Coordinate-wise Power Method

Qi Lei, Kai Zhong[1], and Inderjit S. Dhillon[1,2]

[1]Institute for Computational Sciences and Engineering, [2]Department of Computer Science, The University of Texas at Austin

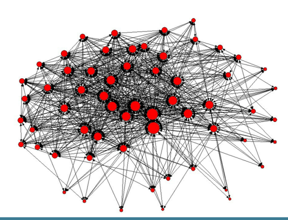
Motivation

Goal: Given a matrix A, we seek to compute its dominant eigenvector $\boldsymbol{v}_1$:

$$\boldsymbol{v}_1 = \underset{\|\boldsymbol{x}\|=1}{\text{argmax}}\, \boldsymbol{x}^T A^T A \boldsymbol{x} \quad (1)$$

Computing the dominant eigenvector of a given matrix/graph is meaningful for:

- Graph Centrality/PageRank
- Sparse PCA
- Spectral Clustering



The classic power method is still powerful in the sense of:

- Simplicity
- Small memory footprint
- Stable: being resistent to noise

We propose two coordinate-wise versions of the power method, from an optimization viewpoint.

A brief review of the Power Method

- Given a matrix A, let its two dominant eigenvalues be λ_1, λ_2, and its dominant eigenvector is $\boldsymbol{v}$. Power iteration conducts:

$$\boldsymbol{x}^{(l+1)} \leftarrow \text{normalize}(A\boldsymbol{x}^{(l)}) \quad (2)$$

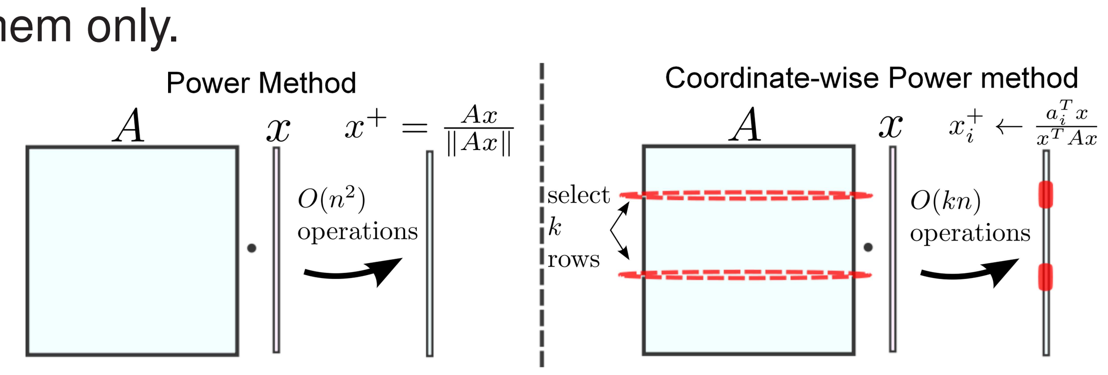
- This is inefficient since some coordinates converge faster than others, e.g.,

$$A = \begin{bmatrix} 2 & 0 & 1 \\ 0 & 3 & 0 \\ 1 & 0 & 2 \end{bmatrix}, \boldsymbol{x}: \begin{bmatrix} 0.71 \\ 0.71 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0.53 \\ 0.80 \\ 0.27 \end{bmatrix} \rightarrow \begin{bmatrix} 0.45 \\ 0.81 \\ 0.36 \end{bmatrix} \rightarrow \begin{bmatrix} 0.42 \\ 0.82 \\ 0.39 \end{bmatrix} \rightarrow \begin{bmatrix} 0.41 \\ 0.82 \\ 0.40 \end{bmatrix}$$

  Therefore we want to select and update important coordinates only.
- *One key question: how to select the coordinates?*
- *Another key problem: how to choose these coordinates without too much overhead?*

Algorithm of Coordinate-wise Power Method (CPM)

MAIN IDEA: Choose k coordinates with the most potential change and update them only.



