A Fast Dual Projected Newton Method for ℓ_1-Regularized Least Squares

Pinghua Gong Changshui Zhang
State Key Laboratory on Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology (TNList)
Department of Automation, Tsinghua University, Beijing 100084, China
{gph08@mails,zcs@mail}.tsinghua.edu.cn

Abstract

L_1-regularized least squares, with the ability of discovering sparse representations, is quite prevalent in the field of machine learning, statistics and signal processing. In this paper, we propose a novel algorithm called Dual Projected Newton Method (DPNM) to solve the ℓ_1-regularized least squares problem. In DPNM, we first derive a new dual problem as a box constrained quadratic programming. Then, a projected Newton method is utilized to solve the dual problem, achieving a *quadratic convergence rate*. Moreover, we propose to utilize some practical techniques, thus it greatly reduces the computational cost and makes DPNM more efficient. Experimental results on six real-world data sets indicate that DPNM is very efficient for solving the ℓ_1-regularized least squares problem, by comparing it with state of the art methods.

1 Introduction

Adding an ℓ_1-norm constraint or an ℓ_1-norm regularization term to an optimization problem, a sparse solution can be achieved in some applications. A sparse solution usually benefits us in some aspects: good interpretation [Tibshirani, 1996] and memory savings.

The lasso [Tibshirani, 1996], a representative ℓ_1-regularized least squares problem[1], has attracted more and more attentions from the field of artificial intelligence. It has a wide range of applications, such as signal reconstruction [Wright *et al.*, 2009], image deblurring [Beck and Teboulle, 2009], gaussian graphical model structure learning [Friedman *et al.*, 2008], sparse coding [Lee *et al.*, 2007], curve-fitting and classification [Bishop, 2006]. In these applications, how to efficiently solve the ℓ_1-regularized least squares problem becomes a critical issue. Most existing optimization methods for the ℓ_1-regularized least squares problem can be broadly classified into three categories.

First, some algorithms are designed by transforming ℓ_1-regularized least squares as a constrained quadratic programming problem. This is achieved by either introducing an auxiliary variable, or splitting the variable into the positive and negative parts. Representative algorithms include interior method (L1LS) [Kim *et al.*, 2007], GPSR [Figueiredo *et al.*, 2007] and ProjectionL1 [Schmidt *et al.*, 2007]. However, these methods double the variable size, making the optimization more costly.

Second, several algorithms are developed in the fixed-point-type framework: A gradient descent operation is first done, and then a soft-thresholding operation is performed. Two most representative algorithms are SpaRSA [Wright *et al.*, 2009] and FISTA [Beck and Teboulle, 2009]. SpaRSA utilizes the Barzilai-Borwein (BB) rule [Figueiredo *et al.*, 2007] to carefully choose the gradient descent step size. FISTA chooses the position of the soft-thresholding opearation by exploiting Nesterov's method [Nesterov, 1983]. Some other fixed-point-type algorithms include FOBOS [Duchi and Singer, 2009], fixed point continuation [Hale *et al.*, 2007], etc. However, these algorithms are first-order methods, not utilizing the second-order information.

Third, a few active-set-type algorithms are studied. A very recent method, called block principal pivoting (BP) [Kim and Park, 2010] is proposed to solve the ℓ_1-regularized least squares problem. It's a further development based on least angle regression (LARS) [Efron *et al.*, 2004] and feature-sign (FS) search algorithm [Lee *et al.*, 2007]. Kim and Park [2010] give a dual problem and then utilize it to obtain KKT conditions. Subsequently, BP is built based on the KKT conditions. However, BP doesn't directly solve the dual problem, since the constraint of the dual problem make it difficult to design a very efficient algorithm (see Section 2.3).

Although various methods mentioned above are designed to solve the ℓ_1-regularized least squares problem, none of them reports a quadratic convergence rate[2]. In this paper, we propose a second-order algorithm called Dual Projected Newton Method (DPNM) to efficiently solve the ℓ_1-regularized least squares problem. By skillfully introducing an auxiliary variable and exploiting the characteristic of the ℓ_1-norm,

[1] In fact, the lasso is an ℓ_1-constrained least squares problem, but it can be recast as an ℓ_1-regularized least squares problem under mild conditions. In the subsequent text, we only focus on the ℓ_1-regularized least squares problem.

[2] The quadratic convergence rate here refers to $\lim_{k\to\infty}||\mathbf{x}^{k+1}-\mathbf{x}^\star||/||\mathbf{x}^k-\mathbf{x}^\star||^2=\mu$ with $\mu>0$. It achieves a δ-accurate solution $\mathbf{x}$ ($||\mathbf{x}-\mathbf{x}^\star||\leq\delta$) in $O(\log\log(1/\delta))$ steps.

we derive a new dual formulation of the ℓ_1-regularized least squares as a box constrained quadratic programming problem. Due to the simple box constraint in our new dual problem, a projected Newton method is used to solve the dual problem, achieving a *quadratic convergence rate*. Moreover, we apply some practical techniques to DPNM, greatly reducing the cost per iteration. In the sequel, we will see that the new dual form we derive doesn't increase the variable size, although we introduce an auxiliary variable. Low computational cost per iteration and less iterative steps make DPNM quite efficient. Empirical studies demonstrate that DPNM converges much faster than several state of the art algorithms by comparing the CPU time consumed.

