# Preconditioned ADMM on (Convolutional) Sparse Coding

**Yao Li**

August 6, 2019

Mentor: *Youzuo Lin, Brendt Wohlberg*

# Problem Description

- (Convolutional) Sparse Coding considers a (convolution) dictionary to represent a signal in the following form:

$$Dx\left(=\sum_m d_m * x_m\right) \approx s.$$

- It contains problems like (Convolutional) Basis Pursuit DeNoising, (convolutional) elestic net, mask decoupling, etc.
- Most of problems can be formulated as

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & f(x) + g(y) \\ \text{subject to} \quad & Ax = y \end{aligned}$$

$f$ and $g$ are all proper, closed and convex.

# ADMM for Separable Problems

- Alternating Direction Method of Multipliers (ADMM) is good at solving problem with separable structure.
- It considers augmented Lagrangian function

$$L_\rho(x, y, u) := f(x) + g(y) + \rho u^T(Ax - y) + \frac{\rho}{2}\|Ax - y\|_2^2$$

- It follows the iteration:

$$\begin{cases} x^{k+1} = \arg\min_x L_\rho(x, y^k, u^k), \\ y^{k+1} = \arg\min_y L_\rho(x^{k+1}, y, u^k), \\ u^{k+1} = u^k + (Ax^{k+1} - y^{k+1}). \end{cases}$$

- $e_p^{k+1} = \|Ax^{k+1} - y^{k+1}\|, e_d^{k+1} = \|\rho A^T(y^{k+1} - y^k)\|$ are primal and dual residuals as stopping criteria.

# Motivation of Preconditioning

- Theoretically, ADMM converges at rate of $O(1/k)$ and if $f$ is strongly convex w.r.t $H \succ 0$ and Lipschitz smooth w.r.t $M \succ 0$, it converges at rate of $O\Big((\frac{\sqrt{\tau}}{1+\sqrt{\tau}})^k\Big)$, where

$$\tau := \frac{\lambda_{\max}(AH^{-1}A^T)}{\lambda_{\min}(AM^{-1}A^T)}$$

- In particular, if $M = H = I$, $\tau = \kappa(AA^T)$. Empirically, the smaller $\tau$ is, the better the perfomance of ADMM will get even in general cases.
- The preconditioning is to find a diagonal positive matrix $E$ so that $\frac{\lambda_{\max}(EAH^{-1}A^TE)}{\lambda_{\min}(EAM^{-1}A^TE)}$ is minimized and to solve the equivalent problem

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & f(x) + g(y) \\ \text{subject to} \quad & EAx = Ey \end{aligned}$$

# Matrix Equilibration

- Matrix Equilibration is trying to minimize $\kappa(A)$ for some matrix $A$ by using two diagonal positive matrices $D$ and $E$ such that $\kappa(DAE) < \kappa(A)$.
- Theorems shown in [6] guarantee that if $\|(DAE)_{:,j}\| = \alpha$ and $\|(DAE)_{i,:}\| = \beta$ for some $\alpha, \beta$, the upper bound of $\kappa(DAE)$ is minimized.
- I consider two equilibration algorithms mentioned in [7]: Sinkhorn-Knopp algorithm and Ruiz algorithm and a regularized matrix-free algorithm proposed in [2].

# Numerical Experiments

- I compare the effectiveness of three algorithms on matrices selected from *The SuiteSparse Matrix Collection* [1].

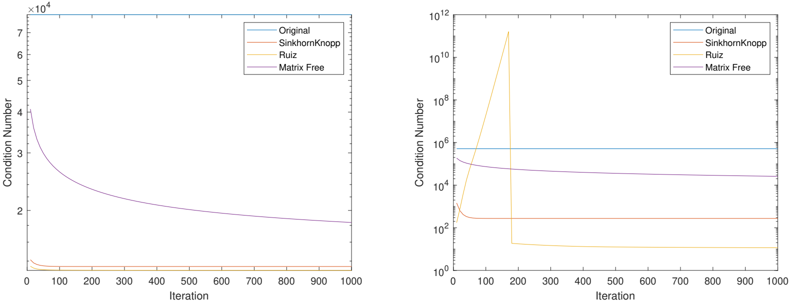


Figure: Left: 628 by 628 sparse matrix indexed as 2561 with condition number 7.9443e+4. Right: 625 by 1506 sparse matrix indexed as 653 with condition number 5.1327e+5.

# Precondition on BPDN

- For non-convolutional type problem, e.g., BPDN

$$\min_{x} \frac{1}{2}\|Dx - s\|_2^2 + \lambda\|x\|_1,$$

I use Ruiz algorithm to generate precondition matrix $E$ for $D^T D$, i.e., reducing $\kappa(ED^T DE)$, and solve the equivalent problem

$$\min_{x,y} \frac{1}{2}\|Dx - s\|_2^2 + \lambda\|y\|_1$$
$$\text{s.t. } E^{-1}x = E^{-1}y$$

# A Simulation of BPDN

$D \in \mathbb{R}^{500\times 800}$ is generated randomly with $\kappa(D^T D) =$5.9101e+18 and after precondition, $\kappa(ED^T DE) \approx \frac{1}{3}\kappa(D^T D)$.

