Maximum Margin Matrix Factorization using Smooth Semidefinite Optimization

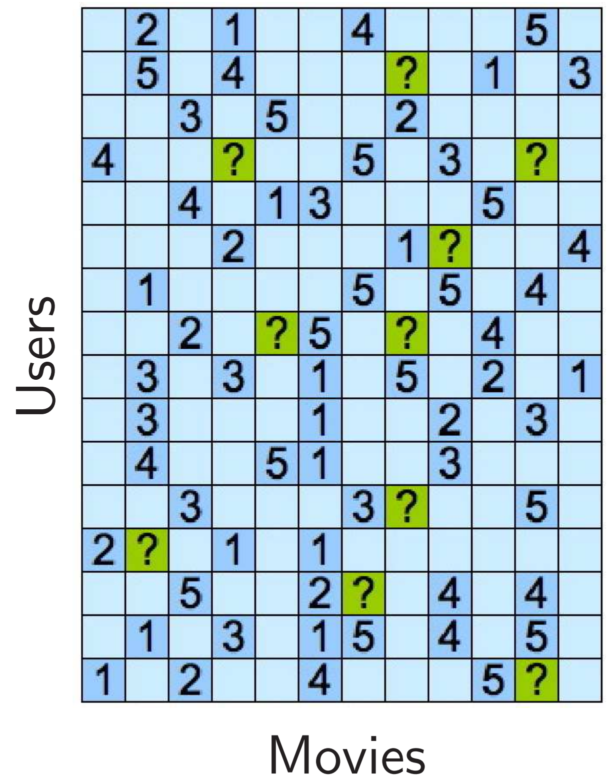
Alexandre d'Aspremont, Nathan Srebro

ORFE, Princeton University & CS, University of Toronto

Thanks to Yurii Nesterov for numerous suggestions!

Introduction

- Users assign *ratings* to a certain number of movies:



- Objective: make recommendations for other movies. . .

Collaborative prediction

- Infer *user preferences* and *movie features* from user ratings.
- We use a linear prediction model:

$$rating_{ij} = u_i^T v_j$$

  where u_i represents user characteristics and v_j movie features.

- This makes collaborative prediction a *matrix factorization* problem
- Overcomplete representation. . .

Collaborative prediction

- **Inputs**: a matrix of ratings $M_{ij} = \{-1, +1\}$ for $(i, j) \in S$, where S is a subset of all possible user/movies combinations.

- We look for a linear model by factorizing $M \in \mathbf{R}^{n \times m}$ as:

$$M = U^T V$$

  where $U \in \mathbf{R}^{n \times k}$ represents user characteristics and $V \in \mathbf{R}^{k \times m}$ movie features.

- *Parsimony*. . . We want k to be as small as possible.

- **Output**: a matrix $X \in \mathbf{R}^{n \times m}$ which is a low-rank approximation of the ratings matrix M.

Least-Squares

- Choose Means Squared Error as measure of discrepancy.
- Suppose S is the full set, our problem becomes:

$$\min_{\{X:\ \mathbf{Rank}(X)=k\}} \|X - M\|^2$$

- This is just a *singular value decomposition* (SVD)...

Problem: Not true when S is not the full set (partial observations). Also, MSE not a good measure of prediction performance...

Soft Margin

$$\text{minimize} \quad \mathbf{Rank}(X) + c \sum_{(i,j)\in S} \max(0, 1 - X_{ij}M_{ij})$$

non-convex and numerically hard. . .

- Relaxation result in Fazel, Hindi & Boyd (2001): replace $\mathbf{Rank}(X)$ by its convex envelope on the spectahedron to solve:

  $$\text{minimize} \quad \|X\|_* + c \sum_{(i,j)\in S} \max(0, 1 - X_{ij}M_{ij})$$

  where $\|X\|_*$ is the *nuclear norm*, *i.e.* sum of the singular values of X.

- Srebro (2004): This relaxation also corresponds to multiple large margin SVM classifications.

Soft Margin

- The dual of this program:

$$\begin{array}{ll} \text{maximize} & \sum_{ij} Y_{ij} \\ \text{subject to} & \|Y \odot M\|_2 \leq 1 \\ & 0 \leq Y_{ij} \leq c \end{array}$$

in the variable $Y \in \mathbf{R}^{n \times m}$, where $Y \odot M$ is the Schur (componentwise) product of Y and M and $\|Y\|_2$ the largest singular value of Y.

