15-826: Multimedia Databases and Data Mining

Lecture #20: SVD - part III (more case studies)

C. Faloutsos

Must-read Material

- MM Textbook Appendix D
- Graph Mining Textbook, chapter 15.

Must-read Material, cont’d


Outline

Goal: ‘Find similar / interesting things’

- Intro to DB
- Indexing - similarity search
- Data Mining
Indexing - Detailed outline

- primary key indexing
- secondary key / multi-key indexing
- spatial access methods
- fractals
- text
- Singular Value Decomposition (SVD)
- multimedia
- ...

SVD - Detailed outline

- Motivation
- Definition - properties
- Interpretation
- Complexity
- Case studies
- SVD properties
- More case studies
- Conclusions

SVD - detailed outline

- ...
- Case studies
- SVD properties
- more case studies
  - google/Kleinberg algorithms
  - query feedbacks
- Conclusions

SVD - Other properties - summary

- can produce orthogonal basis (obvious) (who cares?)
- can solve over- and under-determined linear problems (see C(1) property)
- can compute ‘fixed points’ (= ‘steady state prob. in Markov chains’) (see C(4) property)
Properties – sneak preview:

A(0): $A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} V^T_{[r \times m]}$

B(5): $(A^T A)^k v' \sim \text{(constant)} v_1$

C(1): $A_{[n \times m]} x_{[m \times 1]} = b_{[n \times 1]}$
then, $x_0 = V \Lambda^{(-1)} U^T b$: shortest, actual or least-squares solution

C(4): $A^T A v_1 = \lambda_1^2 v_1$
Properties – sneak preview:

A(0): \( A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} V^T_{[r \times m]} \)

B(5): \( (A^T A)^k \mathbf{v}' \sim \text{constant} \mathbf{v}_1 \)

C(1): \( A_{[n \times m]} \mathbf{x}_{[m \times 1]} = \mathbf{b}_{[n \times 1]} \)
\[ \text{then, } x_0 = V \Lambda^{(-1)} U^T \mathbf{b} : \text{shortest, actual or least-squares solution} \]

C(4): \( A^T A \mathbf{v}_1 = \lambda_1^2 \mathbf{v}_1 \)

IMPORTANT!

SVD - outline of properties

- (A): obvious
- (B): less obvious
- (C): least obvious (and most powerful)
Properties - by defn.:

A(0): \( A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} V^T_{[r \times m]} \)

A(1): \( U^T_{[r \times n]} U_{[n \times r]} = \mathbf{I}_{[r \times r]} \) (identity matrix)

A(2): \( V^T_{[r \times n]} V_{[n \times r]} = \mathbf{I}_{[r \times r]} \)

A(3): \( \Lambda^k = \text{diag}(\lambda_1^k, \lambda_2^k, \ldots, \lambda_r^k) \) (k: ANY real number)

A(4): \( \Lambda^T = V \Lambda U^T \)

---

Less obvious properties

A(0): \( A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} V^T_{[r \times m]} \)

B(1): \( A_{[n \times m]} (A^T)_{[m \times n]} = ?? \)

---

Less obvious properties

A(0): \( A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} V^T_{[r \times m]} \)

B(1): \( A_{[n \times m]} (A^T)_{[m \times n]} = U \Lambda^2 U^T \)

symmetric; Intuition?

B(2): symmetrically, for ‘\( V \)’

\( (A^T)_{[m \times n]} A_{[n \times m]} = V \Lambda^2 V^T \)

Intuition?
Reminder: ‘column orthonormal’

\[ \mathbf{V}^\mathsf{T} \mathbf{V} = \mathbf{I}_{[r \times r]} \]

- \( \mathbf{v}_1 \)
- \( \mathbf{v}_2 \)

\[ \mathbf{v}_1^\mathsf{T} \times \mathbf{v}_1 = 1 \]
\[ \mathbf{v}_1^\mathsf{T} \times \mathbf{v}_2 = 0 \]

Less obvious properties

A: term-to-term similarity matrix

B(3): \((\mathbf{A}^\mathsf{T} \mathbf{A})_{[m \times n]} \mathbf{A}_{[n \times m]}\)^k = \mathbf{V} \mathbf{\Lambda}^{2k} \mathbf{V}^\mathsf{T} \]

and

B(4): \((\mathbf{A}^\mathsf{T} \mathbf{A})^k \sim \mathbf{v}_1 \lambda_1^{2k} \mathbf{v}_1^\mathsf{T} \) for \( k \gg 1 \)

where

\( \mathbf{v}_1: [m \times 1] \) first column (singular-vector) of \( \mathbf{V} \)

\( \lambda_1: \) strongest singular value

Proof of (B4)?

