Singular Value Decomposition

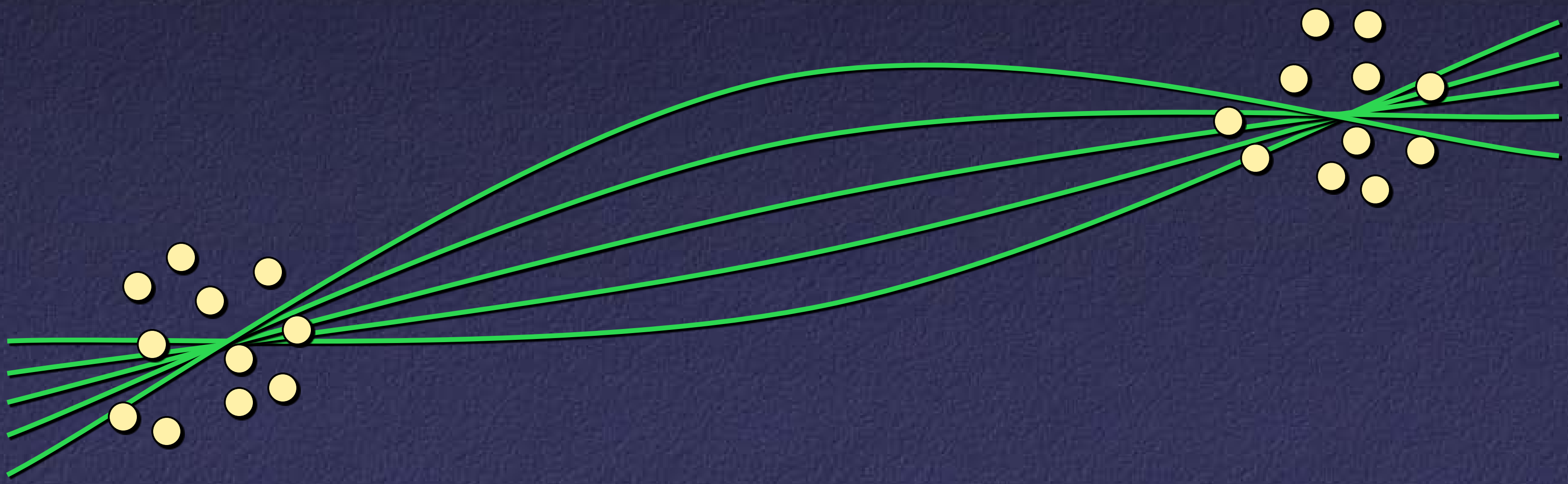
COS 323

Underconstrained Least Squares

- What if you have fewer data points than parameters in your function?
  - Intuitively, can't do standard least squares
  - Recall that solution takes the form $A^TAx = A^Tb$
  - When A has more columns than rows, A^TA is singular: can't take its inverse, etc.

Underconstrained Least Squares

- More subtle version: more data points than unknowns, but data poorly constrains function
- Example: fitting to $y=ax^2+bx+c$



Underconstrained Least Squares

- Problem: if problem very close to singular, roundoff error can have a huge effect
  - Even on "well-determined" values!
- Can detect this:
  - Uncertainty proportional to covariance $C = (A^T A)^{-1}$
  - In other words, unstable if $A^T A$ has small values
  - More precisely, care if $x^T(A^T A)x$ is small for any x
- Idea: if part of solution unstable, set answer to 0
  - Avoid corrupting good parts of answer

Singular Value Decomposition (SVD)

- Handy mathematical technique that has application to many problems
- Given any $m \times n$ matrix $\mathbf{A}$, algorithm to find matrices $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$ such that

$$\mathbf{A} = \mathbf{U}\mathbf{W}\mathbf{V}^T$$

$\mathbf{U}$ is $m \times n$ and orthonormal

$\mathbf{W}$ is $n \times n$ and diagonal

$\mathbf{V}$ is $n \times n$ and orthonormal

SVD

$$\begin{pmatrix} \\ \\ \mathbf{A} \\ \\ \\ \end{pmatrix} = \begin{pmatrix} \\ \\ \mathbf{U} \\ \\ \\ \end{pmatrix} \begin{pmatrix} w_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & w_n \end{pmatrix} \begin{pmatrix} \\ \mathbf{V} \\ \\ \end{pmatrix}^{\mathrm{T}}$$