1. Define auxiliary parameters:
   1.1 $\boldsymbol{z} = A\boldsymbol{x}$ maintained for algorithm efficiency.
   1.2 Coordinate selection criterion: $\boldsymbol{c} = \frac{\boldsymbol{z}}{\boldsymbol{x}^T\boldsymbol{z}} - \boldsymbol{x}$
2. Coordinate selection: let Ω be a set containing k coordinates of $\boldsymbol{c}$ with the largest magnitude.
3. Update the new iterate $\boldsymbol{x}^+$:

$$y_i \leftarrow \begin{cases} \frac{z_i}{\boldsymbol{z}^T\boldsymbol{z}}, & i \in \Omega \\ x_i & i \notin \Omega \end{cases}. \qquad \boldsymbol{x}^+ \leftarrow \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|}.$$

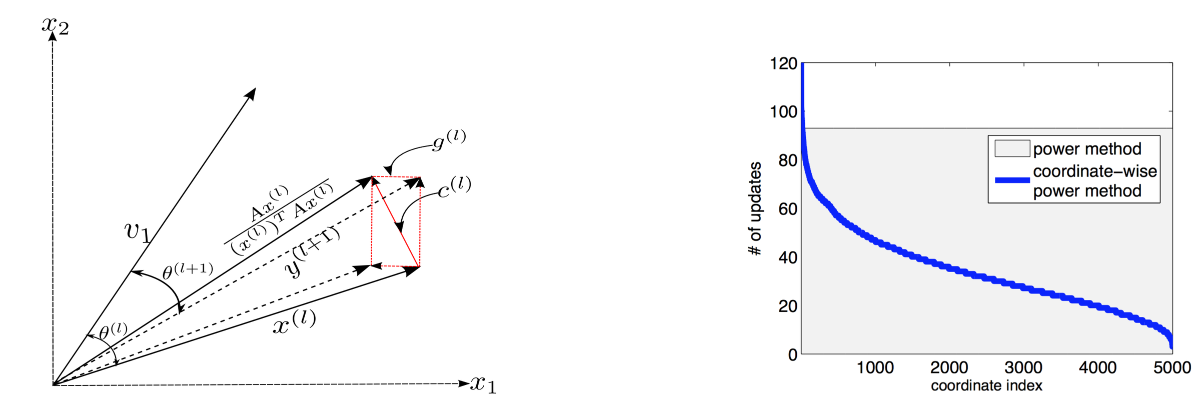
4. Update the auxiliary parameters with the k changes in $\boldsymbol{x}$ with $O(kn)$ operations.

$$\boldsymbol{z}^+ \leftarrow \boldsymbol{z} + A_{:,\Omega}^T(\boldsymbol{y}_\Omega - \boldsymbol{x}_\Omega). \qquad \boldsymbol{z}^+ \leftarrow \boldsymbol{z}^+/\|\boldsymbol{y}\|.$$

$$\boldsymbol{c}^+ = \frac{\boldsymbol{z}^+}{(\boldsymbol{x}^+)^T\boldsymbol{z}^+} - \boldsymbol{x}^+.$$

5. Repeat $2-4$.

Illustration on how CPM works



(a) Illustration on one update of CPM (b) Number of updates of each coordinate

- (a) One iteration in CPM suffices similar result with the Power Method, but with less operations.
- (b) The unevenness of updates suggests that selecting important coordinates saves many useless updates in the Power method.