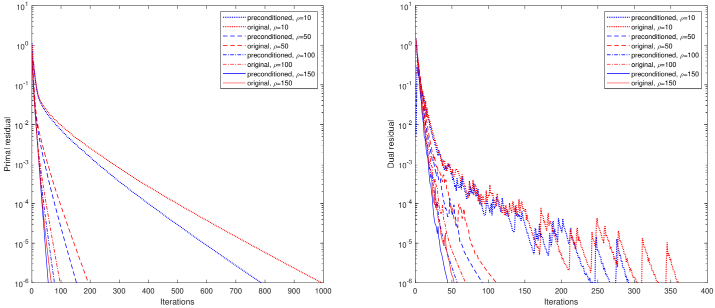


Figure: Residual comparison for different penalty parameters. LEFT: Primal residual, RIGHT: Dual residual.

# Comparison of the Speedup

I test ADMM on a wide range of penalty parameter $\rho \in [1e-2, 5e2]$ and restrict $\rho$ on an optimal range $[0.5, 5]$ to test the number of steps both algorithms require to attain tolerance $5e-4$.

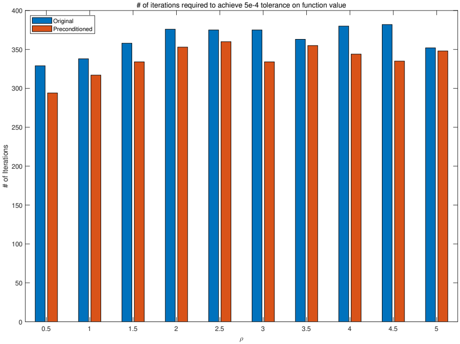


Figure: The number of iterations required to attain tolerance.

# Precondition on CBPDN

- Set $E_1 = uI_M$ and $E_2 = v \otimes I_M \in \mathbb{R}^{mM \times mM}$ if each $D_i \in \mathbb{R}^{M \times M}$.
- Use matrix-free method to generate $E_1$ and $E_2$, and to solve the equivalent problem

$$\min_{x_i, y_i} \frac{1}{2} \| \sum_{i=1}^{m} D_i x_i - s \|_2^2 + \lambda \| y \|_1$$
$$\text{s.t. } E_2^{-1} x = E_2^{-1} y,$$

where $x = [x_1^T, \cdots, x_m^T]^T$, $y = [y_1^T, \cdots, y_m^T]^T \in \mathbb{R}^{mM}$.

# Speedup on CBPDN

I pick the $8 \times 8 \times 96$ dictionary set in SPORCO [8] and the image is 'lena.png' in grey scale.

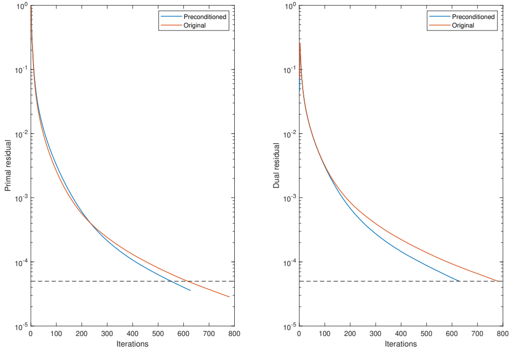


Figure: Residual comparison. LEFT: Primal residual, RIGHT: Dual residual.

# Speedup on CBPDN

- The tolerance is set to $5e-5$, the vanilla ADMM iterates 781 steps to stop, while the preconditioned ADMM only needs 628 steps.
- The actually runtime is 734.8294s for the vanilla ADMM and 537.2272s for preconditioned one.
- The precondition will not increase the computational complexity at each iteration.

# BPDN with Mask Decoupling

Compared with plain Basis Pursuit Denoising, BPDN with mask decoupling is trying to deal with boundary issue using mask $W \in \mathbb{R}^m$, it is formulated as (1-d signal for example):

$$\min_x \frac{1}{2}\|W \otimes Dx - s\|^2 + \lambda\|x\|_1 \qquad \text{(mask decoupling)}$$

The implementation of ADMM follows:

$$\min_{x,y_0,y_1} \underbrace{\frac{1}{2}\|W \otimes y_1 - s\|^2 + \lambda\|y_0\|_1}_{f(y_0,y_1)} + \underbrace{0(x)}_{g(x)} \quad \text{s.t.} \begin{bmatrix} I \\ D \end{bmatrix} x = \begin{bmatrix} y_0 \\ y_1 \end{bmatrix},$$

where $D$ is dictionary, convolutional one or non-convolutional one.

