Application of Alternating Direction Method of Multipliers to Linear Ridge-Regularized Support Vector Machines

Simon Gibson

May 4, 2022

1 Introduction

In this project we develop a solver by the Alternating Direction Method of Multipliers (ADMM) for Support Vector Machines (SVM) of the form *loss* + *penalty*. Specifically, considering SVM's of the form:

$$\underset{\mathbf{w},b}{\text{minimize}} \sum_{i=1}^{N} \max(0, 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)) + \frac{\lambda}{2}||\mathbf{w}||_2^2 \tag{1}$$

where the loss is the hinge-loss, and the penalty term is a ridge regularization with parameter λ. ADMM is not easily applied to this form. In the next section we consider a reformulation of the problem which will be beneficial towards ADMM.

2 Problem Reformulation

In the pursuit of closed form solutions for ADMM, certain functions are more analytically friendly. Ideally, $f(x), g(z)$ should be easily differentiated. However, for those functions which are not differentiable, we try to reformulate them into isomorphic functions which have known minimization procedures (such as proximal mapping).

We aim to convert the above minimization problem into the general ADMM form:

$$\underset{\mathbf{x},\mathbf{z}}{\text{minimize}}\, f(\mathbf{x}) + g(\mathbf{z}) \text{ s.t. } \mathbf{Ax} + \mathbf{Bz} = \mathbf{c} \tag{2}$$

2.1 Sum Reformulation

Let us introduce the temporary constraint $a_i = (1 - y_i(\mathbf{w}^\top \mathbf{x} + b))_+$, then our the summation term of our objective function can be expressed as $\sum_{i=1}^{N}(a_i) = \mathbf{a} = \mathbf{1} - \mathbf{y}(\mathbf{X}\mathbf{W} + b\mathbf{1})$, where $\mathbf{1} \in \mathbb{R}^N$, $\mathbf{y} \in \mathbb{R}^N$, $\mathbf{X} \in \mathbb{R}^{p \times N}$, and $\mathbf{W} \in \mathbb{R}^p$. Let us introduce the following predefined variables in order to simplify $\mathbf{a}$.

$$\begin{aligned}
\mathbf{X}_1 &= \begin{bmatrix} \mathbf{X} & \mathbf{1} \end{bmatrix} \in \mathbb{R}^{N \times (p+1)} \\
\hat{\mathbf{I}} &= \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0}^\top & 0 \end{bmatrix} \in \mathbb{R}^{(p+1) \times (p+1)} \\
\mathbf{W}_b &= \begin{bmatrix} \mathbf{W} \\ b \end{bmatrix} \in \mathbb{R}^{(p+1) \times 1} \\
\mathbf{M} &= \mathbf{y} \odot \mathbf{X}_1 \in \mathbb{R}^{N \times (p+1)} \\
\mathbf{S} &= \mathbf{1} - \mathbf{M}\mathbf{W}_b \in \mathbb{R}^{N \times 1}
\end{aligned}$$

where $\mathbf{X}_1$ is our data matrix with an additional column of 1's, $\hat{\mathbf{I}}$ is the $\mathbf{I}_{p+1}$ matrix with the end of the diagonal set 0, $\mathbf{W}_b$ is the column vector which combines our weights and bias, $\mathbf{M}$ is the element-wise multiplication matrix of the labels y_i and the rows of $\mathbf{X}_1$, and $\mathbf{S}$ is the simplification of $\mathbf{a}$ using the above variables. Thus we have the new summation term of our objective equal to

$$\sum_{i=1}^{N}(a_i) = \sum_{i=1}^{N}(\mathbf{1} - \mathbf{y} \odot \mathbf{X}_1\mathbf{W})_+ = ||(\mathbf{S})_+||_1$$

