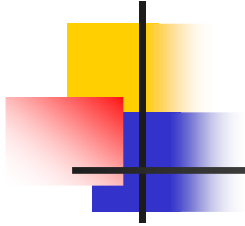


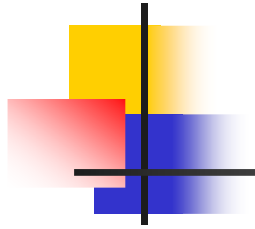


Algorithms 2005

Ramesh Hariharan



Algebraic Methods



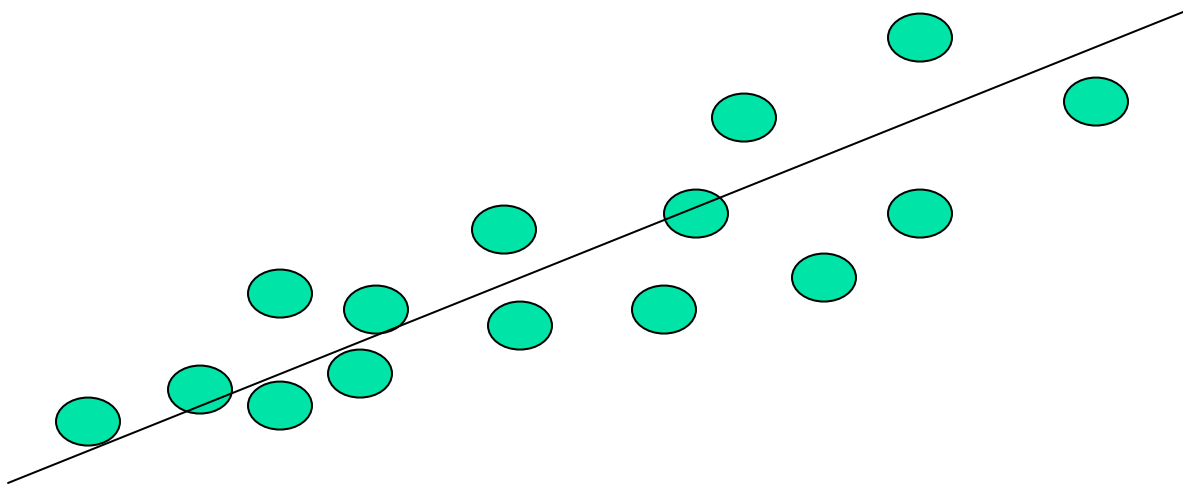
Principal Component Analysis

- Given m points in a n dimensional space, for large n , how does one project on to a 2 or 3 dimensional space while preserving broad trends in the data and allowing it to be visualized?



Principal Component Analysis

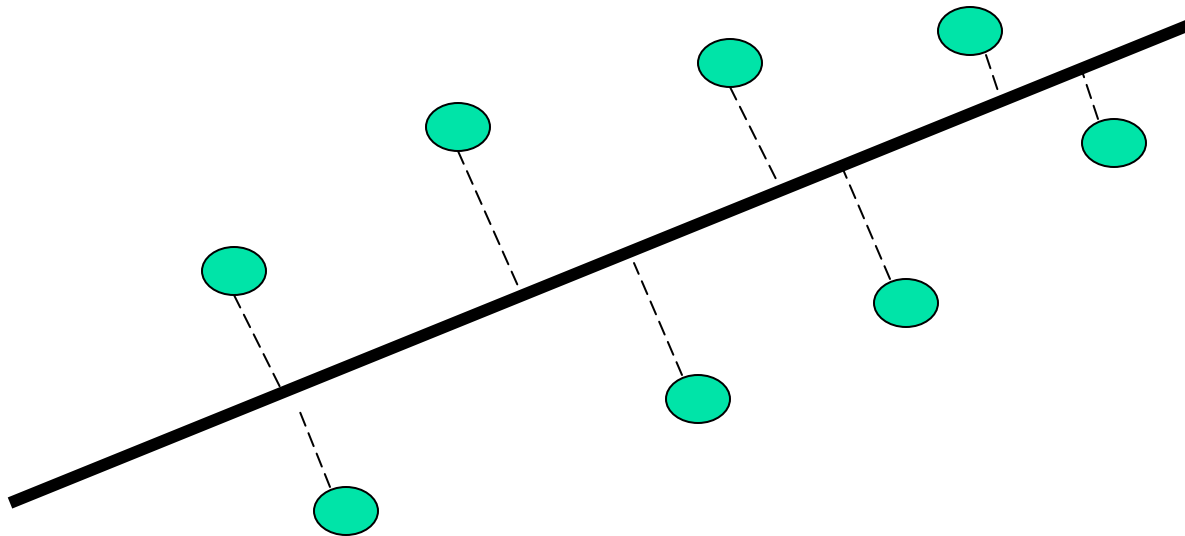
- Given m points in a n dimensional space, for large n , how does one project on to a 1 dimensional space?



