

# Latent Feature Lasso

Ian E.H. Yen<sup>\*</sup>, Wei-Cheng Lee<sup>†</sup>, Sung-En Chang<sup>†</sup>, Arun Suggala<sup>\*</sup>,  
Shou-De Lin<sup>†</sup> and Pradeep Ravikumar<sup>\*</sup>.

<sup>\*</sup> Carnegie Mellon University

<sup>†</sup> National Taiwan University

Presenter: Wei-Cheng Lee

# Outline

- 1 Latent-Feature Models
- 2 Brief Survey of Previous Research
- 3 Latent Feature Lasso—A Convex Estimator
  - Convex Formulation via Atomic Norm
  - Greedy Coordinate Descent via MAX-CUT
- 4 Theoretical Results
- 5 Empirical Results
- 6 Some Details

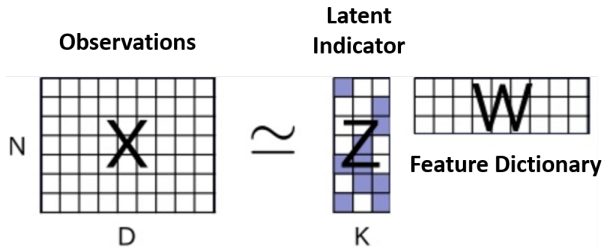
# Latent Feature Models

- In **Latent Feature Model**, each observation

$$\mathbf{x}_n = W^T \mathbf{z}_n + \epsilon_n$$

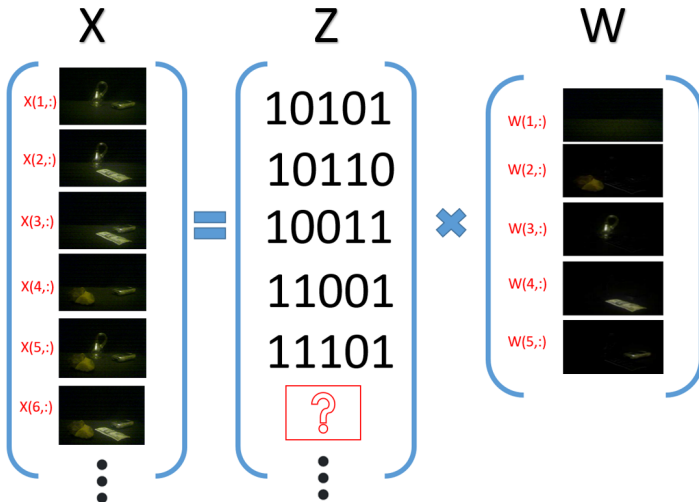
where  $\mathbf{x}_n \in \mathbb{R}^D$ : observation,  $W \in \mathbb{R}^{K \times D}$ : feature dictionary,  $\mathbf{z}_n \in \{0, 1\}^K$ : binary latent indicators, and  $\epsilon_n \in \mathbb{R}^D$ : noise.

- Mixture Model** is a special case with  $\|\mathbf{z}_n\|_0 = 1$ .



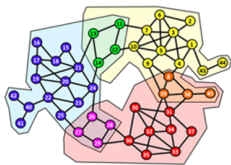
# Latent Feature Model: Tabletop Dataset

- $X \in \mathbb{R}^{ND}$ : dataset,  $Z \in \{0, 1\}^{NK}$ : binary latent indicators,  $W \in \mathbb{R}^{KD}$ : feature dictionary, here  $K=5$ ,  $D$ =picture size.

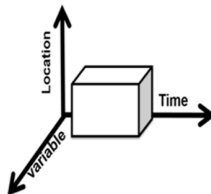


# Latent Feature Model: Why Binary and Applications

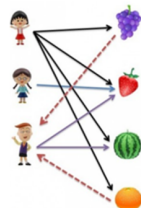
- Why binary? (interpretability, semi-supervision, and computational efficiency.)



Community Detection



Spatial-Temporal Factorization



Recommendation

N	People	Location	Movies
D	People	Meteorological data	People
X value	0/1 for being friends or paper coauthorship	real value for temperature, precipitation, wind scale ...	Ratings
possible latent K	Groups, Affiliations	Weather events (typhoon, blizzard, cold current...)	Film genres

# Let's Play a Game

- Under Latent Feature Model assumptions, Can you identify the latent features by your own eyes?



