Co-clustering as Multilinear Decomposition with Sparse Latent Factors

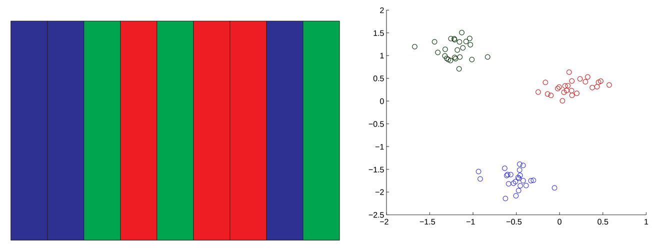
Evangelos. E. Papalexakis[1] Nicholas. D. Sidiropoulos[1]

[1]Department of Electronic & Computer Engineering, Technical University of Crete, Chania, Greece

K-means / VQ as constrained outer product decomposition

Cluster set of vectors $\{\mathbf{x}_j \in \mathbb{R}^I\}_{j=1}^J$ in K clusters:

- Find $K << J$ cluster means $\{\boldsymbol{\mu}_k \in \mathbb{R}^I\}_{k=1}^K$ and an assignment of each $\mathbf{x}_j$ to a best-matching cluster $k^*(j)$ such that $\sum_j \|\mathbf{x}_j - \boldsymbol{\mu}_{k^*(j)}\|^2$ (or other suitable mismatch cost) is minimized



- $\mathbf{X} := [\mathbf{x}_1, \cdots, \mathbf{x}_J]$ $(I \times J)$, $\mathbf{M} := [\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_K]$ $(I \times K)$, and $\mathbf{A} := [\mathbf{a}_1, \cdots, \mathbf{a}_K]$ $(J \times K)$, with: $\mathbf{A}(j,k) = \mathbf{a}_k(j) \in \{0,1\}$ and $\sum_{k=1}^K \mathbf{A}(j,k) = 1, \forall j$ (i.e., each row sums to 1; $\mathcal{RS}$ constraint)
- K-*means* clustering:

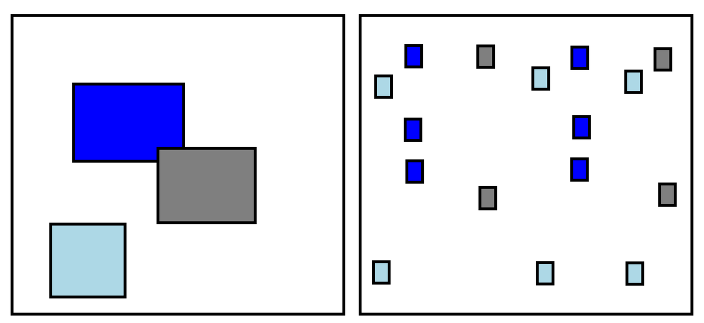
$$\min_{\mathbf{M},\mathbf{A} \in \{0,1\}^{J\times K} \bigcap \mathcal{RS}} \|\mathbf{X} - \mathbf{M}\mathbf{A}^T\|_F^2,$$

- K-*means* $\leftrightarrow$ low-rank "decomposition": $\min \|\mathbf{X} - (\boldsymbol{\mu}_1\mathbf{a}_1^T + \cdots + \boldsymbol{\mu}_K\mathbf{a}_K^T)\|_F^2$ i.e., $\mathbf{X} \simeq \boldsymbol{\mu}_1\mathbf{a}_1^T + \cdots + \boldsymbol{\mu}_K\mathbf{a}_K^T$
- $\exists$ important difference: $\mathbf{A} \in \{0,1\}^{J\times K} \bigcap \mathcal{RS}$
- NP-hard; popular approximation: Lloyd-Max
- Binary $\{0,1\}$ constraint $\leftrightarrow$ *hard* clustering. Relax to $[0,1]$ interval (or simply ≥ 0) $\leftrightarrow$ *soft clustering* weights
- $\mathcal{RS}$ constraint: *every* vector is classified (*lossless* clustering). Drop $\mathcal{RS}$ $\leftrightarrow$ *lossy* (exploratory) clustering; [all-zero rows in $\mathbf{A}$ OK]: spot *important* clusters

Co-clustering

Introducing co-clustering: Amazon.com

- Each customer $\leftrightarrow$ vector, across list of products (and vice-versa): matrix $\mathbf{X}$
- **Not** interested in grouping customers (or products); but rather in ...
- ... spotting *co-clusters*: subsets of customers that tend to buy same subset of products
- ... even though their overall buying patterns **can otherwise be very different**.
- Don't know which subset(s) are of interest; had we known, problem would have been reduced to K-means
- Regular clustering fails to capture such patterns **because it postulates similarity in all dimensions**



Prior art

- J.A. Hartigan, JASA 1972, 1975. Hard co-clustering NP-hard (K-means is special case)
- $\exists$ many ad-hoc (re)formulations and algorithms, e.g., I. Dhillon: spectral, information-theoretic; A. Banerjee, I. Dhillon, *et al* max entropy Bregman co-clustering
- Many applications: social network analysis, data & web mining, medicine, biology (gene expression), market basket analysis, census.

