Generalized Low Rank Models

Madeleine Udell

Cornell ORIE

Based on joint work with

Stephen Boyd, Corinne Horn, Reza Zadeh, Nathan Kallus, Alejandro Schuler, Nigam Shah, Anqi Fu, Nandana Sengupta, Nati Srebro, James Evans, Damek Davis, and Brent Edmunds

MIIS Tutorial 12/17/2016

Outline

Data table

age	gender	state	diabetes	education	⋯
22	F	CT	?	college	⋯
57	?	NY	severe	high school	⋯
?	M	CA	moderate	masters	⋯
41	F	NV	none	?	⋯
⋮	⋮	⋮	⋮	⋮	

- detect demographic groups?
- find typical responses?
- identify related features?
- impute missing entries?

Data table

m examples (patients, respondents, households, assets)
n features (tests, questions, sensors, times)

$$\begin{bmatrix} & A & \end{bmatrix} = \begin{bmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & \ddots & \vdots \\ A_{m1} & \cdots & A_{mn} \end{bmatrix}$$

- ith row of A is feature vector for ith example
- jth column of A gives values for jth feature across all examples

Low rank model

given: A, $k \ll m, n$
find: $X \in \mathbf{R}^{m \times k}$, $Y \in \mathbf{R}^{k \times n}$ for which

$$\begin{bmatrix} \\ X \\ \\ \end{bmatrix} \begin{bmatrix} \quad Y \quad \end{bmatrix} \approx \begin{bmatrix} \\ A \\ \\ \end{bmatrix}$$

i.e., $x_i y_j \approx A_{ij}$, where

$$\begin{bmatrix} \\ X \\ \\ \end{bmatrix} = \begin{bmatrix} —x_1— \\ \vdots \\ —x_m— \end{bmatrix} \qquad \begin{bmatrix} \quad Y \quad \end{bmatrix} = \begin{bmatrix} | & & | \\ y_1 & \cdots & y_n \\ | & & | \end{bmatrix}$$

interpretation:

- X and Y are (compressed) representation of A
- $x_i^T \in \mathbf{R}^k$ is a point associated with example i
- $y_j \in \mathbf{R}^k$ is a point associated with feature j
- inner product $x_i y_j$ approximates A_{ij}

Why use a low rank model?

- reduce storage; speed transmission
- understand (visualize, cluster)
- remove noise
- infer missing data
- simplify data processing

Outline

Principal components analysis

PCA:

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} (A_{ij} - x_i y_j)^2$$

with variables $X \in \mathbf{R}^{m \times k}$, $Y \in \mathbf{R}^{k \times n}$

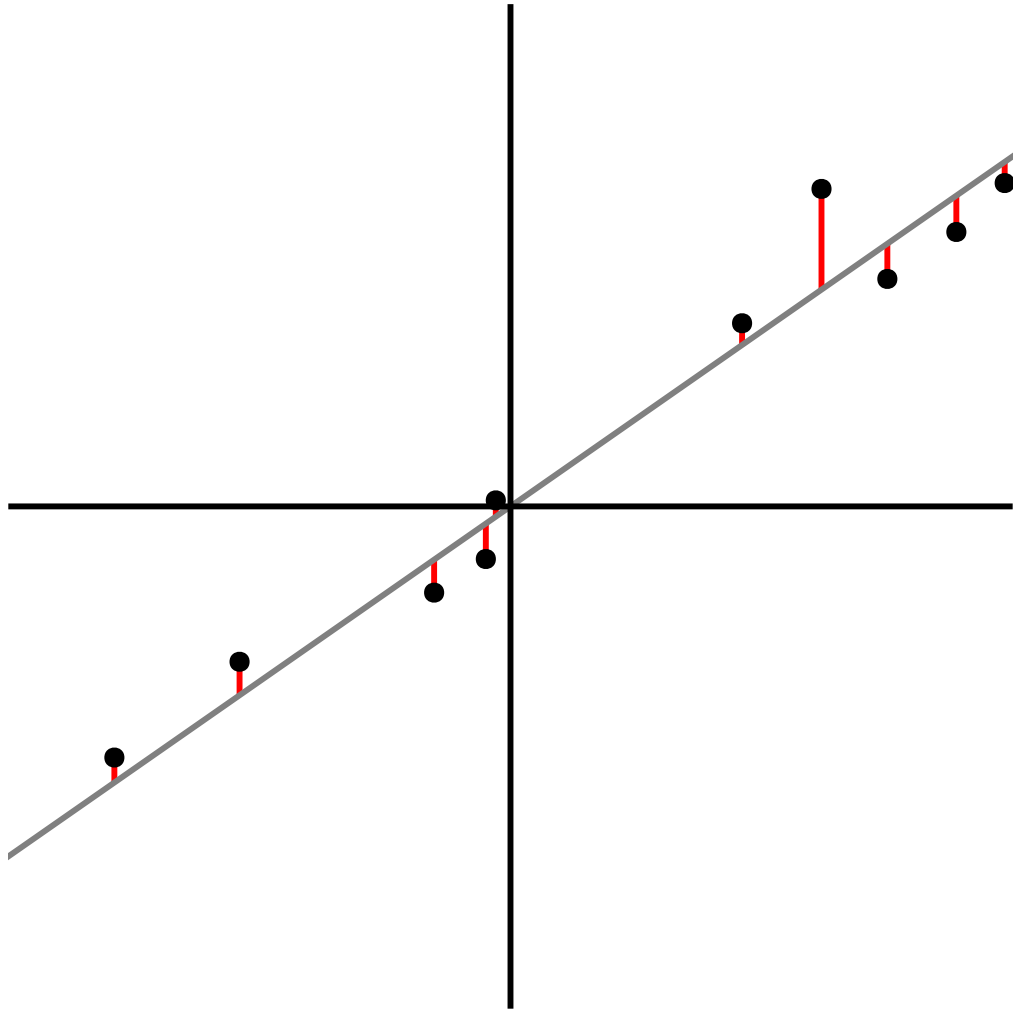
- old roots [Pearson 1901, Hotelling 1933]
- least squares low rank fitting

PCA finds best covariates

regression:

$$\text{minimize} \quad \|A - XY\|_F^2,$$

$n = 2$, $k = 1$, fix $X = A_{:,1:k}$ (first k columns of A), variable Y

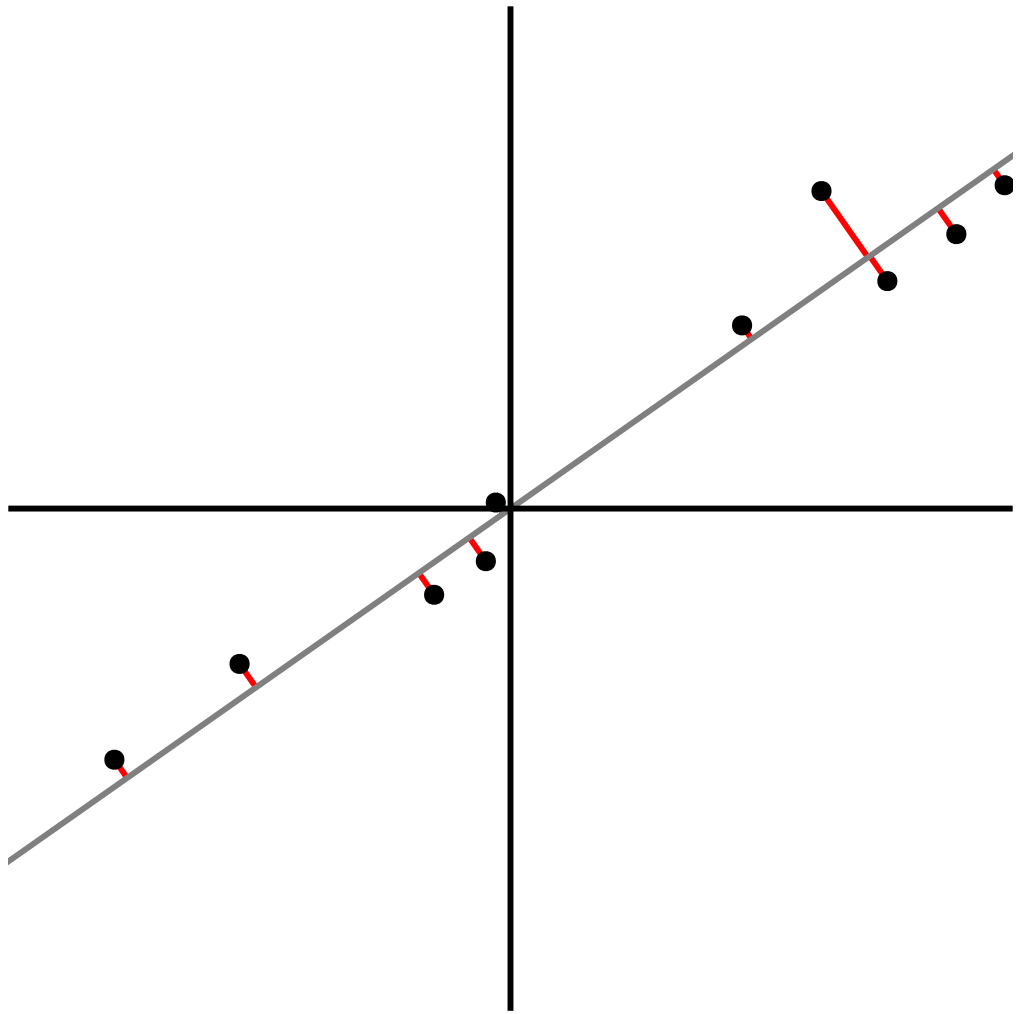


PCA finds best covariates

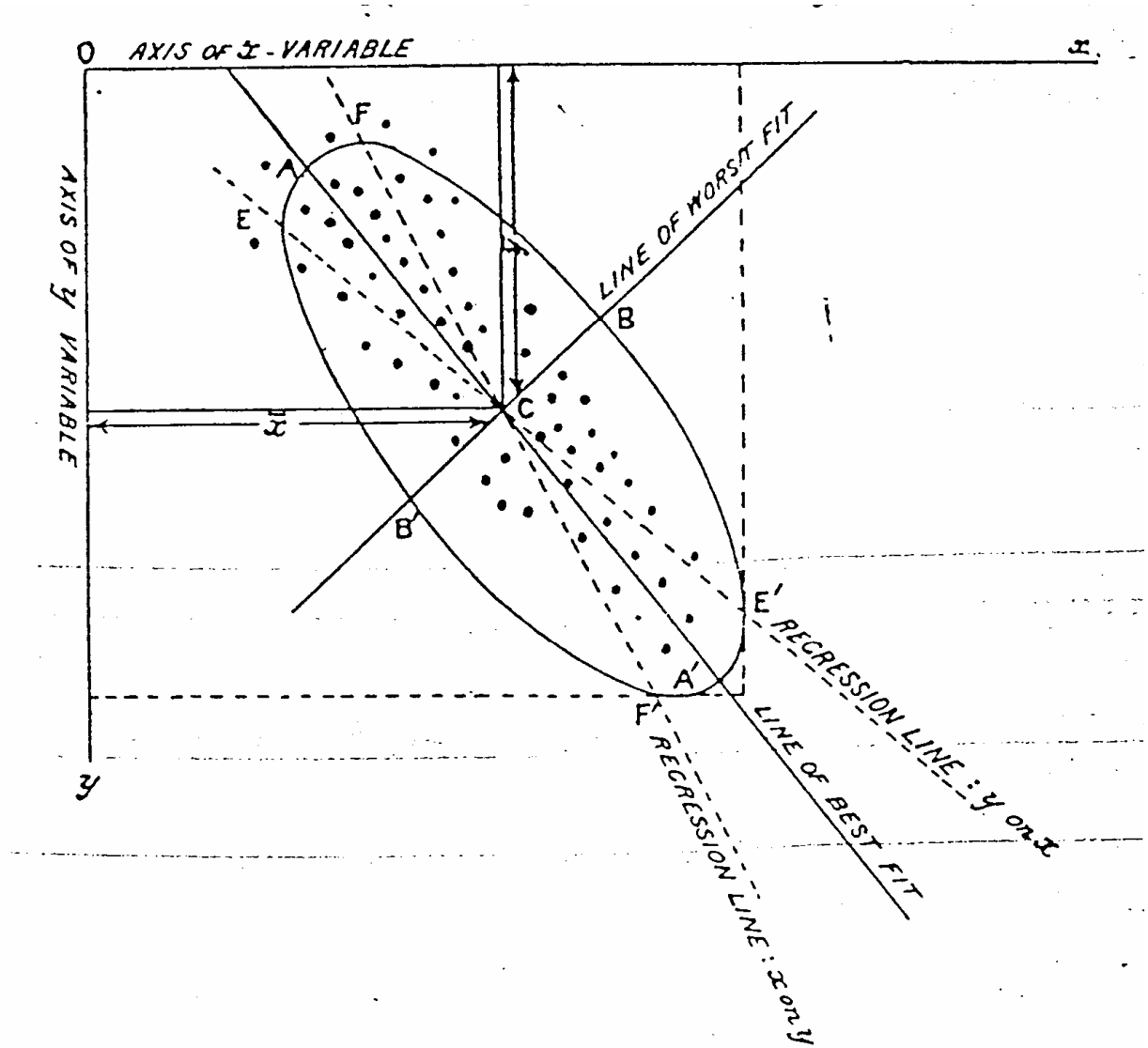
PCA:

$$\text{minimize} \quad \|A - XY\|_F^2,$$

$n = 2$, $k = 1$, variables X and Y



On lines and planes of best fit



[Pearson 1901]

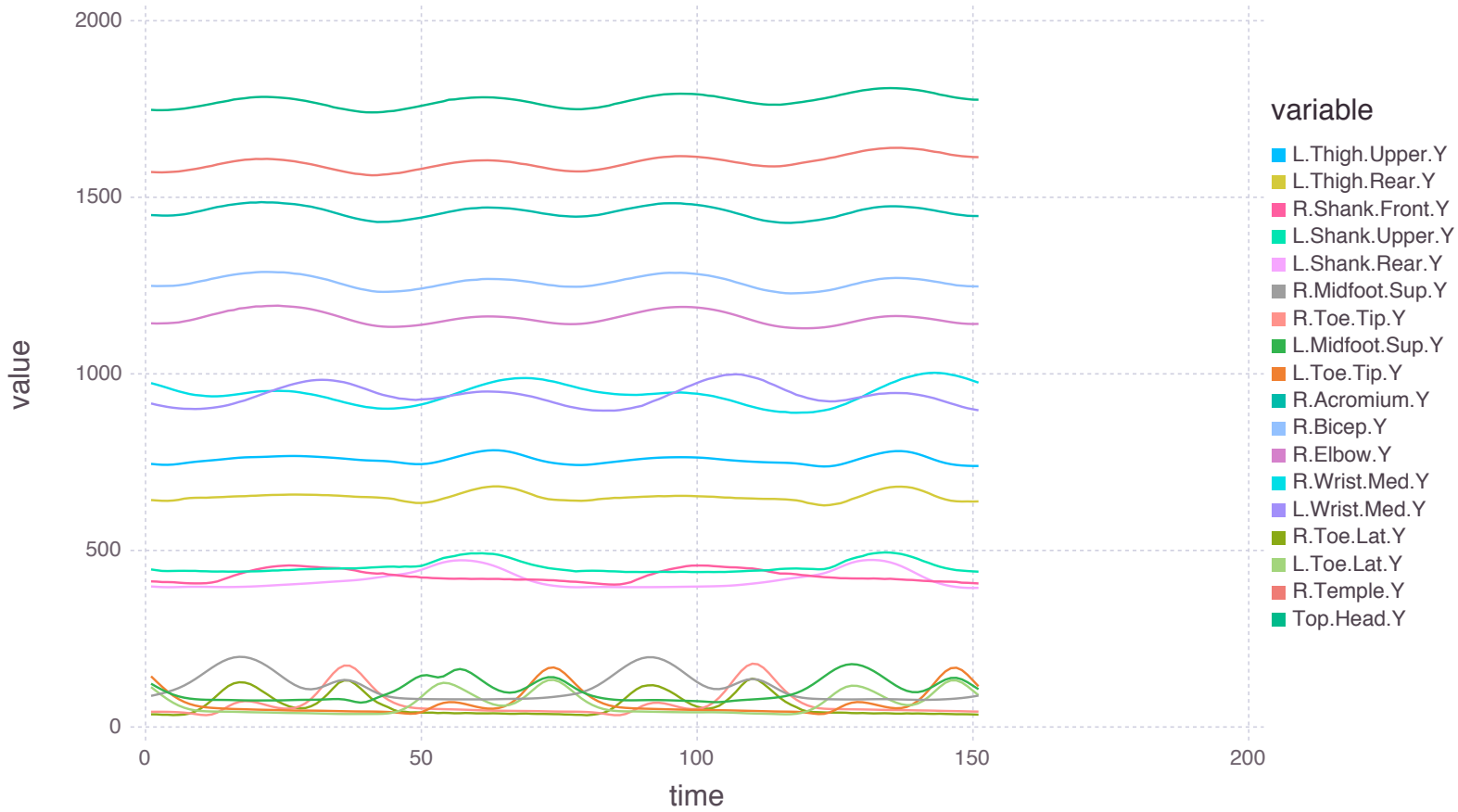
Low rank models for gait analysis

time	forehead (x)	forehead (y)	$\cdots$	right toe (y)	right toe (z)
t_1	1.4	2.7	$\cdots$	-0.5	-0.1
t_2	2.7	3.5	$\cdots$	1.3	0.9
t_3	3.3	-.9	$\cdots$	4.2	1.8
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

- rows of Y are principal stances
- rows of X decompose stance into combination of principal stances

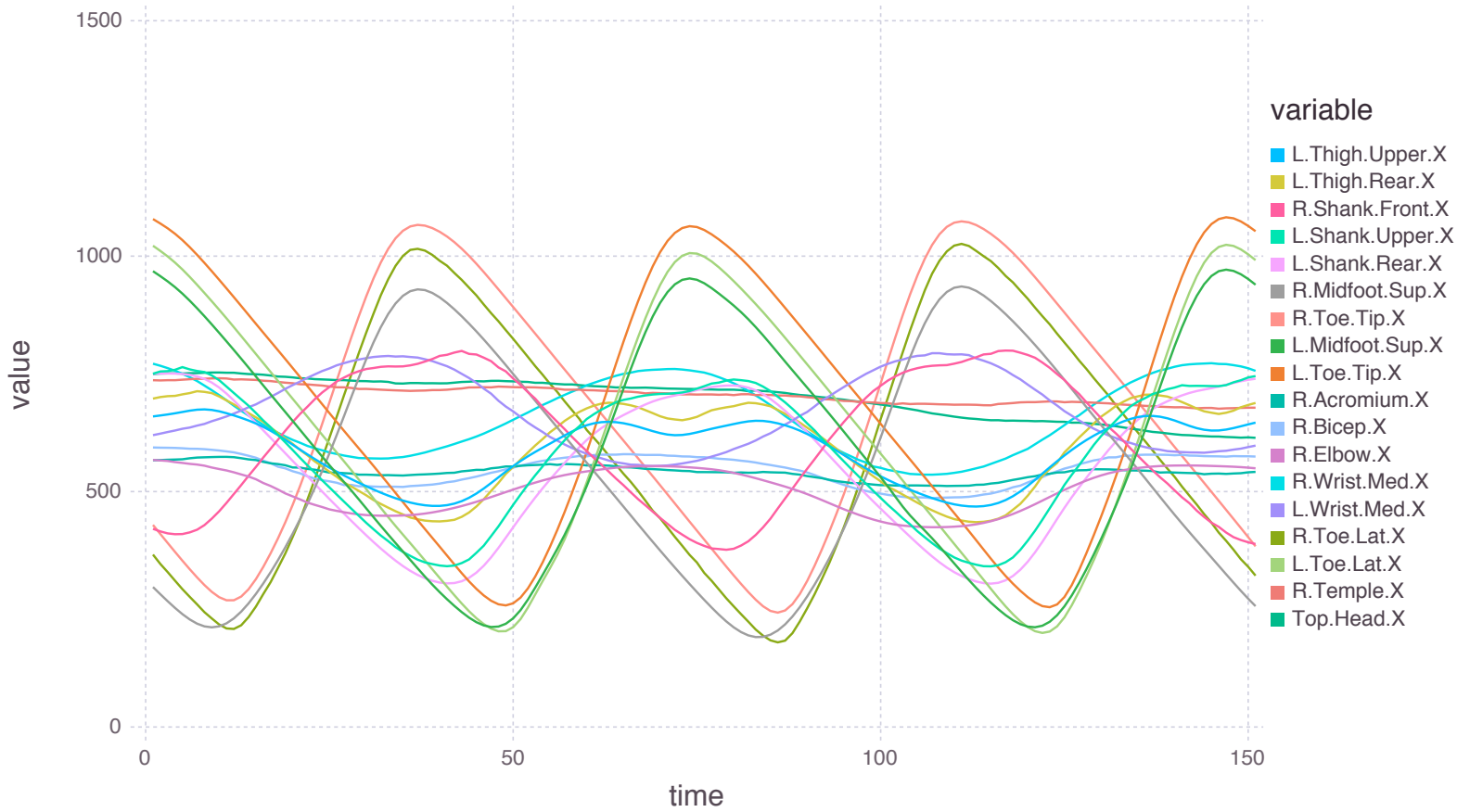
Interpreting principal components

columns of A (features) (height of point over time)