2.2 Ridge Reformulation

Next, consider the ridge regularization term $||\mathbf{w}||_2^2 = \mathbf{w}^\top\mathbf{w}$. This form is reminiscent of the quadratic form $\frac{1}{2}\mathbf{x}^\top\mathbf{Q}\mathbf{x}$. Using the predefined variables $\mathbf{W}_b, \hat{\mathbf{I}}$ we consider the following calculations:

$$\begin{aligned}
\mathbf{W}_b^\top \hat{\mathbf{I}} \mathbf{W}_b &= \begin{bmatrix} \mathbf{w} & b \end{bmatrix} \begin{bmatrix} \mathbf{I}_p & \mathbf{0} \\ \mathbf{0}^\top & 0 \end{bmatrix} \begin{bmatrix} \mathbf{w} & b \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{w}^\top & 0 \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix} \\
&= \mathbf{w}^\top \mathbf{w} \\
&= ||\mathbf{w}||_2^2
\end{aligned}$$

Then we see we can reformulate the ridge regularization term of our objective as the modified quadratic term

$$\frac{\lambda}{2}||\mathbf{w}||_2^2 = \frac{\lambda}{2}\mathbf{W}_b^\top \hat{\mathbf{I}} \mathbf{W}_b$$

Dividing both terms of our newly found objective, we know have the reformulated problem

$$\underset{\mathbf{W}_b,\mathbf{S}}{\text{minimize}}\, \frac{1}{2}\mathbf{W}_b\hat{\mathbf{I}}\mathbf{W}_b + \frac{1}{\lambda}||\mathbf{S}_+||_1 \text{ s.t. } \mathbf{S} + \mathbf{M}\mathbf{W}_b = \mathbf{1} \tag{3}$$

where our constraint is given by the definition of $\mathbf{S}$. In order to affirm that this form is consistent with that of (2), we let $\mathbf{W}_b = \mathbf{x}, \mathbf{S} = \mathbf{z}, \mathbf{A} = \mathbf{M}, \mathbf{B} = \mathbf{I}, \mathbf{c} = \mathbf{1}$. Then it is easy to see that $f(\mathbf{W}_b) = \frac{1}{2}\mathbf{W}_b^\top\hat{\mathbf{I}}\mathbf{W}_b$ and $g(\mathbf{S}) = \frac{1}{\lambda}||(\mathbf{S})_+||_1$ and we have a problem to which ADMM is applicable.

3 ADMM Setup

3.1 Augmented Lagrangian

In order to find the update formulas for $\mathbf{W}_b, \mathbf{S}$, we consider the general augmented lagrangian for ADMM

$$\mathcal{L}_\beta(x, z, u) = f(x) + g(z) + u^\top(Ax + Bz - c) + \frac{\beta}{2}||Ax + Bz - c||_2^2$$

Application to our objective yields our augmented lagrangian

$$\mathcal{L}_\beta(\mathbf{W}_b, \mathbf{S}, \mathbf{u}) = \frac{1}{2}\mathbf{W}_b^\top\hat{\mathbf{I}}\mathbf{W}_b + \frac{1}{\lambda}||(\mathbf{S})_+||_1 + \mathbf{u}^\top(\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1}) + \frac{\beta}{2}||\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1}||_2^2 \tag{4}$$

3.2 Optimality Conditions

Next, we derive the Karush-Kuhn-Tucker (KKT) optimality conditions of constrained optimization for our problem. Notice that $\frac{1}{2}\mathbf{W}_b^\top\hat{\mathbf{I}}\mathbf{W}_b$ is easily differentiated, while $\frac{1}{\lambda}||(\mathbf{S})_+||_1$ is not differentiable (since it is non-smooth at $\mathbf{S} = \mathbf{0}$). Then our dual feasibility conditions will consist of a gradient, and a subgradient. Taking our primal feasibility condition as our constraint set equal to 0, we form the KKT system as:

Primal Feasibility: $\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1} = \mathbf{0}$

Dual Feasibility 1: $\hat{\mathbf{I}}\mathbf{W}_b + \mathbf{M}^\top\mathbf{u} = \mathbf{0}$

Dual Feasibility 2: $\mathbf{0} \in \partial g(\mathbf{S}) + \mathbf{I}^\top\mathbf{u}$

Where the gradient of the second dual feasibility condition is replaced with the subgradient.