R36



R37



R38



R39



R40



R45



R46



R47



R48



R49



R54



R55



R56



R57



R58



R63



R64



R65



R66



R67



R72



R73



R74



R75



R76



W1



W2



W3



W4



W5



W6



W7



W8



W9



W10



W11



W12



W13



W14



W15

# Latent Feature Models: Result Summary

- **Goal:** Find **dictionary**  $W_{K \times D}$  and **latent indicators**  $Z : N \times K$  that best approximates **observation**  $X : N \times D$ .
- **Existing Approaches:**
  - **MCMC, Variational (Indian Buffet Process):**  
No finite-time guarantee.
  - **Spectral Method (Tung 2014):**  
 $O(DK^6)$  sample complexity. ( $z \sim \text{Ber}(\pi)$ ,  $x \sim N(W^T z, \sigma)$ ).
  - **Matrix Factorization (Slawski et al., 2013):**  
 $O(NK2^K)$  runtime complexity for exact recovery (noiseless).
- **This Paper:**
  - A convex estimator — Latent Feature Lasso.
  - Low-order polynomial **runtime** and **sample complexity**.
  - **No restrictive assumption** on  $p(X)$ , even allows **model mis-specification**.

# Outline

- 1 Latent-Feature Models
- 2 Brief Survey of Previous Research
- 3 Latent Feature Lasso—A Convex Estimator
  - Convex Formulation via Atomic Norm
  - Greedy Coordinate Descent via MAX-CUT
- 4 Theoretical Results
- 5 Empirical Results
- 6 Some Details



# Previous Research

- Indian Buffet Process is a standard probability prior for the latent feature model.
- Previous works focus on using Gibbs sampler or Variational Inference to solve it.
- Spectral Method and BMF use different ways, but they have their own limitations.
- Sorry, I don't have time to survey previous works in details. So we just skip it QQ.

# Outline

- 1 Latent-Feature Models
- 2 Brief Survey of Previous Research
- 3 Latent Feature Lasso—A Convex Estimator
  - Convex Formulation via Atomic Norm
  - Greedy Coordinate Descent via MAX-CUT
- 4 Theoretical Results
- 5 Empirical Results
- 6 Some Details

# Latent Feature Model: Problem Formulation

- Goal: Empirical Risk Minimization:

$$\min_{Z \in \{0,1\}^{N \times K}} \left\{ \min_{W \in \mathbb{R}^{K \times D}} \underbrace{\frac{1}{2N} \|X - ZW\|_F^2}_L + \frac{\tau}{2} \|W\|_F^2 \right\},$$

- Given  $Z$ , the Lagrangian dual problem w.r.t. primal variables  $W$  is:

$$\min_{M=ZZ^T \in \{0,1\}^{N \times N}} \underbrace{\left\{ \max_{A \in \mathbb{R}^{N \times D}} \frac{-1}{2\tau} \text{tr}(AA^T \mathbf{ZZ}^T) - \underbrace{L^*}_{\text{conjugate } L}(-A) \right\}}_{g(M)}.$$

- **Tricks:** Introducing dummy variable  $E = ZW$ , where  $A$  is its dual variable.
- **Key insight:** the function is convex w.r.t.  $M = \mathbf{ZZ}^T$ .

# Latent Feature Model: Estimator

- Let  $\mathcal{S} := \{\mathbf{z}\mathbf{z}^T \mid \mathbf{z} \in \{0,1\}^N\}$ .
- The "Latent-Feature" Atomic Norm:

$$\|M\|_{\mathcal{S}} := \min_{\mathbf{c} \geq 0} \sum_{\mathbf{z}\mathbf{z}^T \in \mathcal{S}} c_{\mathbf{z}} \quad \text{s.t.} \quad M = \sum_{\mathbf{z}\mathbf{z}^T \in \mathcal{S}} c_{\mathbf{z}} \mathbf{z}\mathbf{z}^T.$$

- The Latent Feature Lasso estimator:

$$\min_M g(M) + \lambda \|M\|_{\mathcal{S}}.$$

- Equivalently, one can solve the estimator by

$$\min_{\mathbf{c} \in \mathbb{R}_+^{|\mathcal{S}|}} g\left(\sum_{k=1}^{2^N} c_k \mathbf{z}_k \mathbf{z}_k^T\right) + \lambda \|\mathbf{c}\|_1$$

**Question:** How to optimize with  $|\mathcal{S}| = 2^N$  variables?

# Outline

- 1 Latent-Feature Models
- 2 Brief Survey of Previous Research
- 3 Latent Feature Lasso—A Convex Estimator
  - Convex Formulation via Atomic Norm
  - Greedy Coordinate Descent via MAX-CUT
- 4 Theoretical Results
- 5 Empirical Results
- 6 Some Details

