Efficient Learning of Nonconvex Sparse and Low-rank Models

姚权铭

http://www.cse.ust.hk/~qyaoaa/
Covered Papers (mine)

• Large-Scale Low-Rank Matrix Learning with Nonconvex Regularizers. TPAMI. 2018.
• Efficient Learning with Nonconvex Regularizers by Nonconvexity Redistribution. JMLR. 2018.
• Efficient Learning with a Family of Nonconvex Regularizers by Redistributing Nonconvexity. ICML. 2016.
• Fast Low-Rank Matrix Learning with Nonconvex Regularization. ICDM. 2015.
Related Papers in Computer Vision (classical)

- Image denoising via sparse and redundant representations over learned dictionaries. TIP. 2006.
- Robust face recognition via sparse representation. TPAMI. 2009
- Robust subspace segmentation by low-rank representation. ICML. 2010.
- Robust video denoising using low rank matrix completion. CVPR. 2010.
- Fast and accurate matrix completion via truncated nuclear norm regularization. TPAMI. 2013.
- Weighted nuclear norm minimization with application to image denoising. ICCV. 2014.
- Hyperspectral image restoration using low-rank matrix recovery. TGRS. 2014
Related Papers in Computer Vision (recent)

- Scalable Robust Matrix Factorization with Nonconvex Loss. NIPS. 2018.
- Fast randomized singular value thresholding for nuclear norm minimization. TPAMI. 2018.
- Tensor Robust Principal Component Analysis with A New Tensor Nuclear Norm. TPAMI. 2018.
- Efficient, Sparse Representation of Manifold Distance Matrices for Classical Scaling. CVPR. 2018.
- Efficient Low Rank Tensor Ring Completion. ICCV. 2017.
Agenda

• Introduction
  • Example Applications
  • Nonconvex Regularization
  • Preliminary : Proximal Gradient Algorithm
  • N2C Transformation \( \text{(sparse)} \)
  • FaNCL Algorithm \( \text{(low-rank)} \)
  • Conclusion and Future Works
Sparse Coding for Image Denoising

Overlapping patches

Group similar patches and reshape into vectors

\{x_i\}

Sparse coding on these patches
Sparse Coding for Image Denoising

Objective: \[
\min_{\{\alpha_i\}} \frac{1}{2} \sum_{i=1}^{N} \| x_i - D \alpha_i \|_2^2 + \lambda \| \alpha_i \|_1
\]

- \( D \) is dictionary learnt from noisy image/external data
- clean patch is recovered by \( D \alpha_i \)

loss \quad \text{sparse} \quad \text{regularization}

Highly redundant dictionary is key for good performance
- \( \alpha_i \) is very sparse, convex \( l_1 \)-norm is popularly used to encouraging sparsity
- more complex structure can be encoded, e.g., group structure and tree-structure
Matrix Completion for Recommender System

Predicting unknown ones based on existing observations
Matrix Completion for Recommender System

Similarity among users and items: low-rank assumption

Low-rank assumption $k \ll \min(m, n)$
- much less variables are needed \(\rightarrow\) capture relatedness and redundancy
Matrix Completion for Recommender System

Objective:
\[
\min_{\{X\}} \frac{1}{2} \sum_{i=1}^{N} \| P_\Omega (X - O) \|_F^2 + \lambda \| X \|_*
\]

- \( P_\Omega \) is a mask operator, if corresponding positions are not observed, their losses will be always set to zero
- \( \| X \|_* = \sum_{i=1}^{m} \sigma_i(X) \), where \( \sigma_i(X) \) denotes \( i \)th singular value of \( X \) (extension of \( l_1 \)-norm from vector to matrix case)

Capturing redundancy in \( O \) is key for good performance
- if a matrix is of rank \( k \), it has \( k \) nonzero singular values
- singular value threshold (SVD) is need to solve \( \| \cdot \|_* \), which costs \( O(m^2n) \) (very expensive)
Optimization Objectives

\[
\begin{align*}
\text{sparse:} & \quad \min_{\{\alpha_i\}} \frac{1}{2} \sum_{i=1}^{N} \| x_i - D \alpha_i \|_2^2 + \lambda \| \alpha_i \|_1 \\
\text{low-rank:} & \quad \min_{\{X\}} \frac{1}{2} \sum_{i=1}^{N} \| P_\Omega (X - O) \|_F^2 + \lambda \| X \|_*
\end{align*}
\]

\[\min_{x} f(x) + \lambda g(x)\]

