermitian), there exists an orthogonal (unitary) T that makes $T^{-1}\hat{A}T$ diagonal with real eigenvalues
  - e.g. $T = V$ for $A^T A = V\Sigma^T\Sigma V^T$ and $T = U$ for $AA^T = U\Sigma\Sigma^T U^T$

- Other interesting facts:
  - When $\hat{A}$ has distinct eigenvalues, a T exists to make $T^{-1}\hat{A}T$ diagonal
  - <u>Schur form</u>: For any (square) matrix, a unitary T exists to make $T^{-1}\hat{A}T$ upper triangular with eigenvalues on the diagonal
  - <u>Jordan form</u>: Any (square) matrix can be put into a form with eigenvalues on the diagonal and nonzero off-diagonal elements only occurring on the band above the diagonal and only for defective eigenvalues (which are repeated eigenvalues that don’t possess a full set of eigenvectors)

Similarity Transforms via QR Iteration

- Starting with $\hat{A}^0 = \hat{A}$
- Compute the factorization $\hat{A}^q = Q^q R^q$ with orthogonal Q^q
- Then define $\hat{A}^{q+1} = R^q Q^q$

- Note: $R^q Q^q = (Q^q)^T Q^q R^q Q^q = (Q^q)^T \hat{A}^q Q^q$ is a similarity transform of $\hat{A}^q$

- When the eigenvalues are distinct, $\hat{A}^q$ converges to a triangular matrix
- When $\hat{A}$ is symmetric, $\hat{A}^q$ converges to a diagonal matrix

Power Method

- Computes the largest eigenvalue (great for rank 1 updates)
- Start with a $c^0 \neq 0$, and iterate $c^{q+1} = \hat{A}c^q$
- Suppose c^0 is a linear combination of eigenvectors: $c^0 = \sum_k \alpha_k v_k$
- Then $c^q = \hat{A}^q c^0 = \sum_k \alpha_k \hat{A}^q v_k = \sum_k \alpha_k \lambda_k^q v_k = \lambda_{max}^q \sum_k \alpha_k \left(\frac{\lambda_k}{\lambda_{max}}\right)^q v_k$
- As $q \to \infty$, $\left(\frac{\lambda_k}{\lambda_{max}}\right)^q \to 0$ for $\lambda_k < \lambda_{max}$; so, $c^q \to \lambda_{max}^q \alpha_{max} v_{max}$
- As $q \to \infty$, $\frac{(c^{q+1})_i}{(c^q)_i} \to \frac{\lambda_{max}^{q+1}\alpha_{max}(v_{max})_i}{\lambda_{max}^q \alpha_{max}(v_{max})_i} = \lambda_{max}$ for every component i of c
- *Deflation* removes an eigenvalue from $\hat{A}$ by subtracting off its rank 1 update
  - The deflated $A^TA - \sigma_k^2 v_k v_k^T$ or $AA^T - \sigma_k^2 u_k u_k^T$ can then be used to compute the next largest eigenvalue (repeatedly)

Power Method

- If $c^0 = \sum_k \alpha_k v_k$ happens to have $\alpha_{max} = 0$, the method might fail (but roundoff errors can help)
- c^q needs to be periodically renormalized to stop it from growing too large
- When c^0 and $\hat{A}$ are real valued, cannot obtain complex numbers
- When the largest eigenvalue is repeated, one needs to determine a basis for the multiple associated eigenvectors

- Inverse Iteration can be used to find the smallest eigenvalue of $\hat{A}$, since the largest eigenvalue of $\hat{A}^{-1}$ is the smallest eigenvalue of $\hat{A}$
  - $c^{q+1} = \hat{A}^{-1}c^q$ is updated by solving $\hat{A}c^{q+1} = c^q$ to find c^{q+1}
  - Useful for finding the condition number $\frac{\sigma_{max}}{\sigma_{min}}$