A Scalable and Extensible Framework for Superposition-Structured Models

Shenjian Zhao and **Cong Xie** and **Zhihua Zhang**
Department of Computer Science and Engineering
Shanghai Jiao Tong University
{zhao1014,xcgoner,zhihua}@sjtu.edu.cn

Abstract

In many learning tasks, structural models usually lead to better interpretability and higher generalization performance. In recent years, however, the simple structural models such as lasso are frequently proved to be insufficient. Accordingly, there has been a lot of work on "superposition-structured" models where multiple structural constraints are imposed. To efficiently solve these "superposition-structured" statistical models, we develop a framework based on a proximal Newton-type method. Employing the smoothed conic dual approach with the LBFGS updating formula, we propose a scalable and extensible proximal quasi-Newton (SEP-QN) framework. Empirical analysis on various datasets shows that our framework is potentially powerful, and achieves super-linear convergence rate for optimizing some popular "superposition-structured" statistical models such as the fused sparse group lasso.

1 Introduction

In this paper, we consider the "superposition-structured" statistical models (Yang and Ravikumar 2013) where multiple structural constraints are imposed. Examples of such structural constraints include sparsity constraint, graph-structure, group-structure, etc. We could leverage such structural constraints via specific regularization functions. Consequently, many problems of relevance in "superposition-structured" statistical learning can be formulated as minimizing a composite function:

$$\min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) \triangleq g(\mathbf{x}) + \Psi(\mathbf{x}), \tag{1}$$

where g is a convex and continuously differentiable loss function, and Ψ is a hybrid regularization, usually defined as sum of N convex (non-smooth) functions. More specifically,

$$\Psi(\mathbf{x}) \triangleq \sum_{i=1}^{N} \psi_i(\mathbf{W}_i\mathbf{x} + \mathbf{b}_i),$$

each ψ_i is convex but not necessarily differentiable, $\mathbf{W}_i \in \mathbb{R}^{q_i\times p}$ and $\mathbf{b}_i \in \mathbb{R}^{q_i}$ are available. For example, $\Psi(\mathbf{x}) \triangleq \lambda_1\|\mathbf{x}\|_1 + \lambda_2\|\mathbf{F}\mathbf{x}\|_1 + \lambda_3\sum_{j=1}^{N-2}\|\mathbf{G}_j\mathbf{x}\|_2$ defines a fused sparse group penalty (Zhou et al. 2012) when $\mathbf{F}$ is the difference matrix and $\mathbf{G}_j$ indicates the group.

Indeed, there are plenty of machine learning models, which can be cast into the formulation in (1).

- Generalized lasso model: all generalized lasso models such as the fused lasso (Tibshirani et al. 2005), the sparse group lasso (Simon et al. 2013), the group lasso for logistic regression (Meier, Van De Geer, and Bühlmann 2008) can be written as the following form:

$$\min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) \triangleq g(\mathbf{x}) + \lambda_1\|\mathbf{F}\mathbf{x}\|_1 + \sum_{j=2}^{N}\lambda_j\|\mathbf{G}_j\mathbf{x}\|_2.$$

- Multi-task learning: given r tasks, each with sample matrix $\mathbf{A}^{(k)} \in \mathbb{R}^{(n_k\times p)}$ (samples in the k-th task) and labels $\mathbf{y}^{(k)}$, Jalali et al. proposed minimizing the following objective:

$$\min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) \triangleq \sum_{k=1}^{r} l(\mathbf{y}^{(k)}, \mathbf{A}^{(k)}(\mathbf{S}^{(k)} + \mathbf{B}^{(k)})) + \lambda_1\|\mathbf{S}\|_1 + \lambda_2\|\mathbf{B}\|_{1,\infty}, \tag{2}$$

  where $l(\cdot)$ is the loss function and $\mathbf{S}^{(k)}$ is the k-th column of S. Besides, more multi-task learning like the model in (Kim and Xing 2010) also could be cast into (1).

- Gaussian graphical model with latent variables: Chandrasekaran et al. showed that the precision matrix will have a low rank + sparse structure when some random variables are hidden, thus the "superposition-structured" model will be much helpful.

Moreover, many real-world problems benefit from these models such as Gene expression, time-varying network and disease progression. In this paper we mainly study the computational issue of the model in (1).

There are some generic methods that can be used to solve these models theoretically. The CVX (Grant and Boyd 2014) is able to solve these models, but it is not scalable. The Primal-Dual approach proposed by Combettes and Pesquet (2012) can deal with these models, but it converges slowly. The smoothed conic dual (SCD) approach was studied in (Lan, Lu, and Monteiro 2011; Nesterov 2005) and Becker, Candès, and Grant (2012) could obtain $\mathcal{O}(\frac{1}{\epsilon})$ iteration-complexity, but it needs to find the minimizer related to

Copyright © 2016, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

$g(\mathbf{x})$ in each iteration. In addition, the alternating direction method of multipliers (ADMM) (Boyd et al. 2011) can also be used to solve this kind of problems. However, ADMM still suffers from the same bottleneck as the methods mentioned earlier. Additionally, as we know that disk I/O is the bottleneck of computation, so it is important to reduce the number of evaluating $g(\mathbf{x})$. In summary, it is challenging to efficiently solve the model on large-scale datasets.