- This problem is *sparse*: $Y_{ij}^* = c$ for $(i, j) \in S^c$

Semidefinite Program

- How do we solve it?
- Rewrite the dual

$$\begin{array}{ll} \text{maximize} & \sum_{ij} Y_{ij} \\ \text{subject to} & \|Y \odot M\|_2 \leq 1 \\ & 0 \leq Y_{ij} \leq c \end{array}$$

as:

$$\begin{array}{ll} \text{maximize} & \sum_{ij} Y_{ij} \\ \text{subject to} & \left[\begin{array}{cc} I & -(Y \odot M) \\ -(Y \odot M)^T & I \end{array}\right] \succeq 0 \\ & 0 \leq Y_{ij} \leq c \end{array}$$

which is a sparse *semidefinite program* in $Y \in \mathbf{R}^{n \times m}$.

Complexity

Complexity?

- Small subset S: the dual in Y is sparse, primal (in ratings X) is *dense*.
- Interior point solvers work fine for problem sizes up to 400...
- We need to solve much larger instances.
- High precision is not necessary. . .

Smoothing Technique

- Solution, formulate this as a saddle problem using binary search:

$$\begin{array}{ll} \text{minimize} & \lambda^{\max}\left(\begin{bmatrix} I & -(Y \odot M) \\ -(Y \odot M)^T & I \end{bmatrix}\right) \\ \text{subject to} & \sum_{ij} Y_{ij} = t \\ & 0 \leq Y_{ij} \leq c \end{array}$$

  for some $t > 0$.

- Use the smoothing technique in Nesterov (2005): first-order algorithm with optimal complexity of $O(1/\epsilon)$.

- *Homogeneity* means we also get a solution to:

$$\begin{array}{ll} \text{maximize} & \sum_{ij} Y_{ij} \\ \text{subject to} & \|Y \odot M\|_2 \leq 1 \\ & 0 \leq Y_{ij} \leq c^* \end{array}$$

Nesterov's method

Assuming problem has a particular min-max structure:

- *Regularization*. Add strongly convex penalty inside the min-max representation to produce an ϵ-approximation of f with Lipschitz continuous gradient (generalized Moreau-Yosida regularization step, see Lemaréchal & Sagastizábal (1997) for example).

- *Optimal first order minimization*. Use optimal first order scheme for Lipschitz continuous functions detailed in Nesterov (1983) to the solve the regularized problem.

Caveat: Only efficient if the subproblems involved in these steps can be solved explicitly or very efficiently. . . Change of *granularity*: larger number of cheaper iterations.

Regularization

Replace $\lambda^{\max}(X)$ by

$$f_\mu(X) = \mu \log\left(\sum_{i=1}^{k} e^{\frac{\lambda_i}{\mu}}\right).$$

For a good choice of μ:

- $f_\mu(X)$ is an ϵ-approximation of f.
- $f_\mu(X)$ has a *Lipschitz continuous gradient* with constant $L = O(1/\epsilon)$.

First-Order Minimization

The minimization algorithm in Nesterov (1983) then involves the following steps:

Choose $\epsilon > 0$ and set $X_0 = \beta I_n$, **For** $k = 0, \ldots, N(\epsilon)$ **do**

1. Compute f_μ and ∇f_μ
2. Find $Y_k = \arg\min_Y \{\mathbf{Tr}(\nabla f_\epsilon(X_k)(Y - X_k)) + \frac{1}{2} L_\epsilon \|Y - X_k\|_F^2 \ : \ Y \in \mathcal{Q}_1\}$.
3. Find $Z_k = \arg\min_X \left\{ L_\epsilon \beta^2 \|X\| + \sum_{i=0}^{k} \frac{i+1}{2} \mathbf{Tr}(\nabla f_\epsilon(X_i)(X - X_i)) \ : \ X \in \mathcal{Q}_1 \right\}$.
4. Update $X_k = \frac{2}{k+3} Z_k + \frac{k+1}{k+3} Y_k$.

Numerical Cost

At each iteration:

- **Step 1**: computes f and ∇f and is a (full) eigenvalue decomposition (in fact SVD here, because of structure)

- **Step 2 & 3**: involve projections on a the set:

$$\mathcal{Q}_1 = \{Y : \textstyle\sum_{ij} Y_{ij} = t,\ 0 \leq Y_{ij} \leq c\}$$

  and are numerically easy.

Complexity, $i.e.$ maximum number of iterations to reach absolute precision ϵ

$$\frac{4\sqrt{m+n+mnc^2}}{\epsilon}$$

with each iteration (roughly) costing $O(mn^2 + n^3)$.