- \(f\) loss: smooth loss function
- \(g\) regularization: nonsmooth nonconvex

What's nonconvexity here?
Why we need nonconvex?
Nonconvex Regularization

One dimensional illustration

- Features will large values are more informative, thus need to be less penalized
- All these penalties less penalize top features than the convex L1-norm (thus become nonconvex)

Table 1: Example nonconvex regularizers. Here, $\kappa_0 \equiv \kappa'(0)$ and $\beta > 0$. For SCAD, $\theta > 2$, whereas for others, $\theta > 0$. 

<table>
<thead>
<tr>
<th>Regularizer</th>
<th>$\kappa(\alpha)$</th>
<th>$\kappa'(\alpha)$</th>
<th>$\kappa_0$</th>
<th>$\rho$</th>
</tr>
</thead>
<tbody>
<tr>
<td>GP (Geman and Yang, 1995)</td>
<td>$\frac{\beta \alpha}{\theta + \alpha}$</td>
<td>$\frac{\beta \theta}{(\theta + \alpha)^2}$</td>
<td>$\frac{\beta}{\theta}$</td>
<td>$2\frac{\beta}{\theta^2}$</td>
</tr>
<tr>
<td>LSP (Candès et al., 2008)</td>
<td>$\beta \log(1 + \frac{\alpha}{\theta})$</td>
<td>$\frac{\beta}{\theta + \alpha}$</td>
<td>$\frac{\beta}{\theta}$</td>
<td>$\frac{\beta}{\theta^2}$</td>
</tr>
<tr>
<td>MCP (Zhang, 2010a)</td>
<td>$\begin{cases} \beta \alpha - \frac{\alpha^2}{2\theta} &amp; \alpha \leq \beta \theta \ \frac{1}{2} \beta \theta^2 &amp; \alpha &gt; \beta \theta \end{cases}$</td>
<td>$\begin{cases} \beta - \frac{\alpha}{\theta} &amp; \alpha \leq \beta \theta \ 0 &amp; \alpha &gt; \beta \theta \end{cases}$</td>
<td>$\beta$</td>
<td>$\frac{1}{\theta}$</td>
</tr>
<tr>
<td>Laplace (Trzasko and Manduca, 2009)</td>
<td>$\beta (1 - \exp(-\frac{\alpha}{\theta}))$</td>
<td>$\frac{\beta}{\theta} \exp(-\frac{\alpha}{\theta})$</td>
<td>$\frac{\beta}{\theta}$</td>
<td>$\frac{\beta}{\theta^2}$</td>
</tr>
<tr>
<td>SCAD (Fan and Li, 2001)</td>
<td>$\begin{cases} \beta \alpha &amp; \alpha \leq \beta \ -\frac{\alpha^2 + 2\theta \alpha - \beta^2}{2(\theta - 1)} &amp; \beta &lt; \alpha \leq \theta \beta \ \frac{\beta^2(1+\theta)}{2} &amp; \alpha &gt; \beta \theta \end{cases}$</td>
<td>$\begin{cases} \beta &amp; \alpha \leq \beta \ -\frac{\alpha + \beta^2}{\theta - 1} &amp; \beta &lt; \alpha \leq \theta \beta \ 0 &amp; \alpha &gt; \beta \theta \end{cases}$</td>
<td>$\beta$</td>
<td>$\frac{1}{\theta - 1}$</td>
</tr>
</tbody>
</table>
Nonconvex Regularization

(a). original image, (b). blurry image, (c). deconvolution with L1, (d). with nonconvex regularization [from Zuo et al 2013]