Recently, there has been a flurry of activity about developments of Newton-type methods for minimizing composite functions (1) in the literature. In particular, in (Lee, Sun, and Saunders 2014; Becker and Fadili 2012) the authors focused on minimizing a composite function, which contains a convex smooth function and a convex non-smooth function with a simple proximal mapping. They also analyzed the convergence rate of various proximal Newton-type methods. Schmidt, Kim, and Sra (2011) discussed a projected quasi-Newton algorithm, but the sub-iteration procedure costs too much. Hsieh et al. (2014) further generalized the Newton method to handle some dirty statistical estimators. Their developments "open up the state of the art but forbidding class of M-estimators to very large-scale problems." In addition, there have already been plenty of packages that implement these Newton-type methods such as LIBLINEAR (Fan et al. 2008), GLMNET (Friedman, Hastie, and Tibshirani 2009; Yuan, Ho, and Lin 2012), but are limited to solve simple models such as lasso and elastic net.

To solve the "superposition-structured" models in (1) on the large-scale problem, we resort to a proximal quasi-Newton method which converges superlinearly (Lee, Sun, and Saunders 2014). We develop a Scalable and Extensible Proximal Quasi-Newton (SEP-QN) framework to solve these models. More specifically, we apply a smoothed conic dual (SCD) approach to solving a surrogate of the original model (1). We employ the LBFGS updating formula, so that the surrogate problem could be solved not only efficiently but also robust. Moreover, we present several accelerating techniques including adaptive initial Hessian, warm-start and continuation SCD to solve the surrogate problem more efficiently and gain faster convergence rate.

In the following we start by presenting our SEP-QN framework for solving the "superposition-structured" statistical models. Then we present the approach to solve the surrogate problem, followed by theoretical analysis and concluding empirical analysis.

2 The SEP-QN Framework

In this section we present the SEP-QN framework for solving the "superposition-structured" statistical model in (1). We refer to $g(\mathbf{x})$ as "smooth part" and $\Psi(\mathbf{x})$ as "non-smooth part." Usually, $g(\mathbf{x})$ is a loss function. For example, $g(\mathbf{x}) \triangleq \frac{1}{n}\sum_{i=1}^{n}(y_i - \mathbf{a}_i^T\mathbf{x})^2$ in the least squares regression problem where the $\mathbf{a}_i \in \mathbb{R}^p$ are input vectors and $y_i \in \mathbb{R}$ are the corresponding outputs, and $g(\mathbf{x}) \triangleq \frac{1}{n}\sum_{i=1}^{n}\log(1+\exp(-y_i\mathbf{a}_i^T\mathbf{x}))$ in the logistic regression where the $y_i \in \{-1, 1\}$. We are especially interested in the large-scale case; i.e., the number of training data n is large.

Basic Framework

Roughly speaking, the method is built on a line search strategy, which produces a sequence of points $\{\mathbf{x}_k\}$ according to

$$\mathbf{x}_{k+1} = \mathbf{x}_k + t_k\Delta\mathbf{x}_k,$$

where t_k is a step length calculated by backtrack, and $\Delta\mathbf{x}_k$ is a descent direction. We compute the descent direction by minimizing a surrogate $\hat{f}_k$ of the objective function f. Given the kth estimate $\mathbf{x}_k$ of $\mathbf{x}$, we let $\hat{f}_k(\mathbf{x})$ be a local approximation of f around $\mathbf{x}_k$. The descent direction $\Delta\mathbf{x}_k$ is obtained by solving the following surrogate problem:

$$\min_{\mathbf{x}} \ \hat{f}_k(\mathbf{x}). \tag{3}$$

Proximal Newton-type methods approximate only the smooth part g with a local quadratic form. Thus, in this paper the surrogate function is defined by

$$\begin{aligned}\hat{f}_k(\mathbf{x}) =& \ \hat{g}_k(\mathbf{x}) + \Psi(\mathbf{x}) \\ =& \ g(\mathbf{x}_k) + \nabla g(\mathbf{x}_k)^T(\mathbf{x}-\mathbf{x}_k) \\ &+ \frac{1}{2}(\mathbf{x}-\mathbf{x}_k)^T\mathbf{H}_k(\mathbf{x}-\mathbf{x}_k) + \Psi(\mathbf{x}),\end{aligned} \tag{4}$$

where $\mathbf{H}_k$ is a $p \times p$ positive definite matrix as approximation to the Hessian of g at $\mathbf{x} = \mathbf{x}_k$. There are many strategies for choosing $\mathbf{H}_k$, such as BFGS and LBFGS (Nocedal and Wright 2006). Considering the use in the large-scale problem, we will employ LBFGS to compute $\mathbf{H}_k$.

After we have obtained the minimizer $\hat{\mathbf{x}}_k$ of (3), we use the line search procedure such as backtracking to select the step length t_k such that a sufficient descent condition is satisfied (Lee, Sun, and Saunders 2012). That is,

$$f(\mathbf{x}_k + t_k\Delta\mathbf{x}_k) \le f(\mathbf{x}_k) + \alpha t_k\gamma_k, \tag{5}$$

where $\alpha \in (0, 1/2)$, $\Delta\mathbf{x}_k \triangleq \hat{\mathbf{x}}_k - \mathbf{x}_k$, and

$$\gamma_k \triangleq \nabla g(\mathbf{x}_k)^T\Delta\mathbf{x}_k + \Psi(\mathbf{x}_k + \Delta\mathbf{x}_k) - \Psi(\mathbf{x}_k).$$

Algorithm 1 gives the basic framework of SEP-QN. The key is to solve the surrogate problem (3) when there are multiple structural constraints. In Algorithm 2 we present the method of solving the problem (3). Moreover, we develop several techniques to further accelerate our method. Specifically, we propose an acceleration schema by adaptively adjusting the initial Hessian $\mathbf{H}_0$ in Algorithm 3. We will see that with an appropriate $\mathbf{H}_0$, $\mathbf{H}_k$ can be a better approximation of $\nabla^2 g(\mathbf{x}_k)$, leading to a much faster convergent procedure.