4 Update Schemes

In this section, we derive the closed-form update solutions for $\mathbf{W}_b, \mathbf{S}$. According to ADMM, the update schemes are as follows

$$\begin{aligned}
\mathbf{W}_b^{(k+1)} &= \arg\min_{\mathbf{W}_b} \mathcal{L}_\beta(\mathbf{W}_b, \mathbf{S}^{(k)}, \mathbf{u}^{(k)}) \\
\mathbf{S}^{(k+1)} &= \arg\min_{\mathbf{S}} \mathcal{L}_\beta(\mathbf{W}_b^{(k+1)}, \mathbf{S}, \mathbf{u}^{(k)}) \\
\mathbf{u}^{(k+1} &= \mathbf{u}^{(k)} + \beta(\mathbf{S}^{(k+1)} + \mathbf{M}\mathbf{W}_b^{(k+1)} - \mathbf{1})
\end{aligned}$$

4.1 $\mathbf{W}_b$ Update Scheme

According to ADMM, the update of $\mathbf{W}_b$ is defined as

$$\mathbf{W}_b^{(k+1)} = \arg\min_{\mathbf{W}_b} \mathcal{L}_\beta(\mathbf{W}_b, \mathbf{S}^{(k)}, \mathbf{u}^{(k)})$$

Note that $f(\mathbf{W}_b)$ is differentiable with respect to $\mathbf{W}_b$. Then we do the following calculations (with some abuse of notation)

$$\begin{aligned}
\mathbf{0} &= \nabla_{\mathbf{W}_b}(\frac{1}{2}\mathbf{W}_b^\top \hat{\mathbf{I}} \mathbf{W}_b) + \nabla_{\mathbf{W}_b}(\mathbf{u}^\top(\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1}) + \frac{\beta}{2}||\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1}||_2^2) \\
&= \hat{\mathbf{I}}\mathbf{W}_b + \mathbf{M}^\top \mathbf{u} + \beta\mathbf{M}^\top(\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1}) \\
&= \hat{\mathbf{I}}\mathbf{W}_b + \mathbf{M}^\top \mathbf{u} + \beta\mathbf{M}^\top\mathbf{S} + \beta\mathbf{M}^\top\mathbf{M}\mathbf{W}_b - \beta\mathbf{M}^\top\mathbf{1} \\
&= \mathbf{W}_b(\hat{\mathbf{I}} + \beta\mathbf{M}^\top\mathbf{M}) + \beta\mathbf{M}^\top(\mathbf{S}\mathbf{1}) + \mathbf{M}^\top\mathbf{u} \\
&= \mathbf{W}_b(\hat{\mathbf{I}} + \beta\mathbf{M}^\top\mathbf{M}) + \mathbf{M}^\top(\beta(\mathbf{S} - \mathbf{1}) + \mathbf{u}) \\
&\implies \mathbf{W}_b(\hat{\mathbf{I}} + \beta\mathbf{M}^\top\mathbf{M}) = -\mathbf{M}^\top(\beta(\mathbf{S} - \mathbf{1}) + \mathbf{u})
\end{aligned}$$

So then we have final update

$$\mathbf{W}_b^{(k+1)} = (\hat{\mathbf{I}} + \beta\mathbf{M}^\top\mathbf{M})^{-1}\mathbf{M}^\top(\beta(\mathbf{1} - \mathbf{S}) - \mathbf{u}) \tag{5}$$

4.2 S Update Scheme

According to ADMM, the update of $\mathbf{S}$ is defined as

$$\mathbf{S}^{(k+1)} = \arg\min_{\mathbf{S}} \mathcal{L}_\beta(\mathbf{W}_b^{(k+1)}, \mathbf{S}, \mathbf{u}^{(k)})$$

Since $\frac{1}{\lambda}||(\mathbf{S})_+||_1$ is not differentiable like $\frac{1}{2}\mathbf{W}_b^\top\hat{\mathbf{I}}\mathbf{W}_b$ is, we consider a different approach. Consider (4) with respect to $\mathbf{S}$. Then we have the equation

$$\mathbf{S}^{(k=1)} = \arg\min_{\mathbf{S}} \frac{1}{\lambda}||(\mathbf{S})_+||_1 + \mathbf{u}^\top(\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1}) + \frac{\beta}{2}||\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1}||_2^2$$

Now expanding the lagrangian we get

$$\frac{1}{\lambda}||(\mathbf{S})_+||_1 + \mathbf{u}^\top(\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1}) + \frac{\beta}{2}(\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1})^\top(\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1})$$