Multi-mode / multi-way co-clustering

- Mostly two-mode (aka two-way) *bi-clustering*; very limited multi-mode / multi-way
- Important in numerous applications - unfolding ignores structure
- L. Zhao, M.J. Zaki, triclustering, SIGMOD 2005; Q. Zhou, G. Xu, Y. Zong, 2009
- Don't know which 3-way model to use (Tucker? PARAFAC?) when in fact none of the existing ones fits
- Our contribution: start from first principles, *derive* the right multi-way model

Cluster / co-cluster: rank-1 modeling

- Assume data ≥ 0, variables ≥ 0 (for the moment)
- Standard clustering: single cluster $\Leftrightarrow$ $\mathbf{X} = \boldsymbol{\mu}\mathbf{a}^T +$ noise, where $\mathbf{a}(j) \in \{0,1\}$.
- $\mathbf{a}$ *selects* which columns belong to the given cluster
- When only *relative expression* matters (e.g., gene co-expression; market analysis), generalize as: $\mathbf{X} = \mathbf{b}\mathbf{a}^T +$ noise
- In co-clustering: $\mathbf{b}$ will be $\neq 0$ only for the selected rows; likewise, $\mathbf{a}$ will be $\neq 0$ only for the selected columns
- Co-cluster $\leftrightarrow$ rank-one component ($\mathbf{X} = \mathbf{b}\mathbf{a}^T +$ noise) w/ many (latent) zeros
- Co-clustering $\leftrightarrow$ outer prod decomp (rank = # of co-clusters)

Why sparsity is key

- Sparsity: selects! Improves uniqueness and reduces noise
- Two-way (matrix) case (bi-clustering): Sparse bilinear decomposition
- Can be implemented using (non-neg) Lasso in alternating fashion:

$$\min_{\mathbf{B}\geq 0, \mathbf{A}\geq 0} \|\mathbf{X} - \mathbf{B}\mathbf{A}^T\|_F^2 + \lambda \sum_{i,k} |\mathbf{B}(i,k)| + \lambda \sum_{j,k} |\mathbf{A}(j,k)|$$

- Three- and higher-way case $\rightarrow$ Sparse multi-linear (PARAFAC) decomposition

The Sparse PARAFAC decomposition

- Consider three way array $\underline{\mathbf{X}} \in \mathbb{R}^{I\times J\times N}$.
- PARAFAC w/ SLF (Sparse Latent Factors) decomposition in K rank-one components:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \|\underline{\mathbf{X}} - \sum_{k=1}^K \mathbf{a}_k \odot \mathbf{b}_k \odot \mathbf{c}_k\|_F^2 + \lambda|\mathbf{A}| + \lambda|\mathbf{B}| + \lambda|\mathbf{C}|$$

$\mathbf{A} \in \mathbb{R}^{I\times K}$, $\mathbf{B} \in \mathbb{R}^{J\times K}$ and $\mathbf{C} \in \mathbb{R}^{N\times K}$ contain vectors $\mathbf{a}_k, \mathbf{b}_k, \mathbf{c}_k$ as columns; $|\mathbf{A}| := \sum_{i,k} |\mathbf{A}(i,k)|$

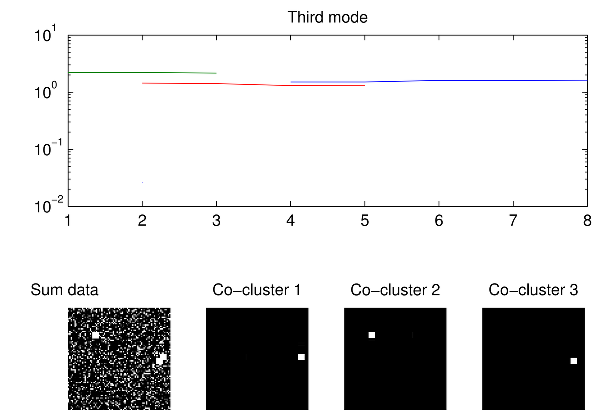
- λ is sparsity-controlling regularization parameter
- Include non-negativity when appropriate
- Solved "a-la" ALS, using *Lasso* steps for $\mathbf{A}$, $\mathbf{B}$ & $\mathbf{C}$
- Can use different λ's for the different modes ... but then solution is trickier :-)