- Choose a line that fits the data so the points are spread out well along the line

Principal Component Analysis

- Formally, minimize sum of squares of distances to the line.

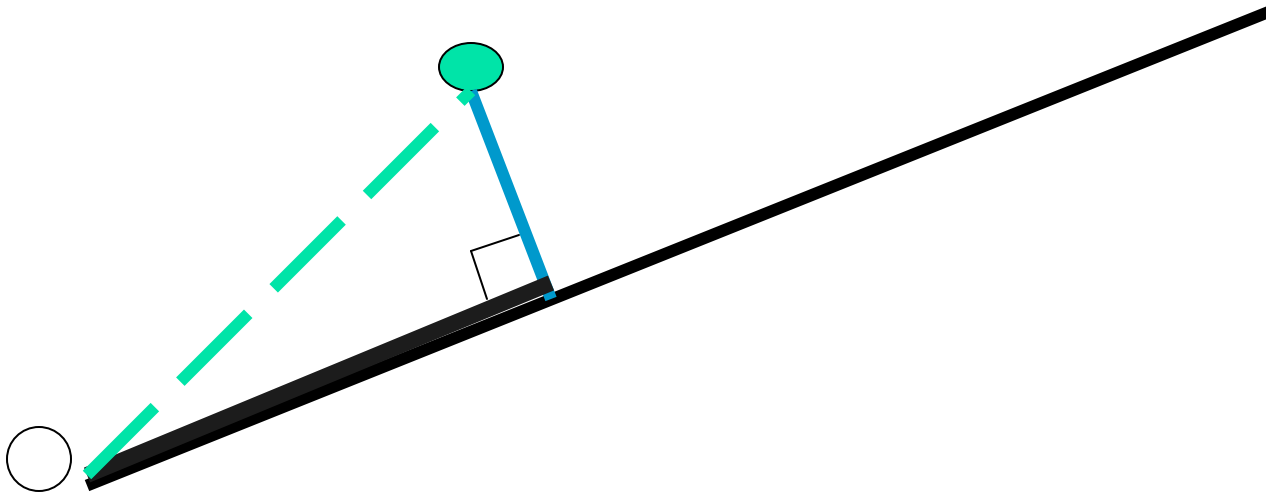


- Why sum of squares? Because it allows fast minimization, assuming the line passes thru 0



Principal Component Analysis

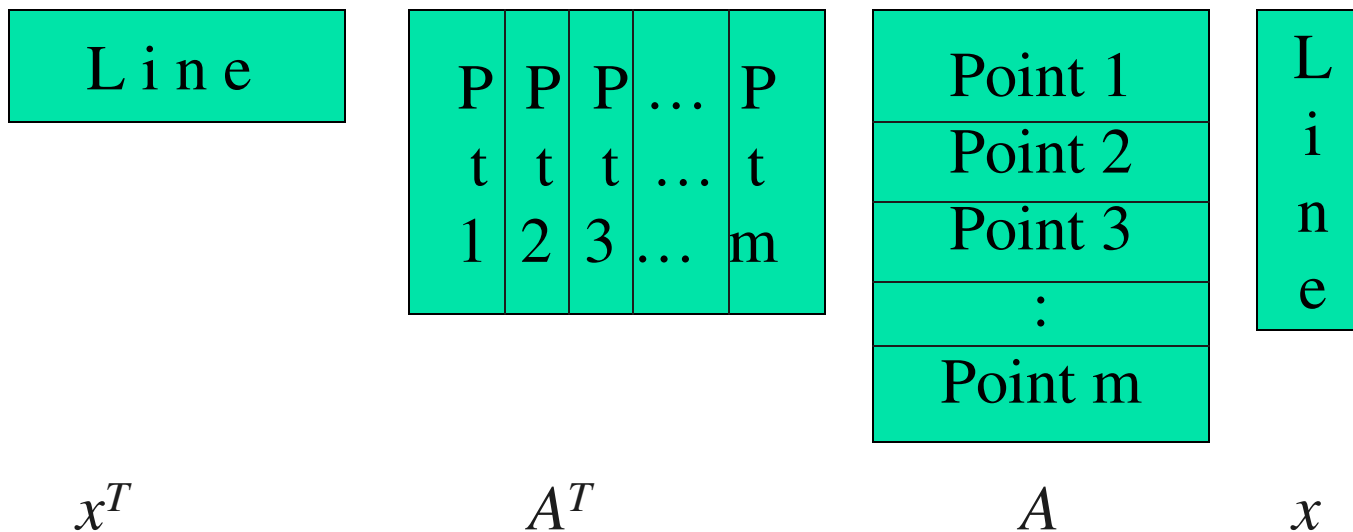
- Minimizing sum of squares of distances to the line is the same as maximizing the sum of squares of the projections on that line, thanks to Pythagoras.