Then using the matrix version of completing the square on the second and third term, which is defined as

$$x^\top I x - 2b^\top x = (x - I^{-1}b)^\top I(x - I^{-1}b) - b^\top I^{-1} b$$

Setting $x = \mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1}$, $b = \frac{-1}{2}\mathbf{u}$, we then have

$$\begin{aligned}\frac{\beta}{2}\mathbf{x}^\top\mathbf{x} - 2\mathbf{b}^\top\mathbf{x} &= \frac{\beta}{2}(\mathbf{x} - \mathbf{b})^\top(\mathbf{x} - \mathbf{b}) - \mathbf{b}^\top\mathbf{b} \\ &= \frac{\beta}{2}||\mathbf{x} - \mathbf{b}||_2^2 - ||\mathbf{b}||_2^2 \\ &= \frac{\beta}{2}||\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1} + \frac{1}{2}\mathbf{u}||_2^2 - \frac{1}{4}||\mathbf{u}||_2^2\end{aligned}$$

Since the second term is constant, it is irrelevant to the minimization of $\mathbf{S}$. Thus we have the update

$$\mathbf{S}^{(k+1)} = \arg\min_{\mathbf{S}} \frac{1}{\lambda}||(\mathbf{S})_+||_1 + \frac{\beta}{2}||\mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1} + \frac{1}{2}\mathbf{u}||_2^2$$

Dividing both terms by β and rearrangement leads to the following proximal mapping

$$\mathbf{S}^{(k+1)} = \arg\min_{\mathbf{S}} \frac{1}{\lambda\beta}||(\mathbf{S})_+||_1 + \frac{1}{2}||\mathbf{S} - (-\mathbf{M}\mathbf{W}_b + \mathbf{1} - \frac{1}{2}\mathbf{u})||_2^2 = \mathbf{prox}_{\frac{1}{\lambda\beta}||(\mathbf{S})_+||_1}(\mathbf{P})$$

Where $\mathbf{P} = -\mathbf{M}\mathbf{W}_b + \mathbf{1} - \frac{1}{2}\mathbf{u}$.Thus we have

$$\mathbf{S}^{(k+1)} = \mathbf{prox}_{\frac{1}{\lambda\beta}||(\mathbf{S})_+||_1}(\mathbf{P}) = \arg\min_{\mathbf{S}} \frac{1}{\lambda\beta}||(\mathbf{S})_+||_1 + \frac{1}{2}||\mathbf{S} - \mathbf{P}||_2^2 \tag{6}$$

Expansion of (6) into component form yields

$$\sum_{i=1}^{N} \frac{1}{\lambda\beta}\max(0, \mathbf{S}_i) + \frac{1}{2}(\mathbf{S}_i - \mathbf{P}_i)^2$$

which we perform piecewise component optimization on. Then

$$\mathbf{S}_i{}^{(k+1)} = \arg\min_{\mathbf{S}_i} \frac{1}{\beta\lambda}\max(0, \mathbf{S}_i) + \frac{1}{2}\mathbf{S}_i{}^2 - \mathbf{S}_i\mathbf{P}_i + \frac{1}{2}\mathbf{P}_i{}^2$$

$$\mathbf{S}_i{}^{(k+1)} = \begin{cases} \frac{1}{2}\mathbf{S}_i{}^2 - \mathbf{S}_i\mathbf{P}_i + \frac{1}{2}\mathbf{P}_i{}^2 & \mathbf{S}_i \leq 0 \\ \frac{1}{\beta\lambda}\mathbf{S}_i + \frac{1}{2}\mathbf{S}_i{}^2 - \mathbf{S}_i\mathbf{P}_i + \frac{1}{2}\mathbf{P}_i{}^2 & \mathbf{S}_i > 0 \end{cases}$$

Taking the partial yields

$$0 = \begin{cases} \mathbf{S}_i - \mathbf{P}_i & \mathbf{S}_i \leq 0 \\ \frac{1}{\beta\lambda} + \mathbf{S}_i - \mathbf{P}_i & \mathbf{S}_i > 0 \end{cases} \tag{7}$$

We thus consider the following cases:

4.2.1 $\mathbf{P}_i \leq 0$

Letting $\mathbf{P}_i \leq 0$, we have two sub cases to check. Let $\mathbf{S}_i \leq 0$, then $0 = \mathbf{S}_i - \mathbf{P}_i \implies \mathbf{S}_i = \mathbf{P}_i \leq 0$. Thus $\mathbf{S}_i$ agrees with (8) when $\mathbf{S}_i \leq 0$. Let $\mathbf{S}_i > 0$, then $0 = \mathbf{S}_i - \mathbf{P}_i \implies \mathbf{S}_i = \mathbf{P}_i < 0$. In this case $\mathbf{S}_i$ does not agree with (8). Then we have found that when $\mathbf{P}_i \leq 0$, $\mathbf{S}_i = \mathbf{P}_i$.

4.2.2 $\mathbf{P}_i > 0$

Letting $\mathbf{P}_i > 0$, we again have two sub cases. Let $\mathbf{S}_i \leq 0$, then $0 = \mathbf{S}_i - \mathbf{P}_i \implies \mathbf{S}_i = \mathbf{P}_i$ which disagrees in sign with our assumption. If we let $\mathbf{S}_i > 0$, then $0 = \mathbf{S}_i - \mathbf{P}_i + \frac{1}{\lambda\beta} \implies \mathbf{S}_i = \mathbf{P}_i - \frac{1}{\lambda}\beta$. The sign of $\mathbf{S}_i$ when $\mathbf{P}_i > 0$ is defined as

$$\text{sign}(\mathbf{S}_i) = \begin{cases} -1 & 0 < \mathbf{P}_i < \frac{1}{\lambda\beta} \\ 0 & \mathbf{P}_i = \frac{1}{\lambda\beta} \\ 1 & \mathbf{P}_i > \frac{1}{\lambda\beta} \end{cases}$$

From inspection, we can see that the sign of $\mathbf{S}_i$ only agrees with our assumption when $\mathbf{P}_i > \frac{1}{\lambda\beta}$, so $\mathbf{S}_i = \mathbf{P}_i - \frac{1}{\lambda\beta}$, and we must consider a third case.

4.2.3 $0 < \mathbf{P}_i < \frac{1}{\lambda\beta}$

AS with the previous two cases, we compare the signs of $\mathbf{S}_i$ to (8). We see that the sign disagrees in both cases. That is, the value of $\mathbf{S}_i$ which is a minimizer is outside the set which contains all values of correct signage. Thus the only ideal value for $\mathbf{S}_i$ is 0.

From the findings of our three sub cases, we formulate in the following way.

$$\mathbf{S}_i{}^{(k+1)} = \begin{cases} \mathbf{P}_i{}^{(k)} & \mathbf{P}_i{}^{(k)} < 0 \\ 0 & 0 \leq \mathbf{P}_i{}^{(k)} \leq \frac{1}{\lambda\beta} \\ \mathbf{P}_i{}^{(k)} - \frac{1}{\lambda\beta} & \mathbf{P}_i{}^{(k)} > \frac{1}{\lambda\beta} \end{cases} \tag{8}$$

5 Residuals

As we have found closed form solutions for our variables we minimize over, the last thing we need before implementing ADMM_SVM is our dual and primal residuals. Finding the primal residual is trivial, as it is defined as the violation of our equality constraint. Finding our dual residual is a bit more tricky. Noting that $g(\mathbf{S})$ is non-differentiable, we aim to find the dual residual using the augmented lagrangian without $g(\mathbf{S})$. Consider the optimality condition