Relation to Optimization & Coordinate selection rules

- Power method $\Longleftrightarrow$ Alternating minimization for Rank-1 matrix approximation:

$$\underset{\boldsymbol{x}\in\mathbb{R},\boldsymbol{y}\in\mathbb{R}}{\text{argmin}} \left\{ f(\boldsymbol{x},\boldsymbol{y}) = \|A - \boldsymbol{x}\boldsymbol{y}^T\|_F^2 \right\} \quad (3)$$

- Updating rule for Alternation minimization:
  $\boldsymbol{x} \leftarrow \text{argmin}_\alpha f(\alpha, \boldsymbol{y}) = \frac{A\boldsymbol{y}}{\|\boldsymbol{y}\|^2}$, $\boldsymbol{y} \leftarrow \text{argmin}_\beta f(\boldsymbol{x}, \beta) = \frac{A^T\boldsymbol{x}}{\|\boldsymbol{x}\|^2}$,

- The following coordinate selecting rules for (3) are equivalent:
  1. largest coordinate value change, denoted as $|\delta x_i|$;
  2. largest partial gradient (Gauss-Southwell rule), $|\nabla_i f(\boldsymbol{x})|$
  3. largest function value decrease, $|f(\boldsymbol{x} + \delta x_i \boldsymbol{e}_i) - f(\boldsymbol{x})|$

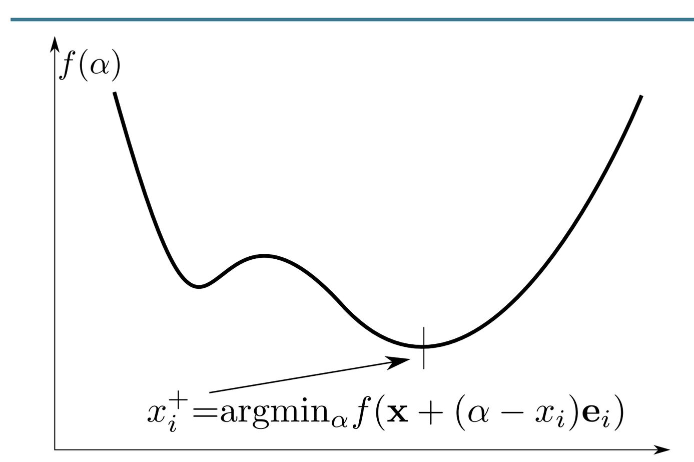
- A simple alternation of the objective function for Rank-1 matrix approximation for symmetric matrices:

Algorithm	Compared to	Objective function
Power Method	Alternating Minimization	$f(\boldsymbol{x},\boldsymbol{y}) = \|A - \boldsymbol{x}\boldsymbol{y}^T\|_F^2$
CPM	Greedy Coordinate Descent	$f(\boldsymbol{x},\boldsymbol{y}) = \|A - \boldsymbol{x}\boldsymbol{y}^T\|_F^2$
SGCD	Greedy Coordinate Descent	$f(\boldsymbol{x}) = \|A - \boldsymbol{x}\boldsymbol{x}^T\|_F^2$

Algorithm of Symmetric Greedy Coordinate Descent(SGCD)

- We also propose a new method we call Symetric Greedy Coordinate Descent (SGCD) for symmetric matrices.
- **MAIN IDEA**: use greedy and exact coordinate descent on $f(\boldsymbol{x}) = \|A - \boldsymbol{x}\boldsymbol{x}^T\|_F^2$.
- Main differences:
  1. A different coordinate selection criterion: $\boldsymbol{c} = \frac{A\boldsymbol{x}}{\|\boldsymbol{x}\|^2} - \boldsymbol{x}$ (parallel to the gradient of $f(\boldsymbol{x})$)
  2. A different update rule of $\boldsymbol{x}^+$ in Ω

$$x_i^+ = \begin{cases} \text{argmin}_\alpha f(\boldsymbol{x} + (\alpha - x_i)\boldsymbol{e}_i), & \text{if } i \in \Omega, \\ x_i, & \text{if } i \notin \Omega. \end{cases}$$



- Exact update:
- Solve $x_i^+ = \alpha$ such that $\nabla f(\boldsymbol{x} + (\alpha - x_i)\boldsymbol{e}_i) = \alpha^3 + p\alpha + q = 0$, where $p = \|\boldsymbol{x}\|^2 - x_i^2 - a_{ii}$, $q = -\boldsymbol{a}_i^T\boldsymbol{x} + a_{ii}x_i$.
- $O(n)$ operations

Convergence guarantees for CPM and SGCD

- For Coordinate-wise Power Method (CPM), we prove global linear convergence for any positive semidefinite matrix A.

