Min-Norm Solutions to Group Lasso Problems

Karthik Seetharaman

June 2024

Abstract

The *group lasso problem*, helpful for achieving sparsity of solutions in situations where features can be naturally grouped together, has been extensively studied. However, in high-dimensional settings with many features, the minimum norm solution to this problem subject to an exact fit to the data is of interest, and this problem has not been thoroughly studied. In this paper, we use and compare several optimization methods such as ADMM and log-barrier methods to solve the min-norm group lasso problem. We theoretically derive their updates and compare their performance on randomly generated, noisy data. We find that the ADMM method far outperforms log-barrier methods on these experiments, although more robust experimentation on a larger scale is likely needed to make a definitive claim. All code is available at `https://github.com/kvscomputing/ee364bproject.git`.

1 Introduction

The standard group lasso problem is typically stated as follows. Given data $X \in \mathbb{R}^{n \times p}$ and $y \in \mathbb{R}^n$, we want to find $\beta \in \mathbb{R}^p$ to fit a regression $y = X\beta$ to this data. Furthermore, we have some predetermined grouping of the features $\beta = [\beta_{(1)}, \ldots, \beta_{(G)}]^T$, where $\beta \in \mathbb{R}^{p_i}$ for all $1 \leq i \leq G$, such that $\sum_{i=1}^{G} p_i = p$. We are thus interested in the optimization problem:

$$\min_{\beta} \frac{1}{2}||y - X\beta||_2^2 + \lambda \sum_{g=1}^{G} ||\beta_{(g)}||_2,$$

where $\sum_{g=1}^{G} ||\beta_{(g)}||_2$ is the so-called *group lasso penalty* to encourage keeping norms of the grouped features low.

If $p > n$ (as is often the case in high-dimensional settings), so that there exist β such that $y = X\beta$, we are interested in the *minimum-norm solution* to the group lasso problem - that is, the solution to

$$\min_{\beta} \sum_{g=1}^{G} ||\beta_{(g)}||_2 \text{ subject to } y = X\beta.$$

In what follows, we refer to this as the *min-norm group lasso problem.* This paper aims to survey different methods to solve the min-norm group lasso problem, theoretically deriving their updates and comparing their computation speed and convergence properties.

2 Related Work

There has been quite a bit of work on the lasso and group lasso problems, some of which we detail here. The original lasso problem was proposed in [6] as a way to encourage sparsity in solutions, only selecting the most important features. In [1], the authors introduce the LARS algorithm, an efficient forward selection algorithm for many problems including the Lasso problem. In [1] and [4], the authors characterize the solution path of Lasso (the solution as λ varies), which is known to be piecewise linear.

The group lasso problem [12] is less completely solved - similar fast algorithms to solve the group lasso problem have been proposed [9, 10, 11]. A software package called Adelie also exists to efficiently solve the problem for various GLMs, as developed by James Yang and Trevor Hastie [8]. Extending work on the solution path of the Lasso problem, Mishkin and Pilanci show continuity of the solution path of the group lasso problem, in addition to constructing an implicit function for the min-norm solution [3]. To the author's best knowledge, this is the only theoretical work on the min-norm group lasso problem: in particular, there has been little to no work on implementation of actual methods to solve the min-norm group lasso problem, either theoretically or in practice, a hole this work aims to fill.

Extensions to the group lasso have been considered as well, as in [5], which combines both an ℓ_1 and ℓ_2 penalty.

3 Algorithms

3.1 ADMM Method

The first algorithm we propose to solve the min-norm group lasso problem is a standard ADMM algorithm. We derive the algorithm here. Let $\mathcal{C} = \{z : y = Xz\}$; this is an affine set. Furthermore, let $I_{\mathcal{C}}(z) \in \{0, \infty\}$ be the indicator for $\mathcal{C}$. We can then rewrite the min-norm group lasso problem as

$$\min_{\beta, \alpha \in \mathbb{R}^p} \sum_{g=1}^{G} ||\beta_{(g)}||_2 + I_{\mathcal{C}}(\alpha) \text{ subject to } \beta - \alpha = 0.$$