Principal Component Analysis

- How is the sum of squares of projection lengths expressed in algebraic terms?





Principal Component Analysis

- How is the sum of squares of projection lengths expressed in algebraic terms?

$$\max(x^T A^T A x), \text{ subject to } x^T x = 1$$



Principal Component Analysis

- Rewriting this:

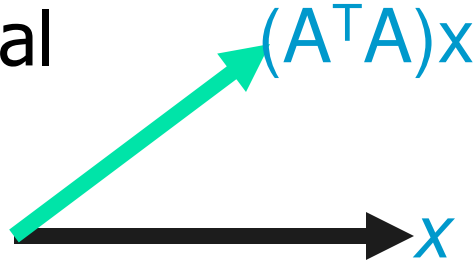
$$\begin{aligned}x^T A^T A x &= e = e x^T x = x^T (e x) \\ \Leftrightarrow x^T (A^T A x - e x) &= 0\end{aligned}$$

- Show that the maximum value of $x^T A^T A x$ is obtained for x satisfying $A^T A x = e x$
- So, find the largest e and associated x such that the matrix $A^T A$ when applied to x yields a new vector which is in the same direction as x , only scaled by a factor e .



Principal Component Analysis

- $(A^T A)x$ points in some other direction in general



x is an eigenvector and e an eigenvalue if

$$ex = (A^T A)x$$


A diagram illustrating the eigenvalue case. A black arrow labeled x points horizontally to the right. A cyan arrow labeled $ex = (A^T A)x$ also points horizontally to the right, overlapping the black arrow.



Principal Component Analysis

- How many eigenvectors are there?
- For Real Symmetric Matrices
 - except in degenerate cases when eigenvalues repeat, there are n eigenvectors
 - $x_1 \dots x_n$ are the eigenvectors*
 - $e_1 \dots e_n$ are the eigenvalues*
 - all eigenvectors are mutually orthogonal and therefore form a new basis
 - Eigenvectors for distinct eigenvalues are mutually orthogonal
 - Eigenvectors corresponding to the same eigenvalue have the property that any linear combination is also an eigenvector with the same eigenvalue; one can then find as many orthogonal eigenvectors as the number of repeats of the eigenvalue.
- Show the above properties



Principal Component Analysis

- For matrices of the form $A^T A$
 - All eigenvalues are non-negative (show this)



Principal Component Analysis

- How are the eigenvectors computed?

Since the eigenvectors form a basis:

$$\begin{aligned}(A^T A) y &= (A^T A) (a_1 x_1 + a_2 x_2 + \dots + a_n x_n) \\ &= a_1 e_1 x_1 + \dots + a_n e_n x_n\end{aligned}$$

$$(A^T A)(A^T A) y = a_1 e_1 e_1 x_1 + \dots + a_n e_n e_n x_n$$

- Repeated application of $A^T A$ on almost any vector y converges to a scaled version of the eigenvector x_1 corresponding to the largest eigenvalue e_1 .
- y should satisfy $a_1 \neq 0$

What happens if e_1 is a repeated eigen value?