Less obvious properties

B(4): \((\mathbf{A}^\mathsf{T} \mathbf{A})^k \sim \mathbf{v}_1 \lambda_1^{2k} \mathbf{v}_1^\mathsf{T} \) for \( k \gg 1 \)

B(5): \((\mathbf{A}^\mathsf{T} \mathbf{A})^k \mathbf{v}' \sim \text{(constant)} \mathbf{v}_1 \)

ie., for (almost) any \( \mathbf{v}' \), it converges to a vector parallel to \( \mathbf{v}_1 \)

Thus, useful to compute first singular vector/value (as well as the next ones, too...)
Proof of (B5)?

- B(5): \((A^T A)^k v' \sim \text{(constant)} v_1\)
- \(\cdots (A^T A) (A^T A)^k v' \sim \text{(constant)} v_1\)

Property (B5)

- Intuition:
  - \((A^T A)^k v'\)
  - \((A^T A)^k v'\)

- \(\cdots\)

Property (B5)

- Intuition:
  - \((A^T A)^k v'\)
  - \((A^T A)^k v'\)

- \(\cdots\)

Property (B5)

- Intuition:
  - \((A^T A)^k v'\)
  - \((A^T A)^k v'\)

- \(\cdots\)
Property (B5)

• Intuition:
  - \((A^TA)^k v'
  - \((A^TA)^k v'

(similarities to Smith)

users

products

users

products

Property (B5)

• Intuition:
  - \((A^TA)^k v'
  - \((A^TA)^k v'

A v'

A^T A v'

users

products

users

products

Less obvious properties - repeated:

\[ A(0): A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} V^T_{[r \times m]} \]
\[ B(1): A_{[n \times m]} (A^T)_{[m \times n]} = U A^2 U^T \]
\[ B(2): (A^T)_{[m \times n]} A_{[n \times m]} = V A^2 V^T \]
\[ B(3): (A^T)_{[m \times n]} A_{[n \times m]} = V A^{2k} V^T \]
\[ B(4): (A^T A)^k v \sim \lambda_v^{2k} v^T \]
\[ B(5): (A^T A)^k v' \sim (constant) v_1 \]
Least obvious properties

A(0): $A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} V^T_{[r \times m]}$

C(1): $A_{[n \times m]} \times_{[m \times 1]} = b_{[n \times 1]}$
let $x_0 = V \Lambda^{-1} U^T b$

- if under-specified, $x_0$ gives ‘shortest’ solution
- if over-specified, it gives the ‘solution’ with the smallest least squares error

(see Num. Recipes, p. 62)

Slowly:

Identity
$U$: column-orthonormal
Slowly:

\[ \square | = | \quad \text{blue} \]

\[ \text{red} = \quad \text{green} \]
Slowly:

Important: DROP small values of $\Lambda$
(say, $< 10^{-6} \lambda_1$)

Verify formula:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \end{bmatrix} \quad \mathbf{b} = [4]$$

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

$$\mathbf{U} = ??$$

$$\Lambda = ??$$

$$\mathbf{V} = ??$$

$$\mathbf{x}_0 = \mathbf{V} \Lambda^{(-1)} \mathbf{U}^T \mathbf{b}$$
Verify formula:

\[ \mathbf{A} = \begin{bmatrix} 1 & 2 \end{bmatrix}, \quad \mathbf{b} = [4] \]
\[ \mathbf{A} = \mathbf{U} \mathbf{\Lambda} \mathbf{V}^T \]
\[ \mathbf{U} = [1] \]
\[ \mathbf{\Lambda} = [\sqrt{5}] \]
\[ \mathbf{V} = \begin{bmatrix} 1/\sqrt{5} & 2/\sqrt{5} \end{bmatrix}^T \]
\[ \mathbf{x}_0 = \mathbf{V} \mathbf{\Lambda}^{-1} \mathbf{U}^T \mathbf{b} \]

Verify formula:

Show that \( w = 4/5, z = 8/5 \) is

(a) A solution to \( 1*w + 2*z = 4 \) and

(b) Minimal (wrt Euclidean norm)
Least obvious properties – cont’d

Illustration: over-specified, eg

\[ \begin{bmatrix} 3 & 2 \end{bmatrix}^T \begin{bmatrix} w \end{bmatrix} = \begin{bmatrix} 1 & 2 \end{bmatrix}^T \] (ie, 3 w = 1; 2 w = 2)

A = ??

b = ??