The standard ADMM updates then proceed as follows: given iterates $\beta^k, \alpha^k, \gamma^k$, we update:

$$\beta^{k+1} = \arg\min_{\beta} \left(\sum_{g=1}^{G} ||\beta_{(g)}||_2 + \frac{\rho}{2} ||\beta - \alpha^k + \gamma^k||_2^2 \right)$$

$$\alpha^{k+1} = \Pi_{\mathcal{C}}(\beta^{k+1} + \gamma^k)$$

$$\gamma^{k+1} = \gamma^k + \beta^{k+1} - \alpha^{k+1}$$

The β-update can be recognized as the proximal operator of $h(z) = \sum_{g=1}^{G} ||z_{(g)}||_2$ with parameter $\lambda = \frac{1}{\rho}$ evaluated at $v = \alpha^k - \gamma^k$. Since the group lasso penalty is block-seaparable into G groups, it suffices to evaluate the proximal operator at each group separately. For each $g = 1, \ldots, G$, we can write

$$\text{prox}_{\lambda h}(v)_{(g)} = \left(1 - \frac{\lambda}{||v_{(g)}||_2}\right)_{+} v_{(g)}$$

[2].

The α-update is a projection onto the affine set $\mathcal{C}$, which is given by, for any vector v,

$$\Pi_{\mathcal{C}}(v) = v - X^T (XX^T)^{-1} (Xv - y).$$

This gives us the ADMM updates for the problem, and repeating these updates will lead to convergence.

3.2 Interior Point Method

We could also consider trying to use interior point methods; however, we cannot use interior point methods directly on the original problem as $\sum_{g=1}^{G} ||\beta_g||_2$ may be a non-differentiable objective in the case that certain groups are not selected (i.e. $\beta_{(g)} = 0$ for some $g = 1, \ldots, G$). To this end, we instead solve the dual problem, which we derive in the following subsection.

3.2.1 Dual Problem Derivation

We write the Lagrangian of the min-norm group lasso problem as

$$L(\beta, \nu) = \sum_{g=1}^{G} ||\beta_{(g)}||_2 + \nu^T(X\beta - y).$$

The Lagrange dual function is then given by

$$g(\nu) = \inf_{\beta} L(\beta, \nu) = \inf_{\beta} \left(\sum_{g=1}^{G} ||\beta_{(g)}||_2 + \nu^T(X\beta - y) \right) = \inf_{\beta} \left(\sum_{g=1}^{G} ||\beta_{(g)}||_2 + \nu^T X\beta \right) - \nu^T y.$$

To find β that minimize the Lagrangian, we must find β such that $0 \in \partial_\beta L(\beta, \nu)$. To do this, decompose

$$X\beta = \sum_{g=1}^{G} X_{(g)}\beta_{(g)}, X_{(g)} \in \mathbb{R}^{n \times p_g},$$

where $X_{(g)}$ consists of the columns of X in group g, which will allow us to focus on each $\beta_{(g)}$ separately. In particular, this means

$$g(\nu) = \inf_{\beta} \left(\sum_{g=1}^{G} \nu^T(X_{(g)}\beta_{(g)}) + ||\beta_{(g)}||_2 \right) - \nu^T y = \sum_{g=1}^{G} \inf_{\beta_{(g)}} \left(\nu^T(X_{(g)}\beta_{(g)}) + ||\beta_{(g)}||_2 \right) - \nu^T y.$$

For each group $g = 1, \ldots, g$, 0 must be in the subdifferential set of $\nu^T X_{(g)}\beta_{(g)} + ||\beta_{(g)}||_2$ with respect to $\beta_{(g)}$. If $||\beta_{(g)}||_2 \neq 0$, the subgradient is equal to $X_{(g)}^T\nu + \frac{\beta_{(g)}}{||\beta_{(g)}||_2}$; else, the subgradient is equal to $\{X_{(g)}^T\nu + v : ||v||_2 \leq 1\}$. In particular, this implies that, if $||X_{(g)}^T\nu||_2 = 1$, we optimally set $\beta_{(g)} = -X_{(g)}^T\nu$, and if $||X_{(g)}^T\nu||_2 < 1$, we set $\beta_{(g)} = 0$ as well. If $||X_{(g)}^T\nu||_2 > 1$, 0 is not in the subdifferential set, so we restrict our attention to the case where $||\nu^T X_{(g)}||_2 \leq 1$ for all $g = 1, \ldots, G$.