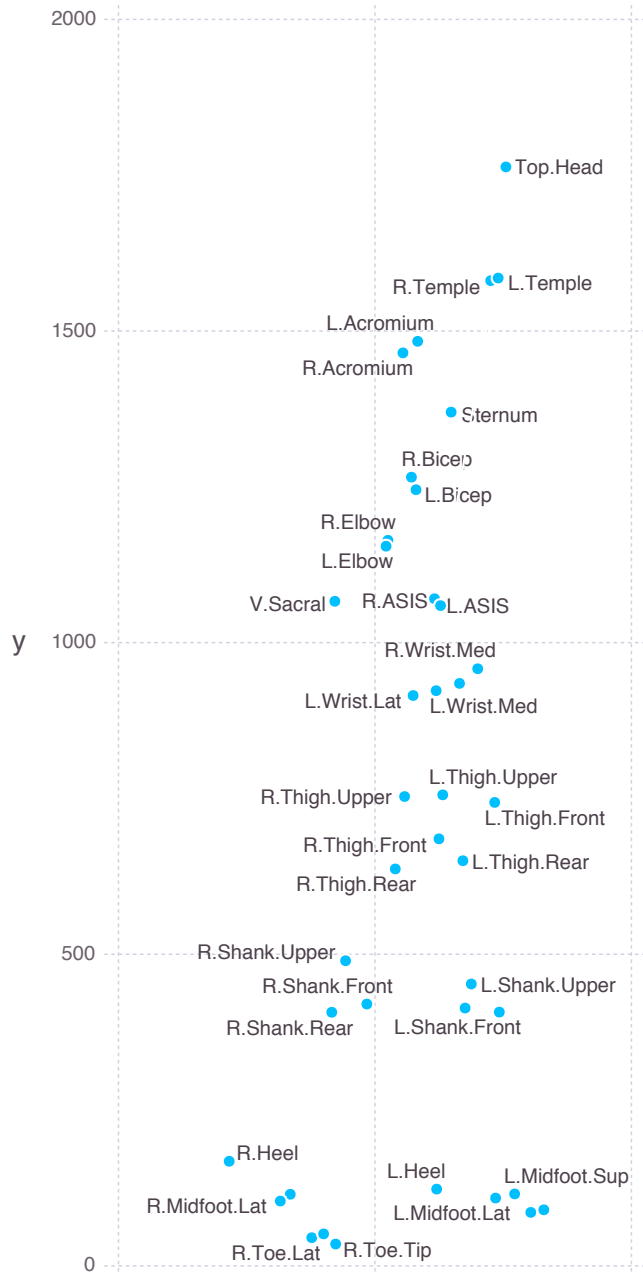
Interpreting principal components

columns of A (features) (depth of point over time)



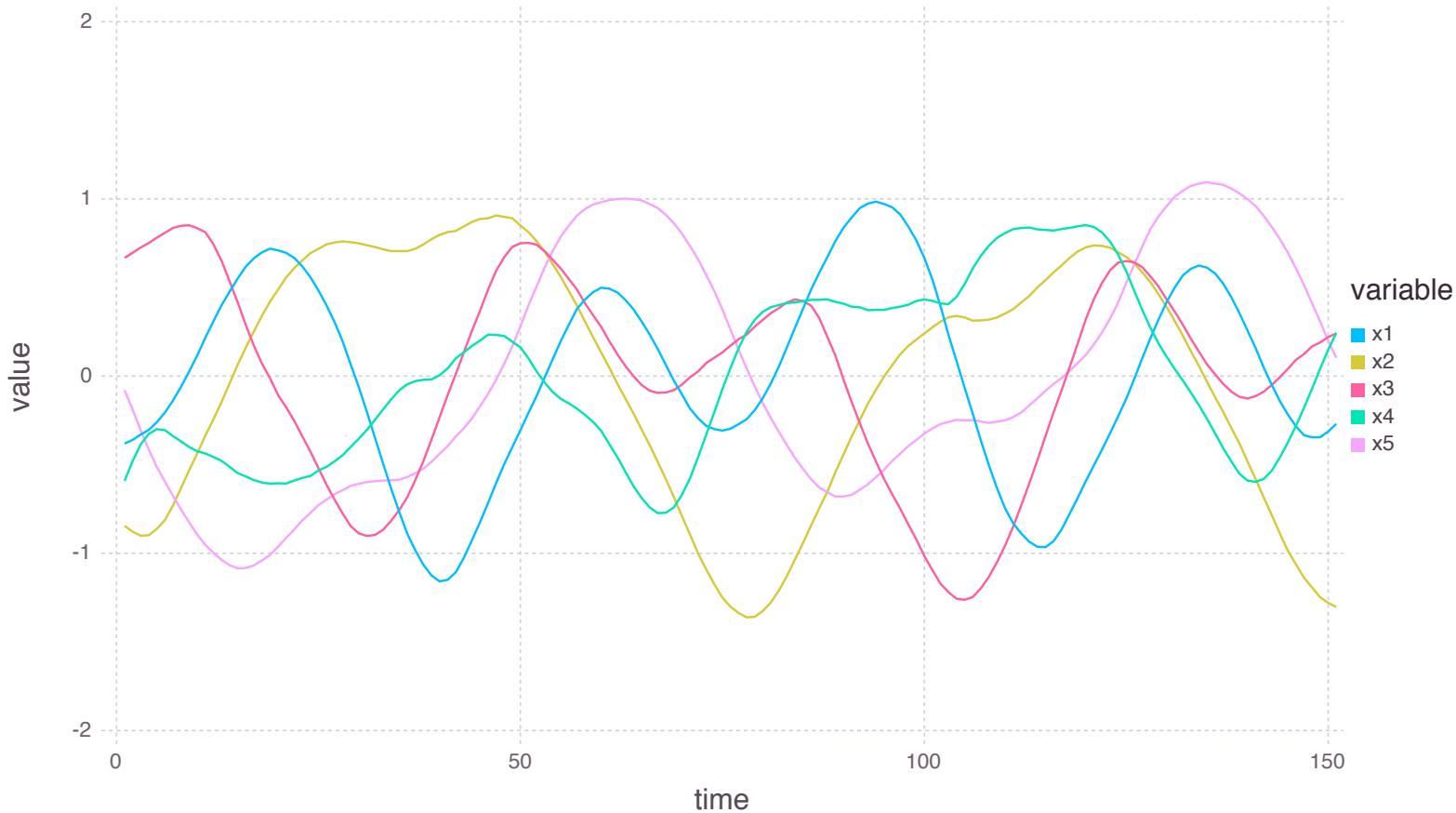
Interpreting principal components

row of Y
(archetypical example)
(principal stance)



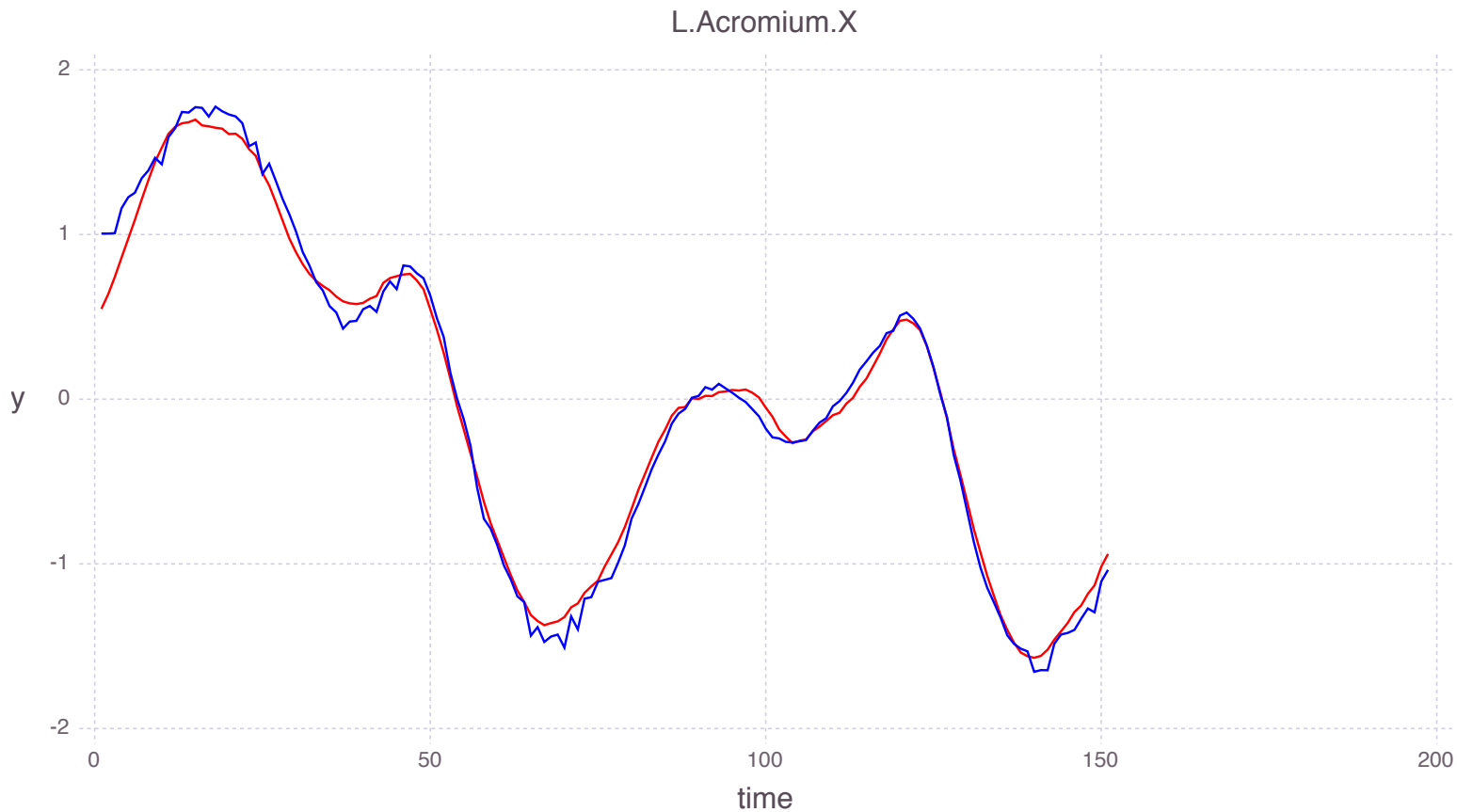
Interpreting principal components

columns of X (archetypical features) (principal timeseries)



Interpreting principal components

column of XY (red) (predicted feature)
column of A (blue) (observed feature)



Principal components analysis (PCA)

Principal components analysis (PCA): Given $A \in \mathbf{R}^{m \times n}$, solve

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} (A_{ij} - x_i y_j)^2$$

with $X \in \mathbf{R}^{m \times k}$, $Y \in \mathbf{R}^{k \times n}$

how should we solve this problem?

Principal components analysis (PCA)

Principal components analysis (PCA): Given $A \in \mathbf{R}^{m \times n}$, solve

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n (A_{ij} - x_i y_j)^2$$

with $X \in \mathbf{R}^{m \times k}$, $Y \in \mathbf{R}^{k \times n}$

how should we solve this problem?

- idea 1: use the SVD
- idea 2: alternating minimization over X and Y

PCA: solution via the SVD

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} (A_{ij} - x_i y_j)^2$$

with $X \in \mathbf{R}^{m \times k}$, $Y \in \mathbf{R}^{k \times n}$

Eckart-Young-Mirsky theorem: if

$$A = U \Sigma V^T = \sum_{i=1}^{\mathrm{Rank}(A)} \sigma_i u_i v_i^T$$

is the SVD of A, then

$$X = U_r, \quad Y = \Sigma_k V_k^T$$

is a solution to PCA, where

$$\Sigma_r = \mathbf{diag}(\sigma_1, \ldots, \sigma_k), \qquad U_r = [u_1 \cdots u_k], \qquad V_r = [v_1 \cdots v_k].$$

PCA: solution via the SVD

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n (A_{ij} - x_i y_j)^2$$

with $X \in \mathbf{R}^{m\times k}$, $Y \in \mathbf{R}^{k\times n}$

Eckart-Young-Mirsky theorem: if

$$A = U\Sigma V^T = \sum_{i=1}^{\mathrm{Rank}(A)} \sigma_i u_i v_i^T$$

is the SVD of A, then

$$X = U_r, \quad Y = \Sigma_k V_k^T$$

is a solution to PCA, where

$$\Sigma_r = \mathbf{diag}(\sigma_1, \ldots, \sigma_k), \qquad U_r = [u_1 \cdots u_k], \qquad V_r = [v_1 \cdots v_k].$$

with this X and Y,

$$\|A - XY\|_F^2 = \|U\Sigma V^T - U_k\Sigma_k V_k^T\|_F^2 = \sum_{i=k+1}^{\mathrm{Rank}(A)} \sigma_i^2$$

The Frobenius norm

the *Frobenius norm*

$$\|A\|_F = \sqrt{\sum_{i=1^m}^{n} \sum_{j=1} A_{ij}^2}$$

some useful identities:

- $\|A\|_F = \|\text{vec}(A)\|$
- $\|A\|_F = \|A^T\|_F$
- $\|A\|_F^2 = \mathbf{tr}(A^T A)$
- if U is orthogonal (*i.e.*, $U^T U = I$), then $\|UA\|_F = \|A\|_F$
  proof:

  $$\|UA\|_F^2 = \mathbf{tr}((UA)^T UA) = \mathbf{tr}(A^T U^T UA) = \mathbf{tr}(A^T A) = \|A\|_F^2$$

Proof of Eckart-Young-Mirsky theorem I

proof step 1: reduce to diagonal.
if $A = U\Sigma V^T$ is the full SVD, then

$$U^T U = UU^T = I \text{ and } V^T V = VV^T = I,$$

so

$$\begin{aligned} \|A - XY\|_F^2 &= \|U\Sigma V^T - XY\|_F^2 \\ &= \|U^T U\Sigma V^T V - U^T XYV\|_F^2 \\ &= \|\Sigma - U^T XYV\|_F^2 \\ &= \|\Sigma - Z\|_F^2 \end{aligned}$$

where $Z = U^T XYV$ is a rank k matrix.
we want to show

$$\sum_{i=k+1}^{\mathrm{Rank}(A)} \sigma_i \leq \|\Sigma - Z\|_F^2$$

for any rank k matrix Z.

Proof of Eckart-Young-Mirsky theorem II

proof step 2: eigenvalue interlacing.
let's use *Weyl's theorem for eigenvalues*:
for any matrices $A,B \in \mathbf{R}^{m\times n}$,

$$\sigma_{i+j-1}(A+B) \leq \sigma_i(A) + \sigma_j(B), \quad 1 \leq i,j \leq n.$$

set $A = \Sigma - Z$, $B = Z$, $j = k+1$ to get

$$\begin{aligned} \sigma_{i+k}(\Sigma) &\leq \sigma_i(\Sigma - Z) + \sigma_{k+1}(Z), \quad 1 \leq i \leq n-k \\ \sigma_{i+k} &\leq \sigma_i(\Sigma - Z), \quad 1 \leq i \leq n-k, \end{aligned}$$

using $\mathrm{Rank}(Z) \leq k$. square and sum from $i = 1$ to $\mathrm{Rank}(A) - k$:

$$\|\Sigma - \Sigma_k\|_F^2 = \sum_{i=k+1}^{\mathrm{Rank}(A)} \sigma_i^2 \leq \sum_{i=1}^{\mathrm{Rank}(A)-k} \sigma_i^2(\Sigma - Z) \leq \|\Sigma - Z\|_F^2.$$

PCA: solution via AM

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n (A_{ij} - x_i y_j)^2$$

Alternating Minimization (AM): fix Y^0. for $t = 1, \ldots,$

- $X^t = \operatorname{argmin}_X \|A - XY^{t-1}\|_F^2$
- $Y^t = \operatorname{argmin}_Y \|A - X^t Y\|_F^2$

properties:

- objective decreases at each iteration
- objective bounded below, so the procedure converges
- (it is true but we won't prove that) with probability 1 over choices of Y^0, AM converges to an optimal solution

PCA: AM subproblem is separable

how would you solve the AM subproblem

$$Y^t = \operatorname*{argmin}_Y \|A - X^t Y\|_F^2 = \operatorname*{argmin}_Y \sum_{j=1}^{n} \|a_j - X^t y_j\|^2$$

where $A = [a_1 \cdots a_d]$, $Y = [y_1 \cdots y_d]$?

PCA: AM subproblem is separable

how would you solve the AM subproblem

$$Y^t = \operatorname*{argmin}_Y \|A - X^t Y\|_F^2 = \operatorname*{argmin}_Y \sum_{j=1}^{n} \|a_j - X^t y_j\|^2$$

where $A = [a_1 \cdots a_d]$, $Y = [y_1 \cdots y_d]$?

- problem separates over columns of Y:

$$y_j^t = \operatorname*{argmin}_w \|a_j - X^t y\|^2$$

- for each column of Y, it's just a least squares problem!
- $y_j = ((X^t)^T X^t)^{-1} (X^t)^T a_j$

PCA: solution via AM

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n (A_{ij} - x_i y_j)^2$$

Alternating Minimization (AM): fix Y^0. for $t = 1, \ldots,$

- for $i = 1, \ldots, m$,

$$x_i^t = A_{i:}(Y^{t-1})^T (Y^{t-1}(Y^{t-1})^T)^{-1}$$

- for $j = 1, \ldots, n$,

$$y_j^t = ((X^t)^T X^t)^{-1} (X^t)^T a_j$$

PCA: solution via AM

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n (A_{ij} - x_i y_j)^2$$

computational tricks:

- cache gram matrix $G = (X^t)^T X^t$
- parallelize over j

Alternating Minimization (AM): fix Y^0. for $t = 1, \ldots,$

- cache factorization of $G = Y^{t-1}(Y^{t-1})^T$
- in parallel, for $i = 1, \ldots, m$,

$$x_i^t = A_{i:}(Y^{t-1})^T(Y^{t-1}(Y^{t-1})^T)^{-1}$$

- cache factorization of $G = (X^t)^T X^t$
- in parallel, for $j = 1, \ldots, n$,

$$y_j^t = ((X^t)^T X^t)^{-1}(X^t)^T a_j$$

PCA: solution via AM

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} (A_{ij} - x_i y_j)^2$$

complexity?

Alternating Minimization (AM): fix Y^0. for $t = 1, \ldots,$

- cache factorization of $G = Y^{t-1}(Y^{t-1})^T$

PCA: solution via AM

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n (A_{ij} - x_i y_j)^2$$

complexity?

Alternating Minimization (AM): fix Y^0. for $t = 1, \ldots,$

- cache factorization of $G = Y^{t-1}(Y^{t-1})^T$ $(\mathcal{O}(nk^2 + k^3))$
- in parallel, for $i = 1, \ldots, m$,

$$x_i^t = (Y^{t-1}(Y^{t-1})^T)^{-1} Y^{t-1} A_{i:}^T$$

PCA: solution via AM

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n (A_{ij} - x_i y_j)^2$$

complexity?

Alternating Minimization (AM): fix Y^0. for $t = 1, \ldots,$

- cache factorization of $G = Y^{t-1}(Y^{t-1})^T$ $(\mathcal{O}(nk^2 + k^3))$
- in parallel, for $i = 1, \ldots, m$,

$$x_i^t = (Y^{t-1}(Y^{t-1})^T)^{-1} Y^{t-1} A_{i:}^T$$

  $(\mathcal{O}(nk + k^2))$
- cache factorization of $G = (X^t)^T X^t$

PCA: solution via AM

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n (A_{ij} - x_i y_j)^2$$

complexity?

Alternating Minimization (AM): fix Y^0. for $t = 1, \ldots,$

- cache factorization of $G = Y^{t-1}(Y^{t-1})^T$ $(\mathcal{O}(nk^2 + k^3))$
- in parallel, for $i = 1, \ldots, m$,

$$x_i^t = (Y^{t-1}(Y^{t-1})^T)^{-1} Y^{t-1} A_{i:}^T$$

  $(\mathcal{O}(nk + k^2))$
- cache factorization of $G = (X^t)^T X^t$ $(\mathcal{O}(mr^2 + k^3))$
- in parallel, for $j = 1, \ldots, n$,

$$y_j^t = ((X^t)^T X^t)^{-1} (X^t)^T a_j$$

PCA: solution via AM

$$\text{minimize} \quad \|A - XY\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n (A_{ij} - x_i y_j)^2$$

complexity?

Alternating Minimization (AM): fix Y^0. for $t = 1, \ldots,$

- cache factorization of $G = Y^{t-1}(Y^{t-1})^T$ $(\mathcal{O}(nk^2 + k^3))$
- in parallel, for $i = 1, \ldots, m$,

$$x_i^t = (Y^{t-1}(Y^{t-1})^T)^{-1} Y^{t-1} A_{i:}^T$$

  $(\mathcal{O}(nk + k^2))$
- cache factorization of $G = (X^t)^T X^t$ $(\mathcal{O}(mr^2 + k^3))$
- in parallel, for $j = 1, \ldots, n$,

$$y_j^t = ((X^t)^T X^t)^{-1} (X^t)^T a_j$$

  $(\mathcal{O}(mk + k^2))$

Outline

Matrix completion

observe A_{ij} only for $(i,j) \in \Omega \subset \{1,\ldots,m\} \times \{1,\ldots,n\}$

$$\text{minimize} \quad \sum_{(i,j)\in\Omega}(A_{ij} - x_i y_j)^2 + \gamma\|X\|_F^2 + \gamma\|Y\|_F^2$$

two regimes:

- **some entries missing:** don't waste data; "borrow strength" from entries that are *not* missing
- **most entries missing:** matrix completion still works!