Verify formula:

\[ A = \begin{bmatrix} 3 & 2 \end{bmatrix}^T \quad b = \begin{bmatrix} 1 & 2 \end{bmatrix}^T \]

\[ A = U \Lambda V^T \]

\[ U = ?? \]

\[ \Lambda = ?? \]

\[ V = ?? \]

\[ x_0 = V \Lambda^{-1} U^T b \]
Verify formula:

\[
[3 \ 2]^T \begin{bmatrix} 7/13 \\ 21/13 \ 14/13 \end{bmatrix}^T \rightarrow \text{‘red point’ - perpendicular?}
\]

Least obvious properties - cont’d

\[A(0): \ A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} V_T^{[r \times m]}\]

\[C(2): \ A_{[n \times m]} v_1_{[m \times 1]} = \lambda_1 u_1_{[n \times 1]}\]

where \(v_1, u_1\) the first (column) vectors of \(V, U\). (\(v_1\) right-singular-vector)

\[C(3): \text{symmetrically: } u_1^T A = \lambda_1 v_1^T\]

\(u_1\) left-singular-vector

Therefore:

\[C(4): \ A^T A v_1 = \lambda_1^2 v_1\]

(fixed point - the defn of eigenvector for a symmetric matrix)
**Least obvious properties - altogether**

\[
A(0): A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} V^T_{[r \times m]}
\]

\[
C(1): A_{[n \times m]} x_{[m \times 1]} = b_{[n \times 1]}
\]
then, \( x_0 = V \Lambda^{-1} U^T b \): shortest, actual or least-squares solution

\[
C(2): A_{[n \times m]} v_1_{[m \times 1]} = \lambda_1 u_1_{[n \times 1]}
\]

\[
C(3): u_1^T A = \lambda_1 v_1^T
\]

\[
C(4): A^T A v_1 = \lambda_1^2 v_1
\]

**Properties - conclusions**

\[
A(0): A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} V^T_{[r \times m]}
\]

\[
B(5): (A^T A)^k v^T \sim (constant) v_1
\]

\[
C(1): A_{[n \times m]} x_{[m \times 1]} = b_{[n \times 1]}
\]
then, \( x_0 = V \Lambda^{-1} U^T b \): shortest, actual or least-squares solution

\[
C(4): A^T A v_1 = \lambda_1^2 v_1
\]

**SVD - detailed outline**

- ...
- Case studies
- SVD properties
- more case studies
  - Kleinberg/google algorithms
  - query feedbacks
- Conclusions

**Kleinberg’s algo (HITS)**

Kleinberg’s algorithm

• Problem dfn: given the web and a query
• find the most ‘authoritative’ web pages for this query

Step 0: find all pages containing the query terms
Step 1: expand by one move forward and backward

Kleinberg’s algorithm

• Step 1: expand by one move forward and backward

Kleinberg’s algorithm

• on the resulting graph, give high score (= ‘authorities’) to nodes that many important nodes point to
• give high importance score (‘hubs’) to nodes that point to good ‘authorities’

Kleinberg’s algorithm

• on the resulting graph, give high score (= ‘authorities’) to nodes that many important nodes point to
• give high importance score (‘hubs’) to nodes that point to good ‘authorities’
**Kleinberg’s algorithm**

**Observations**
- Recursive definition!
- Each node (say, ‘i’-th node) has both an authoritativeness score $a_i$ and a hubness score $h_i$

Let $E$ be the set of edges and $A$ be the adjacency matrix:
the $(i,j)$ is 1 if the edge from $i$ to $j$ exists

Let $h$ and $a$ be $[n \times 1]$ vectors with the ‘hubness’ and ‘authoritativness’ scores.

Then:

\[-a_i = h_k + h_l + h_m\]

that is
\[-a_i = \text{Sum}(h_j) \text{ over all } j \text{ that } (j,i) \text{ edge exists}\]

or
\[-a = A^T h\]

symmetrically, for the ‘hubness’:

\[-h_i = a_n + a_p + a_q\]

that is
\[-h_i = \text{Sum}(q_j) \text{ over all } j \text{ that } (i,j) \text{ edge exists}\]

or
\[-h = A a\]
Kleinberg’s algorithm

In conclusion, we want vectors $h$ and $a$ such that:

$$ h = A \cdot a $$

$$ a = A^T \cdot h $$

Recall properties:

C(2): $A_{[n \times m]} \cdot v_1_{[m \times 1]} = \lambda_1 u_1_{[n \times 1]}$

C(3): $u_1^T \cdot A = \lambda_1 v_1^T$

In short, the solutions to

$$ h = A \cdot a $$

$$ a = A^T \cdot h $$

are the left- and right- singular-vectors of the adjacency matrix $A$.