The remainder of this paper is organized as follows. In Section 2, we present the Dual Projected Newton Method (DPNM) in detail. In Section 3, experimental results are shown on six real-world data sets. Concluding remarks are given in Section 4.

2 Dual Projected Newton Method

2.1 Notations

First of all, we introduce some notations in this paper. Scalars are denoted by lower case letters (e.g., $x \in \mathbb{R}$) and vectors by lower case bold face letters (e.g., $\mathbf{x} \in \mathbb{R}^n$). x_i denotes the i-th element of a vector $\mathbf{x}$. Matrix and Sets are denoted by capital letters (e.g., A, I) and $\sharp(I)$ indicates the number of elements in the set I. A_I denotes a submatrix of A, containing the corresponding rows and columns indexed by I, and $\mathbf{x}_I$ denotes a sub-vector of $\mathbf{x}$ including the corresponding elements indexed by I. ℓ_1, ℓ_2 and ℓ_∞ norms of $\mathbf{x}$ are respectively denoted by $||\mathbf{x}||_1 = \sum_{i=1}^n |x_i|$, $||\mathbf{x}||_2 = \sqrt{\sum_{i=1}^n x_i^2}$ and $||\mathbf{x}||_\infty = \max_{i=1}^n |x_i|$. $\mathbf{x} \odot \mathbf{y} (\mathbf{x}, \mathbf{y} \in \mathbb{R}^n)$ is the element-wise product of $\mathbf{x}$ and $\mathbf{y}$. $\mathbf{y} = \text{sign}(\mathbf{x}) \Leftrightarrow \forall i, y_i = \text{sign}(x_i)$ and $\mathbf{y} = |\mathbf{x}| \Leftrightarrow \forall i, y_i = |x_i|$.

2.2 Problem Statement

The ℓ_1-regularized least squares problem is formulated as the following optimization problem:

$$\min_{\mathbf{x} \in \mathbb{R}^n} \mathcal{P}(\mathbf{x}) = \frac{1}{2}||A\mathbf{x} - \mathbf{b}||_2^2 + \tau||\mathbf{x}||_1, \tag{1}$$

where $A = [\mathbf{a}_1^T; \cdots; \mathbf{a}_m^T] (\mathbf{a}_i \in \mathbb{R}^n, i = 1, \cdots, m)$ is an $m \times n$ data matrix; $\mathbf{b}$ is a regression vector; τ is a tradeoff parameter. In this paper, we focus on the case of $m \geq n$, in which we assume $A^T A$ has full rank. This assumption is also introduced by Kim and Park [2010] when their BP algorithm is proposed. The full rank assumption is usually reasonable in practical applications. For instance, in the polynomial curve-fitting problem [Bishop, 2006], the number of samples is usually larger than the order of the polynomial, in which $A^T A$ has full rank. In the classification task, each row of A is a sample and b_i is the corresponding label. When the data is large-scale but not so high dimensional, the assumption that $A^T A$ has full rank usually holds. Moreover, in the wavelet-based image deblurring problems [Beck and Teboulle, 2009], $A^T A$ always has full rank. Besides, in the gaussian graphical model structure learning problem [Friedman *et al.*, 2008], the ℓ_1-regularized least squares is a subproblem, in which the full rank assumption is satisfied.

2.3 Dual Problem of ℓ_1-regularized Least Squares

In this subsection, we derive a new dual problem of Eq. (1). By introducing an auxiliary variable $\mathbf{y}$, we reformulate Eq. (1) as the following constrained optimization problem:

$$\min_{\mathbf{x}, \mathbf{y} \in \mathbb{R}^n} \frac{1}{2}||A\mathbf{x} - \mathbf{b}||_2^2 + \tau||\mathbf{y}||_1 \quad s.t. \quad \mathbf{x} = \mathbf{y}. \tag{2}$$

We show in the following proposition that this constrained optimization problem holds the strong duality property.

Proposition 1 *Strong duality of Eq. (2) holds.*

Proof Let I_n and $O_{m \times n}$ be an $n \times n$ identity matrix and an $m \times n$ zero matrix, respectively. Denote

$$\mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}, \; C = [A, O_{m \times n}], \\ W = [O_{n \times n}, I_n], \; E = [I_n, -I_n]. \tag{3}$$

Then, Eq. (2) can be equivalently formulated as follows:

$$\min_{\mathbf{z} \in \mathbb{R}^{2n}} \frac{1}{2}||C\mathbf{z} - \mathbf{b}||_2^2 + \tau||W\mathbf{z}||_1 \quad s.t. \quad E\mathbf{z} = \mathbf{0}. \tag{4}$$

The objective function in Eq. (4) is convex and the affine equality constraint satisfies Slater's condition [Boyd and Vandenberghe, 2004]. Therefore, the strong duality of Eq. (4) holds and so does Eq. (2). □

Proposition 1 inspires us to solve the dual problem of Eq. (2). Let $\boldsymbol{\mu}$ be the Lagrange multiplier corresponding with the equality constraint and we get the Lagrange function as follows:

$$\mathcal{L}(\mathbf{x}, \mathbf{y}, \boldsymbol{\mu}) = \frac{1}{2}||A\mathbf{x} - \mathbf{b}||_2^2 + \tau||\mathbf{y}||_1 + \boldsymbol{\mu}^T(\mathbf{x} - \mathbf{y}). \tag{5}$$