Plugging this $\beta = [\beta_{(1)}, \ldots, \beta_{(G)}]$ back into the minimization objective, we get that

$$g(\nu) = -\nu^T y,$$

where $||X_{(g)}^T\nu||_2 \leq 1$ for all $g = 1, \ldots, G$. The dual problem is to maximize this over all $\nu \in \mathbb{R}^n$ satisfying the constraints, which is equivalent to the problem

$$\min_{\nu} \nu^T y \text{ subject to } ||X_{(g)}^T\nu||_2 \leq 1, g = 1, \ldots, G.$$

To get this back in terms of X, we can write, for each $g = 1, \ldots, G$, $X_{(g)}^T = G_{(g)}X^T$, where $G_{(g)} \in \mathbb{R}^{p_g \times p}$ is the *group selection matrix* where each row is all zeros except a 1 for the feature in the group. This means the dual problem is

$$\min_{\nu} \nu^T y \text{ subject to } ||G_{(g)}X^T\nu||_2 \leq 1, g = 1, \ldots, G.$$

This is now a linear program with convex constraints that we can solve via interior point methods.

3.2.2 Log-Barrier Method

We now derive the log-barrier method update for this problem. Rewrite the constraints as $||G_{(g)}X^T\nu||_2^2 - 1 \leq 0$ for $g = 1, \ldots, G$. Then, we can define, for a barrier parameter $t > 0$, the barrier problem

$$\min_{\nu} \nu^T y - \frac{1}{t} \sum_{g=1}^{G} \log(1 - ||G_{(g)}X^T\nu||_2^2).$$

For fixed t, we can solve this problem via (truncated) Newton's Method. In particular, the log-barrier method works as follows. Beginning at some initial $t = t_{\text{init}}$, we repeat the following for some μ:

1. Solve the barrier problem for the current t using some variation of Newton's Method.
2. Set $t := \mu t$.

This sends $t \to \infty$, making the impact of the log-barriers smaller and smaller and eventually leading to convergence. A typical value of μ is $\mu = 1.5$, which we use in this paper.

3.3 Solving The Barrier Problem

We use Newton's Method to solve the barrier problem at each step. This involves, at each step, calculating the Newton descent direction $\Delta x_{\text{nt}} = -\nabla^2 f(x)^{-1} \nabla f(x)$. This requires taking the gradient and Hessian of the log-barrier objective, which we provide below. Let

$$f(\nu) = \nu^T y - \frac{1}{t} \sum_{g=1}^{G} \log(1 - ||G_{(g)} X^T \nu||_2^2)$$

be the log-barrier objective. We then calculate

$$\nabla_\nu f(\nu) = y + \frac{2}{t} \sum_{g=1}^{G} \frac{X G_{(g)}^T G_{(g)} X^T \nu}{1 - ||G_{(g)} X^T \nu||_2^2}$$

and

$$\nabla^2 f(x) = \frac{2}{t} \sum_{g=1}^{G} \frac{X G_{(g)}^T G_{(g)} X^T}{1 - ||G_{(g)} X^T \nu||_2^2} + \frac{2(X G_{(g)}^T G_{(g)} X^T \nu)(X G_{(g)}^T G_{(g)} X^T \nu)^T}{(1 - ||G_{(g)} X^T \nu||_2^2)^2}.$$

Notice that the terms $X G_{(g)}^T G_{(g)} X^T$ are ubiquitous in these expressions, so during implementation, we precompute and store $X G_{(g)}^T G_{(g)} X^T$ for all $g = 1, \ldots, G$.

To calculate a step size which ensures actual descent in the objective, we use the *Armijo line search*, which is to begin from $t = 1$ and, while $f(x + t\Delta x) > f(x) + \alpha t \nabla f(x)^T \Delta x$, set $t := \beta t$. In our experiments, we use $\alpha = 0.25, \beta = 0.5$.

The traditional Newton's Method involves solving for the Newton descent direction exactly. This may be expensive, so it is common to perform a *truncated Newton's Method*, which uses the conjugate gradients algorithm with some set number of maximum iterations to calculate an approximate descent direction.

In general, computing the Hessian can be quite expensive, so we also experiment with an *approximate Newton's Method* where we only recompute the Hessian every 10 steps. We call this the *Limited Hessian Truncated Newton's Method* going forward.