- Treat as black box: code widely available

  In Matlab: `[U,W,V]=svd(A,0)`

SVD

- The w_i are called the singular values of **A**
- If **A** is singular, some of the w_i will be 0
- In general $rank(\mathbf{A})$ = number of nonzero w_i
- SVD is mostly unique (up to permutation of singular values, or if some w_i are equal)

SVD and Inverses

- Why is SVD so useful?
- Application #1: inverses
- $\mathbf{A}^{-1} = (\mathbf{V}^T)^{-1}\, \mathbf{W}^{-1}\, \mathbf{U}^{-1} = \mathbf{V}\, \mathbf{W}^{-1}\, \mathbf{U}^T$
  - Using fact that inverse = transpose for orthogonal matrices
  - Since **W** is diagonal, $\mathbf{W}^{-1}$ also diagonal with reciprocals of entries of **W**

SVD and Inverses

- $\mathbf{A}^{-1} = (\mathbf{V}^T)^{-1}\,\mathbf{W}^{-1}\,\mathbf{U}^{-1} = \mathbf{V}\,\mathbf{W}^{-1}\,\mathbf{U}^T$
- This fails when some w_i are 0
  - It's *supposed* to fail – singular matrix
- Pseudoinverse: if $w_i = 0$, set $1/w_i$ to 0 (!)
  - "Closest" matrix to inverse
  - Defined for all (even non-square, singular, etc.) matrices
  - Equal to $(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$ if $\mathbf{A}^T\mathbf{A}$ invertible

SVD and Least Squares

- Solving **Ax**=**b** by least squares
- **x**=pseudoinverse(**A**) times **b**
- Compute pseudoinverse using SVD
  - Lets you see if data is singular
  - Even if not singular, ratio of max to min singular values (= condition number) tells you how stable the solution will be
  - Set $1/w_i$ to 0 if w_i is small (even if not exactly 0)

SVD and Eigenvectors

- Let $\mathbf{A}=\mathbf{UWV}^T$, and let x_i be i^{th} column of $\mathbf{V}$
- Consider $\mathbf{A}^T\mathbf{A}\,x_i$:

$$\mathbf{A}^T\mathbf{A}x_i = \mathbf{VW}^T\mathbf{U}^T\mathbf{UWV}^T x_i = \mathbf{VW}^2\mathbf{V}^T x_i = \mathbf{VW}^2\begin{pmatrix}0\\ \vdots\\ 1\\ \vdots\\ 0\end{pmatrix} = \mathbf{V}\begin{pmatrix}0\\ \vdots\\ w_i^{\,2}\\ \vdots\\ 0\end{pmatrix} = w_i^{\,2}x_i$$

- So elements of **W** are sqrt(eigenvalues) and columns of **V** are eigenvectors of $\mathbf{A}^T\mathbf{A}$
  - What we wanted for robust least squares fitting!

SVD and Matrix Similarity

- One common definition for the norm of a matrix is the Frobenius norm:

$$\|\mathbf{A}\|_{\mathrm{F}} = \sum_i \sum_j a_{ij}^{2}$$

- Frobenius norm can be computed from SVD

$$\|\mathbf{A}\|_{\mathrm{F}} = \sum_i w_i^{2}$$

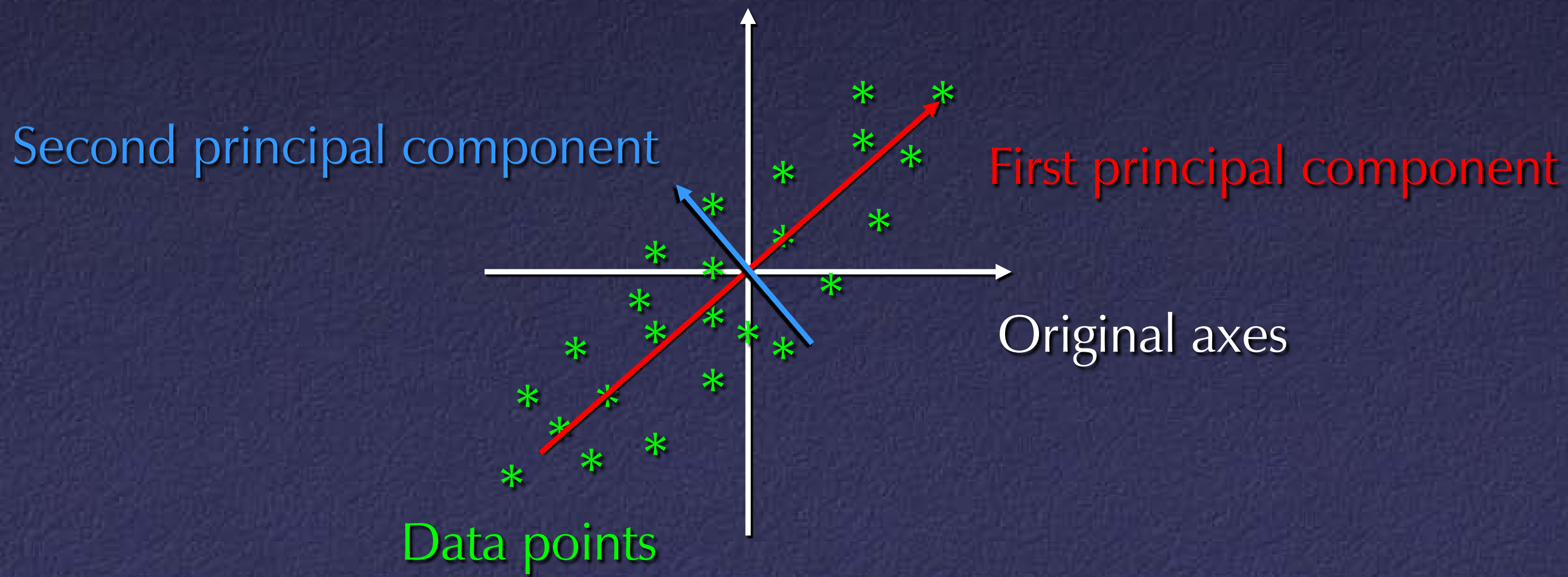
- So changes to a matrix can be evaluated by looking at changes to singular values

SVD and Matrix Similarity

- Suppose you want to find best rank-k approximation to **A**
- Answer: set all but the largest k singular values to zero
- Can form compact representation by eliminating columns of **U** and **V** corresponding to zeroed w_i

SVD and PCA

- Principal Components Analysis (PCA): approximating a high-dimensional data set with a lower-dimensional subspace