# Greedy Coordinate Descent via MAX-CUT

- At each iteration, we find the **coordinate of steepest descent**:

$$j^* = \underset{j}{\operatorname{argmax}} -\nabla_j f(c) = \underset{z \in \{0,1\}^N}{\operatorname{argmax}} \langle -\nabla g(M), zz^T \rangle$$

which is a **Boolean Quadratic problem** that can be reformulated to **MAX-CUT**:

$$\max_{z \in \{0,1\}^N} z^T C z$$

- Can be solved to a **3/5-approximation (Nesterov)** by rounding from a special type of **SDP with  $O(ND)$  iterative solver (Po-Wei Wang's 2016)**.

# Greedy Coordinate Descent via MAX-CUT

0.  $\mathcal{A} = \emptyset$ ,  $\mathbf{c} = \mathbf{0}$ .

**for**  $t = 1 \dots T$  **do**

1. Find an **approximate greedy** atom  $\mathbf{z}\mathbf{z}^T$  by MAX-CUT-like problem:

$$\max_{\mathbf{z} \in \{0,1\}^N} \langle -\nabla g(\mathbf{M}), \mathbf{z}\mathbf{z}^T \rangle.$$

2. Add  $\mathbf{z}\mathbf{z}^T$  to an **active set**  $\mathcal{A}$ .

3. **Refine**  $\mathbf{c}_{\mathcal{A}}$  via Proximal Gradient Method on:

$$\min_{\mathbf{c} \geq 0} g\left(\sum_{k \in \mathcal{A}} c_k \mathbf{z}_k \mathbf{z}_k^T\right) + \lambda \|\mathbf{c}\|_1$$

4. Eliminate  $\{\mathbf{z}_k \mathbf{z}_k^T \mid c_k = 0\}$  from  $\mathcal{A}$ .

**end for**.

- Evaluating  $\nabla g(\mathbf{M})$  requires solving a **least-square problem** of cost  $O(DK^2)$ .
- Each iteration costs  $\underbrace{O(ND)}_{\text{MAX-CUT}} + \underbrace{O(DK^2)}_{\text{Least-Square}}$

# Outline

- 1 Latent-Feature Models
- 2 Brief Survey of Previous Research
- 3 Latent Feature Lasso—A Convex Estimator
  - Convex Formulation via Atomic Norm
  - Greedy Coordinate Descent via MAX-CUT
- 4 Theoretical Results
- 5 Empirical Results
- 6 Some Details



# Convergence Analysis

Given any reference solution  $c^*$ , the  $t$ -th iteration of the Greedy Algorithm satisfies

$$F(c^t) - F(c^*) \leq \frac{2\gamma \|c^*\|_1^2}{\mu^2} \left( \frac{1}{t} \right) + \underbrace{\frac{2(1-\mu)}{\mu} \lambda \|c^*\|_1}_{\Delta(\lambda)},$$

$\mu = 3/5$  is the approximation ratio given by the MAX-CUT-like problem and  $\gamma$  is the Lipschitz-continuous parameter of  $\nabla_j f(c)$ .

- $\Delta(\lambda)$  decreases with  $N$  when  $\lambda$  is chosen to trade off between bias and variance.

# Risk Analysis

Let the **population risk** of a dictionary  $W$  be

$$r(W) := E\left[\min_{z \in \{0,1\}^K} \frac{1}{2} \|\mathbf{x} - W^T \mathbf{z}\|^2\right].$$

Let  $W^*$  be an optimal dictionary of size  $K$ , the algorithm outputs  $\hat{W}$  with bounds

$$r(\hat{W}) \leq r(W^*) + \epsilon$$

under probability  $1 - \rho$  as long as

$$t = \Omega\left(\frac{K}{\epsilon}\right) \quad \text{and} \quad N = \Omega\left(\frac{DK}{\epsilon^3} \log\left(\frac{RK}{\epsilon\rho}\right)\right).$$

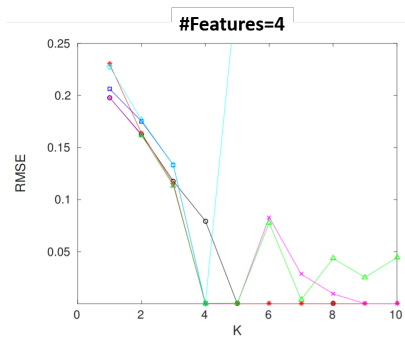
and  $\lambda$   $\tau$  chosen appropriately corresponding to  $N$ .