Principal Component Analysis

- Algorithm for computing the fitting line
 - Compute $A^T A$
 - Start with (almost) any vector y
 - Repeatedly multiply $A^T A$ with y and rescale until convergence (in practice, a fixed number of times)
 - The resulting direction is the fitting line direction!!



Principal Component Analysis

- Time taken for convergence
 - How many iterations i before
 - $[a_1(e_1)^i] / [\sum a_k(e_k)^i] > 1-\epsilon$
 - $a_1(e_1)^i \epsilon > \sum a_{k>1}(e_{k>1})^i$
 - $a_1(e_1)^i \epsilon > n a_{\max}(e_2)^i$
 - $a_1/a_{\max}(e_1/e_2)^i > n/\epsilon$
 - $i > \log(n/\epsilon a_{\max}/a_1) / \log(e_1/e_2)$



Principal Component Analysis

Optimization

- if A is $m \times n$ and $m \ll n$ then $A^T A$ is $n \times n \gg m \times m$
- use AA^T instead, eigenvector of AA^T is easily converted to that of $A^T A$

$$(AA^T)y = e'y$$

$$\Rightarrow A^T(AA^T)y = e'(A^T y)$$

$$\Rightarrow (A^T A)(A^T y) = e'(A^T y)$$

$$\Rightarrow A^T y \text{ is the eigenvector of } A^T A$$



Principal Component Analysis

- New algorithm for computing the fitting line
 - Compute $B = A^T A$ or AA^T whichever is smaller
 - Start with (almost) any vector y
 - Repeatedly multiply B with y and rescale until convergence
 - If $B = A^T A$ then the resulting direction is the fitting line direction
 - If $B = AA^T$ then A^T times the resulting direction is the fitting line direction



Principal Component Analysis

- Complexity

- Compute $B = A^T A$ or $A A^T$: (cols x rows x cols) or (rows x cols x rows), whichever is smaller
- Iterations: cols x cols x numIterations or rows x rows x numIterations, whichever is smaller
- If $B = A A^T$ then multiplication with A^T takes cols x rows



Principal Component Analysis

- The Conundrum in Practice

- For 9000×2000 , $B = A^T A$ takes $2000 \times 9000 \times 2000$
- For 2000×9000 , $B = A A^T$ takes $2000 \times 9000 \times 2000$
- Iterations: $2000 \times 2000 \times \text{numIterations}$ in both cases
- For 2000×9000 , $B = A A^T$ and multiplication with A^T takes 2000×9000

Yet, 9000×2000 takes 100 times what 2000×9000 takes!!!!



Principal Component Analysis

- A careful look at the code
 - *for i=0 to min(rows,cols)*
 - *for j=0 to min(rows,cols)*
 - *if rows<cols*
 - *{for k=0 to cols*
 - *B[i,j]=A[i,k]*A[j,k]*
 - *}*
 - *else*
 - *{for k=0 to rows*
 - *B[i,j]=A[k,i]*A[k,j]*
 - *}*



Principal Component Analysis

- A careful look at the code
 - The AA^T option shows strong locality of reference
 - The $A^T A$ option shows strong non-locality of reference
 - Caching misses cause a greater than 100 fold slowdown in the latter cache



Principal Component Analysis

- The solution

- Prior to doing $A^T A$, move A into a linear array stored in column major order.
- Currently done using an extra array, can one save an array? More space usage is bad. **How does one convert row major to col major order in place for a rectangular matrix?**
- With this, both $A^T A$ and $A A^T$ take the same amount of time.



Principal Component Analysis

- Further dimensions

- Once the line along which projections are made is found, all points are projected to the space orthogonal to this line to find further dimensions, recursively.
- These dimensions are called principal components; usually only a few are computed.
- The strength associated with a dimension is the corresponding eigenvalue = the sum of squares of projection lengths
- The total sum of all eigenvalues over all eigenvectors is equal to the squares of the lengths of the point vectors themselves; the strength of an eigenvalue is expressed as a fraction of this quantity.



Principal Component Analysis

- Why normalize points?
 - Assumption: Fitting line passes through the origin.
 - This assumption is necessary for fast computation.
 - If data is skewed along any dimension then good fitting lines need not pass through the origin; mean centering helps reduce this skew.