The Solution of the Surrogate Problem (3)

If there is only one non-smooth function in $\Psi(\mathbf{x})$ (i.e., N=1) with simple proximal mapping, we can solve the surrogate problem (3) directly and efficiently via various optimal first-order algorithms such as FISTA (Beck and Teboulle 2009) and coordinate descent which is used in LIBLINEAR (Fan et al. 2008) and GLMNET (Friedman, Hastie, and Tibshirani 2009; Yuan, Ho, and Lin 2012). In this paper we mainly consider the case that there are multiple non-smooth functions. In this case, we could use SCD or ADMM to solve the problem. Since we empirically observe that SCD outperforms ADMM, we resort to the SCD approach.

Algorithm 1 The SEP-QN Framework

Require: $\mathbf{x}_0$ and $\mathbf{H}_0$
Ensure: $\mathbf{x}_0 \in \mathrm{dom} f$, and $\mathbf{H}_0$ is a scaled identity matrix (positive definite).
$S \leftarrow []$, $Y \leftarrow []$, and $\beta \leftarrow 2$
repeat
  Update $\mathbf{H}_k$ using LBFGS, where $\mathbf{H}_k$ is symmetric positive definite.
  Solve the problem in (3) for a descent direction:

$$\Delta \mathbf{x}_k \leftarrow \operatorname*{argmin}_{\Delta} \hat{f}_k(\mathbf{x}_k + \Delta) \quad \text{(Alogrithm 2)}$$

  Search t_k with backtracking method.
  Update $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + t_k \Delta \mathbf{x}_k$
  if $(\mathbf{x}_{k+1} - \mathbf{x}_k)^T (\nabla g(\mathbf{x}_{k+1}) - \nabla g(\mathbf{x}_k)) > 0$ **then**
    $S \leftarrow [S, \mathbf{x}_{k+1} - \mathbf{x}_k]$
    $Y \leftarrow [Y, \nabla g(\mathbf{x}_{k+1}) - \nabla g(\mathbf{x}_k)]$
    $\mathbf{H}_0 \leftarrow Ada\_Hess(t_k, \beta, \mathbf{x}_{k+1} - \mathbf{x}_k, \nabla g(\mathbf{x}_{k+1}) - \nabla g(\mathbf{x}_k), \mathbf{H}_0)$ (Algorithm 3)
  end if
until stopping condition is satisfied

The SCD Approach In order to solve the problem (3) efficiently when $N > 1$, we employ the SCD approach. The main idea is to solve the surrogate problem via its dual.

We first reformulate our concerned problem in (3) into the following form:

$$\begin{aligned} \min \quad & \mathcal{P}(\boldsymbol{\nu}) = \hat{g}_k(\mathbf{x}) + \sum_{i=1}^{N} t_i \\ \text{s.t.} \quad & (\mathbf{W}_i \mathbf{x} + \mathbf{b}_i, t_i) \in \mathcal{K}_{\psi_i}, \end{aligned} \tag{6}$$

where $\boldsymbol{\nu} = (\mathbf{x}, t_1, ..., t_N), t_i$ are new scalar variables, and $\mathcal{K}_{\psi_i}$ is a closed convex cone (usually the epigraph $\psi_i(\mathbf{W}_i \mathbf{x} + \mathbf{b}_i) \leq t_i$). Since projection onto the set $\{\mathbf{x} | (\mathbf{W}_i \mathbf{x} + \mathbf{b}_i, t_i) \in \mathcal{K}_{\psi_i}\}$ might be expensive, we address this issue by solving the dual problem.

We denote the dual variables by $\boldsymbol{\lambda} = (\mathbf{z}_1, \tau_1, ..., \mathbf{z}_N, \tau_N)$, $\mathbf{z} = (\mathbf{z}_1, ..., \mathbf{z}_N)$, where $(\mathbf{z}_i, \tau_i) \in \mathcal{K}^*_{\psi_i}$. And $\mathcal{K}^*_{\psi_i}$ is the dual cone defined by

$$\mathcal{K}^*_{\psi_i} = \{\mathbf{x} : \mathbf{x}^T \mathbf{y} \geq 0 \text{ for all } \mathbf{y} \in \mathcal{K}_{\psi_i}\}.$$

Let us take an example in which $\psi_i(\mathbf{x}) = \|\mathbf{W}_i \mathbf{x} + \mathbf{b}_i\|_1 \leq t_i$. Then $\mathcal{K}_{\psi_i} = \{(\mathbf{W}_i \mathbf{x} + \mathbf{b}_i, t_i) : \|\mathbf{W}_i \mathbf{x} + \mathbf{b}_i\|_1 \leq t_i\}$ and $\mathcal{K}^*_{\psi_i} = \{(\mathbf{z}_i, \tau_i) : \|\mathbf{z}_i\|_\infty \leq \tau_i\}$.

The Lagrangian and dual functions are given by

$$\mathcal{L}(\boldsymbol{\nu}; \boldsymbol{\lambda}) = \hat{g}_k(\mathbf{x}) + \sum_{i=1}^{N} (t_i - \mathbf{z}_i^T (\mathbf{W}_i \mathbf{x} + \mathbf{b}_i) - \tau_i t_i),$$

$$\begin{aligned} \mathcal{D}(\boldsymbol{\lambda}) = \inf_{\mathbf{x}, t_i} \Big\{ & \mathcal{L}(\mathbf{x}, t_i; \mathbf{z}_i, \tau_i) \\ & \triangleq \hat{g}_k(\mathbf{x}) + \sum_{i=1}^{N} (t_i - \mathbf{z}_i^T (\mathbf{W}_i \mathbf{x} + \mathbf{b}_i) - \tau_i t_i) \Big\}. \end{aligned} \tag{7}$$

The Lagrangian is unbounded unless $\tau_i = 1$. Because the appropriate Hessian matrix is positive definite, this problem strongly convex, guaranteeing the convergence rate.