$$\begin{aligned}
\mathbf{0} &= \nabla_{\mathbf{W}_b}\mathcal{L}_\beta(\mathbf{W}_b^{(k+1)}, \mathbf{S}^{(k)}, \mathbf{u}^{(k)}) \\
&= \hat{\mathbf{I}}\mathbf{W}_b + \mathbf{M}^\top \mathbf{u}^{(k)} + \beta \mathbf{M}^\top(\mathbf{S}^{(k)} + \mathbf{M}\mathbf{W}_b^{(k+1)} - \mathbf{1}) \\
&= \hat{\mathbf{I}}\mathbf{W}_b + \mathbf{M}^\top(\mathbf{u}^{(k+1)} - \beta(\mathbf{S}^{(k+1)} + \mathbf{M}\mathbf{W}_b^{(k+1)} - \mathbf{1})) + \beta \mathbf{M}^\top(\mathbf{S}^{(k)} + \mathbf{M}\mathbf{W}_b^{(k+1)} - \mathbf{1}) \\
&= \hat{\mathbf{I}}\mathbf{W}_b + \mathbf{M}^\top \mathbf{u}^{(k+1)} - \beta \mathbf{M}^\top(\mathbf{S}^{(k+1)} + \mathbf{M}\mathbf{W}_b^{(k+1)} - \mathbf{1}) + \beta \mathbf{M}^\top(\mathbf{S}^{(k)} + \mathbf{M}\mathbf{W}_b^{(k+1)} - \mathbf{1} \\
&= \hat{\mathbf{I}}\mathbf{W}_b + \mathbf{M}^\top \mathbf{u}^{(k+1)} + \beta \mathbf{M}^\top(\mathbf{S}^{(k)} - \mathbf{S}^{(k+1)})
\end{aligned}$$

From which we can see that the first two terms are the same as our KKT system, and thus the third term (which is not in the KKT system) must be the dual residual. Then our residuals are
Primal Residual: $r_p = \mathbf{S} + \mathbf{M}\mathbf{W}_b - \mathbf{1}$
Dual Residual: $r_d = \beta \mathbf{M}^\top(\mathbf{S}^{(k)} - \mathbf{S}^{(k+1)})$
And we can formulate our stopping conditions as

$$||r_p||_2 \leq \text{tol, and} ||r_d||_2 \leq tol \tag{9}$$

6 Implementation

Due to finding closed forms for $\mathbf{W}_b, \mathbf{S}$, we are able to implement the entire algorithm within a singular loop. The stopping conditions are a combination of the above specified, and a maximum iteration count. Preallocation and computation of all constant values is computed before the loop, in order to reduce computational load; especially within the update step for $\mathbf{W}_b$, which involves a hefty matrix inversion. The update for $\mathbf{S}$ is calculated element-wise, which although creates a nested loop structure, is minimal compared to the computation complexity of matrix multiplications. After the main loop, the weight vector $\mathbf{w}$ and bias b are retrieved from $\mathbf{W}_b$ (I overlooked this step during my implementation, which lead to a VERY painful full-day debugging session). After a cursory glance online, I decided upon the regularization and penalty parameter values of $\lambda = 0.1, \beta = 2.25$. I did not attempt to perform a sort of discrete argument optimization over these values; However, I believe it would have lead to the chosen values regardless.