<table>
<thead>
<tr>
<th>regularizer</th>
<th>convex</th>
<th>nonconvex</th>
</tr>
</thead>
<tbody>
<tr>
<td>optimization</td>
<td>😞 ✓</td>
<td>😞 ×</td>
</tr>
<tr>
<td>performance</td>
<td>😞 ×</td>
<td>☻ ✓</td>
</tr>
</tbody>
</table>

- convex reg leads to easy optimization but worse performance
- better performance can be obtained from nonconvex reg but optimization becomes much harder

Can we have best of both worlds?
Agenda

• Introduction

• Preliminary : Proximal Gradient Algorithm
  • Proximal step
  • N2C Transformation (sparse)
  • FaNCL Algorithm (low-rank)

• Conclusion and Future Works
Proximal Gradient (PG) Algorithm \[\text{[Parikh & Boyd 2013]}\]

PG algorithm is a general optimization framework solving

\[
\min_{\mathbf{x}} f(\mathbf{x}) + \lambda g(\mathbf{x})
\]

\[
\begin{align*}
&\cdot f \text{ loss: smooth loss function} \\
&\cdot g \text{ regularization: nonsmooth nonconvex}
\end{align*}
\]

In each iteration, it generates

\[
\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}} f(\mathbf{x}_t) + (\mathbf{x} - \mathbf{x}_t)\nabla f(\mathbf{x}_t) + \frac{L}{2}\|\mathbf{x} - \mathbf{x}_t\|_2^2 + \lambda g(\mathbf{x})
\]

\[
= \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - \left(\mathbf{x}_t - \frac{1}{L}\nabla f(\mathbf{x}_t)\right)\|_2^2 + \lambda g(\mathbf{x})
\]

\[
\mathbf{x}_{t+1} = \text{prox}_{\lambda g}\left(\mathbf{x} - \left(\mathbf{x}_t - \frac{1}{L}\nabla f(\mathbf{x}_t)\right)\right)
\]

Key concept: Proximal Step
Proximal Gradient (PG) Algorithm [Parikh & Byod 2013]

\[ x = \text{prox}_{\lambda g}(z) \quad \text{cheap solutions} \]

- the sequence is iteratively generated from proximal step

When \( g(x) = \|x\|_1 \), the solution is called \textit{soft-thresholding}, i.e.,

\[ \left[ \text{prox}_{\lambda \| \cdot \|_1}(z) \right]_i = \text{sign}(x_i) \cdot \max(|x_i| - \lambda, 0) \]

one pass of all features, cheap

When \( g(x) = \|X\|_* \), the solution is called \textit{singular value thresholding (SVT)}, i.e.,

\[ \text{prox}_{\lambda \| \cdot \|_1}(z) = U(\Sigma - \lambda I)_+ V^T \]

where is the \( U \Sigma V^T \) is SVD of \( Z \).

SVD is required, expensive
Agenda

• Introduction

• Preliminary: Proximal Gradient Algorithm

• N2C Transformation (sparse)
  • Basic Idea
  • Use with PG
  • Experiments

• FaNCL Algorithm (low-rank)

• Conclusion and Future Works
Tree Structured Lasso – An Example [Rodolphe et al 2011]

How features can be organized

Example of the tree

can be useful for analyzing patterns in data

each red circle is one group
Tree Structured Lasso – An Example [Rodolphe et al. 2011]

\[
\min_{\mathbf{x}} f(\mathbf{x}) + \lambda g(\mathbf{x})
\]

\[
\arg\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \mathbf{z}\|_2^2 + \lambda \|\mathbf{x}\|_1 + \sum_{j=1}^{K} \mu_j \|\mathbf{x}_{g_j}\|_2
\]

- **convex**
  - cheaply solved by PG

- **nonconvex**
  - no closed-form, iterative algorithms are needed for the nonconvex one

\[
\arg\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \mathbf{z}\|_2^2 + \lambda \sum_{i=1}^{d} \kappa(|x_i|) + \sum_{j=1}^{K} \mu_j \kappa(|\mathbf{x}_{g_j}|)
\]

Proximal step is expensive

Transforming the objective!
PG Algorithm – Look inside

\[
\min_{x} f(x) + \lambda g(x)
\]

To guarantee the convergence of PG, the most important thing is the smoothness of \( f \).