4 Experiments

4.1 Data

We randomly generated data $X \in \mathbb{R}^{200 \times 500}$ (so $n = 200, p = 500$) by generating n entries from the multivariate normal distribution with mean $0 \in \mathbb{R}^p$ and covariance matrix $\Sigma \in \mathbb{R}^{p \times p}$, which has 1s on the main diagonal and ρ everywhere else for some $0 < \rho < 1$. For this experiment, we set $\rho = 0.5$.

We sample β by sampling each component separately from a standard normal distribution $\mathcal{N}(0, 1)$. To generate the outputs y, we sample $\varepsilon \sim \mathcal{N}(0, 1)$ and set $\sigma = \frac{||X\beta||_2}{\sqrt{3}}$ (for a signal-to-noise ratio of 3). We then set $y = X\beta + \sigma\varepsilon$. We group the features randomly into groups of size 10 (so there are 50 groups total).

We use CVXPY to calculate the ground truth optimal value for this problem, which we find to be 1366.956.

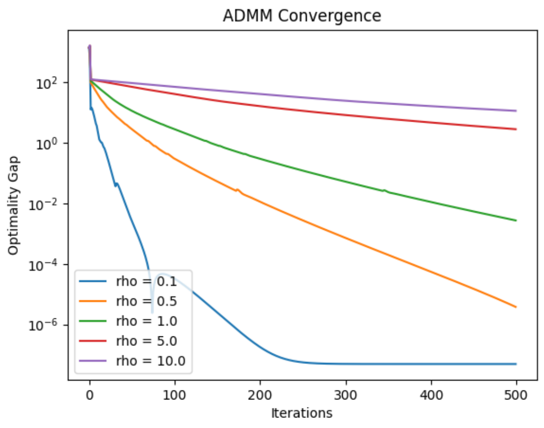

Figure 1: Testing different values of ρ for the ADMM method.

4.2 Sensitivity Analyses

We first perform a few sensitivity analyses to find optimal values for parameters. We test convergence for $\rho = 0.1, 0.5, 1, 5, 10$ in the ADMM method. The results are shown in Figure 1 - we see that $\rho = 0.1$ provides the best convergence to the objective, so we use $\rho = 0.1$ in ADMM tests going forward.

Similarly, for the log-barrier method, we use a truncated Newton's Method to solve the barrier problem for each t. Thus, we perform a sensitivity analysis on the maximum number of iterations for the CG step in calculating the truncated Newton descent direction. We test values of 1, 5, 10, 20, 30, and 50 iterations. The results are shown in Figure 2; as expected, increasing the maximum number of iterations increases convergence accuracy. Table 1 shows the amount of time it takes to run 50 iterations of the log-barrier method with each value of maxiter for the CG step. Going forward, we use 20 as the maxiter parameter.

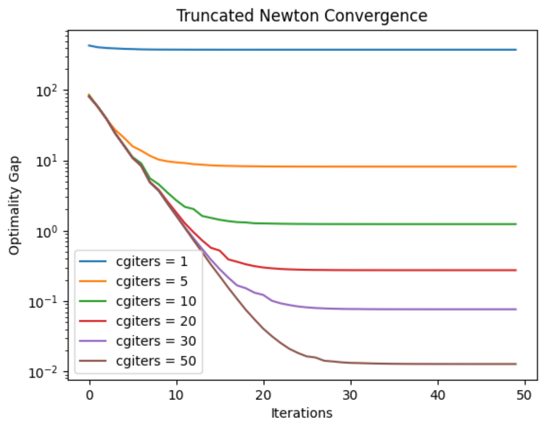

Figure 2: Testing different numbers of maximum iterations for the CG step in the truncated Newton's Method.

4.3 Comparing Methods

Figure 3 compares the convergence of standard Newton's Method, truncated Newton's Method, and the limited Hessian truncated Newton's Method. We see that the limited Hessian Truncated Newton's Method and standard truncated Newton's Method exhibit identical convergence; however, there is very little computation

Max. Iterations For CG Step	Time To Run 50 Iterations (s)
1	19.29
5	16.21
10	12.35
20	11.78
30	12.41
50	12.61

Table 1: Computation times for different numbers of maximum iterations for the CG step in the truncated Newton's Method.

time speedup between the two as well, likely since the precomputation of $XG_{(g)}^T G_{(g)} X^T$ for $g = 1, \ldots, G$ makes the Hessian computation much less expensive than it would be otherwise.