Matrix completion

observe A_{ij} only for $(i,j) \in \Omega \subset \{1,\ldots,m\} \times \{1,\ldots,n\}$

$$\text{minimize} \quad \sum_{(i,j)\in\Omega}(A_{ij} - x_i y_j)^2 + \gamma\|X\|_F^2 + \gamma\|Y\|_F^2$$

two regimes:

- **some entries missing:** don't waste data; "borrow strength" from entries that are *not* missing
- **most entries missing:** matrix completion still works!

Theorem ([Keshavan 2010])

If A has rank $k' \leq k$ and $|\Omega| = O(nk' \log n)$ (and A is incoherent and Ω is chosen UAR), then matrix completion exactly recovers the matrix A with high probability.

Maximum likelihood low rank estimation

noisy data? maximize (log) likelihood of observations by minimizing:

- gaussian noise: $L(u, a) = (u - a)^2$
- laplacian (heavy-tailed) noise: $L(u, a) = |u - a|$
- gaussian + laplacian noise: $L(u, a) = \mathbf{huber}(u - a)$
- poisson (count) noise: $L(u, a) = \exp(()\, u) - au + a \log a - a$
- bernoulli (coin toss) noise: $L(u, a) = \log(1 + \exp(() - au))$

Maximum likelihood low rank estimation works

Theorem (Template)

If a number of samples $|\Omega| = O(n \log(n))$ *drawn UAR from matrix entries is observed according to a probabilistic model with parameter* Z*, the solution to (appropriately) regularized maximum likelihood estimation is close to the true* Z *with high probability.*

examples (not exhaustive!):

- additive gaussian noise [Candes Plan 2009]
- additive subgaussian noise [Keshavan Montanari Oh 2009]
- gaussian + laplacian noise [Xu Caramanis Sanghavi 2012]
- 0-1 (Bernoulli) observations [Davenport et al. 2012]
- entrywise exponential family distribution [Gunasekar Ravikumar Ghosh 2014]
- multinomial logit [Kallus U 2015]

Huber PCA

$$\text{minimize} \quad \sum_{(i,j)\in\Omega} \mathbf{huber}(x_i y_j - A_{ij}) + \sum_{i=1}^{m} \|x_i\|_2^2 + \sum_{j=1}^{n} \|y_j\|_2^2$$

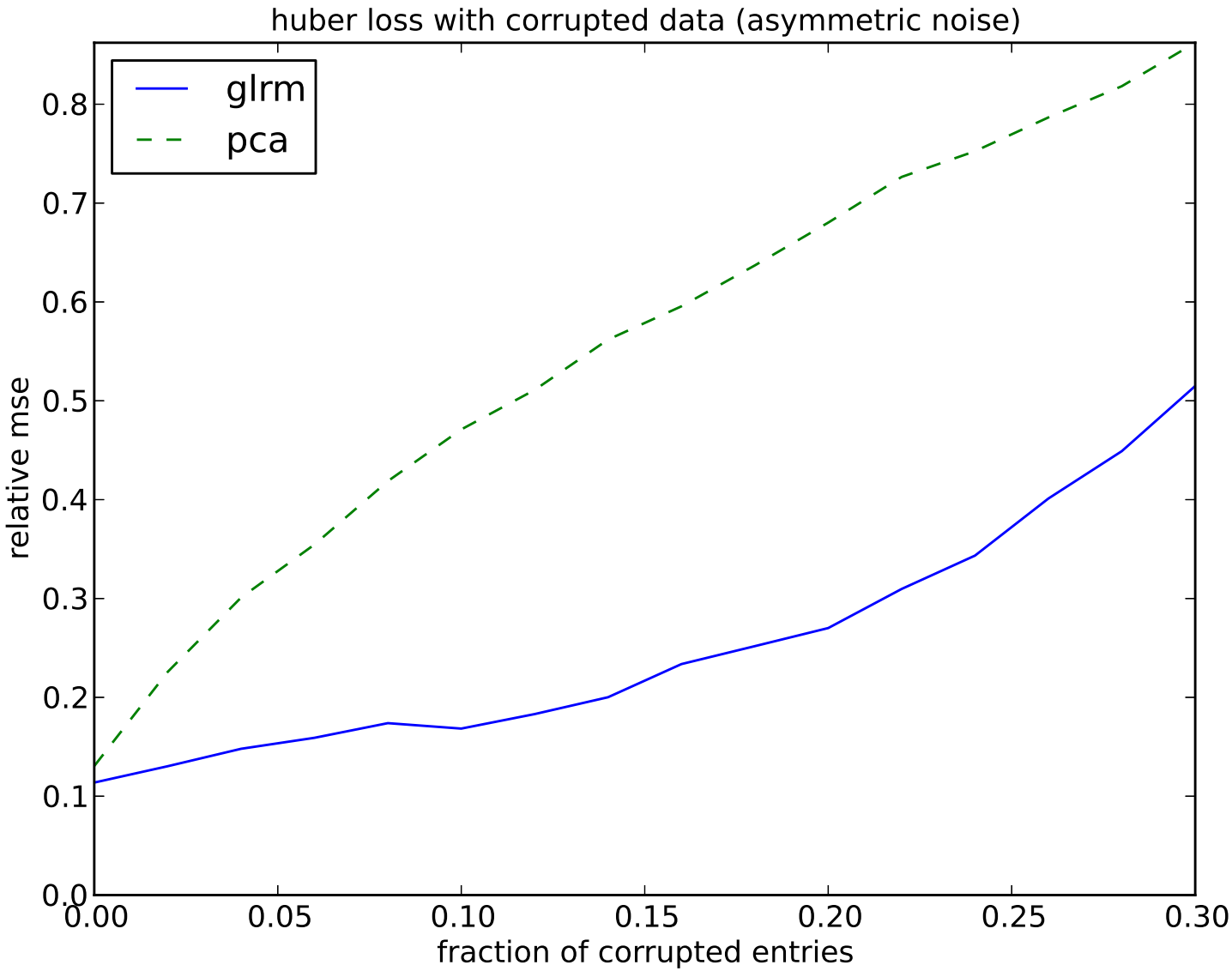
where we define the Huber function

$$\mathbf{huber}(z) = \begin{cases} \frac{1}{2}z^2 & |z| \le 1 \\ |z| - \frac{1}{2} & |z| > 1 \end{cases}.$$

Huber decomposes error into a small (Gaussian) part and large (robust) part

$$\mathbf{huber}(z) = \inf\{|s| + \frac{1}{2}n^2 : z = n + s\}.$$

Huber PCA


huber loss with corrupted data (asymmetric noise)
glrm
pca
relative mse
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.00
0.05
0.10
0.15
0.20
0.25
0.30
fraction of corrupted entries

Generalized low rank model

$$\text{minimize} \quad \sum_{(i,j)\in\Omega} L_j(x_i y_j, A_{ij}) + \sum_{i=1}^m r_i(x_i) + \sum_{j=1}^n \tilde{r}_j(y_j)$$

- loss functions L_j for each column
  - *e.g.*, different losses for reals, booleans, categoricals, ordinals, ...
- regularizers $r : \mathbf{R}^{1\times k} \to \mathbf{R}$, $\tilde{r} : \mathbf{R}^k \to \mathbf{R}$
- observe only $(i,j) \in \Omega$ (other entries are missing)

Outline

Low rank models for finance

factor model of sector returns

ticker	t_1	t_2	$\cdots$
AAPL	.05	-.21	$\cdots$
KRX	.07	-.18	$\cdots$
GOOG	-.11	.24	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\ddots$

- rows of Y are sector return time series
- rows of X are sector exposures

Low rank models for power

electricity usage profiles

household	t_1	t_2	$\cdots$	
1	1.4	0.5	0.1	$\cdots$
2	2.7	1.3	0.9	$\cdots$
3	3.3	4.2	1.8	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

- rows of Y are electricity usage profiles
- rows of X decompose household power usage into distinct usage profiles

Regularizers

$$\text{minimize} \quad \sum_{(i,j)\in\Omega} L_j(A_{ij}, x_i y_j) + \sum_{i=1}^m r_i(x_i) + \sum_{j=1}^n \tilde{r}_j(y_j)$$

choose regularizers r, $\tilde{r}$ to impose structure:

structure	$r(x)$	$\tilde{r}(y)$
small	$\|x\|_2^2$	$\|y\|_2^2$
sparse	$\|x\|_1$	$\|y\|_1$
nonnegative	$\mathbf{1}_+(x)$	$\mathbf{1}_+(y)$
clustered	$\mathbf{1}_1(x)$	0

Nonnegative matrix factorization

$$\text{minimize} \quad \sum_{(i,j)\in\Omega} (A_{ij} - x_i y_j)^2 + \sum_{i=1}^{m} \mathbf{1}_+(x_i) + \sum_{j=1}^{n} \mathbf{1}_+(y_j)$$

- regularizer is indicator of nonnegative orthant

$$\mathbf{1}_+(x) = \begin{cases} 0 & x \geq 0 \\ \infty & \text{otherwise} \end{cases}$$

Nonnegative matrix factorization

$$\text{minimize} \quad \sum_{(i,j)\in\Omega} (A_{ij} - x_i y_j)^2 + \sum_{i=1}^{m} \mathbf{1}_+(x_i) + \sum_{j=1}^{n} \mathbf{1}_+(y_j)$$

- regularizer is indicator of nonnegative orthant

$$\mathbf{1}_+(x) = \begin{cases} 0 & x \geq 0 \\ \infty & \text{otherwise} \end{cases}$$

subproblems are nonnegative least squares problems:

$$x_i^{t+1} = \operatorname*{argmin}_{x>0} \sum_{j:(i,j)\in\Omega} (A_{ij} - x y_j^t)^2 \tag{1}$$

$$y_j^{t+1} = \operatorname*{argmin}_{w>0} \sum_{i:(i,j)\in\Omega} (A_{ij} - x_i^{t+1} y)^2 \tag{2}$$

Clustering

a *clustering* algorithm groups data points into clusters

examples:

- *medical diagnosis.* cluster patients with similar medical histories
- *topic model.* cluster documents with similar patterns of word usage
- *market segmentation.* cluster customers with similar purchase patterns

Quadratic clustering

$$\text{minimize} \quad \sum_{(i,j)\in\Omega}(A_{ij}x_i y_j)^2 + \sum_{i=1}^{m} \mathbf{1}_1(x_i)$$

- $\mathbf{1}_1$ is the indicator function of a selection, *i.e.*,

$$\mathbf{1}_1(x) = \begin{cases} 0 & x = e_l \text{ for some } l \in \{1, \ldots, k\} \\ \infty & \text{otherwise} \end{cases}$$

where e_l is the lth unit vector

Quadratic clustering

$$\text{minimize} \quad \sum_{(i,j)\in\Omega}(A_{ij}x_i y_j)^2 + \sum_{i=1}^{m} \mathbf{1}_1(x_i)$$

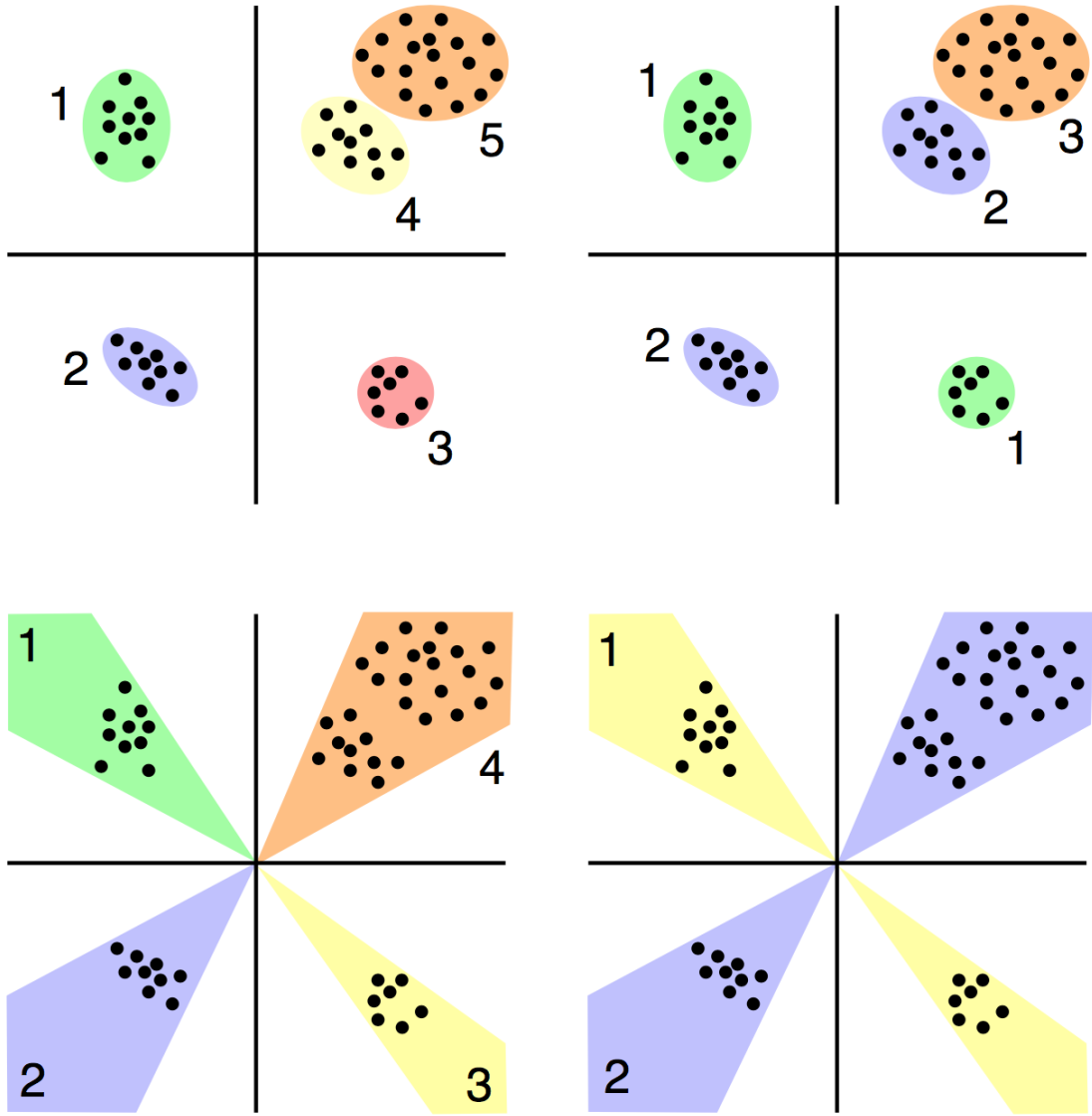
- $\mathbf{1}_1$ is the indicator function of a selection, *i.e.*,

$$\mathbf{1}_1(x) = \begin{cases} 0 & x = e_l \text{ for some } l \in \{1, \ldots, k\} \\ \infty & \text{otherwise} \end{cases}$$

where e_l is the lth unit vector

alternating minimization reproduces k-means
(but allows missing data)

What's a cluster?



[Tropp 2004]

Modifying k-means

different regularizers:

- clusters
- rays
- lines
- planes
- cones

Regularizers

$$\text{minimize} \quad \sum_{(i,j)\in\Omega} L_j(A_{ij}, x_i y_j) + \sum_{i=1}^m r_i(x_i) + \sum_{j=1}^n \tilde{r}_j(y_j)$$

choose regularizers r, $\tilde{r}$ to impose structure:

structure	$r(x)$	$\tilde{r}(y)$
small	$\|x\|_2^2$	$\|y\|_2^2$
sparse	$\|x\|_1$	$\|y\|_1$
nonnegative	$\mathbf{1}_+(x)$	$\mathbf{1}_+(y)$
clustered	$\mathbf{1}_1(x)$	0

Outline

Abstract loss

define *abstract feature space* $\mathcal{F}_j$

e.g., $A_{ij} \in \mathcal{F}_j$ can be

- boolean
- ordinal
- categorical
- ranking

just need a loss function $L_j : \mathbf{R} \times \mathcal{F}_j \to \mathbf{R}$

Boolean losses

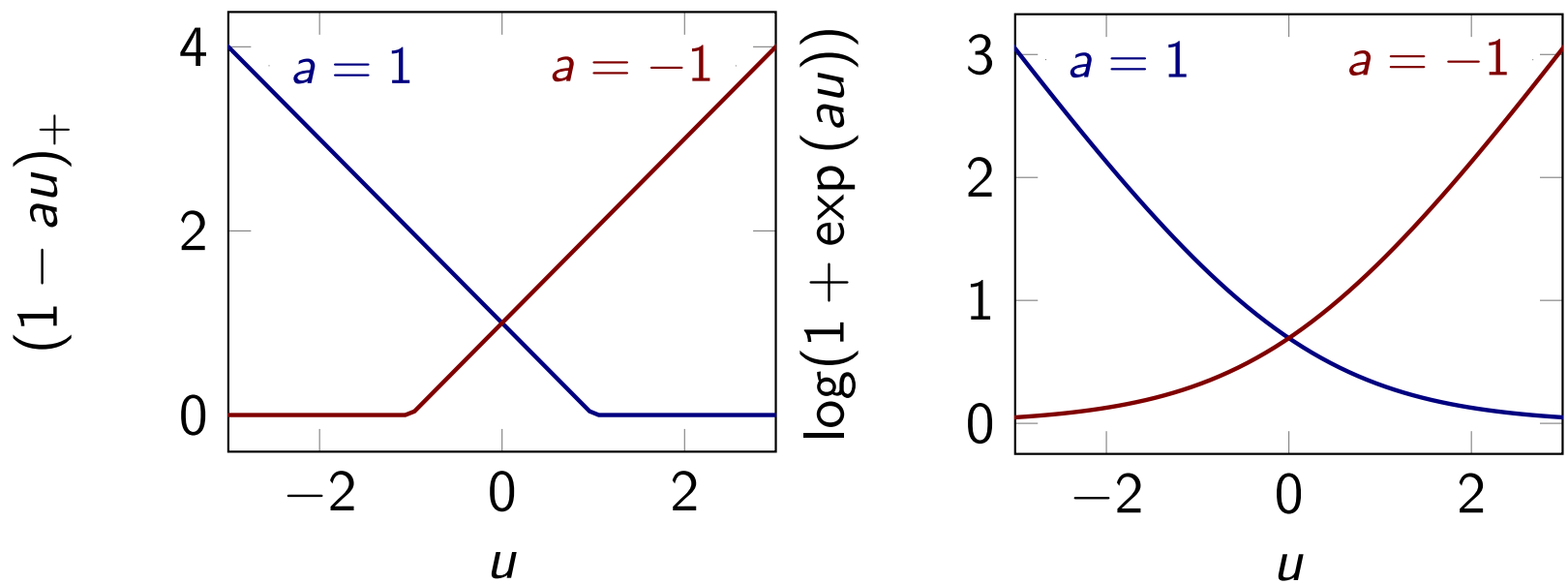
Boolean PCA: $\mathcal{F}_j = \{-1, 1\}$

- hinge loss

$$L(u, a) = (1 - au)_+$$

- logistic loss

$$L(u, a) = \log(1 + \exp(-au))$$



Ordinal loss

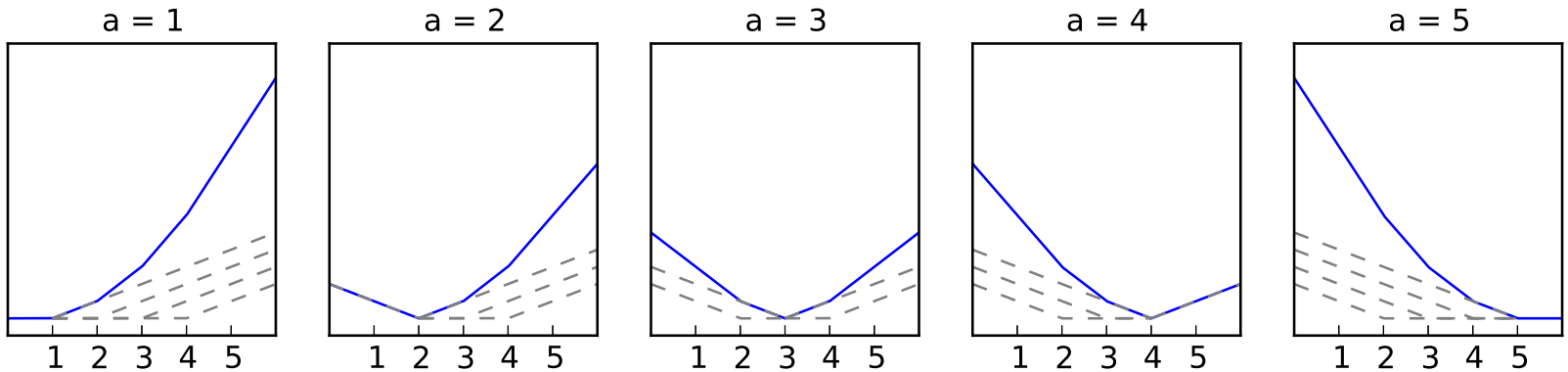
Ordinal PCA: $\mathcal{F}_j = \{1, \ldots, d\}$