Starting from random $a'$ and iterating, we’ll eventually converge

(Q: to which of all the singular-vectors? why?)

Kleinberg’s algorithm - results

(Q: to which of all the singular-vectors? why?)

A: to the ones of the strongest singular-value, because of property B(5):

$$ B(5): (A^T \cdot A)^k \cdot v' \sim \text{(constant)} \cdot v_1 $$

Eg., for the query ‘java’:
0.328 www.gamelan.com
0.251 java.sun.com
0.190 www.digitalfocus.com (“the java developer”)
Kleinberg’s algorithm - discussion

• ‘authority’ score can be used to find ‘similar pages’ (how?)
• closely related to ‘citation analysis’, social networks / ‘small world’ phenomena

SVD - detailed outline

• ...
• Case studies
• SVD properties
• more case studies
  – Kleinberg/google algorithms
  – query feedbacks
• Conclusions

PageRank (google)


Problem: PageRank

Given a directed graph, find its most interesting/central node

A node is important, if it is connected with important nodes (recursive, but OK!)
**Problem: PageRank - solution**

Given a directed graph, find its most interesting/central node

Proposed solution: Random walk; spot most 'popular' node ($\rightarrow$ steady state prob. (ssp))

A node has high ssp, if it is connected with high ssp nodes (recursive, but OK!)

---

**PageRank - algorithm**

- Let $A$ be the adjacency matrix;
- let $B$ be the transition matrix: transpose, column-normalized - then

$$B \cdot p = p$$

- thus, $p$ is the eigenvector that corresponds to the highest eigenvalue ($=1$, since the matrix is column-normalized)

- Why does such a $p$ exist?
  - $p$ exists if $B$ is nxn, nonnegative, irreducible
    [Perron–Frobenius theorem]
(Simplified) PageRank algorithm

- \( B \mathbf{p} = 1 \ast \mathbf{p} \)
- thus, \( \mathbf{p} \) is the eigenvector\(^*\) that corresponds to the highest eigenvalue (\( = 1 \), since the matrix is column-normalized)
- Why does such a \( \mathbf{p} \) exist?
  - \( \mathbf{p} \) exists if \( B \) is \( n \times n \), nonnegative, irreducible
    [Perron–Frobenius theorem]

\(^*\) dfn: a few foils later

Full Algorithm

- With probability \( 1-c \), fly-out to a random node
- Then, we have
  \[ p = c B p + \frac{(1-c)}{n} \mathbf{1} \Rightarrow \]
  \[ p = \frac{(1-c)}{n} \left[ I - cB \right]^{-1} \mathbf{1} \]

Full version of algo: with occasional random jumps
Why? To make the matrix irreducible
Alternative notation – eigenvector viewpoint

\[ M = cB + \frac{(1-c)}{n} \mathbf{1}\mathbf{1}^T \]

Then

\[ p = M p \]

That is: the steady state probabilities = PageRank scores form the first eigenvector of the ‘modified transition matrix’

Parenthesis: intuition behind eigenvectors

- Definition
- 2 properties
- Intuition

Formal definition

If \( A \) is a \((n \times n)\) square matrix

\((\lambda, x)\) is an eigenvector/eigenvector pair of \( A \) if

\[ Ax = \lambda x \]

CLOSELY related to singular values:

Property #1: Eigen- vs singular-values

If

\[ B_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} (V_{[m \times r]})^T \]

then \( A = (B^T B) \) is symmetric and

\[ C(4): B^T B v_i = \lambda_i^2 v_i \]

ie, \( v_1, v_2, ... \): eigenvectors of \( A = (B^T B) \)
Property #2

• If $A_{nxn}$ is a real, symmetric matrix
• Then it has $n$ real eigenvalues

(if $A$ is not symmetric, some eigenvalues may be complex)

Property #3

• If $A_{nxn}$ is a real, symmetric matrix
• Then it has $n$ real eigenvalues
• And they agree with its $n$ singular values, except possibly for the sign

Parenthesis: intuition behind eigenvectors

• Definition
• 2 properties
• Intuition

Intuition

• $A$ as vector transformation

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$
**Intuition**

- By defn., eigenvectors remain parallel to themselves (‘fixed points’)

\[
\lambda_1 \begin{bmatrix} 0.52 \\ 0.85 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} 0.52 \\ 0.85 \end{bmatrix}
\]