By minimizing Eq. (5) with respect to the primal variables $\mathbf{x}, \mathbf{y}$, we obtain the dual objective function as follows:

$$\tilde{\mathcal{D}}(\boldsymbol{\mu}) = \min_{\mathbf{x} \in \mathbb{R}^n} \left\{ \frac{1}{2}||A\mathbf{x} - \mathbf{b}||_2^2 + \boldsymbol{\mu}^T\mathbf{x} \right\} \\ - \max_{\mathbf{y} \in \mathbb{R}^n} \left\{ \boldsymbol{\mu}^T\mathbf{y} - \tau||\mathbf{y}||_1 \right\}. \tag{6}$$

Based on the dual norm introduced in [Boyd and Vandenberghe, 2004], we obtain

$$\max_{\mathbf{y} \in \mathbb{R}^n} \left\{ \boldsymbol{\mu}^T\mathbf{y} - \tau||\mathbf{y}||_1 \right\} = \begin{cases} 0, & ||\boldsymbol{\mu}||_\infty \leq \tau, \\ +\infty, & ||\boldsymbol{\mu}||_\infty > \tau. \end{cases} \tag{7}$$

Then, by maximizing Eq. (6) with respect to the dual variable $\boldsymbol{\mu}$, we get the dual problem as follows:

$$\max_{\boldsymbol{\mu} \in \mathbb{R}^n} \left\{ \tilde{\mathcal{D}}(\boldsymbol{\mu}) = -\frac{1}{2}\boldsymbol{\mu}^T H \boldsymbol{\mu} + (HA^T b)^T \boldsymbol{\mu} + const \right\} s.t. ||\boldsymbol{\mu}||_\infty \leq \tau, \tag{8}$$

where $H = (A^T A)^{-1}$ and $const$ is a constant term. We convert the above maximum problem to the following minimum problem by inverting the sign of $\tilde{\mathcal{D}}(\boldsymbol{\mu})$ and omitting the constant term:

$$\min_{\boldsymbol{\mu} \in \mathbb{R}^n} \left\{ \mathcal{D}(\boldsymbol{\mu}) = \frac{1}{2}\boldsymbol{\mu}^T H \boldsymbol{\mu} - (HA^T b)^T \boldsymbol{\mu} \right\} s.t. -\tau \leq \mu_i \leq \tau. \tag{9}$$

Now, we have transformed the non-differentiable ℓ_1-regularized least squares problem into a quadratic programming problem with a simple box constraint, which is essential to design an efficient optimization algorithm. We should mention that the dual problem is still an optimization problem over an n-dimensional vector space, although we introduce an auxiliary variable. If we can efficiently obtain the optimal solution $\boldsymbol{\mu}^\star$ of dual problem Eq. (9), then we get the optimal solution of the primal problem Eq. (1) via

$$\mathbf{x}^\star = H(A^T b - \boldsymbol{\mu}^\star). \tag{10}$$

As a matter of fact, before our new dual problem, some literatures [Kim *et al.*, 2007; Wright *et al.*, 2009; Kim and Park, 2010] have given the following dual problem:

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^m} \frac{1}{2}\boldsymbol{\alpha}^T\boldsymbol{\alpha} - \mathbf{b}^T\boldsymbol{\alpha} \quad s.t.\ ||A^T\boldsymbol{\alpha}||_\infty \le \tau. \tag{11}$$

The constraint in Eq. (11) is more complex than the constraint in Eq. (9), since $A^T\boldsymbol{\alpha}$ makes all the components (i.e., α_is) of $\boldsymbol{\alpha}$ coupled with each other, hence developing an efficient algorithm to solve Eq. (11) is not so easy. Thus, fewer algorithm is designed by solving the dual problem Eq. (11) directly. Kim *et al.* [2007] and Wright *et al.* [2009] use Eq. (11) to obtain the duality gap and some other properties. Kim and Park [2010] exploit Eq. (11) to get KKT conditions, but not directly solve it. However, the box constraint in Eq. (9) indicates that the components of $\boldsymbol{\mu}$ aren't coupled with each other, which is a very good property for designing an efficient algorithm.

2.4 Algorithm

We design an efficient algorithm called Dual Projected Newton Method (DPNM) to solve the dual problem Eq. (9). Generally speaking, Eq. (9) can be solved by a projected gradient method [Bertsekas, 1999], that is, generating a sequence $\{\boldsymbol{\mu}^k\}$ by $\boldsymbol{\mu}^{k+1} = [\boldsymbol{\mu}^k - \eta^k\nabla\mathcal{D}(\boldsymbol{\mu}^k)]^+$, where $[\mathbf{x}]^+ = \text{sign}(\mathbf{x}) \odot \min(\tau, |\mathbf{x}|)$ is the Euclidean projection of $\mathbf{x}$ onto the box constraint in Eq. (9). However, the convergence rate of the projected gradient method is at most linear. Subsequently, we focus on the following iterations by incorporating the idea of Newton's method, aiming to achieve a quadratic convergence rate:

$$\boldsymbol{\mu}^{k+1} = [\boldsymbol{\mu}^k - \eta^k M^k\nabla\mathcal{D}(\boldsymbol{\mu}^k)]^+, \tag{12}$$

where M^k is a positive definite matrix containing the second-order information and η^k is a step size. How to choose the positive definite matrix M^k is quite critical to the performance of an algorithm. In the unconstrained optimization problem, M^k can be simply chosen as the inverse of the Hessian matrix, but Bertsekas [1982] shows that, in the constrained optimization problem, choosing M^k as the inverse of Hessian matrix can't guarantee objective descent per iteration, even can't guarantee convergence. In the case of simple constraints (e.g., box constraint), Bertsekas [1982] suggests a class of matrices M^k for which objective descent and convergence can be guaranteed.