- The result trades between **risk** and **sparsity**.
- **No assumption** on  $\mathbf{x}$  except that of boundedness.
- The **sample complexity** is (quasi) linear to  $D$  and  $K$ .

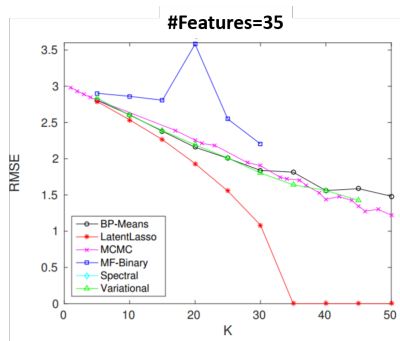
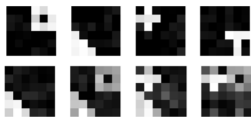
# Outline

- 1 Latent-Feature Models
- 2 Brief Survey of Previous Research
- 3 Latent Feature Lasso—A Convex Estimator
  - Convex Formulation via Atomic Norm
  - Greedy Coordinate Descent via MAX-CUT
- 4 Theoretical Results
- 5 Empirical Results
- 6 Some Details

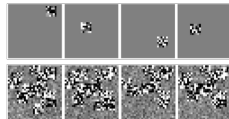
# Results on Synthetic Data



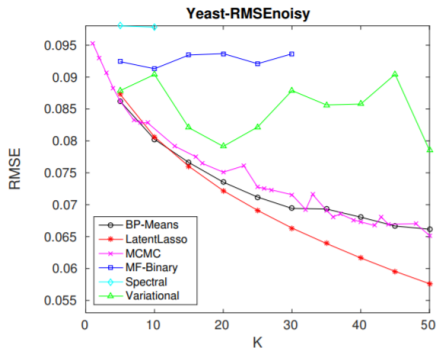
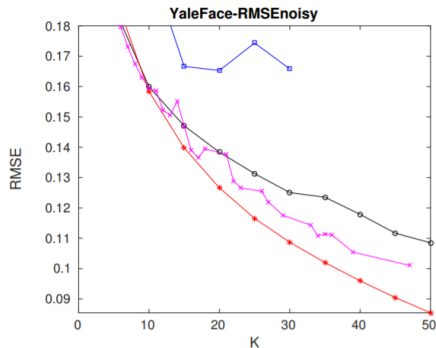
$K_{\text{True}}=4$



$K_{\text{True}}=35$



# Results on Real Data



MCMC	Variational	MF-Binary	BP-Means	Spectral	LatentLasso
$(NDK^2)T$	$(NDK^2)T$	$(NK)2^K$	$(NDK^3)T$	$ND + K^5 \log(K)$	$(ND + K^2 D)T$

- MCMC, Variational, BP-Means take up to 1000s training time, while LatentLasso takes up to 100s.

# Conclusion

- In this work, we propose a novel **convex estimator (Latent Feature Lasso)** for the estimation of Latent Feature Model.
- To best of our knowledge, this is the first method with low-order polynomial **runtime** and **sample complexity without restrictive assumptions** on the data distribution.
- In experiments, the Latent Feature Lasso **significantly outperforms** other methods in terms of **accuracy** and **time**, when there is a **larger number of latent features**.

- No thorough studies of every parameters and design of programs. We even don't do any line search.
- Lack of meaningful real-world dataset and can't think of real world application.
- Need to generalize the model to consider non-linear transforms to rule out location dependency of latent features.
- I believe that the bounds of the risk and convergence rate may not be well-studied. I believe that there is space for theoretically improvements.
- We try to apply our model in Community Detection and Recommendation, but it is not strong. What cause that?

# Outline

- 1 Latent-Feature Models
- 2 Brief Survey of Previous Research
- 3 Latent Feature Lasso—A Convex Estimator
  - Convex Formulation via Atomic Norm
  - Greedy Coordinate Descent via MAX-CUT
- 4 Theoretical Results
- 5 Empirical Results
- 6 Some Details



# How to derive W and E and A

$$\min_{W \in \mathcal{R}^{K \times D}} \frac{1}{2N} \|X - Z_{\mathcal{A}} W\|_F^2 + \frac{\tau}{2} \|W\|_F^2$$