Denote $\mathcal{D}^-(\mathbf{z}) = -\mathcal{D}(\boldsymbol{\lambda}) = -\mathcal{D}(\mathbf{z}_1, 1, ..., \mathbf{z}_N, 1)$, and suppose $\hat{\mathbf{x}}(\mathbf{z})$ is the unique Lagrangian minimizer. Nesterov (2005) proved that $\mathcal{D}^-(\mathbf{z})$ is convex and continuously differentiable, and that $\nabla \mathcal{D}^-(\mathbf{z}) = (\mathbf{W}_1 \hat{\mathbf{x}}(\mathbf{z}) + \mathbf{b}_1, \ldots, \mathbf{W}_N \hat{\mathbf{x}}(\mathbf{z}) + \mathbf{b}_N)^T$ is Lispchitz continuous. Thus, provably convergent and accelerated gradient methods in the Nesterov style are possible.

In particular, we need to minimize $\mathcal{D}^-(\mathbf{z})$. A standard gradient projection step for the smoothed dual problem is

$$\mathbf{z}^{(j+1)} = \operatorname*{arg\,min}_{\mathbf{z}:(\mathbf{z}_i, 1) \in \mathcal{K}^*_{\psi_i}} \|\mathbf{z} - \mathbf{z}^{(j)} + \delta^{(j)} \nabla \mathcal{D}^-(\mathbf{z}^{(j)})\|_2^2. \tag{8}$$

Then we need to obtain $\hat{\mathbf{x}}(\mathbf{z}^{(j)})$ and $\nabla \mathcal{D}^-(\mathbf{z}^{(j)})$. By substituting $\hat{g}_k(\mathbf{x})$ into (7), collecting the linear and quadratic terms, and eliminating the unrelated terms, we get the reduced Lagrangian

$$\hat{\mathcal{D}}(\mathbf{z}) = \inf_{\mathbf{x}} \Big\{ \frac{1}{2} \mathbf{x}^T \mathbf{H}_k \mathbf{x} + \mathbf{x}^T (\nabla g(\mathbf{x}_k) - \mathbf{H}_k \mathbf{x}_k - \sum_{i=1}^{N} \mathbf{W}_i^T \mathbf{z}_i) \Big\}.$$

The minimizer $\hat{\mathbf{x}}(\mathbf{z}^{(j)})$ is given by

$$\hat{\mathbf{x}}(\mathbf{z}^{(j)}) = -\mathbf{H}_k^{-1} (\nabla g(\mathbf{x}_k) - \mathbf{H}_k \mathbf{x}_k - \sum_{i=1}^{N} \mathbf{W}_i^T \mathbf{z}_i^{(j)}). \tag{9}$$

From (7), (8) and (9), the minimization problem over $\mathbf{z}$ is separable, so it can be implemented in parallel. The solution is given by

$$\mathbf{z}_i^{(j+1)} = \operatorname*{arg\,min}_{\mathbf{z}_i : (\mathbf{z}_i, 1) \in \mathcal{K}^*_{\psi_i}} \frac{1}{2\delta^{(j)}} \|\mathbf{z}_i - \mathbf{z}_i^{(j)}\|_2^2 + \mathbf{z}_i^T (\mathbf{W}_i \hat{\mathbf{x}}(\mathbf{z}^{(j)}) + \mathbf{b}_i). \tag{10}$$

From (9) and (10), we obtain the specific AT method (Auslender and Teboulle 2006) to solve the problem (3) in Algorithm 2.

There are many variants of optimal first-order methods (Lan, Lu, and Monteiro 2011; Beck and Teboulle 2009; Nesterov 2007; Tseng 2008). Algorithm 2 is a generic algorithm but may not be the best choice for every model. By using the continuation techniques(Becker, Candès, and Grant 2012), we could obtain the exact solution very quickly.

Acceleration

We further employ several acceleration techniques in our implementation. By applying these techniques we achieve much faster convergence rate which is comparable to the conventional proximal Newton method. Our accelerated implementation behaves much better than the original proximal quasi-Newton method in various aspects.

Adaptive Initial Hessian LBFGS sets the initial Hessian $\mathbf{H}_0$ as $\frac{\mathbf{y}_k^T \mathbf{y}_k}{\mathbf{s}_k^T \mathbf{y}_k} \mathbf{I}$. However, we find that this setting results in a much slower convergence procedure than the proximal Newton method. Thus, it is desirable to give a better initial Hessian $\mathbf{H}_0$, which in turn yields a better approximation of $\nabla^2 g(\mathbf{x}_k)$.