In the framework of [Bertsekas, 1982], we propose the projected Newton method to solve the dual problem Eq. (9) and we prove in the sequel that the convergence rate is at least quadratic. Define

$$\begin{aligned} I^k = &\{i \mid -\tau \le \mu_i^k \le -\tau + \epsilon^k, (\nabla\mathcal{D}(\boldsymbol{\mu}^k))_i > 0 \\ &\text{or } \tau - \epsilon^k \le \mu_i^k \le \tau, (\nabla\mathcal{D}(\boldsymbol{\mu}^k))_i < 0\}, \end{aligned} \tag{13}$$

where $\epsilon^k = \min(||\boldsymbol{\mu}^k - [\boldsymbol{\mu}^k - \nabla\mathcal{D}(\boldsymbol{\mu}^k)]^+||_2, \epsilon)$ and ϵ is a small positive scalar. Denote

$$H_{ij}^k = \begin{cases} 0, & i \in I^k \text{ or } j \in I^k,\ i \ne j, \\ H_{ij}, & \text{otherwise.} \end{cases} \tag{14}$$

Then, we obtain the following proposition:

Proposition 2 *H^k is a positive definite matrix.*

Proof $\forall\ \mathbf{x} \in \mathbb{R}^n, \mathbf{x} \ne \mathbf{0}$,

$$\begin{aligned} \mathbf{x}^T H^k \mathbf{x} &= \sum_{i\notin I^k}\sum_{j\notin I^k} H_{ij}x_i x_j + \sum_{i\in I^k} H_{ii}x_i^2 \\ &= \bar{\mathbf{x}}^T H \bar{\mathbf{x}} + \sum_{i\in I^k} H_{ii}x_i^2, \end{aligned} \tag{15}$$

where $\bar{x}_i = x_i$, if $i \notin I^k$, 0 otherwise. Under the assumption that A^TA has full rank, we get that H is positive definite. Then $H_{ii} > 0, \sum_{i\in I^k} H_{ii}x_i^2 \ge 0$ and $\bar{\mathbf{x}}^T H\bar{\mathbf{x}} \ge 0$. We note that $\mathbf{x} \ne \mathbf{0}$, so at least one of $\sum_{i\in I^k} H_{ii}x_i^2$ and $\bar{\mathbf{x}}^T H\bar{\mathbf{x}}$ is strictly positive. Thus, $\mathbf{x}^T H^k\mathbf{x} > 0$ is satisfied and the proposition is verified. □

In our method, M^k is chosen as $M^k = (H^k)^{-1}$, satisfying the condition that M^k is a positive definite matrix. Denote

$$\mathbf{p}^k = M^k\nabla\mathcal{D}(\boldsymbol{\mu}^k),\ \boldsymbol{\mu}^k(\eta) = [\boldsymbol{\mu}^k - \eta\mathbf{p}^k]^+. \tag{16}$$

The step size η^k is chosen as $\eta^k = \alpha^{m^k}$ with $\alpha \in (0,1)$, satisfying m^k is the first nonnegative integer m such that

$$\begin{aligned} &\mathcal{D}(\boldsymbol{\mu}^k) - \mathcal{D}\left(\boldsymbol{\mu}^k(\eta^k)\right) \ge \\ &\sigma\left\{\eta^k\sum_{i\notin I^k}\frac{\partial\mathcal{D}(\boldsymbol{\mu}^k)}{\partial\mu_i}p_i^k + \sum_{i\in I^k}\frac{\partial\mathcal{D}(\boldsymbol{\mu}^k)}{\partial\mu_i}[\mu_i^k - \mu_i^k(\eta^k)]\right\}, \end{aligned} \tag{17}$$

where σ is a constant. Then $\boldsymbol{\mu}^{k+1}$ is given by $\boldsymbol{\mu}^{k+1} = [\boldsymbol{\mu}^k - \eta^k\mathbf{p}^k]^+$. The detailed procedure of DPNM is listed in Algorithm 1. Next, we declare that Algorithm 1 not only guarantees convergence, but also has a quadratic convergence rate in the following theorem.