Theorem 1

Convergence rate: require $T = O(\frac{\lambda_1}{\lambda_1 - \lambda_2}\log(\frac{1}{\epsilon}))$ to achieve $\tan\theta_{\boldsymbol{x}^{(l)},\boldsymbol{v}_1} \leq \epsilon$ provided the "noise rate" $\frac{\|\boldsymbol{c}_{[n]-\Omega}\|}{\|\boldsymbol{c}\|} \lesssim \frac{\lambda_1 - \lambda_2}{\lambda_1}$.

- For the method of Symmetric Greedy Coordinate Descent (SGCD), we prove local linear convergence:

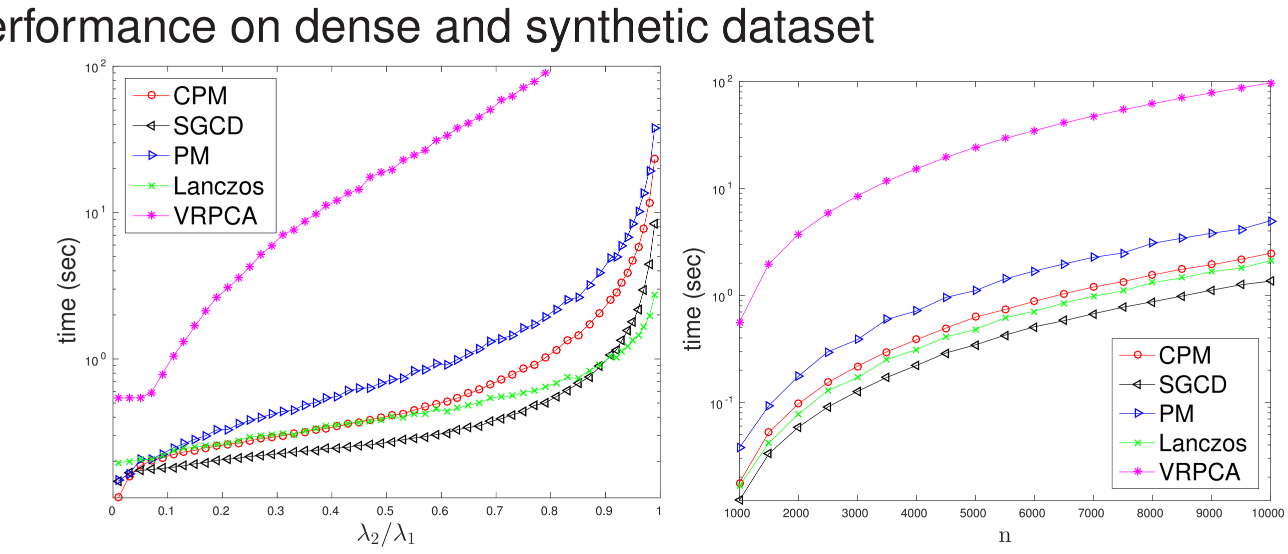
Theorem 2

Convergence rate: require $T = O(\frac{\lambda_1}{\lambda_1 - \lambda_2}\log(\frac{1}{\epsilon}))$ to achieve $f(\boldsymbol{x}^{(l)}) - f(\boldsymbol{v}) \leq \epsilon$ provided $\boldsymbol{x}^{(0)}$ sufficiently close to $\boldsymbol{v}_1$: $\|\boldsymbol{x}^{(0)} - \boldsymbol{v}_1\| \lesssim \frac{\lambda_1 - \lambda_2}{\sqrt{\lambda_1}}$