Algorithm 2 Solve Problem (3) via SCD

Require: $\mathbf{x}_k, S, Y, \mathbf{H}_0, \nabla g(\mathbf{x}_k), \mathbf{z}_i^{(0)}, \mathbf{x}_0$

$\theta^{(0)} \leftarrow 1, \mathbf{v}_i^{(0)} \leftarrow \mathbf{z}_i^{(0)}, j \leftarrow 0$
repeat
  $\mathbf{y}_i^{(j)} \leftarrow (1-\theta^{(j)})\mathbf{v}_i^{(j)} + \theta^{(j)}\mathbf{z}_i^{(j)}$
  $\hat{\mathbf{x}} \leftarrow -\mathbf{H}_k^{-1}(\nabla g(\mathbf{x}_k) - \mathbf{H}_k\mathbf{x}_k - \sum_{i=1}^N \mathbf{W}_i^T\mathbf{y}_i^{(j)})$ by LBFGS method.
  for $i \leftarrow 1, N$ **do**
    $\mathbf{z}_i^{(j+1)} \leftarrow \underset{\mathbf{z}_i:(\mathbf{z}_i,1)\in\mathcal{K}_{\psi_i}^*}{\operatorname{argmin}} \frac{\theta^{(j)}}{2\delta^{(j)}}\|\mathbf{z}_i - \mathbf{z}_i^{(j)}\|_2^2 + \mathbf{z}_i^T(\mathbf{W}_i\hat{\mathbf{x}} + \mathbf{b}_i)$
    $\mathbf{v}_i^{(j+1)} \leftarrow (1-\theta^{(j)})\mathbf{v}_i^{(j)} + \theta^{(j)}\mathbf{z}_i^{(j+1)}$
  end for
  $\theta^{(j+1)} \leftarrow 2/(1+(1+4/(\theta^{(j)})^2)^{\frac{1}{2}})$
  $j \leftarrow j+1$
until some stopping condition is satisfied
$\Delta \leftarrow \hat{\mathbf{x}} - \mathbf{x}_k$
return Δ

Theorem 1. *If* $(1-\alpha)\mathbf{H}_k \succcurlyeq \nabla^2 g(\mathbf{x}_k)$ *for* $\alpha \in (0, \frac{1}{2})$, $\mathbf{H}_k \succcurlyeq m\mathbf{I}, (m>0)$ *and* $\nabla^2 g$ *is Lipschitz continuous with constant* L_2*, then the unit step length satisfies the sufficient decrease condition* (5) *after sufficiently many iterations.*

Theorem 2. *Assume* $\mathbf{H}_k^a$ *and* $\mathbf{H}_k^b$ *are generated by the same procedure* $\{(\mathbf{s}_k, \mathbf{y}_k) : \mathbf{s}_k^T\mathbf{y}_k > 0\}$ *but with different initial Hessians* $\mathbf{H}_0^a$ *and* $\mathbf{H}_0^b$*, respectively. If* $\mathbf{H}_0^a \succ \mathbf{H}_0^b \succ \mathbf{0}$*, then* $\mathbf{H}_k^a \succ \mathbf{H}_k^b \succ \mathbf{0}$.

Based on Theorems 1 and 2, we can decrease $\mathbf{H}_0$ more aggressively. Once the unit step fails, we know that $(1-\alpha)\mathbf{H}_k \succ \nabla^2 g(\mathbf{x}_k)$ is broken; hence we need to increase $\mathbf{H}_0$. We propose our adaptive initial Hessian strategy in Algorithm 3. In practice, we set α to a small number like 0.0001.

Algorithm 3 Adaptive Initial Hessian

procedure ADA_HESS($t_k, \beta, \mathbf{s}_k, \mathbf{y}_k, \mathbf{H}_0$)
  if $t_k < 1$ **then**
    $\mathbf{H}_0 = \mathbf{H}_0/t_k$
    $\beta = \frac{2}{1+1/\beta}$
  end if
  $\mathbf{H}_0 = \text{elementwise\_min}(\frac{\mathbf{H}_0}{\beta}, \frac{\mathbf{y}_k^T\mathbf{y}_k}{\mathbf{y}_k^T\mathbf{s}_k}\mathbf{I})$
  return $\mathbf{H}_0$
end procedure

Warm start and continuation SCD We use the optimal dual value $\mathbf{z}_k^*$ which is obtained in solving dual of $\hat{f}_k(\mathbf{x})$ as the initial dual value to solve dual of $\hat{f}_{k+1}(\mathbf{x})$. This leads to a warm start in solving the problem (3), and the iteration complexity will be dramatically reduced.

By employing continuation SCD to solve the problem (3), the dual of the original problem (6) could reach ϵ optimal within $\mathcal{O}(\sqrt{\frac{1}{\lambda_{min}\epsilon}}\|\mathbf{z}_k^* - \mathbf{z}_{k+1}^*\|_2)$ iterations which shows in (Nesterov 2005).

3 Theoretical Analysis

In this section we conduct analysis about the convergence rate of SEP-QN method. Because of space limitations, we give the detailed proofs in the supplementary. In order to provide the global convergence and solve the problem efficiently, we make the following assumptions:

Assumption 3. *f is a closed convex function and* $\inf_{\mathbf{x}}\{f(\mathbf{x})|\mathbf{x} \in \operatorname{dom} f\}$ *is attained at some* $\mathbf{x}^*$.

Assumption 4. *The smooth part g is a closed, proper convex, continuously differentiable function, and its gradient ∇g is Lipschitz continuous with L_1.*

Assumption 5. *The non-smooth part Ψ should be closed, proper, and convex. The projection onto the dual cone associated with each ψ_i is tractable, or equivalently, easy to solve problem* (10).

First, we analyze the global convergence behavior of SEP-QN under these assumptions.

Theorem 6. *If the problem* (3) *is solved by continuation SCD, then* $\{\mathbf{x}_k\}$ *generated by the SEP-QN method converges to an optimal solution* $\mathbf{x}^*$ *starting at any* $\mathbf{x}_0 \in \operatorname{dom} f$.

Under the stronger assumptions, we could derive the local superlinear convergence rate as shown in the following theorem.