Sparse PARAFAC: Uniqueness

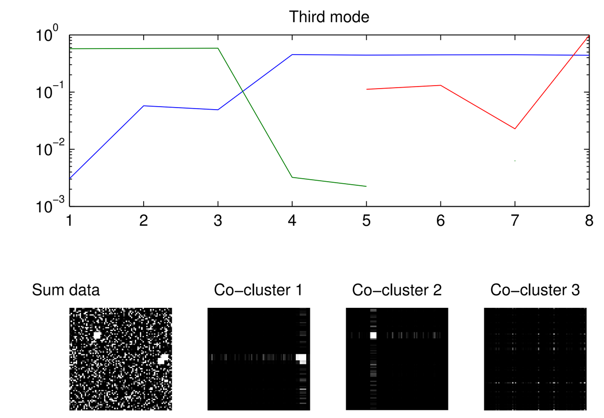
- Even without non-negativity or sparsity, the PARAFAC decomposition is unique under mild conditions (big advantage over the matrix case)
- Kruskal, 1977: $k_{\mathbf{A}} + k_{\mathbf{B}} + k_{\mathbf{C}} \geq 2K + 2$: Sidiropoulos *et al*, 2000: Jiang & Sidiropoulos, 2004; De Lathauwer *et al*, Stegeman *et al*, ...
- Sparsity & non-negativity improve uniqueness
- **Why is this important?**
  - ▷ Can unravel large # of possibly overlapping co-clusters! - *very important*
- Impossible to do as well in the matrix case

Experimental Results

Synthetic Co-clusters



(c) PARAFAC w/ NN SLF, $\lambda = 12$

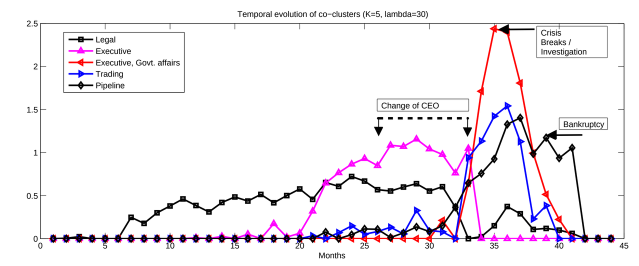


(d) NN PARAFAC (no sparsity)

- Support of each co-cluster is perfectly recovered, and noise has been effectively removed. (Sparse PARAFAC)
- Loss of localization and merging of overlapping co-clusters (NN PARAFAC).

The Enron e-mail corpus

- $168 \times 168 \times 44$: sender $\times$ receiver $\times$ month ('98-'02)



(e) Temporal co-cluster profiles for ENRON, $K = 5$ and $\lambda = 30$.

$K = 1$	Legal	-	-	-	-
$K = 2$	Legal	Executive, Govt. affairs	-	-	-
$K = 3$	Legal	Executive, Govt. affairs	Trading	-	-
$K = 4$	Legal	Executive, Govt. affairs	Trading	Pipeline	-
$K = 5$	Legal	Executive	Executive, Govt. affairs	Trading	Pipeline

(f) Extracted co-clusters for ENRON ($\lambda = 30$)

- Our analysis is consistent (in terms of clique labels) with earlier ones, such as Bader *et al*; but our co-clusters are much cleaner
- Results are 'nested': $K \rightarrow K + 1 \Longrightarrow$ previously extracted co-clusters remain stable as new co-cluster is added!
- Not true for PARAFAC, NN-PARAFAC; only PARAFAC w/ NN SLF (Sparse Latent Factors)

On-going work

Nesting property

- Requires positive λ, becomes more accurate with increasing λ (and K). Co-cluster support remains intact even for moderate λ
- Corollary: greedy SPARAFAC is just as good
- Much faster and cheaper - important for scalability (very large datasets)
- Theoretical investigation of nesting

Other issues

- Asymmetric sparsity penalties
- Better, simpler, faster algorithms
- Data dependent, automatic ways of choosing λ
- Uniqueness issues - how is PARAFAC uniqueness improved by sparsity?