Theorem 1 *The sequence $\{\boldsymbol{\mu}^k\}$ generated by Algorithm 1 converges to the optimal solution $\boldsymbol{\mu}^\star$ and the convergence rate of $\{||\boldsymbol{\mu}^k - \boldsymbol{\mu}^\star||_2\}$ is at least quadratic.*

Proof The objective function $\mathcal{D}(\boldsymbol{\mu})$ in Eq. (9) is twice continuously differentiable and the Hessian matrix $\nabla^2\mathcal{D}(\boldsymbol{\mu}) = H$ is positive definite under the assumption that A^TA has full rank. Therefore, the objective function $\mathcal{D}(\boldsymbol{\mu})$ is convex and Eq. (9) has a unique solution $\boldsymbol{\mu}^\star$. Furthermore, there exists positive scalars λ_1, λ_2 (λ_1 and λ_2 are the smallest and largest eigenvalues of the Hessian matrix respectively.) such that $\lambda_1||\mathbf{z}||_2^2 \le \mathbf{z}^T\nabla^2\mathcal{D}(\boldsymbol{\mu})\mathbf{z} \le \lambda_2||\mathbf{z}||_2^2$ for any $\mathbf{z} \in \mathbb{R}^n$. According to the KKT conditions [Bertsekas, 1999], $\frac{\partial\mathcal{D}(\boldsymbol{\mu}^\star)}{\partial\mu_i} > 0$, if

Algorithm 1: DPNM-Dual Projected Newton Method

Input : $A \in \mathbb{R}^{m\times n},\ \mathbf{b} \in \mathbb{R}^n, \tau \in \mathbb{R},\ \boldsymbol{\mu_0} \in \mathbb{R}^n$

Initialize $\sigma = 10^{-3};\ \alpha = 0.5;\ \epsilon = 10^{-4}$;
Compute $H = (A^TA)^{-1};\ Atb = A^T\mathbf{b}$;
for $k = 0, 1, \cdots$ **do**
  $w^k = ||\boldsymbol{\mu}^k - [\boldsymbol{\mu}^k - \nabla\mathcal{D}(\boldsymbol{\mu}^k)]^+||_2$;
  $\epsilon^k = \min(w^k, \epsilon)$;
  Compute I^k by Eq. (13);
  Compute $\mathbf{p}^k, \boldsymbol{\mu}^k(\eta^k)$ by Eq. (16);
  $m = 0$;
  while *Eq. (17) is not satisfied* **do**
    $m = m + 1; \eta^k = \alpha^m$;
  end
  $\boldsymbol{\mu}^{k+1} = [\boldsymbol{\mu}^k - \eta^k\mathbf{p}^k]^+$;
  if *convergence criterion is satisfied* **then**
    $iter = k + 1$; break;
  end
end
$\boldsymbol{\mu}^\star = \boldsymbol{\mu}^{k+1};\ \mathbf{x}^\star = H(Atb - \boldsymbol{\mu}^\star)$;

Output: $\boldsymbol{\mu}^\star,\ \mathbf{x}^\star,\ iter$

$\mu_i^\star = -\tau$; $\frac{\partial\mathcal{D}(\boldsymbol{\mu}^\star)}{\partial\mu_i} < 0$, if $\mu_i^\star = \tau$. Obviously, $\nabla^2\mathcal{D}(\boldsymbol{\mu}) = H$ is Lipschitz continuous. Based on the properties above and the Proposition 4 (and its extensions) in [Bertsekas, 1982], Algorithm 1 converges to the optimal solution $\boldsymbol{\mu}^\star$ and the convergence rate of $\{||\boldsymbol{\mu}^k - \boldsymbol{\mu}^\star||_2\}$ is at least quadratic. □

The quadratic convergence rate is remarkable, which indicates that Algorithm 1 can achieve a δ-accurate solution $\boldsymbol{\mu}$ ($||\boldsymbol{\mu} - \boldsymbol{\mu}^\star||_2 \le \delta$) for Eq. (9) in $O(\log\log(1/\delta))$ steps.

2.5 Implementation Details

We present the implementation details of Algorithm 1, in which some practical techniques are utilized to reduce the cost per iteration.

- Since H and Atb are used several times in Algorithm 1, they can be computed when we initialize the algorithm, which greatly reduces the computational time. Here $O(n^3)$ operations are needed and we focus on the problem where n is not very large.
- The computational bottleneck in Algorithm 1 is for computing $\mathbf{p}^k$ in Eq. (16), which requires an inverse operation for an $n \times n$ matrix H^k. But in fact, we only need to compute the inverse of a smaller $r \times r$ matrix, where $r = n - \sharp(I^k)$. Denote

$$\tilde{I}^k = \{i | i \notin I^k, 1 \le i \le n\}, \tag{18}$$

$$B_{ij} = \begin{cases} H_{ii}, & i = j, 1 \le i \le n \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

Then there exists a permutation matrix P such that

$$G^k = PH^kP^T = \begin{bmatrix} H_{\tilde{I}^k} & \mathbf{0} \\ \mathbf{0} & B_{I^k} \end{bmatrix}. \tag{20}$$

Therefore,

$$(H^k)^{-1} = P^T(G^k)^{-1}P = P^T \begin{bmatrix} H_{\tilde{I}^k}^{-1} & \mathbf{0} \\ \mathbf{0} & B_{I^k}^{-1} \end{bmatrix} P. \tag{21}$$

- We note that

$$\begin{aligned} \mathbf{p}^k =& M^k\nabla\mathcal{D}(\boldsymbol{\mu}^k) = P^T \begin{bmatrix} H_{\tilde{I}^k}^{-1} & \mathbf{0} \\ \mathbf{0} & B_{I^k}^{-1} \end{bmatrix} P\nabla\mathcal{D}(\boldsymbol{\mu}^k) \\ =& P^T \begin{bmatrix} H_{\tilde{I}^k}^{-1}\nabla\mathcal{D}(\boldsymbol{\mu}^k)_{\tilde{I}^k} \\ B_{I^k}^{-1}\nabla\mathcal{D}(\boldsymbol{\mu}^k)_{I^k} \end{bmatrix}. \end{aligned} \tag{22}$$