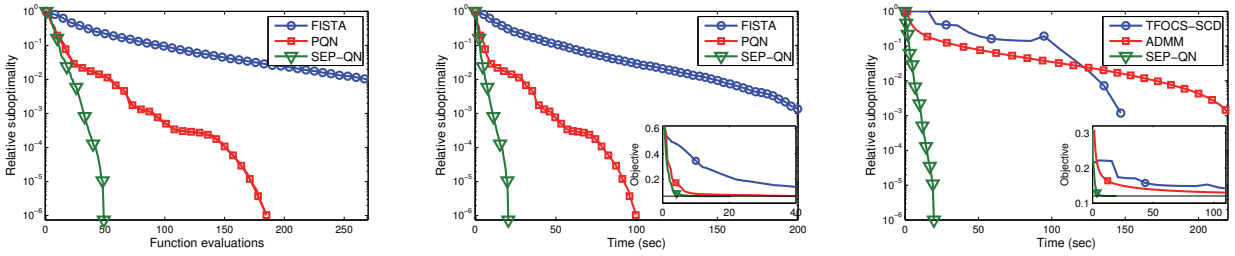
Theorem 7. *Suppose g is twice-continuously differentiable and strongly convex with constant l, and $\nabla^2 g$ is Lipschitz continuous with constant L_2. If $\mathbf{x}_0$ is sufficiently close to $\mathbf{x}^*$, the sequence $\{\mathbf{H}_k\}$ satisfies the Dennis-More criterion, and $l\mathbf{I} \preceq \mathbf{H}_k \preceq L\mathbf{I}$ for some $0 < l \leq L$, then SEP-QN with the continuation SCD converges superlinearly after sufficiently many iterations.*

Remark 8. *Suppose SEP-QN converges within T iterations. If the dataset is dense, then the complexity of SEP-QN is $T\mathcal{O}(np) + T\mathcal{O}(\frac{1}{\epsilon_s}(Mp + \sum_{i=1}^N q_i p))$; if the dataset is sparse, the complexity is $T\mathcal{O}(\text{nnz}) + T\mathcal{O}(\frac{1}{\epsilon_s}(Mp + \sum_{i=1}^N q_i p))$, where M is the history size of LBFGS, ϵ_s is the tolerance of the problem* (3) *and* nnz *is the amount of non-zero entries in the sparse dataset.*

We require that n or nnz are relatively large, otherwise the complexity of the problem (3) will go over the complexity of evaluating the loss function. In this case, it would be better to use some first-order methods instead of SEP-QN. If ignoring the impact of T and the dataset is dense, the convergence time of SEP-QN is linear with respect to the number of features, the amount of data size, and the number of non-smooth terms. We will empirically validate the scalability and extensibility of SEP-QN in the following section.

4 Empirical Analysis

We implement all the experiments on a single machine running the 64-bit version of Linux with an Intel Core i5-3470 CPU and 8 GB RAM. We test the SEP-QN framework on various real-world datasets such as `gisette` ($n = 6,000$ and $p = 5,000$) and `epsilon` ($n = 300,000$ and $p =$



(a) epochs of ℓ_1-logistic regression (b) runtime of ℓ_1-logistic regression (c) fused ℓ_1-logistic regression

Figure 1: Convergence comparison

$2,000$) which can be downloaded from LIBSVM website[1]. The dataset characteristics are provided in the Table 1.

Table 1: Details of the datasets in our experiments

Dataset	p	n (train)	n (test)	nnz (train)
epsilon	2,000	300,000	100,000	600,000,000
gisette	5,000	6,000	1,000	29,729,997
usps	649	1000	1000	649,000

"Superposition-structured" Logistic Regression

We consider the "superposition-structured" logistic regression problem:

$$\min_{\mathbf{x}\in\mathbb{R}^p} \frac{1}{n}\sum_{i=1}^{n} \log(1+\exp(-y_i\mathbf{a}_i^T\mathbf{x})) + \lambda\|\mathbf{x}\|_1 + \gamma\|\mathbf{W}\mathbf{x}\|_q.$$

We first set $\mathbf{W}=\mathbf{0}$ for comparison with the result of PQN in (Lee, Sun, and Saunders 2012). For fairness of comparison, we use the same dataset `gisette` and the same setting of the tuning parameter λ as (Lee, Sun, and Saunders 2012; Yuan, Ho, and Lin 2012). The results are shown in the Figures 1(a) and 1(b). We can see that the SEP-QN method has the fastest convergence rate, which agrees with Theorems 6 and 7.

In order to verify the effectiveness and efficiency when the model has multiple structural constraints.We compare SEP-QN with ADMM and the direct SCD in TFOCS (Becker, Candès, and Grant 2012) on the fused sparse logistic regression by setting $\|\mathbf{W}\mathbf{x}\|_q = \|\mathbf{x}\|_{TV}$ and $\gamma=\lambda$. Figure 1(c) shows that the three algorithms converge to the same optimal value, but SEP-QN performs much better.