- still have the convergence guarantee
- has cheap proximal step as convex regularizers
Lasso - Illustration

Rewrite $g$ as $g(x) = \left( \sum_{i=1}^{d} \kappa(|x_i|) - \kappa_0 \|x\|_1 \right) + \kappa_0 \|x\|_1$

$z \in \mathbb{R}, \|z\|_p = |z|$, then

$\kappa(|z|) - \kappa_0 |z|$ is smooth and concave

where $\kappa_0 = \kappa'(0)$
Nonconvex to Convex (N2C) Transformation

Problem $F(x) = f(x) + g(x)$ can then be rewritten as

$$F(x) = \tilde{f}(x) + \tilde{g}(x),$$

where $\tilde{f}(x) \equiv f(x) + \tilde{g}(x)$

- $\tilde{f}$ (augmented loss): smooth, nonconvex
- $\tilde{g}$ (convexified regularizer): convex, nonsmooth

**Redistributing Nonconvexity:**

- nonconvexity is shifted from regularizer to loss, while still ensuring that the augmented loss is smooth
Back to Tree Structured Lasso

After transformation (left)

\[
\begin{align*}
\bar{f} + \bar{g} &= \arg\min_x \frac{1}{2} \| x - z \|_2^2 + \\
&\quad + \lambda \| x \|_1 + \sum_{j=1}^K \mu_j \| x_{g_j} \|_2 \\

f + g &= \arg\min_x \frac{1}{2} \| x - z \|_2^2 + \\
&\quad + \lambda \sum_{i=1}^d \kappa (|x_i|) + \\
&\quad + \sum_{j=1}^K \mu_j \kappa (\| x_{g_j} \|_2) \\
\end{align*}
\]

cheap closed-from  no closed-form, iterative algorithms are needed

Proximal algorithms on transformed problems can be very fast
Experiments

(a) Objective vs CPU time (minutes).
(b) Objective vs number of iterations.

Figure 8: Convergence of objective on nonconvex tree-structured group lasso. Note that the curves for nmAPG and N2C overlap in Figure 8(b).
Experiments

Figure 9: Convergence of testing loss on nonconvex tree-structured group lasso.
 Whenever using a convex regularizer for sparse learning, you can

• use nonconvex penalty for better performance; and
• N2C to efficiently solve your new model
Agenda

• Introduction

• Preliminary: Proximal Gradient Algorithm

• N2C Transformation (sparse)

• FaNCL Algorithm (low-rank)
  • Thresholding property
  • Sparse plus Low-rank Structure
  • Experiments

• Conclusion and Future Works
Why we need Nonconvex Low-rank Regularizers?

- factorization, $X = UV^T$:
  - set singular values outside selected rank to 0

- nuclear norm, $\|X\|_* = \sum_i \sigma_i(X)$:
  - equally penalize all singular values
What’s the Problem for Matrix

\[ \min_{\mathbf{x}} f(\mathbf{x}) + \lambda r(\mathbf{x}) \]

nonconvex low-rank regularization

GSVT (Generalized Singular Value Thresholding operator) [Lu et al 2014]

The optimal solution of

\[ \text{prox}_{\mu r}(Z) = \arg \min_{X} \frac{1}{2} \|X - Z\|_F + \lambda \sum_{i=1}^{m} \hat{r}(\sigma_i(X)) \]

is \( U \text{Diag}(y^*) V^T \), where \( U \Sigma V^T \) is the SVD of \( Z \), and \( y^* = [y_i^*] \) with

\[ y_i^* = \arg \min_{y_i \geq 0} \frac{1}{2} (y_i - \sigma_i)^2 + \lambda \hat{r}(y_i) \]

using nonconvex penalties on singular values

PG algorithm can be directly used

- GSVT can be computed in closed-form using SVD \( \Theta(m^2n) \), expensive
Cut Down time on Proximal Step

FaNCL (Fast Nonconvex Low-rank algorithm)