  To further reduce the computational cost, the inverse and multiplication operation $H_{\tilde{I}^k}^{-1}\nabla\mathcal{D}(\boldsymbol{\mu}^k)_{\tilde{I}^k}$ can be replaced by the Cholesky decomposition, which can be efficiently obtained requiring $O(r^3)$ $(r < n)$ operations. Moreover, B_{I^k} is a diagonal matrix, $B_{I^k}^{-1}\nabla\mathcal{D}(\boldsymbol{\mu}^k)_{I^k}$ requires only $O(n - r)$ operations.
- P is a permutation matrix, which indicates that PH^kP^T in Eq. (20) is just the permutation of the rows and columns for H^k, according to the index sets $\tilde{I}^k$ and I^k, without requiring any multiplication operation. This conclusion can be also applied to Eq. (22).

3 Experiments

3.1 Experimental Settings

All algorithms in the experiments are implemented in Matlab and they are tested on six real-world data sets: Extended Yale Face Database B (ExtYaleB), USPS, Yale Face Database B (YaleB), PIE, Isolet and dna. Their basic information is listed in Table 1 (m is the number of samples and n is the dimension. ExtYaleB and YaleB are respectively resized to 32×32 and 40×30.).

In the experiments, every row of A is a sample in each data set and b_i is the corresponding label. For the ℓ_1-regularized least squares problem in Eq. (1), the optimal solution $\mathbf{x}^\star$ equals $\mathbf{0}$ if $\tau \ge ||A^Tb||_\infty$ [Wright *et al.*, 2009]. In order to avoid a trivial solution, we set $\tau = tol \times ||A^Tb||_\infty$ with different tols ($tol = 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}$).

To demonstrate the effectiveness and efficiency of DPNM, we compare DPNM with four state of the art methods: BPR[3] [Kim and Park, 2010], L1LS[4] [Kim *et al.*, 2007], FISTA[5] [Beck and Teboulle, 2009] and SpaRSA[6] [Wright *et al.*, 2009]. BPR is an active-set-type algorithm based on obtaining KKT conditions from Eq. (11). L1LS, FISTA and SpaRSA directly solve the ℓ_1-regularized least squares problem in Eq. (1). Since different algorithms have different terminated criteria, for fair comparisons, we first run the L1LS algorithm until its relative duality gap is less than or equal to 10^{-4} and then record its objective value $\mathcal{P}^\star$ in Eq. (1). We next run DPNM, SpaRSA, FISTA and BPR until the objective

[3] BPR is an improved algorithm based on BP.

[4] http://www.stanford.edu/~boyd/l1_ls/

[5] http://www.eecs.berkeley.edu/~yang/software/l1benchmark/

[6] http://www.lx.it.pt/~mtf/SpaRSA/

Table 1: The average computational time (The time of computing H and Atb for DPNM is also included.) and the average number of iterations over 10 independent executions (with different random starting points) for solving the ℓ_1-regularized least squares problem. The regularized parameter τ is set as $tol \times ||A^T\mathbf{b}||_\infty$. More details about the results are shown in the text.