When  $Z_{\mathcal{A}}$  is fixed,  $W$  has closed form solution

$$W^* = (Z_{\mathcal{A}}^T Z_{\mathcal{A}} + N\tau I)^{-1} Z_{\mathcal{A}}^T X$$

Given Lagrangian

$$L(W, E, A) = \frac{1}{2N} \|X - E\|_F^2 + \frac{\tau}{2} \|W\|_F^2 + \langle A, E - ZW \rangle$$

$$0 \in \partial L \text{ (stationary)}$$

$$E = ZW \text{ (primal feasibility)}$$

# Subgradient

A subgradient of a convex function  $f : R^n \rightarrow R$  at a given point  $x \in R^n$  is any  $g \in R^n$  satisfies

$$\partial f(x) = \{g | f(y) \geq f(x) + g^T(y - x) \forall y\}$$

- $\partial f(x)$  always exists in this case.
- When  $f(x)$  is differentiable,  $\partial f(x) = \{\nabla f(x)\}$
- $x^*$  is optimal  $\iff 0 \in \partial f(x^*)$

# Proximal Gradient Descent-1

When convex  $f(x)$  is not totally differentiable but we can factorize it to

$$f(x) = g(x) + h(x)$$

where  $g(x)$  is convex/differentiable but  $h(x)$  is only convex. Same as what we do in gradient descent, we define

$$x^+ = x - t\nabla f(x)$$

but we need a proximal mapping operator defined as

$$\text{prox}_t(x) = \underset{z}{\operatorname{argmin}} \frac{1}{2t} \|z - x\|^2 + h(z)$$

and in Proximal Gradient Descent, our update rule is given by

$$x^+ = \text{prox}_t(x - t\nabla g(x))$$

## Proximal Gradient Descent-2

- We need to have an efficient way to compute the proximal operator itself (in closed form), otherwise we need to solve a quadratic optimization problem at every steps.
- Obtains convergence rate between subgradient methods and gradient methods.

# Danskin's theorem

Suppose  $\phi(x, z)$  is a continuous function

$$\phi : \mathcal{R}^n \times Z \rightarrow \mathcal{R}$$

where  $Z \in \mathcal{R}^m$  is a compact set, and  $\phi(x, z)$  is convex in  $x$  for every  $z \in Z$   
define

$$f(x) = \max_{z \in Z} \phi(x, z)$$

$$Z_0(x) = \{\bar{z} : \phi(x, \bar{z}) = \max_{z \in Z} \phi(x, z)\}$$

then  $f(x)$  is convex and

$$D_y f(x) = \max_{z \in Z_0(x)} \phi'(x, z; y)$$

# Max-Cut Solver-1

We can replace  $X \succeq 0$  constraints with  $X = V^T V$  for some  $V \in \mathcal{R}^{k \times n}$  then  $X_{ii} = 1$  translates to  $\|v_i\| = 1$  leads to non-convex optimization problem

$$\min_{V \in \mathcal{R}^{k \times n}} \langle C, V^T V \rangle \text{ subject to } \|v_i\| = 1, i = 1, \dots, n$$

Solve it by coordinate descent method, minimizing  $v_i$  that depends on  $v_i^T (\sum_{j=1}^n C_{ij} v_j)$  since  $\|v_i\| = 1$  we can assume that  $C_{ii} = 0$  without affecting the solution, so the problem is equivalent to

$$\min_{v_i \in \mathcal{R}^k} v_i^T g \text{ subject to } \|v_i\| = 1$$

The solution has closed form

$$v_i = \frac{-g}{\|g\|}$$

This is from "The Mixing method: coordinate descent for low-rank semi-definite programming (Po-Wei Wang) NIPS 2016"

**Algorithm 1:** The Mixing method

```
1 Initialize  $v_i$  randomly on a unit sphere;  
2 while not yet converged do  
3   for  $i = 1, \dots, n$  do  
4      $v_i := \text{normalize}(-\sum_{j=1}^n C_{ij}v_j)$ ;  
5   end  
6 end
```

This way, we can initialize  $v_i$  on unit sphere and perform cyclic update over all the  $i = 1, \dots, n$  in closed-form. We called it the mixing method, because for each  $v_i$  it mixes and normalizes the remaining vectors  $v_j$  according to weight  $C_{ij}$ . Thus, in the case of sparse  $C$  (which is the normal case for any large data problem) the time complexity for updating all variable once is  $O(k\#\text{nnz})$ , which is significantly cheaper than the interior point method. However, the details

for efficient computation differ depending on the precise nature of the SDP, so we will describe these in more detail in the subsequent application sections. A complete description of the generic algorithm is shown in Algorithm 1.