Numerical Results

- No movies to recommend but. . .
- Compare CPU time and memory usage for CSDP and smooth optimization code.
- Both codes are C/MEX with calls to (dense) LAPACK/BLAS.

Numerical Results

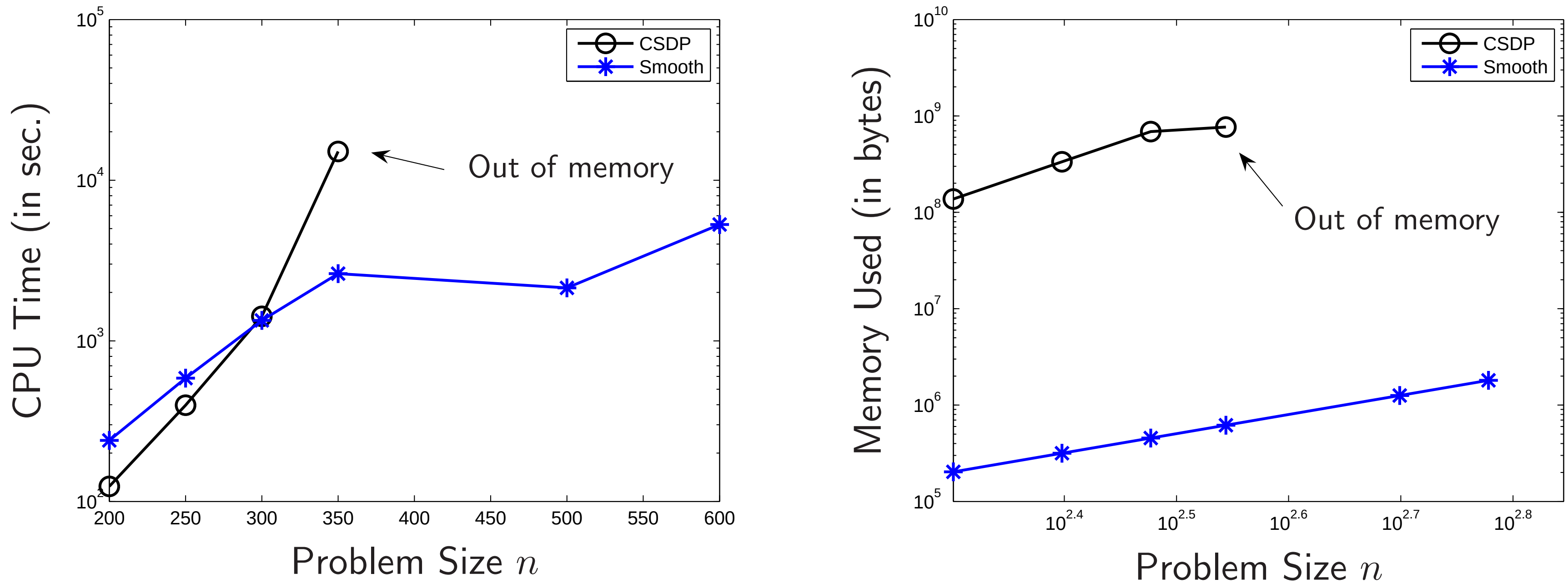


Figure 1: CPU time and memory usage versus n.

Numerical Results

Large scale tests on a 3,06 Ghz CPU with 2Gb RAM:

n	1% observed	10% observed	50% observed
100	2 sec	3 sec	10 sec
178	2 sec	18 sec	35 sec
316	19 sec	2:34 min	2:41 min
562	3:27 min	3:37 min	19:11 min
1000	34:35 min	41:15 min	1:35:28 hours
1778	5:44:07 hours	6:40:06 hours	19:09:49 hours
3162	57:23:09 hours	67:35:34 hours	62:12:21 hours

References

Fazel, M., Hindi, H. & Boyd, S. (2001), 'A rank minimization heuristic with application to minimum order system approximation', *Proceedings American Control Conference* **6**, 4734–4739.

Lemaréchal, C. & Sagastizábal, C. (1997), 'Practical aspects of the Moreau-Yosida regularization: theoretical preliminaries', *SIAM Journal on Optimization* **7**(2), 367–385.

Nesterov, Y. (1983), 'A method of solving a convex programming problem with convergence rate $O(1/k^2)$', *Soviet Mathematics Doklady* **27**(2), 372–376.

Nesterov, Y. (2005), 'Smooth minimization of nonsmooth functions', *Mathematical Programming, Series A* **103**, 127–152.

Srebro, N. (2004), Learning with Matrix Factorization, PhD thesis, Massachusetts Institute of Technology.