- quadratic loss

$$L(u, a) = (u - a)^2$$

- ordinal hinge loss

$$L(u, a) = \sum_{a'=1}^{a-1} (1 - u + a')_+ + \sum_{a'=a+1}^{d} (1 + u - a')_+$$



Multi-dimensional loss

- approximate using *vectors* $x_i Y_j \in \mathbf{R}^{1 \times d_j}$ instead of numbers
- need $L_j : \mathbf{R}^{1 \times d_j} \times \mathcal{F}_j \to \mathbf{R}$

$$\text{minimize} \quad \sum_{(i,j) \in \Omega} L_j(x_i Y_j, A_{ij}) + \sum_{i=1}^m r_i(x_i) + \sum_{j=1}^n \tilde{r}_j(Y_j)$$

- useful for approximating *categorical* variables
  - columns of Y_j represent different labels of categorical variable
- gives more flexible/accurate models for *ordinal* variables
- number of columns of Y_j is *embedding dimension* of $\mathcal{F}_j$

Losses

$$\text{minimize} \quad \sum_{(i,j)\in\Omega} L_j(x_i y_j, A_{ij}) + \sum_{i=1}^{m} r_i(x_i) + \sum_{j=1}^{n} \tilde{r}_j(y_j)$$

choose loss $L(u, a)$ adapted to data type:

data type	**loss**	$L(u, a)$
real	quadratic	$(u-a)^2$
real	absolute value	$\lvert u-a \rvert$
real	huber	$\mathbf{huber}(u-a)$
boolean	hinge	$(1-ua)_+$
boolean	logistic	$\log(1+\exp(-au))$
integer	poisson	$\exp(u) - au + a\log a - a$
ordinal	ordinal hinge	$\sum_{a'=1}^{a-1}(1-u+a')_+ + \sum_{a'=a+1}^{d}(1+u-a')_+$
categorical	one-vs-all	$(1-u_a)_+ + \sum_{a'\neq a}(1+u_{a'})_+$
categorical	multinomial logit	$\frac{\exp(u_a)}{\sum_{a'=1}^{d}\exp(u_{a'})}$

Scaling losses

Analogue of standardization for GLRMs:

$$\begin{aligned} \mu_j &= \operatorname*{argmin}_{\mu} \sum_{i:(i,j)\in\Omega} L_j(\mu, A_{ij}) \\ \sigma_j^2 &= \frac{1}{n_j - 1} \sum_{i:(i,j)\in\Omega} L_j(\mu_j, A_{ij}) \end{aligned}$$

- μ_j generalizes column mean
- σ_j^2 generalizes column variance

To fit a standardized GLRM, solve

$$\text{minimize} \quad \textstyle\sum_{(i,j)\in\Omega} L_j(A_{ij}, x_i y_j + \mu_j)/\sigma_j^2 + \sum_{i=1}^m r_i(x_i) + \sum_{j=1}^n \tilde{r}_j(y_j)$$

Outline

Impute missing data

impute most likely true data $\hat{A}_{ij}$

$$\hat{A}_{ij} = \operatorname*{argmin}_{a} L_j(x_i y_j, a)$$

- implicit constraint: $\hat{A}_{ij} \in \mathcal{F}_j$
- MLE interpretation: if $L_j(x_i y_j, a) = -\log P(a \mid x_i y_j)$, then $\hat{A}_{ij}$ is *most probable* $a \in \mathcal{F}_j$ given $x_i y_j$.

Impute missing data

impute most likely true data $\hat{A}_{ij}$

$$\hat{A}_{ij} = \operatorname*{argmin}_a L_j(x_i y_j, a)$$

- implicit constraint: $\hat{A}_{ij} \in \mathcal{F}_j$
- MLE interpretation: if $L_j(x_i y_j, a) = -\log P(a \mid x_i y_j)$, then $\hat{A}_{ij}$ is *most probable* $a \in \mathcal{F}_j$ given $x_i y_j$.

examples:

- when L_j is quadratic, ℓ_1, or Huber loss,

Impute missing data

impute most likely true data $\hat{A}_{ij}$

$$\hat{A}_{ij} = \operatorname*{argmin}_a L_j(x_i y_j, a)$$

- implicit constraint: $\hat{A}_{ij} \in \mathcal{F}_j$
- MLE interpretation: if $L_j(x_i y_j, a) = -\log P(a \mid x_i y_j)$, then $\hat{A}_{ij}$ is *most probable* $a \in \mathcal{F}_j$ given $x_i y_j$.

examples:

- when L_j is quadratic, ℓ_1, or Huber loss, then $\hat{A}_{ij} = x_i y_j$

Impute missing data

impute most likely true data $\hat{A}_{ij}$

$$\hat{A}_{ij} = \operatorname*{argmin}_a L_j(x_i y_j, a)$$

- implicit constraint: $\hat{A}_{ij} \in \mathcal{F}_j$
- MLE interpretation: if $L_j(x_i y_j, a) = -\log P(a \mid x_i y_j)$, then $\hat{A}_{ij}$ is *most probable* $a \in \mathcal{F}_j$ given $x_i y_j$.

examples:

- when L_j is quadratic, ℓ_1, or Huber loss, then $\hat{A}_{ij} = x_i y_j$
- if $\mathcal{F} \neq \mathbf{R}$, $\operatorname{argmin}_a L_j(x_i y_j, a) \neq x_i y_j$
  - *e.g.*, for hinge loss $L(u, a) = (1 - ua)_+$,

Impute missing data

impute most likely true data $\hat{A}_{ij}$

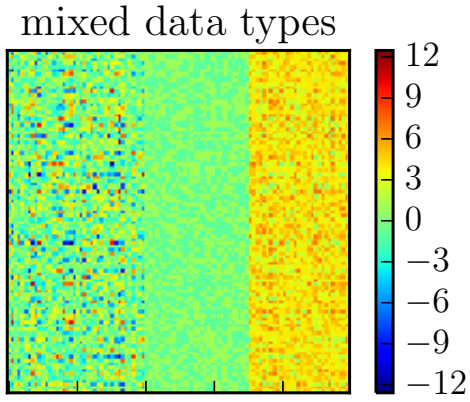
$$\hat{A}_{ij} = \operatorname*{argmin}_a L_j(x_i y_j, a)$$

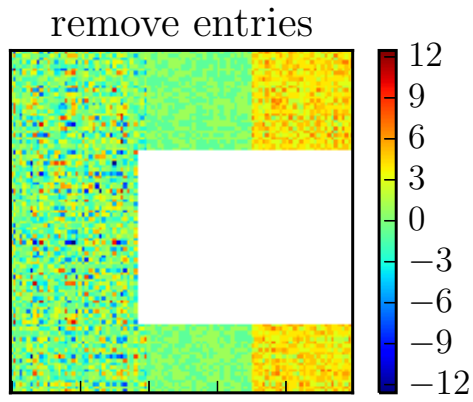
- implicit constraint: $\hat{A}_{ij} \in \mathcal{F}_j$
- MLE interpretation: if $L_j(x_i y_j, a) = -\log P(a \mid x_i y_j)$, then $\hat{A}_{ij}$ is *most probable* $a \in \mathcal{F}_j$ given $x_i y_j$.

examples:

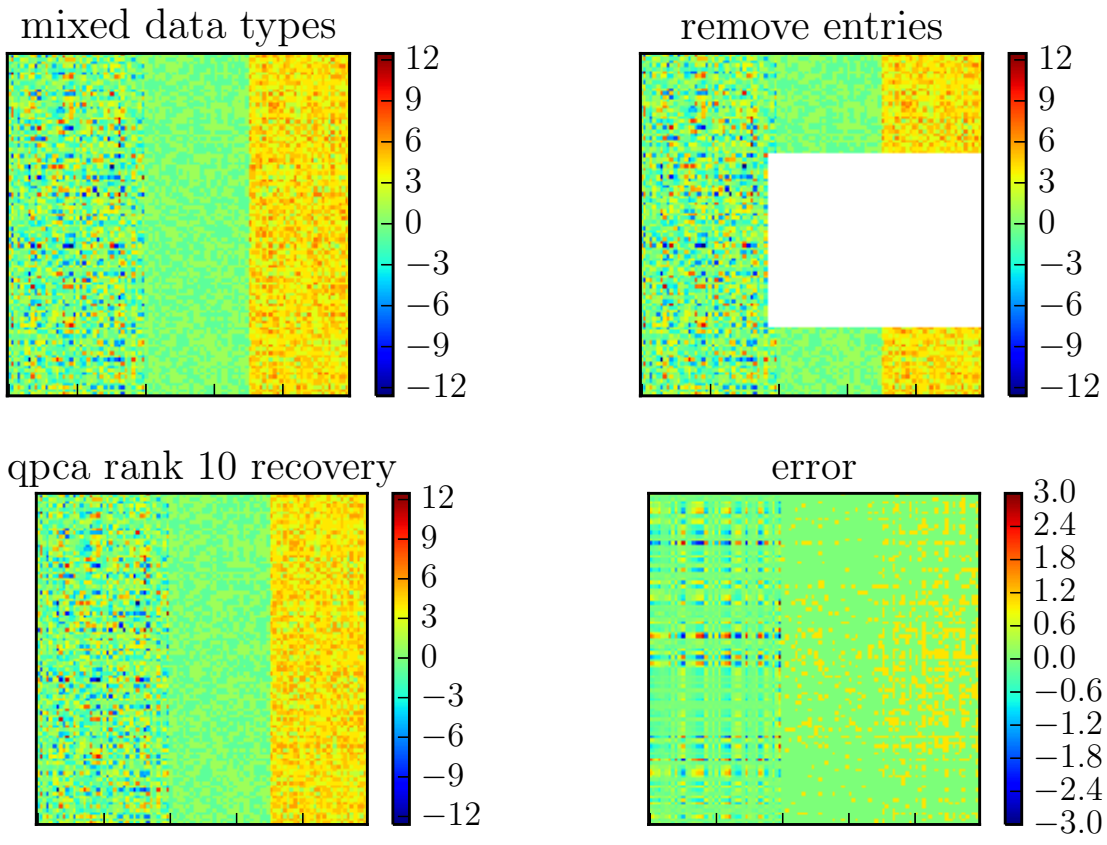
- when L_j is quadratic, ℓ_1, or Huber loss, then $\hat{A}_{ij} = x_i y_j$
- if $\mathcal{F} \neq \mathbf{R}$, $\operatorname{argmin}_a L_j(x_i y_j, a) \neq x_i y_j$
  - *e.g.*, for hinge loss $L(u, a) = (1 - ua)_+$, $\hat{A}_{ij} = \mathbf{sign}(x_i y_j)$

Impute heterogeneous data

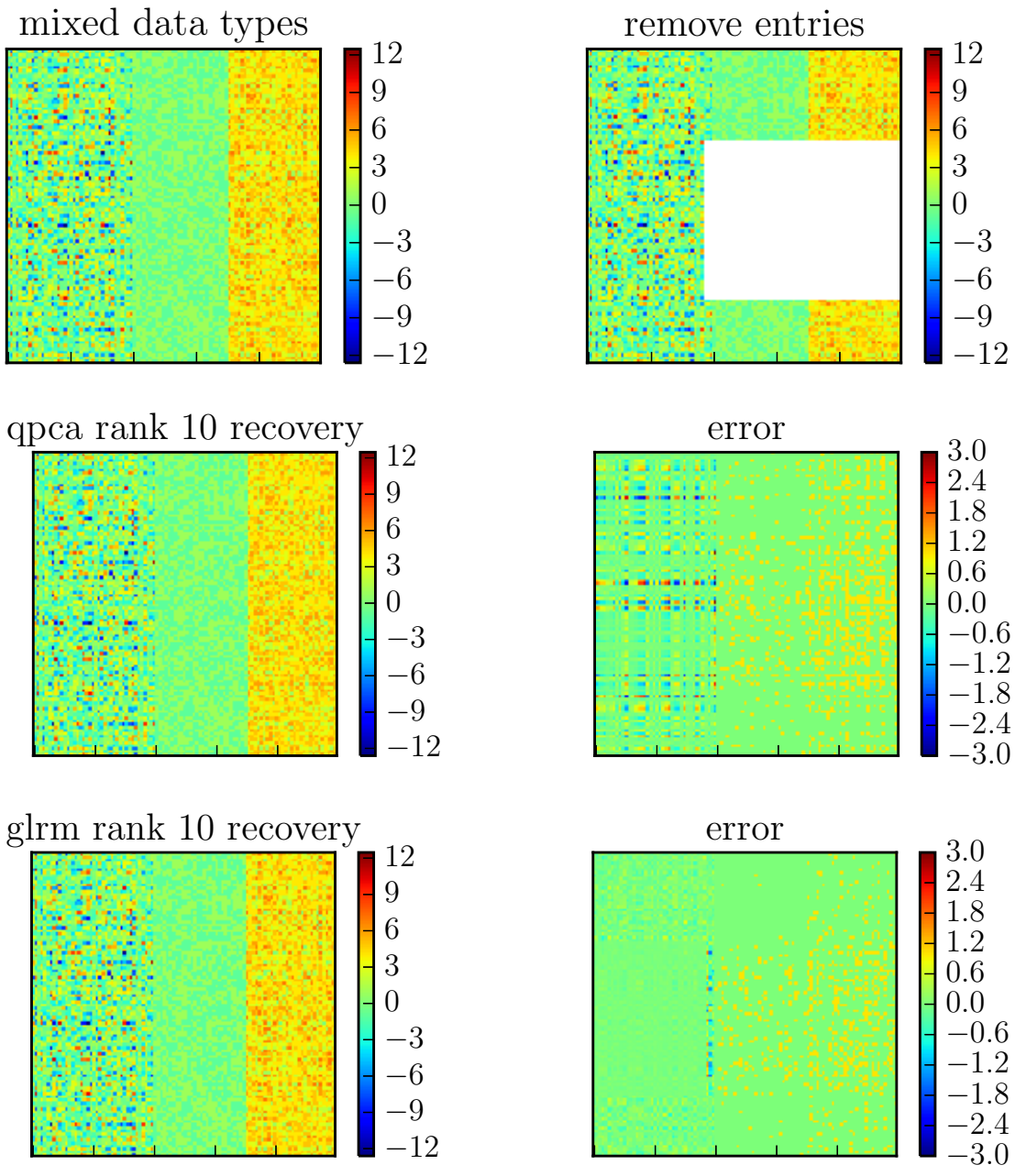




Impute heterogeneous data



Impute heterogeneous data



Validate model

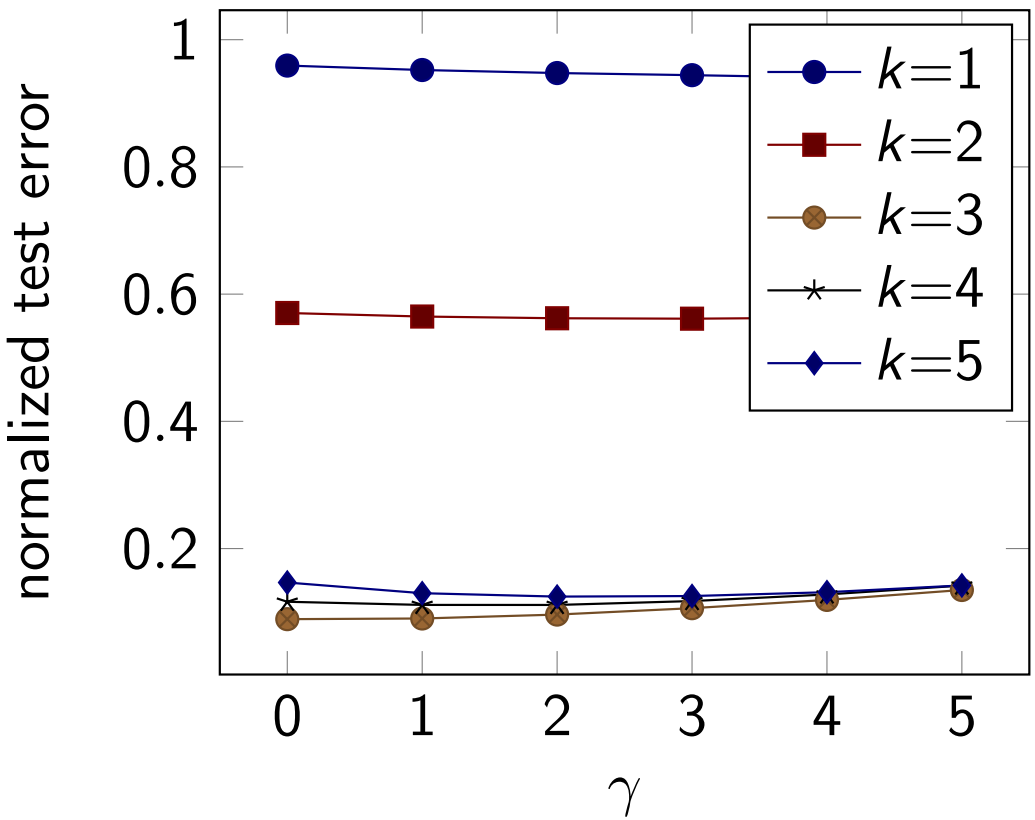
$$\text{minimize} \quad \sum_{(i,j)\in\Omega} L_{ij}(A_{ij}, x_i y_j) + \sum_{i=1}^{m} \gamma r_i(x_i) + \sum_{j=1}^{n} \gamma \tilde{r}_j(y_j)$$

How to choose model parameters (k, γ)?

Validate model

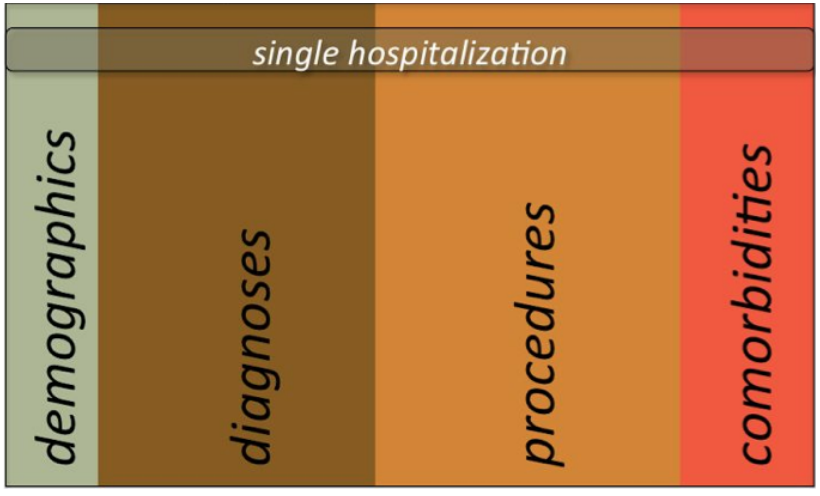
$$\text{minimize} \quad \sum_{(i,j)\in\Omega} L_{ij}(A_{ij}, x_i y_j) + \sum_{i=1}^{m} \gamma r_i(x_i) + \sum_{j=1}^{n} \gamma \tilde{r}_j(y_j)$$

How to choose model parameters (k, γ)?
Leave out 10% of entries, and use model to predict them

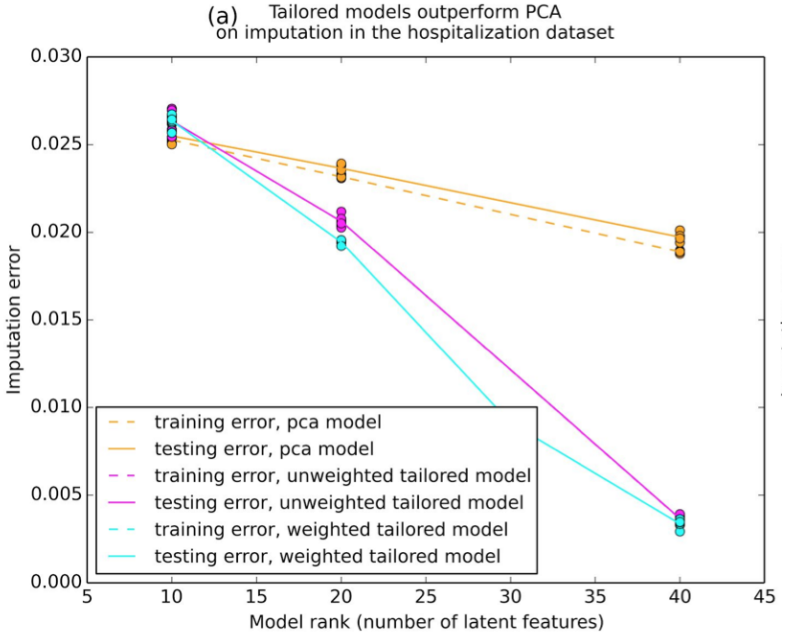


Hospitalizations are low rank

hospitalization data set



GLRM outperforms PCA



[Schuler Liu, Wan, Callahan, U, Stark, Shah 2016]

American community survey

2013 ACS:

- 3M respondents, 87 economic/demographic survey questions
  - income
  - cost of utilities (water, gas, electric)
  - weeks worked per year
  - hours worked per week
  - home ownership
  - looking for work
  - use foodstamps
  - education level
  - state of residence
  - ...
- 1/3 of responses missing

Using a GLRM for exploratory data analysis

$$\begin{bmatrix} | & & | \\ y_1 & \cdots & y_n \\ | & & | \end{bmatrix}$$

age	gender	state	$\cdots$
29	F	CT	$\cdots$
57	?	NY	$\cdots$
?	M	CA	$\cdots$
41	F	NV	$\cdots$
$\vdots$	$\vdots$	$\vdots$	

$$\approx \begin{bmatrix} — x_1 — \\ \vdots \\ — x_m — \end{bmatrix}$$

- cluster respondents?

