Organizational issues

• 1st prog. assignment’s DL this Sunday
  • tomorrow’s last tutorial on that one
• Next prog. assignment released later this week
  • DL 12 July
• If you want to meet with tutor outside tutorial, send email to agree the meeting
Chapter 3
Column and Column-Row Decompositions
Column Decompositions

Boutsidis et al. 2010; Strauch 2014
Motivation

• SVD is often hard to interpret and yields dense factorizations
  • NMF tries to address these problems with varying success
  • But if original data is sparse & easy to interpret, why not use it in the decompositions?
The CX decomposition

- In the **CX decomposition** we are given a matrix $A$ and a rank $k$, and we need to select $k$ columns of $A$ into matrix $C$ and build matrix $X$ s.t. we minimize $||A - CX||_\xi$

- $\xi$ is either $F$ or 2

- A.k.a. **column subset selection problem** (CSSP)
Why CX?

• The columns of $C$ preserve the original interpretation of columns of $A$
  • Even complex constraints are satisfied if the original data satisfied them

• Feature selection
  • Selects the columns that can be used to explain the rest
  • Compare to the dimensionality reduction
Alternative target function

• Building $\mathbf{C}$ is the hard part of $\mathbf{CX}$ decompositions

• Given $\mathbf{A}$ and $\mathbf{C}$, $\mathbf{X}$ can be computed with the pseudo-inverse

• $\mathbf{X} = \mathbf{C}^+ \mathbf{A}$

• Alternative target function for $\mathbf{CX}$:
  
  minimize $||\mathbf{A} - \mathbf{CC}^+ \mathbf{A}||_\xi = ||\mathbf{A} - \mathbf{PcA}||_\xi$
How to select C?

• Exhaustive: try all $\binom{m}{k}$ subsets of columns
  • Not very scalable

• Try to select the columns in a clever way
  • But how?

• Sample columns w.r.t. carefully selected probabilities
  • Avoids deterministic worst-case scenarios
Related idea: RRQR

- The **rank-revealing QR** (RRQR) factorization of matrix $A$ is

$$A \Pi = QR = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}$$

- $k$-by-$k$ upper-triangular with positive diagonal
- $k$-by-$(m-k)$
- $(n-k)$-by-$(m-k)$
- $n$-by-$n$ orthogonal
- $n$-by-$m$
- $k$th singular value of $A$
- Some polynomial on $k$ and $m$

$$\frac{\sigma_k(A)}{p_1(k, m)} \leq \sigma_{\min}(R_{11}) \leq \sigma_k(A)$$

$$\sigma_{k+1}(A) \leq \sigma_{\max}(R_{22}) \leq p_2(k, m) \sigma_{k+1}(A)$$

DMM, summer 2015

Pauli Miettinen
CX and RRQR

• Let $\mathbf{A} \mathbf{\Pi} = \mathbf{QR}$ and let $\mathbf{\Pi}_k$ be the first $k$ columns of $\mathbf{\Pi}$ and $\mathbf{C} = \mathbf{A} \mathbf{\Pi}_k$ some $k$ columns of $\mathbf{A}$

• Now $\|\mathbf{A} - \mathbf{P}_c \mathbf{A}\|_\xi = \|\mathbf{R}_{22}\|_\xi$, $\xi = F$ or 2

• In particular $\|\mathbf{A} - \mathbf{P}_c \mathbf{A}\|_2 \leq p_2(k, m)\|\mathbf{A} - \mathbf{A}_k\|_2$

• $\mathbf{A}_k = \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^T$ (truncated SVD)

• $\mathbf{CX}$ is $p_2(k, m)$-approximation to SVD
Computing CX by sampling

• Let $A = UΣV^T$ be the input and its SVD and $V_k$ the truncated $V$

• Sample columns of $A$ with replacement
  • Probability $p_j$ for selecting column $j$ is
    $p_j = \| (V_k^T)_j \|^2 / k$
  • Sample $O(k^2 \log(1/δ)/ε^2)$ columns and repeat $\log(1/δ)$ times returning the least-error sample
Notes on sampling

• We can prove that

\[ \|A - P_c A\|_F \leq (1 + \epsilon)\|A - A_k\|_F \]

with probability at least 1 − δ

• Notice that C has much more than k columns

• \(O(k^2 \log(1/\delta)/\epsilon^2)\) with large hidden constants
Why does sampling work?