Multi-task Learning

Next we solve the multi-task learning problem where the parameter matrix $\mathbf{X}$ will have a sparse + group sparse structure. In our framework, there is no need to seperate the parameter matrix $\mathbf{X}$ into $\mathbf{S}+\mathbf{B}$ as in (Jalali et al. 2010; Hsieh et al. 2014). Instead of using the square loss (as in (Jalali et al. 2010)), we consider the logistic loss, which

[1]http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets

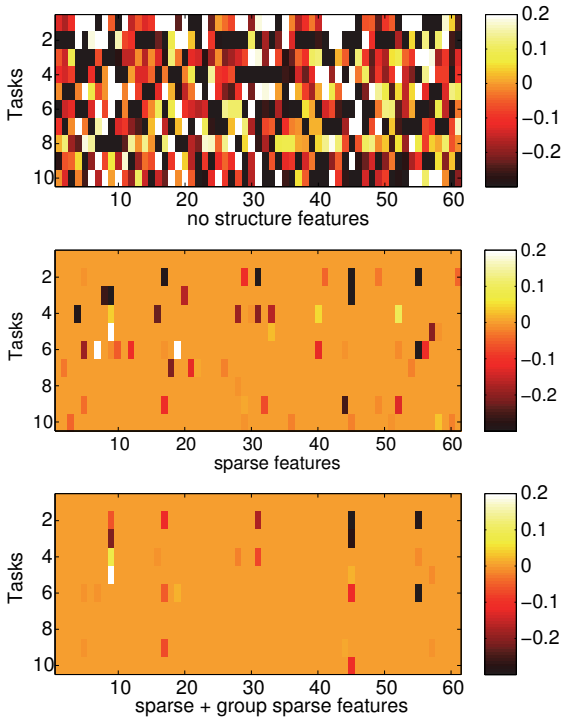


Figure 2: Feature visualization. As shown in the colorbar, orange color indicates that the value of corresponding feature is 0.

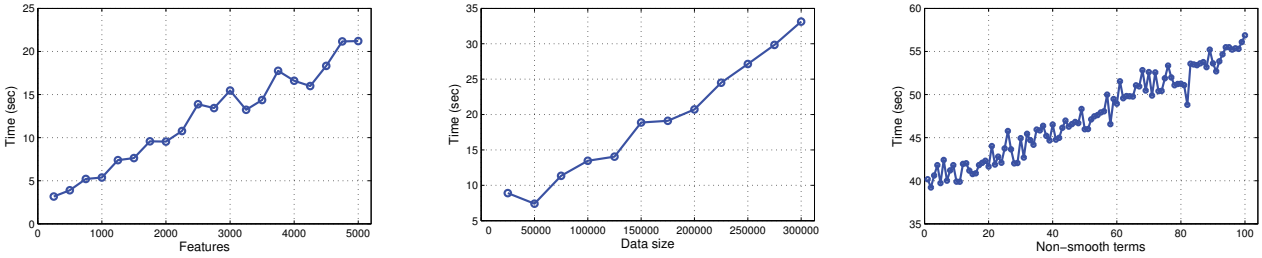
gives better performance. Thus, $\mathbf{X}$ could be estimated by the following objective function,

$$\min_{\mathbf{X}\in\mathbb{R}^{p\times r}} \sum_{k=1}^{r} l_{logistic}(\mathbf{y}^{(k)}, \mathbf{A}^{(k)}\mathbf{X}^{(k)}) + \lambda\|\mathbf{X}\|_1 + \gamma\|\mathbf{X}\|_{1,2}.$$

We follow (Jalali et al. 2010; Hsieh et al. 2014) and transform multi-class problems into multi-task problems. For fairness of comparison, we test on the same dataset USPS which was first collected in (Van Breukelen et al. 1998) and subsequently widely used in multi-task papers as a reliable dataset for handwritten recognition algorithms. There are $r=10$ tasks, and each handwritten sample consists of $p=649$ features. In (Jalali et al. 2010; Hsieh et al. 2014), the authors demonstrated that on USPS, using sparse and group

Table 2: The comparisons on multi-task problems.

n	relative error	sparse + group sparse (test error rate / training time)			Other Models	
		SEP-QN	QUIC & DIRTY	ADMM	Lasso	Group Lasso
100	10^{-1}	7.3% / 0.32s	8.3% / 0.42s	8.3% / 1.5s	7.9%	7.4%
	10^{-4}	6.4% / 0.93s	7.4% / 0.75s	7.5% / 4.3s		
400	10^{-1}	3.0% / 1.2s	2.9% / 1.01s	3.0% / 3.6s	3.0%	3.1%
	10^{-4}	2.6% / 2.0s	2.5% / 1.55s	2.6% / 11.0s		



(a) feature-number scalability (b) data-size scalability (c) nonsmooth-terms extensibility

Figure 3: Scalability and extensibility

sparse regularizations together outperforms the models with a single regularizer.

We visualize features that estimated by our SEP-QN framework in Figure 2, and we just plot the first sixty features to provide a clear visualization. Figure 2 shows that the feature structure is well maintained by the regularizer. The promising results of "sparse + group sparse structure" further validate the effectiveness of our SEP-QN framework. As shown in Table 2, our SEP-QN framework is comparable to QUIC & DIRTY which is the state-of-art method. Unlike QUIS & DIRTY, our implementation is straightforward in the SEP-QN framework. Because of broad interest of our framework, it may be slower than QUIC & DIRTY on some specifical datasets. However, we will show that our framework is scalable by the experiments in the following section.

Scalability and Extensibility

We consider the group generalized lasso problem (Tibshirani et al. 2005; Simon et al. 2013; Meier, Van De Geer, and Bühlmann 2008), but use the logistic loss function instead for classification. Specifically,

$$\min_{\mathbf{x}\in\mathbb{R}^p} \frac{1}{n}\sum_{i=1}^{n}\log(1+\exp(-y_i\mathbf{a}_i^T\mathbf{x})) + \lambda_1\|\mathbf{x}\|_1 + \lambda_2\|\mathbf{F}\mathbf{x}\|_1 + \sum_{j=1}^{N-2}\gamma_j\|\mathbf{G}_j\mathbf{x}\|_2.$$

As far as we know, there is no an efficient algorithm to solve this model. Note that this dirty model may not be a good choice for `gisette` and `epsilon` datasets. We just use this model to validate the scalability and extensibility of SEP-QN framework.