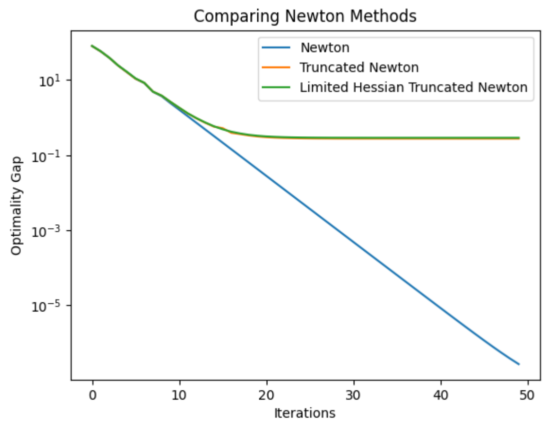

Figure 3: Comparing convergence for different variations of Newton's Method.

To compare the actual convergence times of different methods, refer to Figure 4. We see that the ADMM method converges much faster than either the truncated Newton method or standard Newton method and with high accuracy, so the ADMM method seems to be the best choice for this problem, in general.

5 Conclusion

This work surveys applying standard methods such as ADMM and the log-barrier method to the min-norm group lasso problem in a high-dimensional setting of $p > n$, filling a hole in the literature as this problem is relatively unstudied. After theoretically deriving the updates for the ADMM and log-barrier interior point methods to solve the dual problem, we ran experiments on randomly generated data to explore the effect of changing parameters such as ρ in ADMM and the maximum number of CG iterations in the truncated Newton's Method. After these sensitivity analyses, we compared these methods to each other to find that, both in accuracy and convergence speed, the ADMM method is the preferred method to solve the min-norm group lasso problem.

There are many avenues for future work. Repeating the analysis in this paper on much larger synthetic data, or on real data, would be useful to see how well methods scale into the real world. There are several other methods worth exploring, such as quasi-Newton methods (detailed in the Appendix) and interior point methods on the primal problem rather than the dual problem. Another avenue worth exploring is the case of $n > p$, where an exact fit to the regression may not exist. In this case, solving the problem

$$\min_{\beta} \sum_{g=1}^{G} ||\beta_{(g)}||_2 \text{ subject to } \frac{1}{2n}||y - X\beta||_2^2 \leq b^2$$

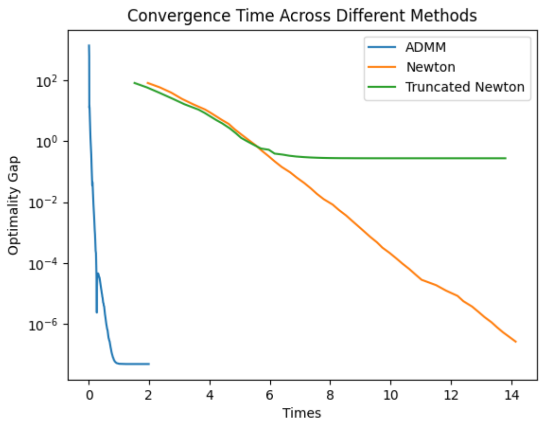

Figure 4: Comparing convergence times across methods.

for some chosen error tolerance b could be of interest - preliminary work in this direction is in the appendix.

6 Acknowledgements

We would like to acknowledge Professor Mert Pilanci, Aaron Mishkin, and Indu Subramaniam for a great quarter in EE 364B! We would also like to thank Chris Mitchell for providing an implementation of the truncated Newton's Method for inspiration.

References

[1] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. 2004.

[2] Jan Klosa, Noah Simon, Pål Olof Westermark, Volkmar Liebscher, and Dörte Wittenburg. Seagull: lasso, group lasso and sparse-group lasso regularization for linear regression models via proximal gradient descent. *BMC bioinformatics*, 21:1–8, 2020.

[3] Aaron Mishkin and Mert Pilanci. The solution path of the group lasso. In *OPT 2022: Optimization for Machine Learning (NeurIPS 2022 Workshop)*, 2022.