Using a GLRM for exploratory data analysis

$$\begin{bmatrix} | & & | \\ y_1 & \cdots & y_n \\ | & & | \end{bmatrix}$$

age	gender	state	$\cdots$
29	F	CT	$\cdots$
57	?	NY	$\cdots$
?	M	CA	$\cdots$
41	F	NV	$\cdots$
$\vdots$	$\vdots$	$\vdots$	

$$\approx \begin{bmatrix} — x_1 — \\ \vdots \\ — x_m — \end{bmatrix}$$

- cluster respondents? **cluster rows of** X

Using a GLRM for exploratory data analysis

$$\begin{bmatrix} | & & | \\ y_1 & \cdots & y_n \\ | & & | \end{bmatrix}$$

age	gender	state	$\cdots$
29	F	CT	$\cdots$
57	?	NY	$\cdots$
?	M	CA	$\cdots$
41	F	NV	$\cdots$
$\vdots$	$\vdots$	$\vdots$	

$$\approx \begin{bmatrix} — x_1 — \\ \vdots \\ — x_m — \end{bmatrix}$$

- cluster respondents? **cluster rows of** X
- demographic profiles?

Using a GLRM for exploratory data analysis

$$\begin{bmatrix} | & & | \\ y_1 & \cdots & y_n \\ | & & | \end{bmatrix}$$

age	gender	state	$\cdots$
29	F	CT	$\cdots$
57	?	NY	$\cdots$
?	M	CA	$\cdots$
41	F	NV	$\cdots$
$\vdots$	$\vdots$	$\vdots$	

$$\approx \begin{bmatrix} — x_1 — \\ \vdots \\ — x_m — \end{bmatrix}$$

- cluster respondents? **cluster rows of** X
- demographic profiles? **rows of** Y

Using a GLRM for exploratory data analysis

$$\begin{bmatrix} | & & | \\ y_1 & \cdots & y_n \\ | & & | \end{bmatrix}$$

age	gender	state	$\cdots$
29	F	CT	$\cdots$
57	?	NY	$\cdots$
?	M	CA	$\cdots$
41	F	NV	$\cdots$
$\vdots$	$\vdots$	$\vdots$	

$$\approx \begin{bmatrix} — x_1 — \\ \vdots \\ — x_m — \end{bmatrix}$$

- cluster respondents? **cluster rows of** X
- demographic profiles? **rows of** Y
- which features are similar?

Using a GLRM for exploratory data analysis

$$\begin{bmatrix} | & & | \\ y_1 & \cdots & y_n \\ | & & | \end{bmatrix}$$

age	gender	state	$\cdots$
29	F	CT	$\cdots$
57	?	NY	$\cdots$
?	M	CA	$\cdots$
41	F	NV	$\cdots$
$\vdots$	$\vdots$	$\vdots$	

$$\approx \begin{bmatrix} — x_1 — \\ \vdots \\ — x_m — \end{bmatrix}$$

- cluster respondents? **cluster rows of** X
- demographic profiles? **rows of** Y
- which features are similar? **cluster columns of** Y

Using a GLRM for exploratory data analysis

$$\begin{bmatrix} | & & | \\ y_1 & \cdots & y_n \\ | & & | \end{bmatrix}$$

age	gender	state	$\cdots$
29	F	CT	$\cdots$
57	?	NY	$\cdots$
?	M	CA	$\cdots$
41	F	NV	$\cdots$
$\vdots$	$\vdots$	$\vdots$	

$$\approx \begin{bmatrix} — x_1 — \\ \vdots \\ — x_m — \end{bmatrix}$$

- cluster respondents? **cluster rows of** X
- demographic profiles? **rows of** Y
- which features are similar? **cluster columns of** Y
- impute missing entries?

Using a GLRM for exploratory data analysis

$$\begin{bmatrix} | & & | \\ y_1 & \cdots & y_n \\ | & & | \end{bmatrix}$$

age	gender	state	$\cdots$
29	F	CT	$\cdots$
57	?	NY	$\cdots$
?	M	CA	$\cdots$
41	F	NV	$\cdots$
$\vdots$	$\vdots$	$\vdots$	

$$\approx \begin{bmatrix} — x_1 — \\ \vdots \\ — x_m — \end{bmatrix}$$

- cluster respondents? **cluster rows of** X
- demographic profiles? **rows of** Y
- which features are similar? **cluster columns of** Y
- impute missing entries? $\operatorname{argmin}_a L_j(x_i y_j, a)$

Fitting a GLRM to the ACS

- construct a rank 10 GLRM with loss functions respecting data types
  - huber for real values
  - hinge loss for booleans
  - ordinal hinge loss for ordinals
  - one-vs-all hinge loss for categoricals
- scale losses and regularizers
- fit the GLRM

American community survey

most similar features (in *demography space*):

- Alaska: Montana, North Dakota
- California: Illinois, cost of water
- Colorado: Oregon, Idaho
- Ohio: Indiana, Michigan
- Pennsylvania: Massachusetts, New Jersey
- Virginia: Maryland, Connecticut
- Hours worked: weeks worked, education

Low rank models for dimensionality reduction[1]

U.S. Wage & Hour Division (WHD) compliance actions:

company	zip	violations	···
Holiday Inn	14850	109	···
Moosewood Restaurant	14850	0	···
Cornell Orchards	14850	0	···
Lakeside Nursing Home	14850	53	···
⋮	⋮	⋮	

- 208,806 rows (cases) $\times$ 252 columns (violation info)
- 32,989 zip codes...

[1] labor law violation demo: `https://github.com/h2oai/h2o-3/blob/master/h2o-r/demos/rdemo.census.labor.violations.large.R`

Low rank models for dimensionality reduction

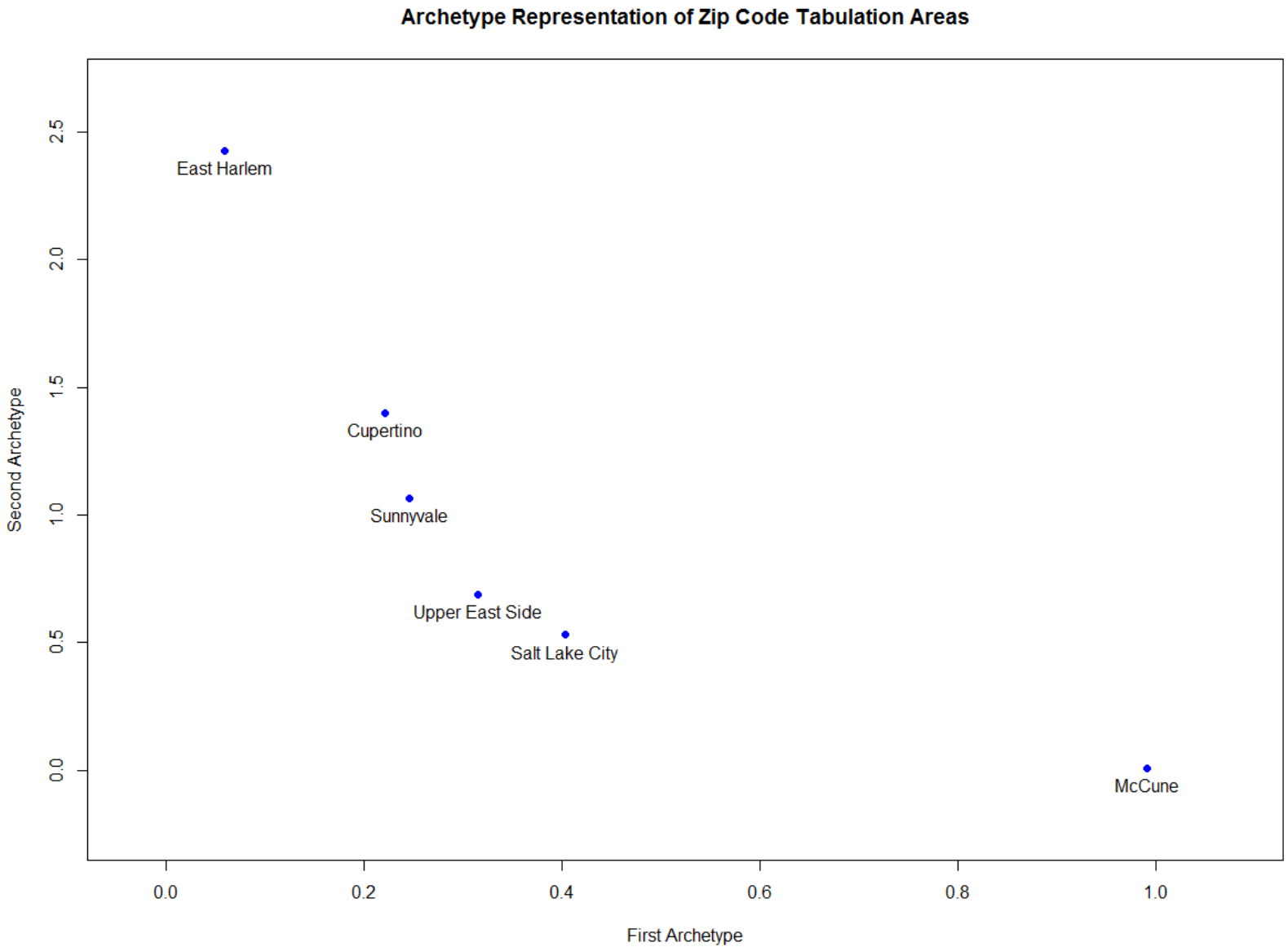
ACS demographic data:

zip	unemployment	mean income	$\cdots$
94305	12%	\$47,000	$\cdots$
06511	19%	\$32,000	$\cdots$
60647	23%	\$23,000	$\cdots$
94121	4%	\$178,000	$\cdots$
$\vdots$	$\vdots$	$\vdots$	

- 32,989 rows (zip codes) $\times$ 150 columns (demographic info)
- GLRM embeds zip codes into (low dimensional) *demography space*

Low rank models for dimensionality reduction

Zip code features:



Low rank models for dimensionality reduction

build 3 sets of features to predict violations:

- categorical: expand zip code to categorical variable
- concatenate: join tables on zip
- GLRM: replace zip code by low dimensional zip code features

fit a supervised (deep learning) model:

method	train error	test error	runtime
categorical	0.2091690	0.2173612	23.7600000
concatenate	0.2258872	0.2515906	4.4700000
GLRM	0.1790884	0.1933637	4.3600000

Missing data

examples:

- weather data: missing data due to sensor failures
- survey data: missing data due to non-response
- purchase/click/like data: missing data due to lack of purchase/click/like
- drug trial: missing data due to subjects leaving trial

How to cope with missing data?

strategy 1:

- drop rows or columns with missing data

How to cope with missing data?

strategy 1:

- drop rows or columns with missing data

how well would this work for

- weather data
- survey data
- purchase/click/like data
- drug trial

How to cope with missing data?

strategy 2:

- fill in missing entries with row or column mean

How to cope with missing data?

strategy 2:

- fill in missing entries with row or column mean

how well would this work for

- weather data
- survey data
- purchase/click/like data
- drug trial

How to cope with missing data?

strategy 3:

- use observed data to *predict* missing entries

How to cope with missing data?

strategy 3:

- use observed data to *predict* missing entries

how well would this work for

- weather data
- survey data
- purchase/click/like data
- drug trial

Correct biased sample

two types of people

- type A always fill out all questions
- type B leave question 3 blank half the time

question 1	question 2	question 3	question 4	$\cdots$
2.7	yes	4	yes	$\cdots$
2.7	yes	4	yes	$\cdots$
9.2	no	?	no	$\cdots$
2.7	yes	4	yes	$\cdots$
9.2	no	1	no	$\cdots$
9.2	no	?	no	$\cdots$
2.7	yes	4	yes	$\cdots$
9.2	no	1	no	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	

estimate population mean of question 3

Correct biased sample

two types of people

- type A always fill out all questions
- type B leave question 3 blank half the time

question 1	question 2	question 3	question 4	$\cdots$
2.7	yes	4	yes	$\cdots$
2.7	yes	4	yes	$\cdots$
9.2	no	?	no	$\cdots$
2.7	yes	4	yes	$\cdots$
9.2	no	1	no	$\cdots$
9.2	no	?	no	$\cdots$
2.7	yes	4	yes	$\cdots$
9.2	no	1	no	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	

estimate population mean of question 3

- excluding missing entries: 3
- imputing missing entries: 2.5

Impute censored data

market segmentation

customer	apples	oranges	pears	$\cdots$
1	yes	?	yes	$\cdots$
2	yes	yes	?	$\cdots$
3	?	?	yes	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	

- rows of Y are purchasing patterns for market segments
- rows of X classify customers into market segment(s)
- imputation: recommend new products, target advertising campaign

Impute censored data

synthetic data:

- generate rank-5 matrix of probabilities, $p \in \mathbf{R}^{300 \times 300}$

customer	apples	oranges	pears	$\cdots$
1	.28	.22	.76	$\cdots$
2	.97	.55	.36	$\cdots$
3	.13	.47	.62	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	

Impute censored data

synthetic data:

- entry (i, j) is $+$ with probability p_{ij}

customer	apples	oranges	pears	$\cdots$
1	$+$	-	$+$	$\cdots$
2	$+$	$+$	-	$\cdots$
3	-	$+$	$+$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	

Impute censored data

synthetic data:

- but we only observe +s...

customer	apples	oranges	pears	$\cdots$
1	+	?	+	$\cdots$
2	+	+	?	$\cdots$
3	?	+	+	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	

Impute censored data

synthetic data:

- . . . and we only observe 10% of the +s

customer	apples	oranges	pears	$\cdots$
1	+	?	?	$\cdots$
2	?	+	?	$\cdots$
3	?	?	?	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	

Impute censored data

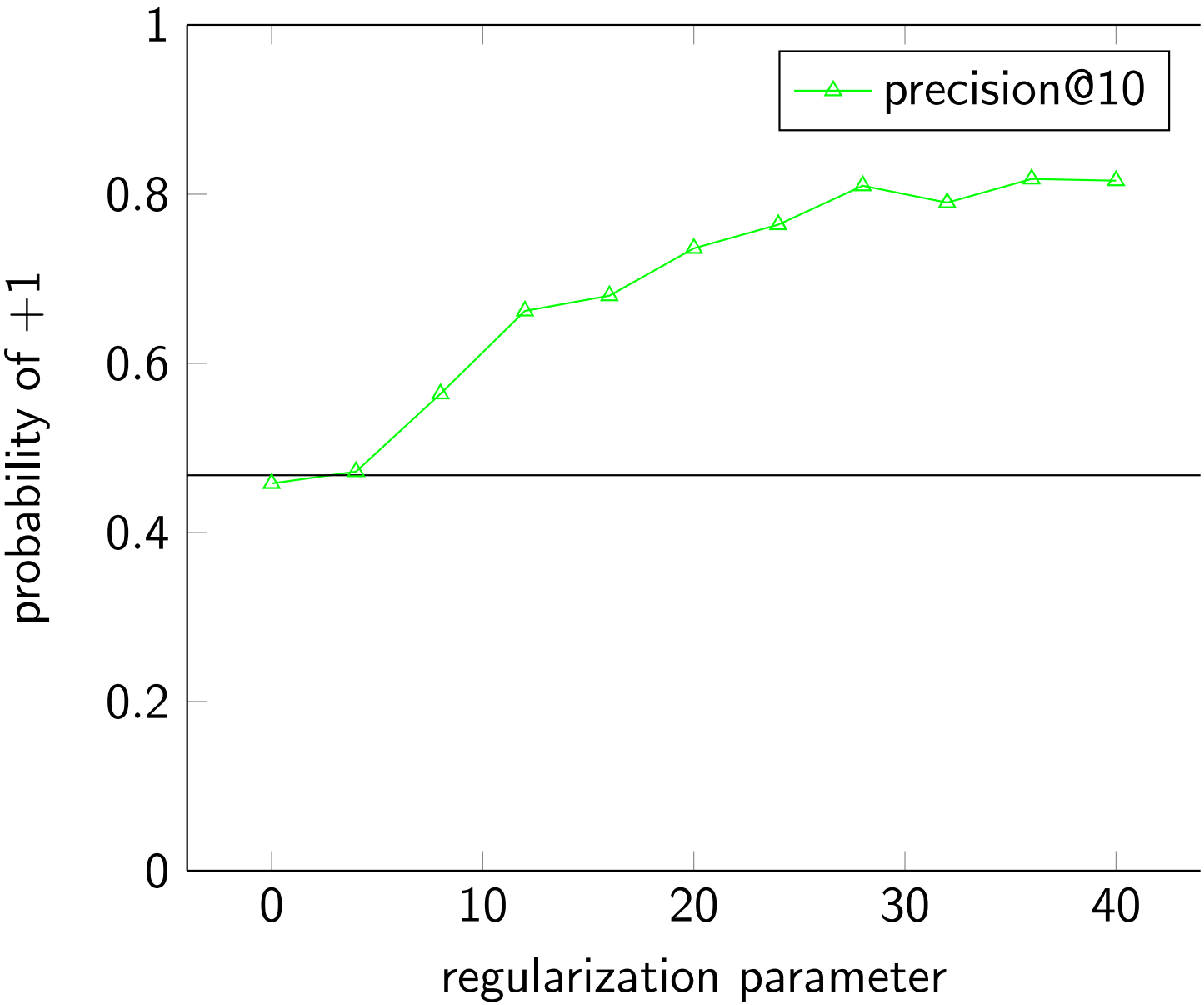
synthetic data:

- . . . and we only observe 10% of the +s

customer	apples	oranges	pears	$\cdots$
1	+	?	?	$\cdots$
2	?	+	?	$\cdots$
3	?	?	?	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	

can we predict 10 more +s?