• Intuitively, if $\mathbf{A}$ is of low rank ($k \ll n$), $\mathbf{A}$ should have many almost-similar columns

• If we sample many columns enough, we should get a representative for each set of similar columns

⇒ We need to sample more columns than the rank

• Or our error depends on the rank...
CX with exact $k$

• Construct larger-than-$k$ CX decomposition as above for $c = O(k \log k)$ columns
  
  • Let $\Pi_1$ be the $m$-by-$c$ matrix that selects $c$ columns s.t. $C = A\Pi_1$
  
  • Let $D_1$ be $c$-by-$c$ diagonal s.t. if $j$th column is selected on round $i$, $(D_1)_{ii} = (cp_j)^{-1/2}$
  
  • Run RRQR algorithm for $V_k^T \Pi_1 D_1$ to select exactly $k$ columns of $V_k^T \Pi_1 D_1$ with matrix $\Pi_2$ (c-by-$k$)
    
    • return $C = A\Pi_1\Pi_2$
Notes on the exact-$k$ CX

- $\Pr[\|A - P_cA\|_F \leq \Theta(k \log^{1/2} k)\|A - A_k\|_F] \geq 0.8$

- The sampling phase still requires really many columns (high hidden constants)
  - But in practice something like $c = 5k$ works

- Any RRQR algorithm can be used for the second step
  - But the analysis depends on the chosen algorithm
Non-Negative CX
Motivation

• If data is non-negative, so is \( C \)
  
  • But \( X \) can contain negative values in standard \( CX \)
  
  • Non-negative \( X \) yields “parts-of-whole” interpretation similar to NMF
    
    • Selected columns are “pure” while others are mixtures of the pure columns
  
  • Non-negativity also improves sparsity
The non-negative CX decomposition

• In the non-negative CX decomposition (NNCX) we are given a non-negative matrix $A$ and a rank $k$, and we need to select $k$ columns of $A$ into matrix $C$ and build a non-negative matrix $X$ s.t. we minimize $||A - CX||_F$
Geometry of NNCX

Columns in $\mathcal{C}$
Columns not in $\mathcal{C}$
Convex cone
Projections
Cones and columns

• Consider the cone spanned by columns of $A$, $\text{cone}(A)$

  • If removing column $j$ of $A$ changes the cone, that column is **extremal**
  
  • Otherwise it is **internal**

• Selecting all extremal columns to $C$ gives us $A = CX$ with nonnegative $X$
Algorithm for NNCX

• When we cannot select all extremal columns, we must choose which of them to select
  • Our goal is to maximize the volume of the convex cone
  • Finding the extremal columns is not easy
  • Given the columns, we must compute the non-negative projection
The convex_cone algorithm

• Set $R \leftarrow A$

• repeat

  • Select column $c$ with highest norm in the residual $R$

  • Normalize $c$ to unit norm

  • Solve nonnegative $x$ that minimizes $\| R - cx \|_T$

  • Set $R \leftarrow R - cx$

• until $k$ columns are selected

• Set $C$ to the columns of $A$ corresponding to the selected $c$ and solve nonnegative $X$ minimizing $\| A - CX \|_F$
Solving for non-negative $X$

- Given $C$, finding non-negative $X$ is the same as with NMF
  - Convex optimization with linear constraints
  - Or truncated-to-zero pseudo-inverse
Application: Neuroimaging

- Record brain cell activity over time
  - Every row is one frame
- Assume some columns contain the pure glomerulus signal
  - \( C \) identifies these signals
  - \( X \) explains how the signals are mixed in the brain images

Movie frame with real and found locations marked

- Human expert
- Algorithm
Applications to neuroimaging data

9.1.2 Signal selection with Convex cone NNCX on imaging data

For processing an imaging movie from the honeybee AL, we are interested in a factorisation of the movie matrix into factors with a temporal and factors with a spatial interpretation (Equation 4.1 from Section 4.2): We want to approximate the movie matrix $A$ (with $m$ time points and $n$ pixels) as

$$A = T^{m \times c} S^{c \times n} + N,$$

where matrix $T$ contains $c$ time series and matrix $S$ contains $c$ images. In addition, consider the noise term $N$.