We use the fused sparse logistic regression ($\lambda_1 = \frac{2}{n}, \lambda_2 = \frac{2}{n}, N = 2$) and `gisette` dataset to test the feature-number scalability of SEP-QN as shown in Figure 3(a). Then we test the data-size scalability on `epsilon` dataset as shown in Figure 3(b). We can see that the convergence time is linear with respect to the number of features as well as the amount of data.

Then we use the group sparse logistic model ($\lambda_1 = \frac{2}{n}, \lambda_2 = 0, \gamma_j = \frac{2}{n}$) and `epsilon` dataset to test the nonsmooth-terms extensibility of SEP-QN. As shown in Figure 3(c), the convergence time is linear with respect to the number of non-smooth terms. These experiments further verify Remark 8 under the assumptions.

5 Conclusion

In this paper, we have generalized the proximal quasi-Newton method to handle "superposition-structured" statistical models and devised a SEP-QN framework. With the help of the SCD approach and LBFGS updating formula, we can solve the surrogate problem in an efficient and feasible way. We have explored the global convergence and the super-linear convergence both theoretically and empirically. Compared with prior methods, SEP-QN converges significantly faster and scales much better, and the promising experimental results on several real-world datasets have further validated the scalability and extensibility of the SEP-QN framework.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 61572017) and the Natural Science Foundation of Shanghai City (No. 15ZR1424200).

References

Auslender, A., and Teboulle, M. 2006. Interior gradient and proximal methods for convex and conic optimization. *SIAM Journal on Optimization* 16(3):697–725.

Beck, A., and Teboulle, M. 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences* 2(1):183–202.

Becker, S., and Fadili, J. 2012. A quasi-Newton proximal splitting method. In *Advances in Neural Information Processing Systems*, 2618–2626.

Becker, S.; Candès, E.; and Grant, M. 2012. TFOCS: Templates for first-order conic solvers.

Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; and Eckstein, J. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning* 3(1):1–122.

Chandrasekaran, V.; Parrilo, P.; Willsky, A. S.; et al. 2010. Latent variable graphical model selection via convex optimization. In *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*, 1610–1613. IEEE.

Combettes, P. L., and Pesquet, J.-C. 2012. Primal-dual splitting algorithm for solving inclusions with mixtures of composite, Lipschitzian, and parallel-sum type monotone operators. *Set-Valued and variational analysis* 20(2):307–330.

Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; and Lin, C.-J. 2008. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research* 9:1871–1874.

Friedman, J.; Hastie, T.; and Tibshirani, R. 2009. glmnet: Lasso and elastic-net regularized generalized linear models. *R package version* 1.

Grant, M., and Boyd, S. 2014. CVX: Matlab software for disciplined convex programming, version 2.1.

Hsieh, C.-J.; Dhillon, I. S.; Ravikumar, P. K.; Becker, S.; and Olsen, P. A. 2014. QUIC & DIRTY: A quadratic approximation approach for dirty statistical models. In *Advances in Neural Information Processing Systems*, 2006–2014.

Jalali, A.; Sanghavi, S.; Ruan, C.; and Ravikumar, P. K. 2010. A dirty model for multi-task learning. In *Advances in Neural Information Processing Systems*, 964–972.

Kim, S., and Xing, E. P. 2010. Tree-guided group lasso for multi-task regression with structured sparsity. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 543–550.

Lan, G.; Lu, Z.; and Monteiro, R. D. 2011. Primal-dual first-order methods with $\mathcal{O}(1/\epsilon)$ iteration-complexity for cone programming. *Mathematical Programming* 126(1):1–29.

Lee, J.; Sun, Y.; and Saunders, M. 2012. Proximal Newton-type methods for convex optimization. In *Advances in Neural Information Processing Systems*, 836–844.

Lee, J. D.; Sun, Y.; and Saunders, M. A. 2014. Proximal Newton-type methods for minimizing composite functions. *SIAM Journal on Optimization* 24(3):1420–1443.

Meier, L.; Van De Geer, S.; and Bühlmann, P. 2008. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 70(1):53–71.

Nesterov, Y. 2005. Smooth minimization of non-smooth functions. *Mathematical programming* 103(1):127–152.

Nesterov, Y. 2007. Gradient methods for minimizing composite objective function.

Nocedal, J., and Wright, S. 2006. Numerical optimization, series in operations research and financial engineering. *Springer, New York, USA*.

Schmidt, M.; Kim, D.; and Sra, S. 2011. Projected Newton-type methods in machine learning.

Simon, N.; Friedman, J.; Hastie, T.; and Tibshirani, R. 2013. A sparse-group lasso. *Journal of Computational and Graphical Statistics* 22(2):231–245.

Tibshirani, R.; Saunders, M.; Rosset, S.; Zhu, J.; and Knight, K. 2005. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67(1):91–108.

Tseng, P. 2008. On accelerated proximal gradient methods for convex-concave optimization. *submitted to SIAM Journal on Optimization*.

Van Breukelen, M.; Duin, R. P.; Tax, D. M.; and Den Hartog, J. 1998. Handwritten digit recognition by combined classifiers. *Kybernetika* 34(4):381–386.

Yang, E., and Ravikumar, P. K. 2013. Dirty statistical models. In *Advances in Neural Information Processing Systems*, 611–619.

Yuan, G.-X.; Ho, C.-H.; and Lin, C.-J. 2012. An improved glmnet for l1-regularized logistic regression. *The Journal of Machine Learning Research* 13(1):1999–2030.

Zhou, J.; Liu, J.; Narayan, V. A.; and Ye, J. 2012. Modeling disease progression via fused sparse group lasso. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 1095–1103. ACM.