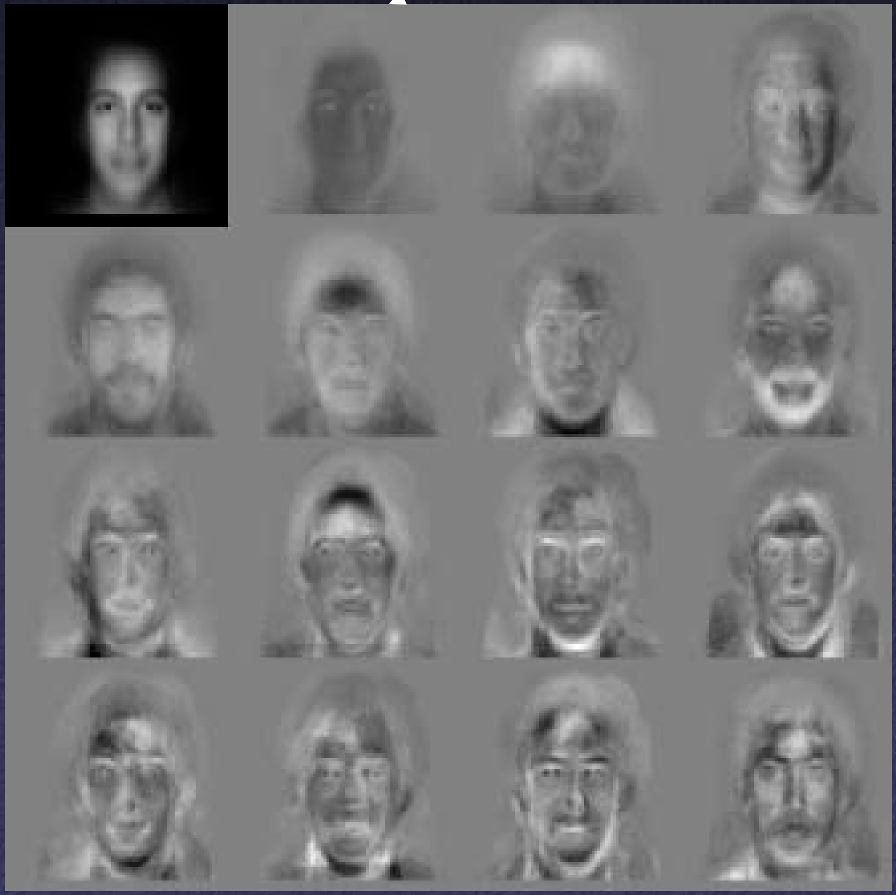
SVD and PCA

- Data matrix with points as rows, take SVD
  - Subtract out mean ("whitening")
- Columns of $\mathbf{V}_k$ are principal components
- Value of w_i gives importance of each component

PCA on Faces: "Eigenfaces"

Average face

First principal component



Other components

For all except average,
"gray" = 0,
"white" > 0,
"black" < 0

Using PCA for Recognition

- Store each person as coefficients of projection onto first few principal components

$$\text{image} = \sum_{i=0}^{i_{\max}} a_i \text{Eigenface}_{\text{i}}$$

- Compute projections of target image, compare to database ("nearest neighbor classifier")

Total Least Squares

- One final least squares application
- Fitting a line: vertical vs. perpendicular error

Total Least Squares

- Distance from point to line:

$$d_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} \cdot \vec{n} - a$$

where n is normal vector to line, a is a constant

- Minimize:

$$\chi^2 = \sum_i d_i^2 = \sum_i \left[\begin{pmatrix} x_i \\ y_i \end{pmatrix} \cdot \vec{n} - a \right]^2$$

Total Least Squares

- First, let's pretend we know n, solve for a

$$\chi^2 = \sum_i \left[\begin{pmatrix} x_i \\ y_i \end{pmatrix} \cdot \vec{n} - a \right]^2$$

$$a = \frac{1}{m} \sum_i \begin{pmatrix} x_i \\ y_i \end{pmatrix} \cdot \vec{n}$$

- Then

$$d_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} \cdot \vec{n} - a = \begin{pmatrix} x_i - \frac{\Sigma x_i}{m} \\ y_i - \frac{\Sigma y_i}{m} \end{pmatrix} \cdot \vec{n}$$

Total Least Squares

- So, let's define

$$\begin{pmatrix} \tilde{x}_i \\ \tilde{y}_i \end{pmatrix} = \begin{pmatrix} x_i - \frac{\Sigma x_i}{m} \\ y_i - \frac{\Sigma y_i}{m} \end{pmatrix}$$

and minimize

$$\sum_i \left[\begin{pmatrix} \tilde{x}_i \\ \tilde{y}_i \end{pmatrix} \cdot \vec{n} \right]^2$$

Total Least Squares

- Write as linear system

$$\begin{pmatrix} \tilde{x}_1 & \tilde{y}_1 \\ \tilde{x}_2 & \tilde{y}_2 \\ \tilde{x}_3 & \tilde{y}_3 \\ & \vdots & \end{pmatrix} \begin{pmatrix} n_x \\ n_y \end{pmatrix} = \vec{0}$$

- Have An=0
  - Problem: lots of n are solutions, including n=0
  - Standard least squares will, in fact, return n=0

Constrained Optimization

- Solution: constrain n to be unit length
- So, try to minimize $|An|^2$ subject to $|n|^2=1$

$$\|\mathbf{A}\vec{n}\|^2 = (\mathbf{A}\vec{n})^{\mathrm{T}}(\mathbf{A}\vec{n}) = \vec{n}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\,\vec{n}$$

- Expand in eigenvectors e_i of A^TA:

$$\vec{n} = \mu_1\mathbf{e}_1 + \mu_2\mathbf{e}_2$$

$$\vec{n}^{\mathrm{T}}\left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)\vec{n} = \lambda_1\mu_1^2 + \lambda_2\mu_2^2$$

$$\|\vec{n}\|^2 = \mu_1^2 + \mu_2^2$$

where the λ_i are eigenvalues of A^TA

Constrained Optimization

- To minimize $\lambda_1\mu_1^2 + \lambda_2\mu_2^2$ subject to $\mu_1^2 + \mu_2^2 = 1$ set $\mu_{\text{min}} = 1$, all other $\mu_{\text{i}} = 0$
- That is, n is eigenvector of A^TA with the smallest corresponding eigenvalue