[4] Michael R Osborne, Brett Presnell, and Berwin A Turlach. A new approach to variable selection in least squares problems. *IMA journal of numerical analysis*, 20(3):389–403, 2000.

[5] Noah Simon, Jerome Friedman, Trevor Hastie, and Robert Tibshirani. A sparse-group lasso. *Journal of computational and graphical statistics*, 22(2):231–245, 2013.

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

[7] Stephen J Wright. Numerical optimization, 2006.

[8] James Yang and Trevor Hastie. Adelie. `https://github.com/JamesYang007/adelie`, 2024.

[9] Yi Yang and Hui Zou. A fast unified algorithm for solving group-lasso penalize learning problems. *Statistics and Computing*, 25:1129–1141, 2015.

[10] Chun Yip Yau and Tsz Shing Hui. Lars-type algorithm for group lasso. *Statistics and Computing*, 27:1041–1048, 2017.

[11] Lei Yuan, Jun Liu, and Jieping Ye. Efficient methods for overlapping group lasso. *Advances in neural information processing systems*, 24, 2011.

[12] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 68(1):49–67, 2006.

7 Appendix: Additional Methods

In this section, we present preliminary work towards a few additional methods that were not implemented, but are still of interest.

7.1 BFGS Quasi-Newton Method

Quasi-Newton Methods are methods that avoid directly computing the Hessian at all, instead working with an approximation at each step. Among the most famous of these methods is the BFGS method [7]. In general, the method works as follows (to use it in this case, we can substitute in expressions for the objective and gradient of our barrier problem):

1. Begin with an initial guess x_0 and approximate *inverse* Hessian matrix H_0. Then, at each iterate x_k, repeat steps 2 through 6. Do this until convergence.
2. Compute $p_k = -H_k \nabla f(x_k)$ to get a descent direction.
3. Perform a Wolfe line search to find an acceptable step size α_k. In particular, begin at $\alpha_k = 1$ and set $\alpha_k = \beta\alpha_k$ for some β (usually 0.5) until both of the following hold:
   (a) $f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k p_k^T \nabla f(x_k)$
   (b) $-p_k^T \nabla f(x_k + \alpha_k p_k) \leq -c_2 p_k^T \nabla f(x_k)$

   For quasi-Newton methods, typical choices of c_1, c_2 are $c_1 = 10^{-4}$ and $c_2 = 0.9$ [7].
4. Set $s_k = \alpha_k p_k$, $x_{k+1} = x_k + s_k$.
5. Compute the change in gradient $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$.
6. Update the inverse Hessian approximation $H_{k+1} = H_k + \frac{s_k^T y_k + y_k^T H_k y_k)(s_k s_k^T)}{(s_k^T y_k)^2} - \frac{H_k y_k s_k^T + s_k y_k^T H_k}{s_k^T y_k}$.

I attempted to implement this method for comparison, but was not able to get the implementation fully working. Attempts are in the GitHub.

7.2 ADMM For $n > p$

As alluded to in the introduction, in the case of $n > p$, it is of interest to prescribe an error tolerance b and consider the problem

$$\min_{\beta} \sum_{g=1}^{G} ||\beta_{(g)}||_2 \text{ subject to } \frac{1}{2n}||y - X\beta||_2^2 \leq b^2.$$

If we were to run an ADMM method on this problem, the only update that would change when compared to the method presented in the main paper is the α-update, which is replaced with a projection onto the convex set $\mathcal{C} = \{z : \frac{1}{2n}||y - Xz||_2^2 \leq b^2\}$. This is done by solving the optimization problem

$$\min_{z} \frac{1}{2}||z - v||_2^2 \text{ subject to } \frac{1}{2n}||y - Xz||_2^2 \leq b^2.$$

The Lagrangian of this is

$$L(z, \lambda) = \frac{1}{2}||z - v||_2^2 + \frac{\lambda}{2}\left(||y - Xz||_2^2 - 2nb^2\right).$$

Taking the gradient and setting to 0, we get

$$\nabla_z L(z, \lambda) = (z - v) + \lambda(X^T X z - y^T x) = 0 \implies z^*(\lambda) = (I - \lambda X^T X)^{-1}(v + y^T x).$$

Substituting this into the dual function leads to a really messy expression and taking the derivative leads to an equation one can solve by bisection search to find the optimal λ.