# Precondition on BPDNMD

- The precondition matrix $E$ is constructed such that

$$\kappa\left(E\begin{bmatrix} I \\ D \end{bmatrix}\begin{bmatrix} I & D^T \end{bmatrix}E\right)$$

is minimized and solve the equivalent problem:

$$\min_{x,y_0,y_1} \frac{1}{2}\|W \otimes y_1 - s\|^2 + \lambda\|y_0\|_1 \quad \text{s.t.} E\begin{bmatrix} I \\ D \end{bmatrix} x = E\begin{bmatrix} y_0 \\ y_1 \end{bmatrix}.$$

# A Simulation on BPDNMD

- $D \in \mathbb{R}^{510\times 800}$ is randomly generated and on 1-d signal.
- For observed signal $s \in \mathbb{R}^{500}$, I add a 10-d zero vector to the end, so $s^{ob} \in \mathbb{R}^{510}$. The mask $w \in \mathbb{R}^{510}$ marks the last 10 entries of $s^{ob}$ as 0.
- $\lambda = 0.01$.
- The stopping criterion is set to be $\|x - x_{optimal}\| < 1\text{e-}3$.

# A Simulation on BPDNMD

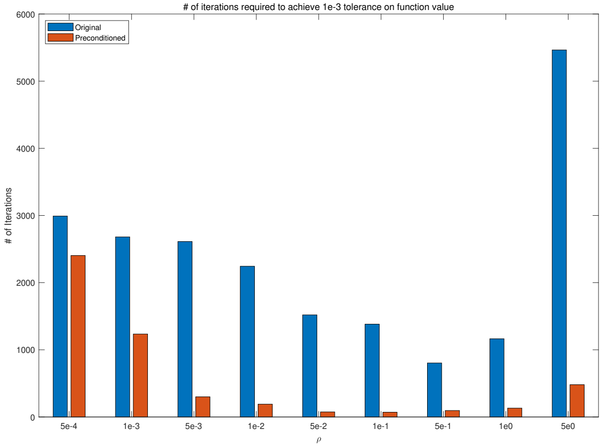


Figure: Iterations required to attain $\|x - x_{optimal}\| < 1e - 3$.

# Convolutional BPDN with Mask Decoupling

- CBPDNMD is similar to BPDNMD except that the dictionary $D$ is replace by a set of convolutional type dictionaries, i.e.,

$$Dx = \sum_{i=1}^{m} D_i x_i$$

- Trying to find a $\gamma$ such that

$$\kappa\left( \begin{bmatrix} I \\ \gamma D \end{bmatrix} \begin{bmatrix} I & \gamma D^T \end{bmatrix} \right)$$

is minimized.

# Speedup on CBPDNMD

I pick the $8 \times 8 \times 96$ dictionary set in SPORCO [8] and the image is 'lena.png' in grey scale.

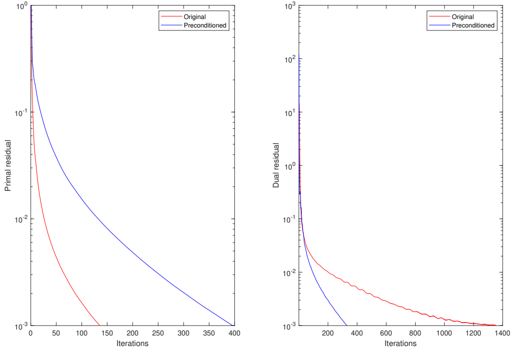


Figure: Residual comparison. LEFT: Primal residual, RIGHT: Dual residual.

# Speedup on CBPDNMD

- The dual residual dominates the convergence of algorithm. The precondition balances the decreasing rate of two residuals to speed up ADMM.
- The tolerance is set to $1e-3$, the vanilla ADMM iterates 1355 steps to stop, while the preconditioned ADMM only needs 397 steps.
- The actually runtime is $4.4561e+03$s for the vanilla ADMM and $1.2520e+03$s for preconditioned one.

# Conclusion

- Precondition as a heuristic method is effective on some (convolutional) sparse coding problems.
- Its performance is comparable with the prevailing residual balancing method on mask decoupling problems.
- In cases where the residual balancing method fails, precondition method is a good complement.

# Reference I


- Timothy A. Davis and Yifan Hu.
  The university of florida sparse matrix collection.
  *ACM Trans. Math. Softw.*, 38(1):1:1–1:25, December 2011.
- Steven Diamond and Stephen Boyd.
  Stochastic Matrix-Free Equilibration.
  *Journal of Optimization Theory and Applications*, 172(2):436–454, feb 2017.
- Pontus Giselsson and Stephen Boyd.
  Linear Convergence and Metric Selection for Douglas-Rachford Splitting and ADMM.
  *IEEE Transactions on Automatic Control*, 62(2):532–544, feb 2017.

# Reference II


Daniel Ruiz.
A scaling algorithm to equilibrate both rows and columns norms in matrices.
Technical report, CM-P00040415, 2001.

Richard Sinkhorn.
A relationship between arbitrary positive matrices and doubly stochastic matrices.
*The annals of mathematical statistics*, 35(2):876–879, 1964.

AVD Sluis.
Condition numbers and equilibration of matrices.
*Numerische Mathematik*, 14(1):14–23, 1969.

# Reference III


- Reza Takapoui and Hamid Javadi.
  Preconditioning via Diagonal Scaling.
  oct 2016.
- Brendt Wohlberg.
  SParse Optimization Research COde (SPORCO).
  Software library available from
  http://purl.org/brendt/software/sporco, 2017.

Thank You!