Following the mixture model for imaging movies (Section 4.2.3), assume further that $S$ is non-negative, and that $T^2 A$, i.e. the pure glomerulus signals are contained in the columns of the movie matrix and can be selected. Without the noise term, that will be dealt with below, this is a NNCX factorisation (Section 2.1.2) that can be solved with Convex cone (Algorithm 8).

As seen in the evaluation in Chapter 7, Convex cone succeeds in selecting pure signal columns on (imaging) data with mixtures, which renders it a suitable algorithm for the task. For signal selection from honeybee imaging movies we can thus compute $T := C, S := X^0_+ = \text{Convex cone } (A, c)$.

Then, $T$ contains $c$ time series that can be used as input for higher-level data analysis as demonstrated later in Section 9.3. Matrix $S$ (see Figure 9.1 for an example) contains information about glomerulus shape, overlap and where in space the signals from $T$ are located. Successful selection of pure signals into $T$ leads to sparse images in $S$ (cp. Section 7.4.5) that allow for interpretation of the time series and images as the signal and shape of glomeruli.

Further, $A^c = T S$ is a rank-$c$ reconstruction of the original movie $A$ that can be used for visualisation purposes.

Fig. 9.1. Top-10 rows of matrix $S$ as obtained by Convex cone (Algorithm 8) performed on a calcium imaging movie of the honeybee AL (as in Section 7.5.1). Top-10 rows of $X$ from NNCX decomposition shows the shape and location of glomeruli.
Column-Row Decompositions

Leskovec et al. chapter 11.4; Skillicorn chapter 3.6.2
The CUR decomposition

• In the CUR decomposition we are given matrix $A$ and integers $c$ and $r$, and our task is to select $c$ columns of $A$ to matrix $C$ and $r$ rows to matrix $R$, and build $c$-by-$r$ matrix $U$ minimizing $\|A - CUR\|_F$

• Often $c = r = k$
Why CUR?

- If selecting the actual columns in CX is good, selecting the actual columns and rows must be even better
  - We find prototypical columns \textit{and rows}
  - $U$ is usually small, so if $C$ and $R$ are sparse, storing CUR takes little space
Solving CUR: general idea

• CUR is two-sided CX

• Simple algorithm idea:
  • Solve CX for $A$ and $A^T$ and solve for $U$ given $C$ and $R$
    • $U = C^+AR^+$
  • Better algorithms take into account the columns selected to $C$ when computing $R$
Simple CUR algorithm

• Sample columns proportional to their $L_2$-norm
• Sample rows proportional to their $L_2$-norm
• Build $W = A[R,C]$ (the sub-matrix of columns in $R$ and rows in $C$)
  • Let $W = X\Sigma Y^T$ be an SVD of $W$, and set $U \leftarrow Y(\Sigma^+)^2X^T$
Fancier CUR algorithm

• Find $C$ similar to exact-$k$ CX earlier
  • Sample $O(k/\epsilon)$ additional columns
• Find $Z \in \text{span}(C)$, $Z^T Z = I$, such that
  $$||A^T - A^T ZZ^T||_F \leq (1 + O(\epsilon))||A - CX^*||_F$$
  • Use $Z$ to get the probabilities for sampling $O(k \log k)$ rows of $A$ and reduce that to $O(k)$ rows
• Sample $O(k/\epsilon)$ additional rows
  • Set $U = X^* Z^T A R^+$
Comments on the Boutsidis & Woodruff algorithm

• Slight variations of the above algorithm achieve:
  • selects the smallest number of rows and columns for \((1+\epsilon)\) approximation
  • matrix \(U\) has the smallest possible rank
CX and CUR summary

- Rows and columns of the original data should be interpretable
  - Also admit local constraints in the data
- CX and CUR decompositions are forms of feature selection
  - Applications when we need “prototypical” rows and columns
Literature

• Drineas, Mahoney & Muthukrishnan (2006): *Subspace sampling and relative-error matrix approximation: Column-based methods*. In APPROX/RANDOM ’06