How to cut down time complexity
• automatic thresholding property
• approximate SVD using power method
• further speedup with “sparse + low-rank” structure in matrix completion

More than 100× faster than state-of-art solvers and better performance than factorization & nuclear norm
Automatic Thresholding Property

\[
\text{prox}_{\lambda r}(Z) = U \text{Diag}(y^*) V^T \quad \rightarrow \quad y_i^* \in \arg \min_{y_i \geq 0} \frac{1}{2} (y_i - \sigma_i)^2 + \lambda \hat{r}(y_i)
\]

Proposition (Automatic Thresholding)

For any \( \hat{r} \) satisfying Assumption A3, there exists a threshold \( \gamma > 0 \) such that once \( \sigma_i \leq \gamma \) then \( y_i^* = 0 \)

(a) nuclear norm  
(b) capped-\( \ell_1 \)  
(c) LSP

Singular values are in non-ascending order, i.e. \( \sigma_1 \geq \cdots \geq \sigma_m \), once \( \sigma_j \leq \gamma \) then for all \( i \geq j \), \( y_i^* = 0 \)

Allow partial SVD
Automatic Thresholding Property

Only top few singular values/vectors are needed $\rightarrow$ approximate SVD by power method

Examples

- capped-$\ell_1$: $\gamma = \min (\mu, \theta + \frac{\mu}{2})$;
- LSP: $\gamma = \min \left( \frac{\mu}{\theta}, \theta \right)$;
- TNN: $\gamma = \max (\mu, \sigma_{\theta+1})$;
- SCAD: $\gamma = \mu$;
- MCP: $\gamma = \sqrt{\theta} \mu$ if $0 < \theta < 1$, and $\mu$ otherwise.
Approximate SVD using Power Method

Power Method [Halko et al., 2011]

Require: matrix $Z \in \mathbb{R}^{m \times n}$, $R \in \mathbb{R}^{n \times k}$.

1: $Y^1 \leftarrow ZR$;
2: for $t = 1, 2, \ldots, T_{pm}$ do
3: \hspace{1em} $Q^{t+1} = QR(Y^t)$; // QR decomposition
4: \hspace{1em} $Y^{t+1} = Z(Z^T Q^{t+1})$;
5: end for
6: $[U, \Sigma, V] = \text{SVD}\left((Q^{T_{pm}+1})^T Z\right)$;
7: return $Q^{T_{pm}+1} U$, $\Sigma$ and $V$.

- reduce from $O(m^2 n)$ to $O(mnk)$
- further speedup to $O(||\Omega||_1 k)$ with "sparse + low rank" structure in matrix completion

$$Z^t = X^t - \frac{1}{\rho} \nabla f(X^t) = \underbrace{U^t V^t}_\text{low-rank} - \frac{1}{\rho} \underbrace{P_\Omega (X^t - O)}_\text{sparse}$$

where $X^t$ is maintained in factorized form, i.e. $X^t = U^t V^t$
Full Algorithm

**FaNCL (Fast NonConvex Lowrank algorithm)**

1. randomly initialize $V_0, V_1 \in \mathbb{R}^{n \times k}$ and $X^1 = 0$;
2. for $t = 1, 2, \ldots, T$ do
3.  \[ \lambda^t \leftarrow (\lambda^{t-1} - \lambda)\nu + \lambda; \]
4.  \[ Z^t \leftarrow X^t - \frac{1}{\tau} \nabla f(X^t); \]
5.  \[ V^{t-1} \leftarrow V^{t-1} - V^t (V^T \nu_{t-1}^T), \text{ and remove any zero columns;} \]
6.  \[ R \leftarrow QR([V^t, V^{t-1}]); \]
7.  for $p = 1, 2, \ldots$ do
8.  \[ [\tilde{X}^p, R] = \text{ApproximateGSVT}(Z^t, R); \]
9.  if $F(\tilde{X}^p) \leq F(X^t) - c_t \| \tilde{X}^p - X^t \|_F^2$ then
10. \[ X^{t+1} \leftarrow \tilde{X}^p, \quad V^{t+1} \leftarrow \tilde{V}^p; \text{ break;} \]
11. else $R^{p+1} = V_A^p$; end if
12. end for
13. end for
14. return $X^{T+1}$.