data set	size $m \times n$	coeff tol	average time(CPU seconds) DPNM	BPR	L1LS	FISTA	SpaRSA	♯ average iterations DPNM	BPR	L1LS	FISTA	SpaRSA
ExtYaleB	2414 × 1024	10^{-1}	**0.9062**	2.0817	8.7543	18.3742	2.9490	**15.0**	83.0	46.0	3258.7	526.5
		10^{-2}	1.0378	**0.8750**	26.8962	42.9980	6.0558	18.0	**16.0**	35.0	8037.7	1082.1
		10^{-3}	**0.4971**	1.0604	92.1205	49.8914	8.5170	**10.0**	**10.0**	37.0	10000.0	1564.3
		10^{-4}	**0.2746**	1.6095	279.9859	50.1130	55.3159	**3.0**	9.0	38.0	10000.0	10001.0
		10^{-5}	**0.2710**	2.1223	325.1329	50.1931	57.0251	**2.0**	10.0	30.0	10000.0	10001.0
Isolet	7797 × 617	10^{-1}	**0.3560**	2.2915	3.1007	0.6595	0.5114	**12.0**	53.0	33.0	31.6	36.2
		10^{-2}	**0.2544**	0.9970	15.3178	1.2751	0.9600	**8.0**	12.0	54.0	92.2	79.9
		10^{-3}	**0.2031**	1.1891	21.0388	3.3981	4.9525	**6.0**	9.0	35.0	283.0	412.0
		10^{-4}	**0.1649**	2.0951	29.2915	14.3518	86.9161	**3.0**	11.0	37.0	1323.6	7098.0
		10^{-5}	**0.1513**	1.8817	31.0686	47.2366	123.6213	**1.0**	9.0	32.0	4400.4	10001.0
PIE	11154 × 1024	10^{-1}	**1.2425**	126.5265	91.2420	74.9511	7.8936	**14.0**	720.0	115.0	2687.0	251.0
		10^{-2}	**1.0813**	3.0447	106.5602	204.9801	7.0325	**11.0**	14.0	70.0	7428.6	219.7
		10^{-3}	**0.9106**	3.7257	206.4877	275.4339	28.7151	**10.0**	**10.0**	32.0	10000.0	956.0
		10^{-4}	**0.5900**	5.3669	1332.5470	275.4096	287.1595	**3.0**	9.0	62.0	10000.0	9374.9
		10^{-5}	**0.5598**	7.1639	1463.5147	275.4168	320.7441	**1.0**	10.0	35.0	10000.0	10001.0
USPS	7291 × 256	10^{-1}	**0.0617**	0.2466	1.9089	0.2585	0.1741	**8.0**	11.0	28.0	57.9	33.6
		10^{-2}	**0.0595**	0.3149	3.9277	0.7283	0.2934	**6.0**	10.0	25.0	212.3	57.2
		10^{-3}	**0.0421**	0.3917	5.2646	0.7591	0.7786	**4.0**	9.0	18.0	227.1	166.8
		10^{-4}	**0.0432**	0.4112	5.2152	2.8532	21.6712	**2.0**	8.0	12.0	930.2	4504.2
		10^{-5}	**0.0372**	0.4825	6.5782	10.2410	48.6217	**1.0**	9.0	12.0	3251.0	10001.0
YaleB	5850 × 1200	10^{-1}	**2.8930**	62.8636	28.9774	58.9738	4.7258	**26.0**	968.0	39.0	3443.2	256.5
		10^{-2}	**1.8639**	2.7727	81.4151	169.9488	20.0788	**18.0**	22.0	43.0	10000.0	1162.1
		10^{-3}	**1.7421**	2.9562	139.2338	170.2495	33.2614	**17.0**	19.0	32.0	10000.0	2017.5
		10^{-4}	**0.9744**	2.7449	378.8279	169.9587	155.3790	**10.0**	11.0	26.0	10000.0	9052.3
		10^{-5}	**0.6419**	3.4733	1295.9871	169.7176	185.0566	**6.0**	9.0	41.0	10000.0	10001.0
dna	3186 × 180	10^{-1}	**0.0135**	0.0776	0.3058	0.0255	0.0172	**2.0**	7.0	22.0	23.6	17.0
		10^{-2}	**0.0158**	0.0826	0.4766	0.0424	0.0363	**2.0**	6.0	23.0	58.7	43.8
		10^{-3}	**0.0164**	0.0871	0.5249	0.0956	0.0660	**2.0**	6.0	17.0	184.3	82.4
		10^{-4}	**0.0138**	0.0879	0.3110	0.1936	0.0904	**1.0**	6.0	17.0	431.3	121.4
		10^{-5}	**0.0161**	0.0832	0.2097	0.4012	0.1061	**1.0**	6.0	13.0	920.6	140.3

value in Eq. (1) are less than or equal to $\mathcal{P}^\star$, or the iterative steps exceeds 10000. We independently run the five algorithms 10 times respectively with different random starting points. To further speed up SpaRSA and FISTA algorithms, the continuation technique [Hale *et al.*, 2007] is adopted.

3.2 Experimental Analysis

The average computational time (The time of computing H and Atb for DPNM is also included.) and the average number of iterations over 10 independent executions (with different random starting points) are listed in Table 1. From these results, we can get: (a) DPNM is the most efficient among all the algorithms, both on CPU time and iterative steps, especially for small τs. (b) The CPU time (iterative steps) for DPNM tends to become less when the regularization parameter τ decreases. When τ is small, the feasible region in Eq. (9) becomes small. Thus, it's easier and cheaper to find the optimal solution in a smaller feasible region. On the contrary, L1LS, FISTA and SpaRSA tends to consume more CPU time and iterative steps when τ becomes smaller and this phenomenon can be also observed in [Wright *et al.*, 2009]. For BPR, not such an obvious phenomenon is observed. On some data sets (e.g., PIE and YaleB), when $\tau = 10^{-1} \times ||A^T\mathbf{b}||_\infty$, the iterative steps and CPU time are very large, which indicates that BPR is not so stable when the regularization τ varies. This is due to the frequent variables exchanging among different groups. (c) For BPR, sometimes less iterative steps even leads to more CPU time. The computational cost for BPR varies greatly for each iteration, more iterative steps with cheaper cost per-iteration consuming less CPU time is possible. (d) Most of the time, the iterative steps for L1LS are much smaller than that for SpaRSA and FISTA, but the CPU time is more, since the computational cost per iteration for L1LS is more expensive than SpaRSA and FISTA. Whereas too large iterative steps for SpaRSA and FISTA make them less efficient than DPNM and BPR. (e) L1LS, SpaRSA and FISTA are very slow when τ is small, although continuation technique [Hale *et al.*, 2007] is adopted for SpaRSA and FISTA. For DPNM, the CPU time consumed doesn't vary too much when τ changes.