Impute censored data



What do we know about missing data?

missingness of entry (i, j) can depend on

- nothing ("missing completely at random")
- row i and column j
- values of other entries in the table ("missing at random")
- value of entry (i, j)

examples

- income in survey
- lab tests in electronic health record
- clicks in online advertising

Multiple imputation

multiple imputation [Rubin 1987, 1966]

- generate multiple imputed data sets
- perform analysis on each one
- average results

proper imputation: imputation procedure "matches" data generation and analysis procedures

- when data is MAR and imputation is *proper*, results of analysis are consistent [Rubin 1966]
- when data is not MAR, proper imputation techniques can fail *badly* [Allison 1999]

Methods for multiple imputation

for each imputation:

- subsample from the data
- fit a model
- sample from the model

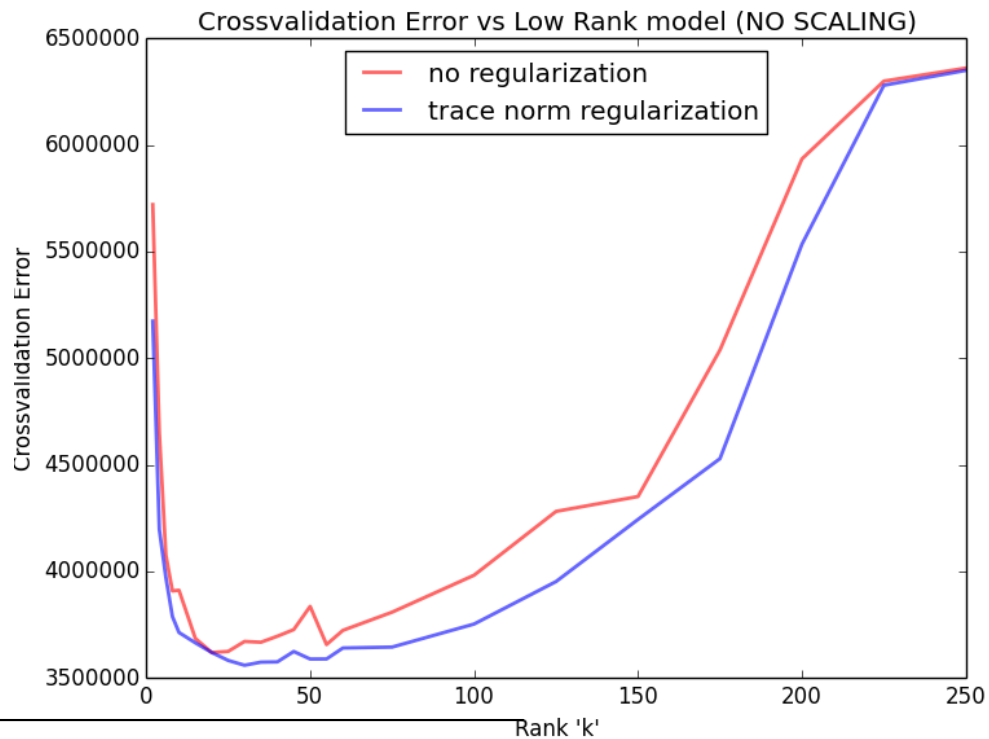
many models:

- Amelia II: [Blackwell King Honaker 2012]
  multivariate normal prior
- MICE: [Van Buren 2011]
  multiple imputation via chained equations
- GLRMs:
  - one imputation: impute with the conditional *mode*
  - multiple imputation: impute with random draws from the conditional *distribution*

Low rank models on GSS data[2]

General Social Survey (GSS):

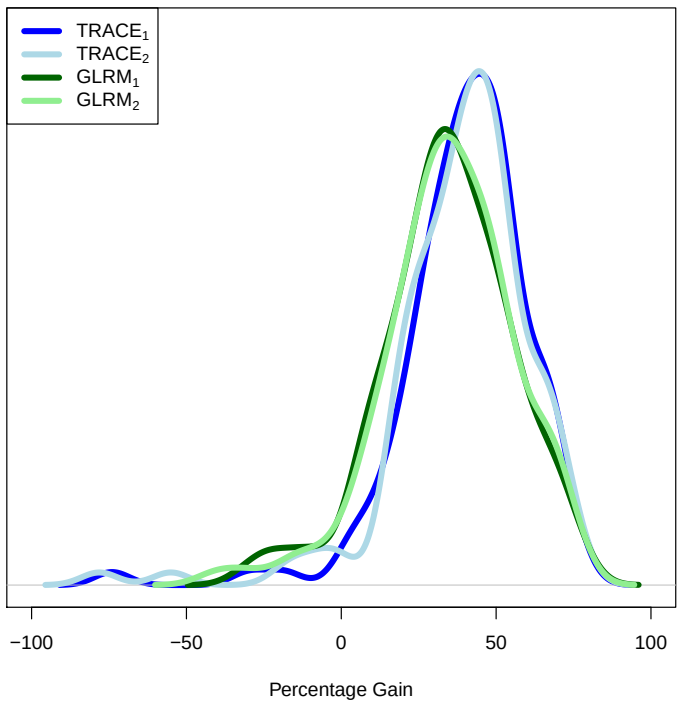
- survey adults in randomly selected US households about attitudes and demographics
- 7422 rows, 69 columns
- $>$ 33% missing data



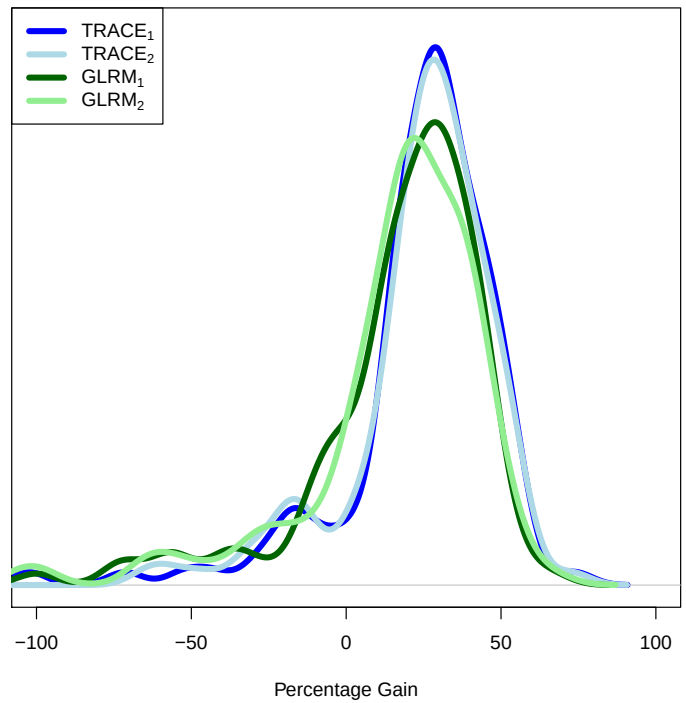
[2][Sengupta U Srebro Evans, in prep]

Multiple imputation comparison

GLRM vs. Amelia



GLRM vs. MICE



[Sengupta U Srebro Evans, in prep]

Outline

Outline

Fitting GLRMs with alternating minimization

$$\text{minimize} \quad \sum_{(i,j)\in\Omega} L_j(x_i y_j, A_{ij}) + \sum_{i=1}^m r_i(x_i) + \sum_{j=1}^n \tilde{r}_j(y_j)$$

repeat:

1. minimize objective over x_i (in parallel)
2. minimize objective over y_j (in parallel)

properties:

- subproblems easy to solve
- objective decreases at every step, so converges if losses and regularizers are bounded below
- (not guaranteed to find global solution, but) usually finds good model in practice
- naturally parallel, so scales to *huge* problems

Alternating updates

given X^0, Y^0
for $t = 1, 2, \ldots$ **do**
  for $i = 1, \ldots, m$ **do**
    $x_i^t = \textbf{update}_{L,r}(x_i^{t-1}, Y^{t-1}, A)$
  for $j = 1, \ldots, n$ **do**
    $y_j^t = \textbf{update}_{L,\tilde{r}}(y_j^{(t-1)T}, X^{(t)T}, A^T)$

- no need to exactly minimize
- choose fast, simple update rules

Proximal operator

define the *proximal operator*

$$\mathbf{prox}_f(z) = \operatorname*{argmin}_x (f(x) + \frac{1}{2}\|x - z\|_2^2)$$

- **generalized projection:** if $\mathbf{1}_() C$ is the indicator function of a set C, then
$$\mathbf{prox}_{\mathbf{1}_() C}(z) = \Pi_C(z)$$
- **implicit gradient step:** if $x = \mathbf{prox}_f(z)$, then
$$\begin{aligned} \nabla f(x) + x - z &= 0 \\ x &= z - \nabla f(x) \end{aligned}$$
- **simple to evaluate:** closed form solutions for
  - $f = \|\cdot\|_2^2$
  - $f = \|\cdot\|_1$
  - $f = \mathbf{1}_+$
  - ...

more info: Amir Beck's tutorial...!

Proximal gradient method

want to solve

$$\text{minimize} \quad f(x) + g(x)$$

- $f : \mathbf{R}^k \to \mathbf{R}$ smooth
- $g : \mathbf{R}^k \to \mathbf{R}$ with a fast prox operator

proximal gradient method.

- pick step size sequence $\{\alpha_t\}_{t=1}^{\infty}$ and $x^0 \in \mathbf{R}^k$
- repeat
  - $x^{t+1} = \mathbf{prox}_{\alpha_t g}(x^t - \alpha_t \nabla f(x^t))$

PALM [Bolte, Sabach, Teboulle 2014]

proximal alternating linearized minimization (PALM):
alternate between proximal gradient updates on X and on Y

- pick $Y^0 = 0$
- repeat
  - for $i = 1, \ldots, n$,
    - let
$$g = \sum_{j:(i,j)\in\Omega} \nabla L_j(x_i y_j, A_{ij}) y_j$$
    - update
$$x_i^{t+1} = \mathbf{prox}_{\alpha_t r}(x_i^t - \alpha_t g)$$
  - (Y update is similar)

- **simple:** only requires ability to evaluate ∇L and $\mathbf{prox}_r$
- **parallelizable:** $O(\frac{(n+m+|\Omega|)k}{p})$ flops/iteration on p workers

Outline

Waiting for lazy workers

problem: the long tail

- do flops/iteration/worker predict time/iteration?
- no: have to wait for the last worker to finish

solution: asynchrony

- remove synchronization steps
- ignore read/write conflicts

hope: more cpu cycles compensate for sloppy division of labor

Asynchronous algorithms: a brief history

hard to debug $\implies$ nothing more practical than a good theory

- Parallel and Distributed Computation [Bertsekas Tsitsiklis 1989]
- Hogwild [Recht Ré Niu Wright 2011]
  convex async parallel SGD
- ARock [Peng Yu Yan Yin 2015]
  async parallel coordinate updates for fixed point problems
- [Lian Huang Li Liu 2015]
  nonconvex smooth async parallel SGD
- APALM [Davis 2016] (async PALM)
- **SAPALM** [Davis Edmunds U 2016]
  stochastic async PALM

Adding noise

why add noise?

- *avoiding saddles.* achieve convergence to a *better* stationary point by injecting noise
- *speed up iterations.* noise captures, *e.g.*, stochastic gradients or computational approximations

Stochastic gradient = gradient + noise

- define $n_j = |\{j : (i,j) \in \Omega\}|$
- suppose j' chosen at random from among $\{j : (i,j) \in \Omega\}$

then

$$\begin{aligned} \mathbb{E}\left[n_{j'} \nabla L_{j'}(x_i y_{j'}, A_{ij'}) y_{j'}\right] &= \sum_{j:(i,j)\in\Omega} \nabla L_j(x_i y_j, A_{ij}) y_j =: g \\ n_{j'} \nabla L_{j'}(x_i y_{j'}, A_{ij'}) y_{j'} &= g + \text{noise} \end{aligned}$$

minibatch:

- can estimate gradient with *less noise* by summing over *more* random observations j'

$$\sum_{j'} n_{j'} \nabla L_{j'}(x_i y_{j'}, A_{ij'}) y_{j'} = g + \text{noise}$$

- minibatch size = number of random observations j' in sum

Generalizing generalized low rank models

SAPALM solves the problem

$$\text{minimize} \quad f(z_1, \ldots, z_\ell) + \sum_{j=1}^{\ell} r_j(z_j)$$

- f is smooth and L-Lipshitz (not necessarily convex)
- r_j is lower semicontinuous (not necessarily convex or smooth)
- $z_j \in \mathbf{R}^{\ell_j}$, $j = 1, \ldots, \ell$ is a coordinate or coordinate block
- partial gradients $\frac{\partial f}{\partial z_j}$ are L_j-Lipshitz continuous

example: to optimize GLRMs, set

- $\ell = m + n$
- $z = (x_1, \ldots, x_m, y_1, \ldots, y_n)$
- $f(z) = \sum_{(i,j)\in\Omega} L_j(x_i y_j, A_{ij})$
- $\sum_{j=1}^{\ell} r_j(z_j) = \sum_{i=1}^{n} r_i(x_i) + \sum_{j=1}^{n} \tilde{r}_j(y_j)$

SAPALM: local view

Require: $z \in \mathbf{R}^{\sum_{j=1}^{\ell} \ell_j}$

All processors in parallel do
loop
    Randomly select a coordinate block $j \in \{1, \ldots, \ell\}$
    Read z from shared memory
    Compute $g = \nabla_j f(z) + \nu_j$
    Choose stepsize $\gamma_j \in \mathbf{R}_{++}$
    $z_j \leftarrow \mathbf{prox}_{\gamma_j r_j}(z_j - \gamma_j g)$

The delayed iterate

- each processor reads, computes, then writes to z
- in the meantime, z may have changed!

for the analysis (not needed by the algorithm):

- define the iteration counter k (updated whenever z is updated)
- consider the kth update, and the worker executing it
- count the number of updates $d_{k,j}$ made to z_j between reading and writing z_j
- define the (inconsistently) *delayed iterate* $z^{k-d_k} = (z_1^{k-d_{k,1}}, \ldots, z_\ell^{k-d_{k,\ell}})$
- define the maximum delay $\tau = \max_{k,j} d_{k,j}$

note: x^{k-d_k} need never have existed in memory!

SAPALM: global view

Require: $z \in \mathbf{R}^{\sum_{j=1}^{\ell} \ell_j}$

for $k = 1, \ldots$ **do**
  Randomly select a coordinate block $j_k \in \{1, \ldots, \ell\}$
  Read $z^{k-d_k} = (z_1^{k-d_{k,1}}, \ldots, z_\ell^{k-d_{k,\ell}})$ from shared memory
  Compute $g^k = \nabla_{j_k} f(z^{k-d_k}) + \nu_{j_k}^k$
  Choose stepsize $\gamma_{j_k}^k \in \mathbf{R}_{++}$
  for $j = 1, \ldots, \ell$ **do**
    if $j = j_k$ **then**
      $z_{j_k}^{k+1} \leftarrow \mathbf{prox}_{\gamma_{j_k}^k r_{j_k}}(z_{j_k}^k - \gamma_{j_k}^k g^k)$
    else
      $z_j^{k+1} \leftarrow z_j^k$

Convergence in what sense?

Lyapunov function measures *expected violation of stationarity*

$$S_k = \mathbb{E}\left[\sum_{j=1}^{\ell}\left\|\frac{1}{\gamma_j^k}(w_j^k - z_j^k) + \nu_k\right\|^2\right]$$

where

$$w_j^k = \mathbf{prox}_{\gamma_j^k r_j}(z_j^k - \gamma_j^k(\nabla_j f(z^{k-d_k}) + \nu_j^k)), \quad j = 1, \ldots, \ell$$

if $d_k = 0$ and $r = 0$, then

$$S_k = \mathbb{E}\left[\|\nabla f(z^k)\|^2\right]$$

Noise determines step size

noise ν^k determines maximal step size sequence

- let $\sigma_k^2 := \mathbb{E}\left[\|\nu_k\|^2\right]$ and let $a \in (1, \infty)$.
- assume $\mathbb{E}\left[\nu^k\right] = 0$
- pick stepsize decay c_k, stepsizes γ_j^k, so $\forall k = 1, 2, \ldots,$ $\forall j \in \{1, \ldots, m\}$,

$$\gamma_j^k = \frac{1}{a c_k (L_j + 2L\tau m^{-1/2})}.$$

Noise determines step size

noise ν^k determines maximal step size sequence

- let $\sigma_k^2 := \mathbb{E}\left[\|\nu_k\|^2\right]$ and let $a \in (1, \infty)$.
- assume $\mathbb{E}\left[\nu^k\right] = 0$
- pick stepsize decay c_k, stepsizes γ_j^k, so $\forall k = 1, 2, \ldots,$ $\forall j \in \{1, \ldots, m\}$,

$$\gamma_j^k = \frac{1}{a c_k (L_j + 2L\tau m^{-1/2})}.$$

two ways to choose the stepsize decay c:

- **summable.** if $\sum_{k=0}^{\infty} \sigma_k^2 < \infty$, choose $c_k = 1$.
- α**-diminishing.** if $\alpha \in (0, 1)$ & $\sigma_k^2 = O((k+1)^{-\alpha}$, choose $c_k \sim (k+1)^{(1-\alpha)}$.

Convergence (summable noise)

Theorem ([Davis Edmunds U 2016])

If $\sum_{k=1}^{\infty} \nu^k < \infty$, *then for every* $T = 1, \ldots$

$$\min_{k=0,\ldots,T} S_k = O\left(\frac{\ell(\max_j L_j + 2L\tau\ell^{-1/2})}{T+1}\right)$$

- if maximum delay $\tau = O(\sqrt{\ell})$, achieve linear speedup
- usually τ scales with the number of processors
- so, linear speedup on up to $O(\sqrt{\ell})$ processors

Convergence (α-diminishing noise)

Theorem ([Davis Edmunds U 2016])

If noise is α-diminishing, then for every $T = 1, \ldots$

$$\min_{k=0,\ldots,T} S_k = O\left(\frac{\ell(\max_j L_j + 2L\tau\ell^{-1/2}) + \ell\log(T+1)}{(T+1)^{-\alpha}}\right)$$

- if maximum delay $\tau = O(\sqrt{\ell})$, achieve linear speedup
- usually τ scales with the number of processors
- so, linear speedup on up to $O(\sqrt{\ell})$ processors

Convergence (α-diminishing noise)

Theorem ([Davis Edmunds U 2016])

If noise is α-diminishing, then for every $T = 1, \ldots$

$$\min_{k=0,\ldots,T} S_k = O\left(\frac{\ell(\max_j L_j + 2L\tau\ell^{-1/2}) + \ell\log(T+1)}{(T+1)^{-\alpha}}\right)$$

- if maximum delay $\tau = O(\sqrt{\ell})$, achieve linear speedup
- usually τ scales with the number of processors
- so, linear speedup on up to $O(\sqrt{\ell})$ processors

open problem: convergence with non-decreasing noise?

But does it work?

two test problems:

- *Sparse PCA.*

$$\underset{X,Y}{\operatorname{argmin}} \frac{1}{2}||A - X^T Y||_F^2 + \lambda\|X\|_1 + \lambda\|Y\|_1,$$

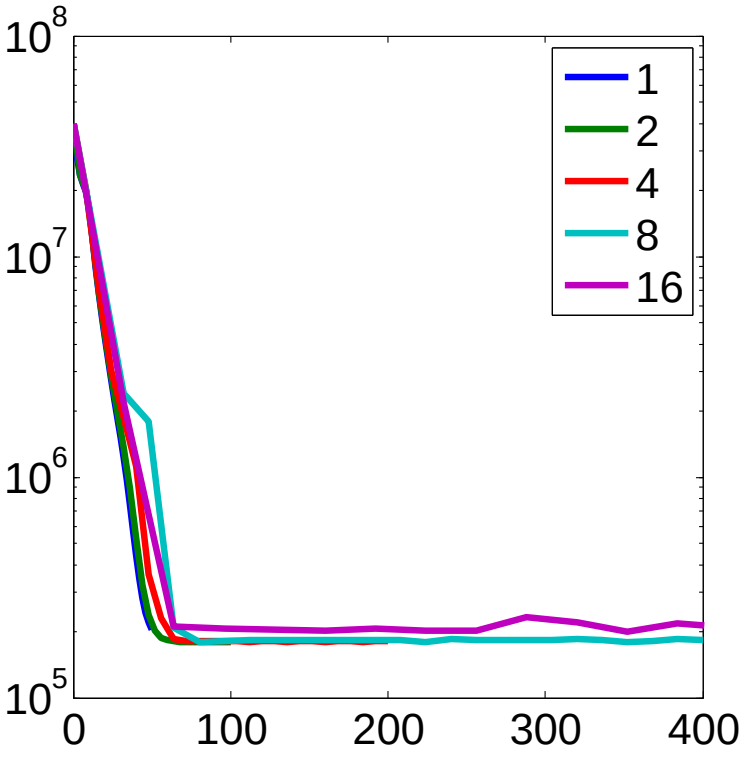
- *Firm Thresholding PCA.* [Woodworth Chartrand 2015]

$$\underset{X,Y}{\operatorname{argmin}} \frac{1}{2}||A-X^T Y||_F^2+\lambda(\|X\|_{\text{Firm}}+\|Y\|_{\text{Firm}})+\frac{\mu}{2}(\|X\|_F^2+\|Y\|_F^2),$$

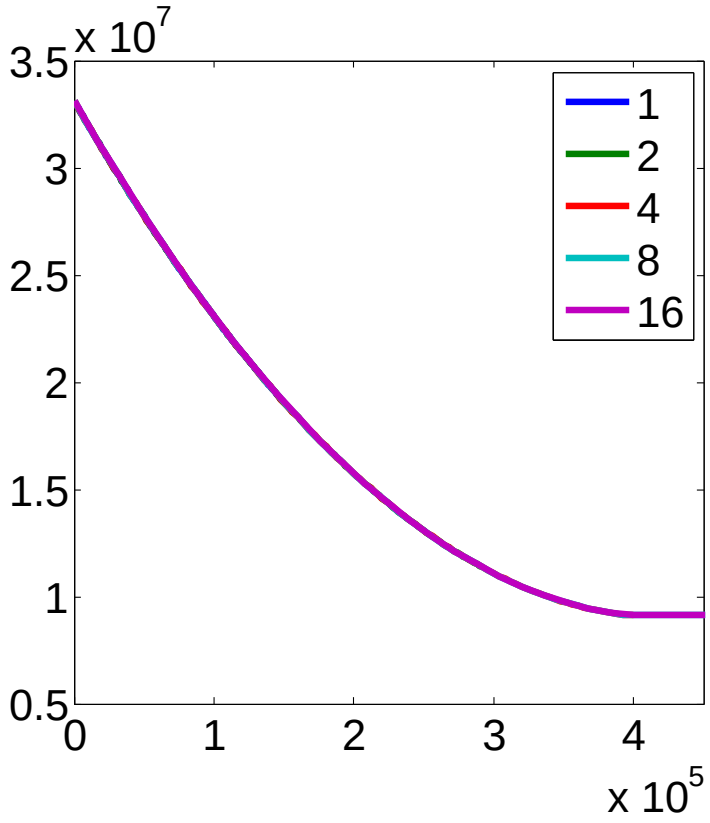
(nonconvex, nonsmooth regularizer)