Experimental Results

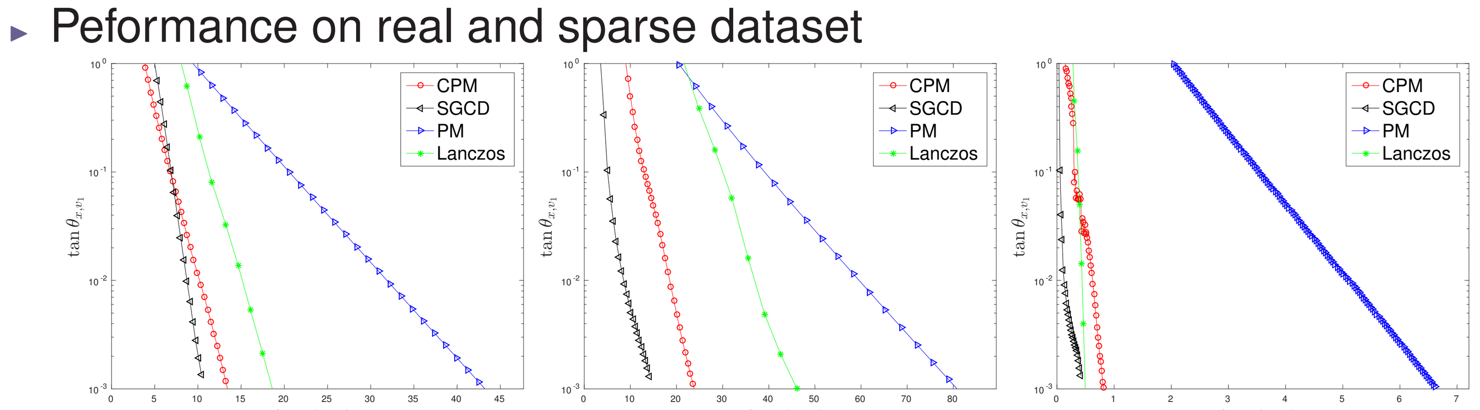
- Scalability experiments between our methods compared to power method, Lanczos method and VRPCA (Ohad Shamir, 2015) conducted with C++ with Eigen library on one machine with 16G memory:

- Performance on dense and synthetic dataset



Convergence time vs $\frac{\lambda_2}{\lambda_1}(A)$ Convergence time vs. n

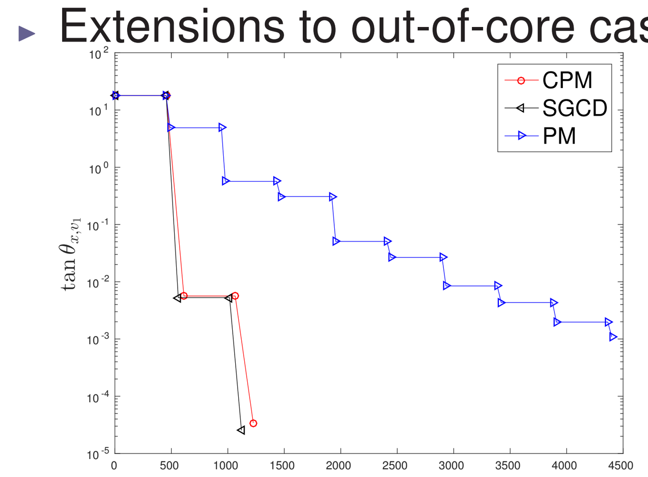
- Peformance on real and sparse dataset



Performance on LiveJournal Performance on com-Orkut Performance on web-Stanford

Dataset	LiveJournal	com-Orkut	web-Stanford
# nodes	4,847,571	3,072,626	281,903
# nonzero	86,220,856	234,370,166	3,985,272

- Extensions to out-of-core case:



- Existing methods can't be easily applied to out-of-core dataset.
- Our methods indicate that updating only k coordinates of iterate $\boldsymbol{x}$ still enhance the target direction
- we can choose a k such that k rows of data fit in memory and then fully update the corresponding coordinates

Mail: {leiqi,zhongkai}@ices.utexas.edu, inderjit@cs.utexas.edu