To further understand the accuracy of the solutions for different algorithms, we plot the average objective values of Eq. (1) for the five algorithms on the six real-world data sets as in Figure 1. From the figures, we can see that BPR and DPNM achieve the smallest objective values and they are very competitive most of the time. In Table 1, we find the iterative steps of SpaRSA and FISTA sometimes exceed 10000, which indicate that their objective values might be still larger than $\mathcal{P}^\star$ when the iterative steps reach 10000. Integrating Table 1 and Figure 1 together, we can see that when an algorithm's iterative steps exceed 10000, the corresponding objective value is truly larger than L1LS's objective value $\mathcal{P}^\star$, which confirms that our conclusion is correct.

In all, DPNM is very efficient and effective for solving the ℓ_1-regularized least squares problem: DPNM converges fast with an at least quadratic convergence rate, which requires

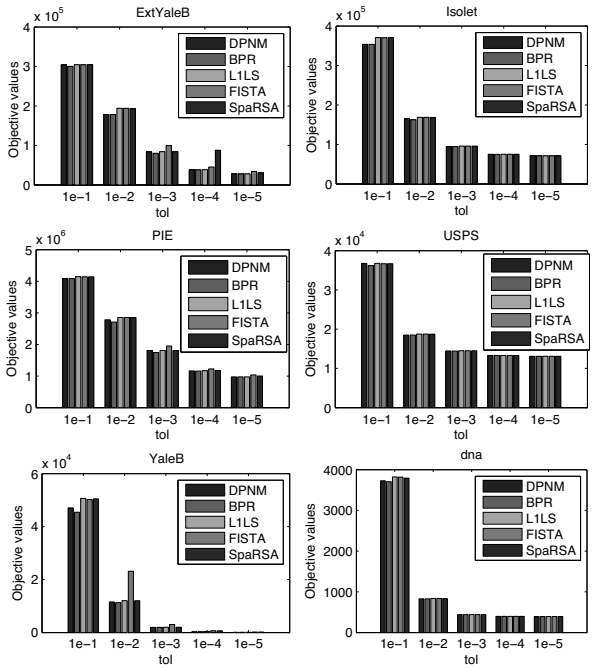

Figure 1: The bar plots of average objective values in Eq. (1) for different algorithms. Each subfigure shows the average objective value vs. tol bar on a real-world data set.

only a dozen even fewer steps, and consumes much less CPU time.

4 Conclusions

In this paper, a novel algorithm called DPNM is proposed to efficiently solve the ℓ_1-regularized least squares problem. Our main contribution is reflected in: (1) We derive a new dual form of the ℓ_1-regularized least squares as a box constrained quadratic optimization problem. (2) Due to the simple box constraint, a projected Newton method is applied to efficiently solve the new dual problem, achieving a *quadratic convergence rate*. Empirical studies on six real-world data sets demonstrate that DPNM is more efficient than several state of the art algorithms. A limitation of DPNM is that it works under the assumption that A^TA has full rank, and so does BPR. Whereas the other three algorithms (L1LS, FISTA, SpaRSA) can still work without the full rank assumption. In our future work, we will extend DPNM to the case without the full rank assumption.

Acknowledgments

This work is supported by NSFC (Grant No. 61075004 and No. 60835002).

References

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

[Bertsekas, 1982] Dimitri P. Bertsekas. Projected newton methods for optimization problems with simple constraints. *SIAM Journal on Control and Optimzation*, 20(2):221–246, 1982.

[Bertsekas, 1999] Dimitri P. Bertsekas. *Nonlinear Programming*. Athena Scientific, September 1999.

[Bishop, 2006] C.M. Bishop. *Pattern recognition and machine learning*. Springer New York, 2006.

[Boyd and Vandenberghe, 2004] S.P. Boyd and L. Vandenberghe. Convex optimization. 2004.

[Duchi and Singer, 2009] J. Duchi and Y. Singer. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, 10:2899–2934, 2009.

[Efron *et al.*, 2004] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least Angle Regression. *Analysis of Statistics*, 32(2):407–499, 2004.

[Figueiredo *et al.*, 2007] M.A.T. Figueiredo, R.D. Nowak, and S.J. Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journal on Selected Topics in Signal Processing*, 1(4):586–597, 2007.

[Friedman *et al.*, 2008] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432, 2008.

[Hale *et al.*, 2007] E.T. Hale, W. Yin, and Y. Zhang. A fixed-point continuation method for ℓ_1-regularized minimization with applications to compressed sensing. *CAAM TR07-07, Rice University*, 2007.

[Kim and Park, 2010] J. Kim and H. Park. Fast Active-set-type Algorithms for ℓ_1-regularized Linear Regression. In *The 13th International Conference on Artificail Intelligence and Statistics*, 2010.

[Kim *et al.*, 2007] S.J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale ℓ_1-regularized least squares. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):606–617, 2007.

[Lee *et al.*, 2007] H. Lee, A. Battle, R. Raina, and A.Y. Ng. Efficient sparse coding algorithms. *Advances in Neural Information Processing Systems*, 19:801, 2007.

[Nesterov, 1983] Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(\frac{1}{k^2})$. 27(2):372–376, 1983.

[Schmidt *et al.*, 2007] M. Schmidt, G. Fung, and R. Rosales. Fast optimization methods for ℓ_1 regularization: A comparative study and two new approaches. In *The 18th European Conference on Machine Learning*, pages 286–297. Springer, 2007.

[Tibshirani, 1996] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[Wright *et al.*, 2009] S.J. Wright, R. Nowak, and M. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.