Same flops, same progress

iterates vs objective



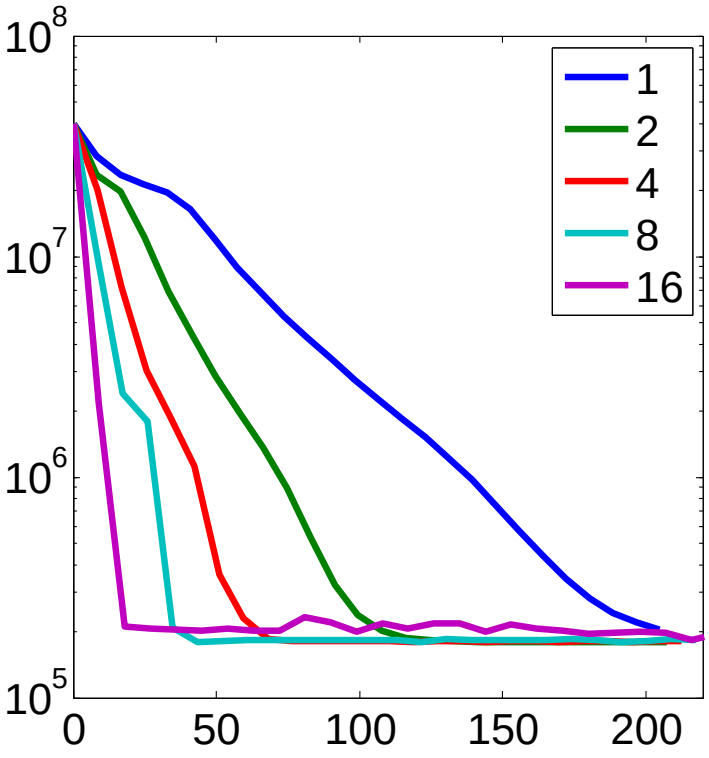
Sparse PCA



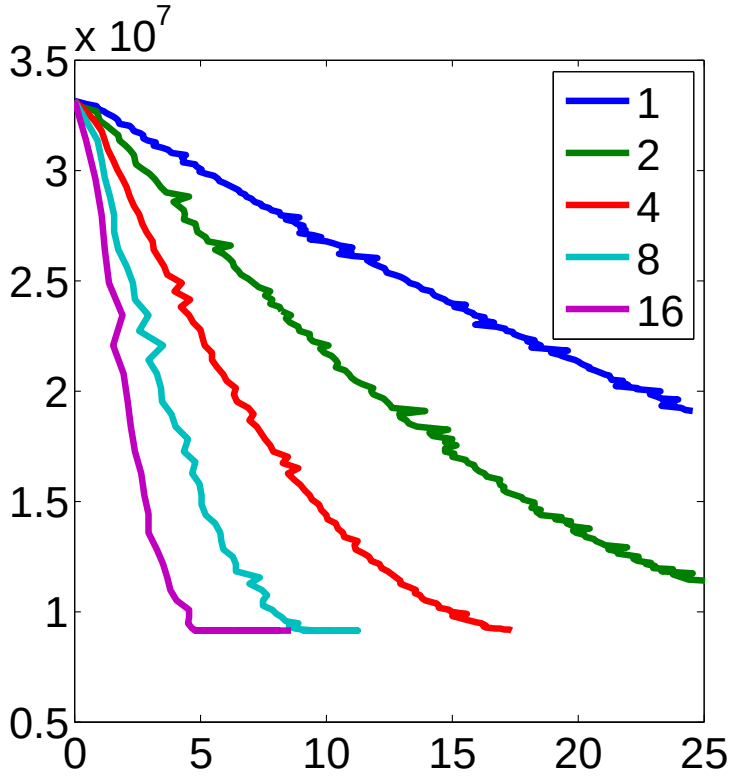
Firm PCA

More workers, faster progress

time (s) vs objective



Sparse PCA

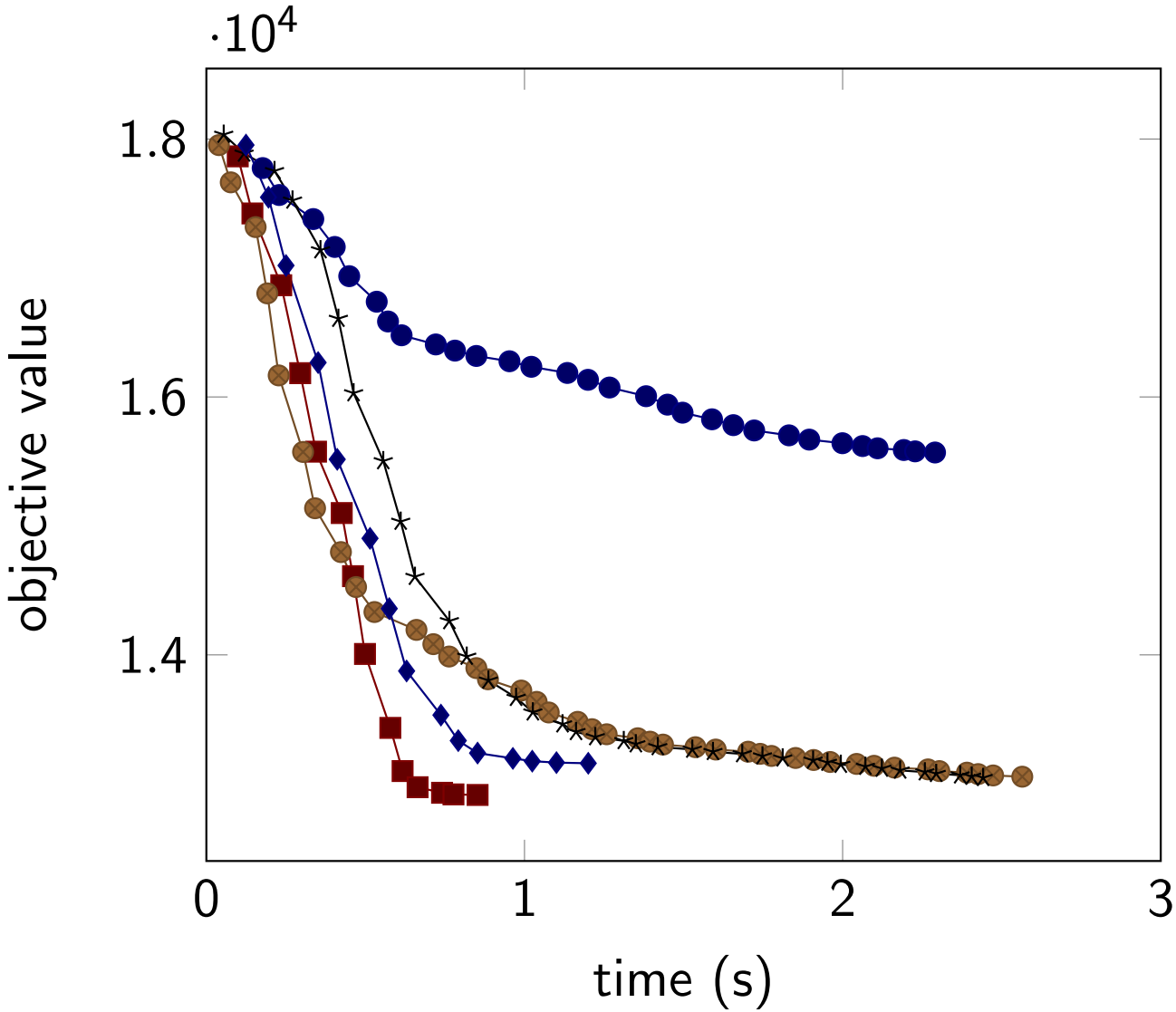


Firm PCA

Outline

Initialization matters

NNMF for $k = 2$: optimal value depends on initialization

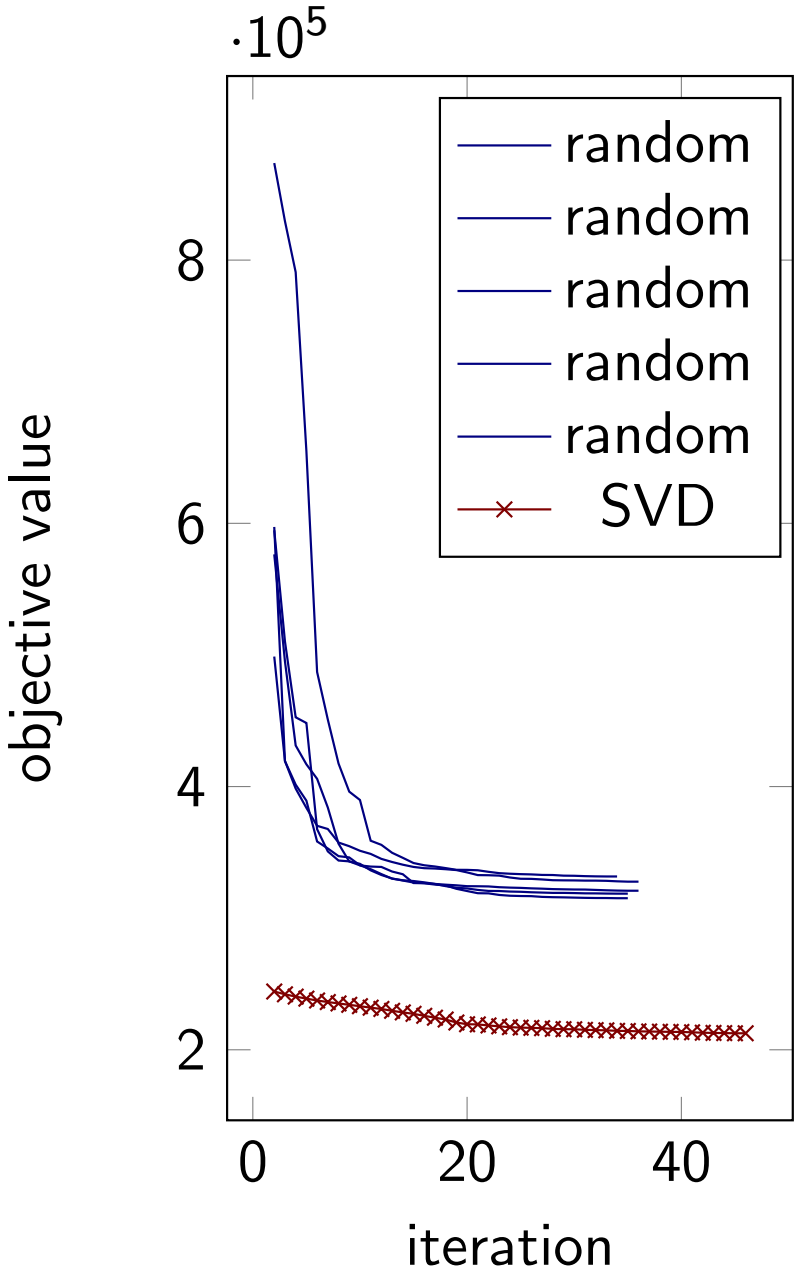


Initializing via SVD

- fit census data set
- random initialization

$$x_i \sim \mathcal{N}(0, I_k)$$
$$y_j \sim \mathcal{N}(0, I_k)$$

- SVD initialization
  - interpret A as numerical matrix M
  - fill in missing entries in M to preserve column mean and variance
  - center and standardize M
  - initialize XY with top k singular tuples of M



Why does SVD initialization work?

Theorem (Tropp 2015 Cor. 6.2.1)

Let $R_1, \ldots, R_n$ be iid random variables with $\mathbb{E}R_i = Z$ for $i = 1, \ldots, n$. Define $\gamma = \|R\|$, $C = \max(\|\mathbb{E}RR^T\|, \|\mathbb{E}R^TR\|)$. Then for every $\delta > 0$,

$$\mathbb{P}\left\{\|\sum_{i=1}^{n} R_i - Z\| \geq \delta\right\} \leq (m+n)\exp\left(\frac{-n\delta^2}{2C + \gamma\delta/3}\right).$$

SVD initialization

- find transformation T so that $\mathbb{E}\,T(A_{ij}) \approx Z$
- top k singular tuples of

$$\frac{1}{|\Omega|} \sum_{(ij)\in\Omega} T(A_{ij})$$

will be close to Z

SVD initialization: examples

if $A_{ij} = Z_{ij} + \epsilon_{ij}$, ϵ_{ij} iid normal,

- random sampling: i, j chosen uniformly at random

$$\mathbb{E} mn A_{ij} e_i e_j^T = Z$$

- row- and column-biased sampling: if i chosen w/prob p_i, j chosen w/prob q_j

$$\mathbb{E} \frac{1}{p_i q_j} A_{ij} e_i e_j^T = Z$$

(can estimate p_i and q_j from empirical distribution...)

SVD initialization: examples

- under random sampling, if $A_{ij} = \alpha_j Z_{ij} + \beta_j + \epsilon_{ij}$ with ϵ_{ij} iid normal,

$$\mathbb{E} mn \left(\frac{A_{ij} - \beta_j}{\alpha_j} \right) e_i e_j^T = Z$$

(can estimate α_j and β_j by empirical mean and variance)

- under random sampling, if

$$A_{ij} = \begin{cases} 1 & \text{with probability } \operatorname{logistic}(Z_{ij}) = (1 + \exp(-Z_{ij}))^{-1} \\ 0 & \text{otherwise} \end{cases}$$

then

$$\begin{aligned} \mathbb{E} mn A_{ij} e_i e_j^T &= \operatorname{logistic}(Z) \\ \operatorname{logit}(\mathbb{E} mn A_{ij} e_i e_j^T) &= Z \end{aligned}$$

near $x = 1/2$, we have $\operatorname{logit}(x) \approx 4(x - 1/2)$, so

$$\mathbb{E} 4mn(A_{ij} - 1/2) e_i e_j^T \approx Z$$

Outline

Time to simplify notation!

rewrite the low rank model

$$\text{minimize} \quad \sum_{(i,j)\in\Omega} L_j(x_i y_j, A_{ij}) + \sum_{i=1}^{m} \|x_i\|^2 + \sum_{j=1}^{n} \|y_j\|^2$$

as

$$\text{minimize} \quad L(XY) + \|X\|_F^2 + \|Y\|_F^2$$

When is a low rank model an SDP?

Theorem

(X, Y) *is a solution to*

$$\textit{minimize} \quad L(XY) + \tfrac{\gamma}{2}\|X\|_F^2 + \tfrac{\gamma}{2}\|Y\|_F^2 \qquad (\mathcal{F})$$

if and only if $Z = XY$ *is a solution to*

$$\begin{array}{ll} \textit{minimize} & L(Z) + \gamma\|Z\|_* \\ \textit{subject to} & Rank(Z) \leq k \end{array} \qquad (\mathcal{R})$$

where $\|Z\|_*$ *is the sum of the singular values of* Z*.*

- if F is convex, then $\mathcal{R}$ is a rank-constrained semidefinite program
- local minima of $\mathcal{F}$ correspond to local minima of $\mathcal{R}$

Proof of equivalence

suppose $Z = XY = U\Sigma V^T$

- $\mathcal{F} \leq \mathcal{R}$: if Z is feasible for $\mathcal{R}$, then

$$X = U\Sigma^{1/2}, \qquad Y = \Sigma^{1/2}V^T$$

  is feasible for $\mathcal{F}$, with the same objective value
- $\mathcal{R} \leq \mathcal{F}$: for any $XY = Z$,

$$\begin{array}{rcl} \|Z\|_* & = & \mathbf{tr}(\Sigma) \\ & = & \mathbf{tr}(U^T XYV) \\ & \leq & \|U^T X\|_F \|YV\|_F \\ & \leq & \|X\|_F \|Y\|_F \\ & \leq & \frac{1}{2}(||X||_F^2 + ||Y||_F^2) \end{array}$$

Convex equivalence

Theorem

For every $\gamma \geq \gamma^\star(k)$, every solution to

$$\begin{array}{ll} \textit{minimize} & L(Z) + \gamma\|Z\|_* \\ \textit{subject to} & Rank(Z) \leq k \end{array} \qquad (\mathcal{R})$$

(with variable $Z \in \mathbf{R}^{m\times n}$) is a solution to

$$\textit{minimize} \quad L(Z) + \gamma\|Z\|_*. \qquad (\mathcal{U})$$

proof: find $\gamma^\star(k)$ so large that there is a Z with rank $\leq k$ satisfying optimality conditions for $\mathcal{U}$

- if γ is sufficiently large (compared to k), rank constraint is *not binding*

Certify global optimality, sometimes

two ways to use convex equivalence:

- **convex:**
  1. solve the unconstrained SDP

$$\text{minimize} \quad L(Z) + \gamma \|Z\|_*$$

  2. see if the solution is low rank
- **nonconvex:**
  1. fit the GLRM with any method, producing (X, Y)
  2. check if $XY = U\Sigma V^T$ satisfies the optimality conditions for the (convex) unconstrained SDP

Check optimality conditions

let $Z = U\Sigma V^T$ with $\mathbf{diag}\,\Sigma > 0$ be the rank-revealing SVD. (*i.e.*, $\mathrm{Rank}(Z) = k$, $\Sigma \in \mathbf{R}^{k\times k}$.)

the subgradient of the objective

$$\mathrm{obj}(Z) = L(Z) + \|Z\|_*$$

is any matrix of the form $G + UV^T + W$ with

- $G \in \partial L(Z)$
- $U^T W = 0$
- $WV = 0$
- $\|W\|_2 \leq 1$.

Check optimality conditions

let $Z = U\Sigma V^T$ with $\mathbf{diag}\,\Sigma > 0$ be the rank-revealing SVD.
(*i.e.*, $\mathrm{Rank}(Z) = k$, $\Sigma \in \mathbf{R}^{k\times k}$.)

the subgradient of the objective

$$\mathrm{obj}(Z) = L(Z) + \|Z\|_*$$

is any matrix of the form $G + UV^T + W$ with

- $G \in \partial L(Z)$
- $U^T W = 0$
- $WV = 0$
- $\|W\|_2 \leq 1$.

for any matrices G and W satisfying these conditions,

$$\begin{aligned}\mathrm{obj}(Z) \geq \mathrm{obj}(Z^\star) &\geq \mathrm{obj}(Z) + \langle G + UV^T + W, Z^\star - Z\rangle \\ &\geq \mathrm{obj}(Z) - \|G + UV^T + W\|_F \|Z^\star - Z\|_F.\end{aligned}$$

Check optimality conditions

let $Z = U\Sigma V^T$ with $\mathbf{diag}\,\Sigma > 0$ be the rank-revealing SVD.
(*i.e.*, $\mathrm{Rank}(Z) = k$, $\Sigma \in \mathbf{R}^{k\times k}$.)

the subgradient of the objective

$$\mathrm{obj}(Z) = L(Z) + \|Z\|_*$$

is any matrix of the form $G + UV^T + W$ with

- $G \in \partial L(Z)$
- $U^T W = 0$
- $WV = 0$
- $\|W\|_2 \leq 1$.

for any matrices G and W satisfying these conditions,

$$\begin{aligned}\mathrm{obj}(Z) \geq \mathrm{obj}(Z^\star) &\geq \mathrm{obj}(Z) + \langle G + UV^T + W, Z^\star - Z\rangle \\ &\geq \mathrm{obj}(Z) - \|G + UV^T + W\|_F \|Z^\star - Z\|_F.\end{aligned}$$

(any two conjugate norms work in the second inequality.)

Check optimality conditions

- $\|G + UV^T + W\|_F$ bounds the suboptimality of the solution.
- if $\|G + UV^T + W\|_F = 0$, then $Z = Z^\star$.
- to find a good bound, solve for G and W:

$$\begin{array}{ll} \text{minimize} & \|G + UV^T + W\|_F^2 \\ \text{subject to} & \|W\|_2 \leq 1 \\ & U^T W = 0 \\ & WV = 0 \\ & G \in \partial L(Z) \end{array}$$

- if loss is differentiable, G is fixed. then using Pythagoras

$$W^\star = \frac{(I - UU^T)G(I - VV^T)}{\|(I - UU^T)G(I - VV^T)\|_2}.$$

Why use the low rank formulation? (statistics)

pro

- low rank factors are
  - easier to interpret
  - smaller to represent
  - more strongly regularized
- theoretical recovery results hold up

con

- low rank constraint too strong(?)

Why use the low rank formulation? (optimization)

pro

- size of problem variable: $(m+n)k$ vs mn
- smooth regularizer: frobenius vs trace norm
- no eigenvalue computations needed
- parallelizable
- (almost) no new local minima if k is large enough
  - solution to rank-constrained SDP is in the relative interior of a face over which the objective is constant [Burer Monteiro]
  - special case: matrix completion has no spurious local minima [Ge Lee Ma, 2016]
- linear convergence, sometimes
  - *e.g.*, if loss is differentiable and strongly convex on the set of rank-k matrices [Bhojanapalli Kyrillidis Sanghavi 2015]

con

- nonconvex (biconvex) formulation
  - local minima
  - saddle points

Implementations

Implementations in Python (serial), Julia (shared memory parallel), Spark (parallel distributed), and H2O (parallel distributed).

- libraries for computing gradients and proxs
- simple user interface
- easy to write new algorithms that work for *all* GLRMs

example: (Julia) forms and fits a k-means model with $k = 5$

```
losses = QuadLoss()                 # minimize squared error
rx = UnitOneSparseConstraint() # one cluster per row
ry = ZeroReg()                      # free cluster centroids
glrm = GLRM(A,losses,rx,ry,k)  # form model
fit!(glrm)                          # fit model
```

Algorithms: summary

- convex methods: (interior point, ADMM, ALM, . . .)
  - guaranteed convergence to global optimum
  - require (at least) (possibly full) SVD at each iteration
  - fast iteration complexity slows convergence
- nonconvex (factored) methods: (alternating minimization, alternating gradient method, alternating proximal gradient method, . . .)
  - guaranteed convergence to local optimum
  - can be initialized provably close to local optimum (sometimes)
  - fast, parallelizable iterations

Outline

Multiclass classification

how to predict *categorical* values?