- step 8: approximate GSVT is done
- step 9: decreasing condition is checked, if it fails, improve approximation by repeatedly calling ApproximateGSVT
FaNCL - Convergence analysis

A limit point $X^*$ can be obtained

**Proposition**

$$\sum_{t=1}^{\infty} \|X^{t+1} - X^t\|_F^2 < \infty.$$  

The limit point is also a critical point

**Theorem**

$\{X^t\}$ converges to a critical point $X^*$ of $F(X)$ in finite iterations.

Converge at $O(1/T)$ rate

**Corollary**

$$\min_{t=1,\ldots,T} \|X^{t+1} - X^t\|_F^2 \leq \frac{1}{c_1 T} \left[ F(X^1) - F(X^*) \right]$$

Can be extended to handle multiple blocks of parameters, such as RPCA.
## Experiments

<table>
<thead>
<tr>
<th>nuclear norm</th>
<th>$m = 500$ (12.43%)</th>
<th>$m = 1000$ (6.91%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMSE</td>
<td>rank</td>
<td>time</td>
</tr>
<tr>
<td>APG</td>
<td>4.26±0.01</td>
<td>50</td>
</tr>
<tr>
<td>AIS-Impute</td>
<td>4.11±0.01</td>
<td>55</td>
</tr>
<tr>
<td>active</td>
<td>5.37±0.03</td>
<td>53</td>
</tr>
<tr>
<td>fixed rank</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LMaFit</td>
<td>3.08±0.02</td>
<td>5</td>
</tr>
<tr>
<td>ER1MP</td>
<td>21.75±0.05</td>
<td>40</td>
</tr>
<tr>
<td>capped $\ell_1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IRNN</td>
<td>1.98±0.01</td>
<td>5</td>
</tr>
<tr>
<td>GPG</td>
<td>1.98±0.01</td>
<td>5</td>
</tr>
<tr>
<td>FaNCL</td>
<td>1.97±0.01</td>
<td>5</td>
</tr>
<tr>
<td>FaNCL-acc</td>
<td>1.97±0.01</td>
<td>5</td>
</tr>
<tr>
<td>LSP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IRNN</td>
<td>1.96±0.01</td>
<td>5</td>
</tr>
<tr>
<td>GPG</td>
<td>1.96±0.01</td>
<td>5</td>
</tr>
<tr>
<td>FaNCL</td>
<td>1.96±0.01</td>
<td>5</td>
</tr>
<tr>
<td>FaNCL-acc</td>
<td>1.96±0.01</td>
<td>5</td>
</tr>
<tr>
<td>TNN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IRNN</td>
<td>1.96±0.01</td>
<td>5</td>
</tr>
<tr>
<td>GPG</td>
<td>1.96±0.01</td>
<td>5</td>
</tr>
<tr>
<td>FaNCL</td>
<td>1.95±0.01</td>
<td>5</td>
</tr>
<tr>
<td>FaNCL-acc</td>
<td>1.96±0.01</td>
<td>5</td>
</tr>
</tbody>
</table>
Experiments

Fig. 5. RMSE vs CPU time on the MovieLens-1M and 10M data sets.
Experiments – Parallel version

(a) Clock time per iteration.  (b) Speedup.
Summary

• Find the cut-off point and use partial SVD
• the SVD can be effectively approximated by power method
• Utilize problem structure
Agenda

• Introduction
• Preliminary : Proximal Gradient Algorithm
• N2C Transformation (sparse)
• FaNCL Algorithm (low-rank)

• Conclusion and Future Works
Conclusion

• Nonconvex penalties are useful to boost performance obtained from convex ones
• N2C is a powerful and general framework to learn sparse regularizers
• FaNCL is an efficient algorithm targeted at low-rank models
• Both N2C and FaNCL take the best from both worlds (efficiency from convex and performance from nonconvex)
Future Works

• Low-rank tensor learning
• Statistical performance of stationary points
• Automatic selection of proper nonconvex penalties