**Convergence**

- Usually, fast:

\[
\frac{\lambda_1}{\lambda_2}
\]
Closing the parenthesis wrt intuition behind eigenvectors

Kleinberg/PageRank - conclusions

SVD helps in graph analysis:
hub/authority scores: strongest left- and right-singular-vectors of the adjacency matrix
random walk on a graph: steady state probabilities are given by the strongest eigenvector of the transition matrix

SVD - detailed outline

• ...
• Case studies
• SVD properties
• more case studies
  – google/Kleinberg algorithms
  – query feedbacks
• Conclusions

Query feedbacks

[Chen & Roussopoulos, sigmod 94]
Sample problem:
estimate selectivities (e.g., ‘how many movies were made between 1940 and 1945’?"
for query optimization,
LEARNING from the query results so far!!
Query feedbacks

• Given: past queries and their results
  – #movies(1925,1935) = 52
  – #movies(1948, 1990) = 123
  – ...
  – And a new query, say #movies(1979,1980)?

• Give your best estimate

For example
F(x) = # movies made until year ‘x’
     = a_1 + a_2 * x + a_3 * x^2 + … a_7 * x^6

GREAT idea #2: adapt your model, as you see the actual counts of the actual queries
Eventually, the problem becomes:
- estimate the parameters $a_1, \ldots, a_7$ of the model
- to minimize the least squares errors from the real answers so far.

Formally:
Query feedbacks

Formally, with $n$ queries and 6-th degree polynomials:

\[
\begin{bmatrix}
X_{11} & X_{12} & \cdots & X_{17} \\
X_{n1} & X_{n2} & \cdots & X_{n7}
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
\vdots \\
a_{14}
\end{bmatrix}
= 
\begin{bmatrix}
b_1 \\
b_2 \\
b_3 \\
\vdots \\
b_{14}
\end{bmatrix}
\]

where $x_{i,j}$ such that $\text{Sum} (x_{i,j} \cdot a_j) = \text{our estimate for the # of movies}$ and $b_j$: the actual

For example, for query ‘find the count of movies during (1920-1932)’:

\[
a_1 + a_2 \cdot 1932 + a_3 \cdot 1932^2 + \ldots - (a_1 + a_2 \cdot 1920 + a_3 \cdot 1920^2 + \ldots )
\]

And thus $X_{11} = 0; X_{12} = 1932-1920$, etc
Query feedbacks

In matrix form:

\[
\begin{bmatrix}
X_{11} & X_{12} & X_{17} \\
X_{21} & X_{22} & X_{27} \\
X_{71} & X_{72} & X_{77}
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_7
\end{bmatrix}
= 
\begin{bmatrix}
b_1 \\
b_2 \\
b_7 \\
b_7
\end{bmatrix}
\]

1st query

n-th query

Query feedbacks - enhancements

The solution

\[ a = V \Lambda^{-1} U^T b \]

works, but needs expensive SVD each time a new query arrives

GREAT Idea #3: Use 'Recursive Least Squares', to adapt \( a \) incrementally.

Details: in paper - intuition:
Query feedbacks - enhancements

Intuition:
least squares fit

\[ a_1 x + a_2 \]

new query

Query feedbacks - enhancements

Intuition:
least squares fit

\[ a'_1 x + a'_2 \]

new query

Query feedbacks - enhancements

the new coefficients can be quickly computed from the old ones, plus statistics in a (7x7) matrix
(no need to know the details, although the RLS is a brilliant method)

Query feedbacks - enhancements

GREAT idea #4: ‘forgetting’ factor - we can even down-play the weight of older queries, since the data distribution might have changed.
(comes for ‘free’ with RLS...)
Query feedbacks - enhancements

Intuition:

![Graph showing least squares fit and new query](image)

Query feedbacks - conclusions

SVD helps find the Least Squares solution, to adapt to query feedbacks
(\(RLS = \) Recursive Least Squares is a great method to incrementally update least-squares fits)

SVD - detailed outline

- Case studies
- SVD properties
- more case studies
  - google/Kleinberg algorithms
  - query feedbacks
- Conclusions

Conclusions

- SVD: a valuable tool
- given a document-term matrix, it finds ‘concepts’ (LSI)
- ... and can reduce dimensionality (KL)
- ... and can find rules (PCA; RatioRules)
Conclusions cont’d

• ... and can find fixed-points or steady-state probabilities (google/ Kleinberg/ Markov Chains)
• ... and can solve optimally over- and under-constraint linear systems (least squares / query feedbacks)

References


References cont’d