Multiclass classification

how to predict *categorical* values?

for this discussion, fix $u = x_i Y_j$, $\mathcal{F} = \mathcal{F}_j$, $a = A_{ij}$ for some i, j

- *idea 1: classification*
  1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
  2. predict entries of $\psi(a)$
  3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$

Multiclass classification

how to predict *categorical* values?

for this discussion, fix $u = x_i Y_j$, $\mathcal{F} = \mathcal{F}_j$, $a = A_{ij}$ for some i, j

- *idea 1: classification*
  1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
  2. predict entries of $\psi(a)$
  3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$
- *idea 2: learning probabilities*
  1. learn the probability $\mathbb{P}(a = a' \mid u)$ for every $a' \in \mathcal{F}$
  2. predict $\hat{a} = \operatorname{argmax}_{a' \in \mathcal{F}} \mathbb{P}(a = a' \mid u)$
  3. u parametrizes probability distribution

Multiclass classification via binary classification

idea 1: classification

1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
2. predict entries of $\psi(a)$
3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$

Q: how to pick $\psi(a)$? (suppose $\mathcal{F} = \{1, \ldots, \ell\}$)

Multiclass classification via binary classification

idea 1: classification

1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
2. predict entries of $\psi(a)$
3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$

Q: how to pick $\psi(a)$? (suppose $\mathcal{F} = \{1, \ldots, \ell\}$)

- one-hot encoding: if $a = i$,

$$\psi(a) = (-1, \ldots, \overbrace{1}^{i\text{th entry}}, \ldots, -1) \in \{-1, 1\}^{\ell}$$

(resulting scheme is called *one-vs-all* classification)

Multiclass classification via binary classification

idea 1: classification

1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
2. predict entries of $\psi(a)$
3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$

Q: how to pick $\psi(a)$? (suppose $\mathcal{F} = \{1, \ldots, \ell\}$)

- one-hot encoding: if $a = i$,

$$\psi(a) = (-1, \ldots, \overbrace{1}^{i\text{th entry}}, \ldots, -1) \in \{-1, 1\}^{\ell}$$

  (resulting scheme is called *one-vs-all* classification)
- binary codes:
  - define binary expansion of y, $\text{bin}(a) \in \{0, 1\}^{\log(\ell)}$
  - let $\psi(a) = 2\,\text{bin}(a) - 1 \in \{-1, 1\}^{\log(\ell)}$

Multiclass classification via binary classification

idea 1: classification

1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
2. predict entries of $\psi(a)$
3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$

Q: how to pick $\psi(a)$? (suppose $\mathcal{F} = \{1, \ldots, \ell\}$)

- one-hot encoding: if $a = i$,

$$\psi(a) = (-1, \ldots, \overbrace{1}^{i\text{th entry}}, \ldots, -1) \in \{-1, 1\}^{\ell}$$

  (resulting scheme is called *one-vs-all* classification)
- binary codes:
  - define binary expansion of y, $\text{bin}(a) \in \{0, 1\}^{\log(\ell)}$
  - let $\psi(a) = 2\,\text{bin}(a) - 1 \in \{-1, 1\}^{\log(\ell)}$
- error-correcting codes

Multiclass classification via binary classification

idea 1: classification

1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
2. predict entries of $\psi(a)$
3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$

Q: how to pick $\psi(a)$? (suppose $\mathcal{F} = \{1, \ldots, \ell\}$)

- one-hot encoding: if $a = i$,

$$\psi(a) = (-1, \ldots, \overbrace{1}^{i\text{th entry}}, \ldots, -1) \in \{-1, 1\}^{\ell}$$

  (resulting scheme is called *one-vs-all* classification)
- binary codes:
  - define binary expansion of y, $\text{bin}(a) \in \{0, 1\}^{\log(\ell)}$
  - let $\psi(a) = 2\,\text{bin}(a) - 1 \in \{-1, 1\}^{\log(\ell)}$
- error-correcting codes

these vary in the *embedding dimension* $\dim(\psi(a))$

Multiclass classification via binary classification

idea 1: classification

1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
2. predict entries of $\psi(a)$
3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$

Q: how to predict entries of $\psi(a) \in \{-1, 1\}^\ell$?

Multiclass classification via binary classification

idea 1: classification

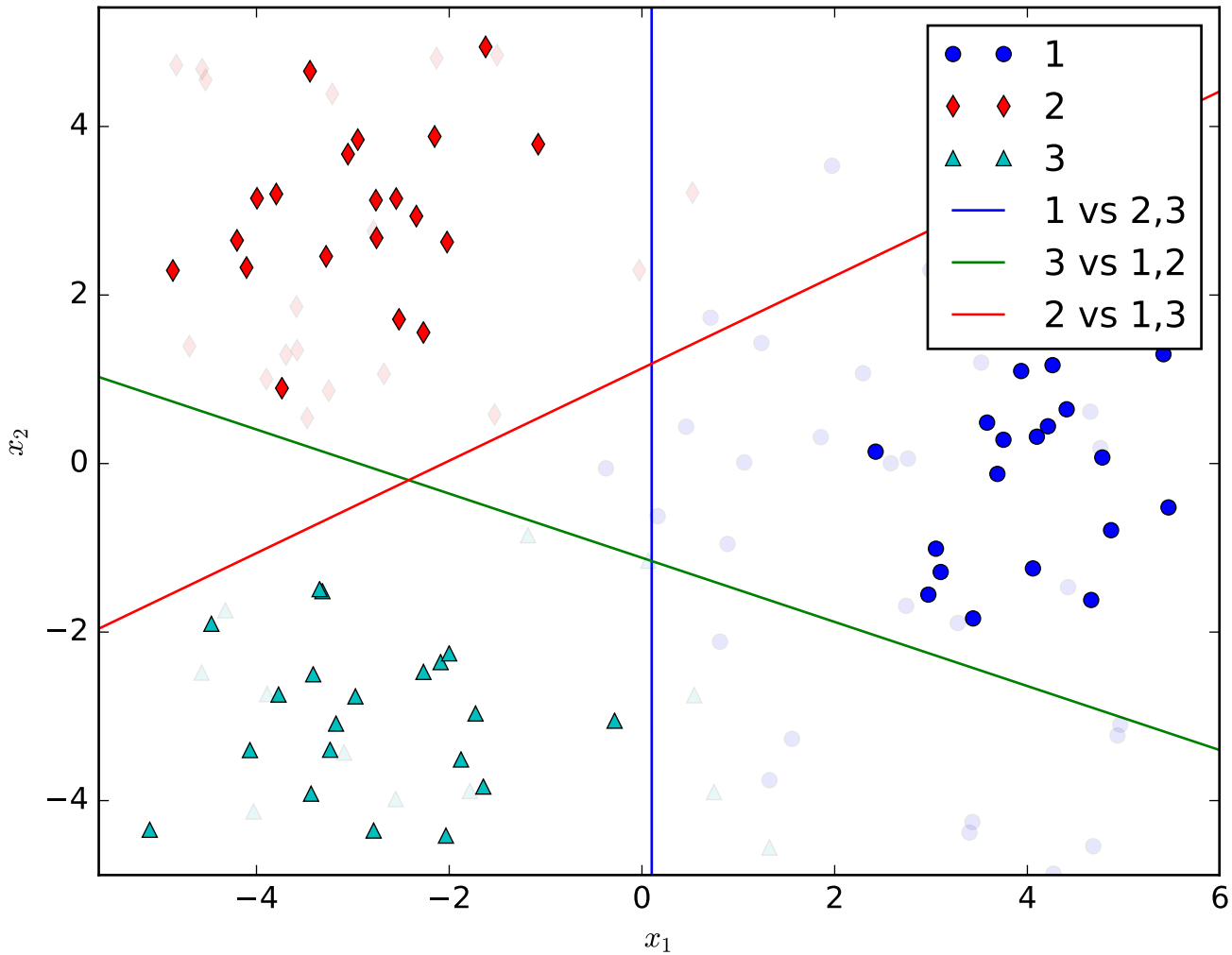
1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
2. predict entries of $\psi(a)$
3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$

Q: how to predict entries of $\psi(a) \in \{-1, 1\}^\ell$?

- reduce to a bunch of binary problems!
- pick your favorite loss function L^{bin} for binary classification
- fit parameter x_i, Y_j by minimizing loss function

$$L(x_i Y_j, a) = \sum_{t=1}^{\ell} L^{\text{bin}}((x_i Y_j)_t, \psi(a)_t)$$

One-vs-All classification



Multiclass classification via learning probabilities

(for concreteness, suppose $\mathcal{Y} = \{1, \ldots, k\}$)

idea 2: learning probabilities

1. learn the probability $\mathbb{P}(a = a' \mid u)$ for every $a' \in \mathcal{F}$
2. predict $\hat{a} = \operatorname{argmax}_{a' \in \mathcal{F}} \mathbb{P}(a = a' \mid u)$
3. u parametrizes probability distribution

Q: how to predict probabilities?

Multiclass classification via learning probabilities

multinomial logit takes a hint from logistic:

- let $u = x_i Y_j$, and suppose

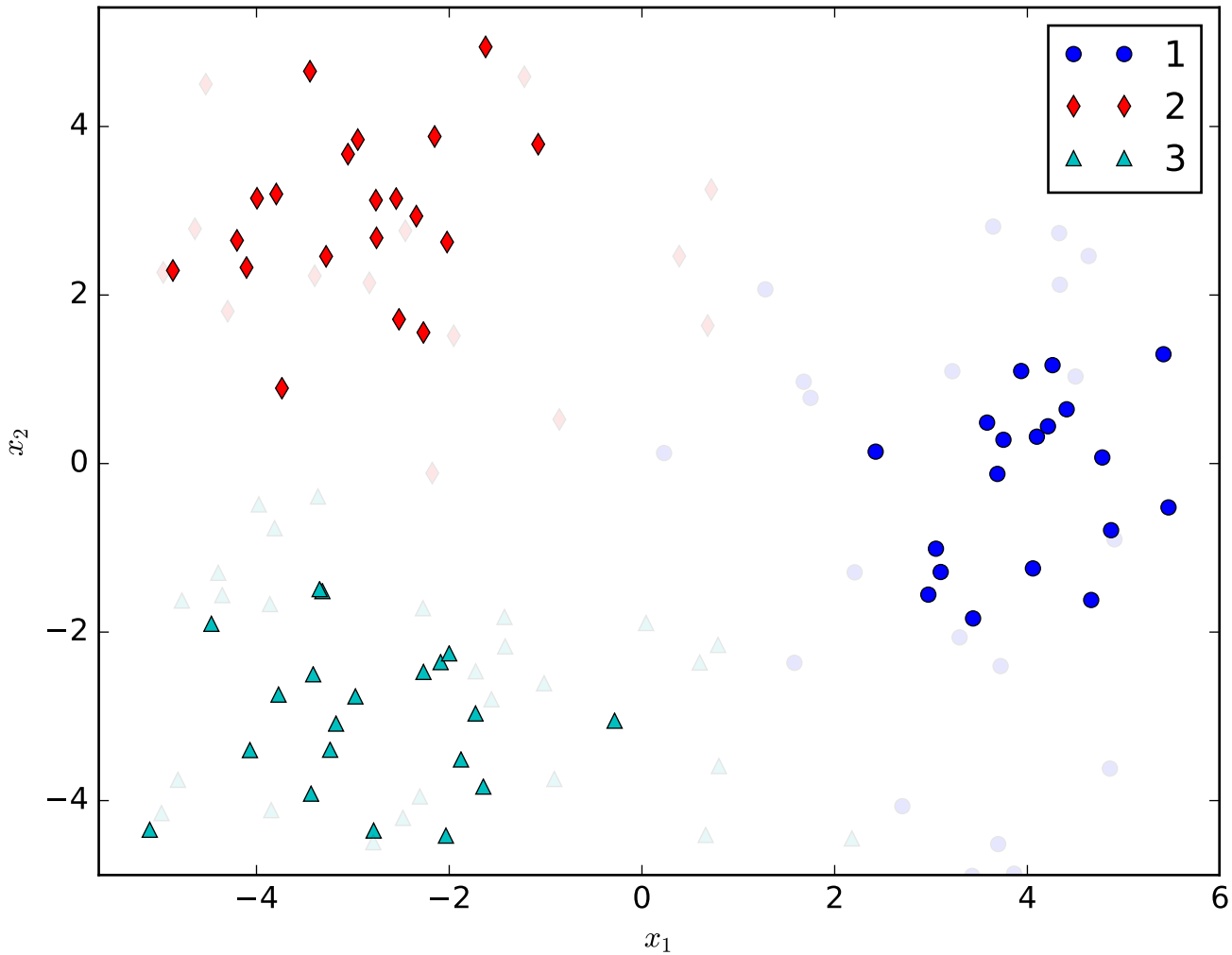
$$\mathbb{P}(a = i \mid u) = \frac{\exp(u_i)}{\sum_{t=1}^{\ell} \exp(u_t)}$$

  (ensures probabilities are positive and sum to 1)

- fit by minimizing negative log likelihood

$$\begin{aligned} L(u, a) &= -\log(\mathbb{P}(a \mid u)) \\ &= -\log\left(\frac{\exp(u_a)}{\sum_{t=1}^{\ell} \exp(u_t)}\right) \end{aligned}$$

Multinomial classification



Ordinal regression

how to predict *ordinal* values?

Ordinal regression

how to predict *ordinal* values?

- *idea 0: regression*
  1. encode $a \in \mathcal{F}$ in $\mathbf{R}$

Ordinal regression

how to predict *ordinal* values?

- *idea 0: regression*
  1. encode $a \in \mathcal{F}$ in $\mathbf{R}$
- *idea 1: classification*
  1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
  2. predict entries of $\psi(a)$
  3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$

Ordinal regression

how to predict *ordinal* values?

- *idea 0: regression*
  1. encode $a \in \mathcal{F}$ in $\mathbf{R}$
- *idea 1: classification*
  1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
  2. predict entries of $\psi(a)$
  3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$
- *idea 2: learning probabilities*
  1. learn the probability $\mathbb{P}(a = a' \mid u)$ for every $a' \in \mathcal{F}$
  2. predict $\hat{a} = \operatorname{argmax}_{a' \in \mathcal{F}} \mathbb{P}(a = a' \mid u)$
  3. u parametrizes probability distribution

Ordinal regression via predicting a vector

idea 1: classification

1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
2. predict entries of $\psi(a)$
3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$

(for concreteness, suppose $\mathcal{F} = \{1, \ldots, \ell\}$)

- how to encode a as a vector?

Ordinal regression via predicting a vector

idea 1: classification

1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
2. predict entries of $\psi(a)$
3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$

(for concreteness, suppose $\mathcal{F} = \{1, \ldots, \ell\}$)

- how to encode a as a vector? how about

$$\psi(a) = (1, \ldots, 1, \overbrace{-1}^{i\text{th entry}}, \ldots, -1) \in \{-1, 1\}^{\ell - 1}$$

Ordinal regression via predicting a vector

idea 1: classification

1. encode $a \in \mathcal{F}$ as a vector $\psi(a)$
2. predict entries of $\psi(a)$
3. each entry of $u = x_i Y_j$ will predict same entry of $\psi(a)$

(for concreteness, suppose $\mathcal{F} = \{1, \ldots, \ell\}$)

- how to encode a as a vector? how about

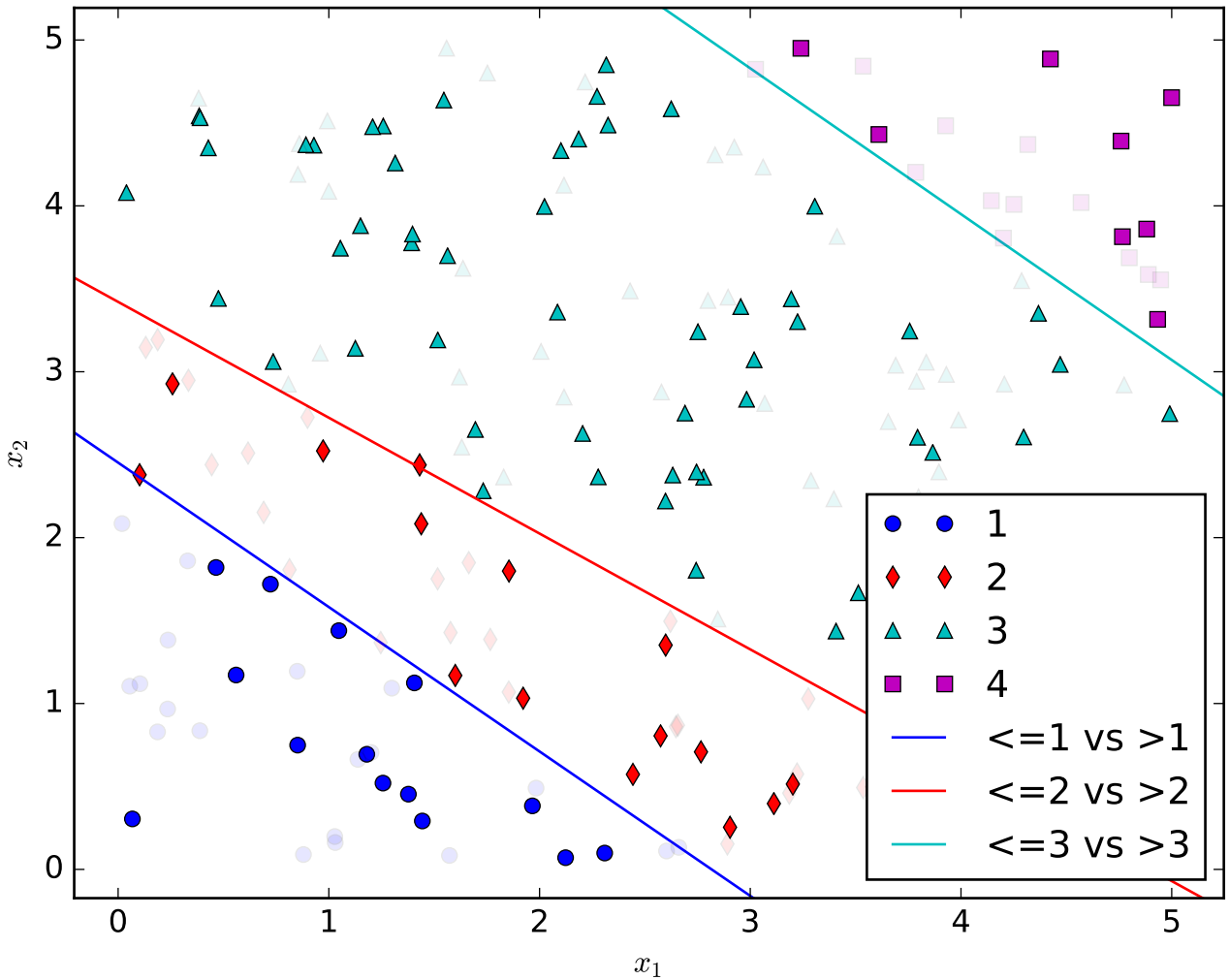
$$\psi(a) = (1, \ldots, 1, \overbrace{-1}^{i\text{th entry}}, \ldots, -1) \in \{-1, 1\}^{\ell-1}$$

- pick your favorite loss function L^{bin} for binary classification
- fit model x_i, Y_j by minimizing loss function

$$L(x_i Y_j, a) = \sum_{t=1}^{\ell-1} L^{\text{bin}}((x_i Y_j)_i, \psi(a)_i)$$

- tth column of Y_j defines a line separating levels $a \leq t$ from

Ordinal regression



Recap: GLRMs

Generalized Low Rank Models are a *framework* that encompasses a bunch of unsupervised learning models

many of these GLRMs have names:

Model	$\ell(\mathbf{y}, \mathbf{z})$	$\mathbf{r}(\mathbf{x})$	$\tilde{\mathbf{r}}(\mathbf{w})$	**reference**
PCA	$(y-z)^2$	0	0	[Pearson 1901]
NNMF	$(y-z)^2$	$\mathbf{1}_+(x)$	$\mathbf{1}_+(w)$	[Lee 1999]
sparse PCA	$(y-z)^2$	$\|x\|_1$	$\|w\|_1$	[D'Aspremont 2004]
sparse coding	$(y-z)^2$	$\|x\|_1$	$\|w\|_2^2$	[Olshausen 1997]
k-means	$(y-z)^2$	$\mathbf{1}_1(x)$	0	[Tropp 2004]
matrix completion	$(y-z)^2$	$\|x\|_2^2$	$\|w\|_2^2$	[Keshavan 2010]
robust PCA	$\|y-z\|$	$\|x\|_2^2$	$\|w\|_2^2$	[Candes 2011]
logistic PCA	$\log(1+\exp(-yz))$	$\|x\|_2^2$	$\|w\|_2^2$	[Collins 2001]
boolean PCA	$(1-yz)_+$	$\|x\|_2^2$	$\|w\|_2^2$	[Srebro 2004]

Conclusion

generalized low rank models

- find structure in data automatically
- can handle huge, heterogeneous data coherently
- transform big messy data into small clean data

papers

GLRMs: `http://arxiv.org/abs/1410.0342`

SAPALM: `http://arxiv.org/abs/1606.02338`

code

`https://github.com/madeleineudell/LowRankModels.jl`