7 Results and Discussion

7.1 rho_02 Dataset

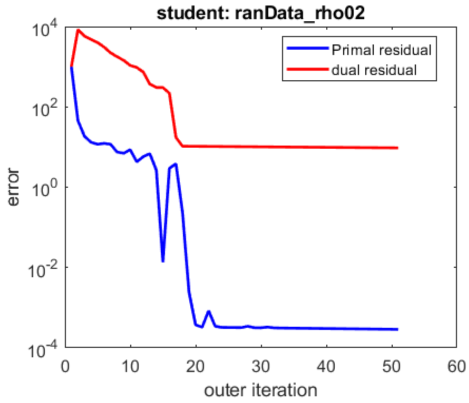

Figure 1: 50 Iterations

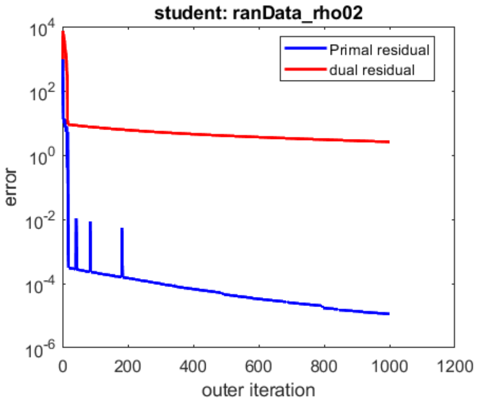

Figure 2: 1000 Iterations

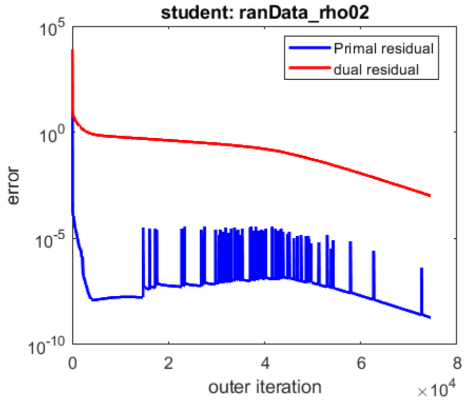

Figure 3: Convergence Iterations

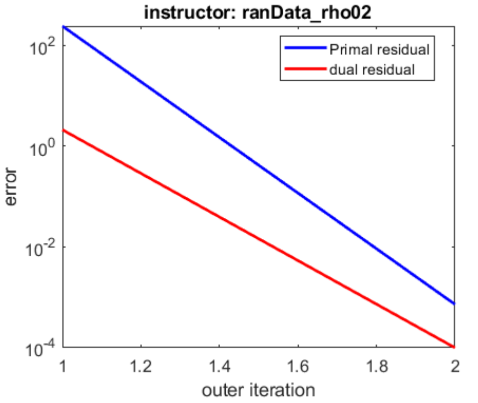

Figure 4: Instructor ALM

Test	Max Iterations	r_p	r_d	Time (s)	Accuracy
ADMM	50	2.8721e-04	9.5316	0.0291	78.50%
ADMM	1000	1.1339e-05	2.5773	0.2150	93.50%
ADMM	None (75000)	1.8403e-09	9.9988e-04	13.298	97.50%
ALM_I	1	7.0856e-04	9.7159e-05	0.1320	97.50%

On the Rho02 dataset, my algorithm featured very slow convergence in order to have 100% correct output. However, minimal iterations were required for relatively decent (75%) classification accuracy. As we can see, the ADMM algorithm converges to near 80% accuracy in around 20 iterations, and takes almost 75000 iterations in order to get the same accuracy as the instructors ALM. For a deviation of 5% accuracy, ADMM has comparable runtime to ALM, but for the same accuracy, it takes an entire magnitude of 10 longer.

7.2 rho_08 Dataset

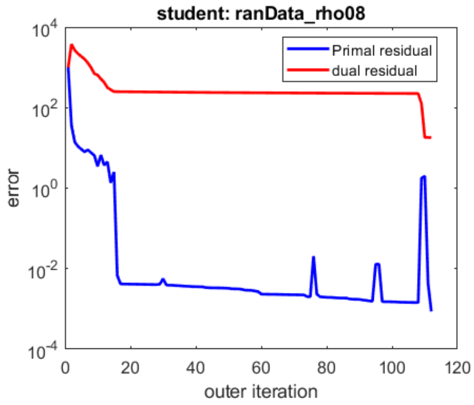

Figure 5: Convergence Iterations

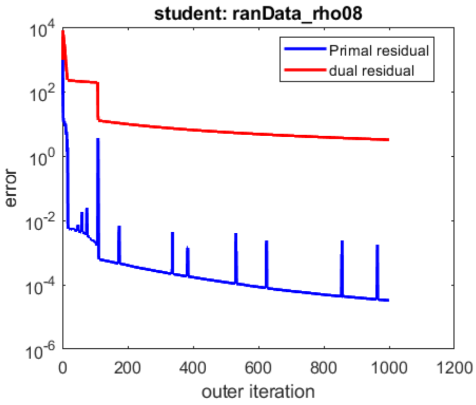

Figure 6: 1000 Iterations

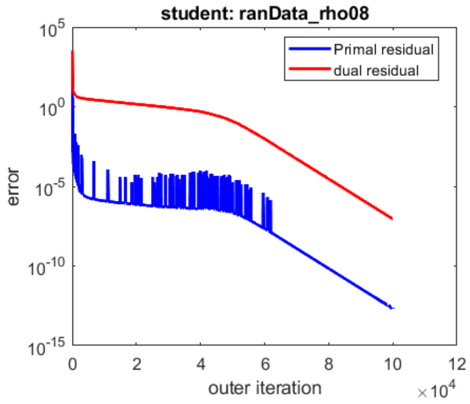

Figure 7: 100,000 Iterations

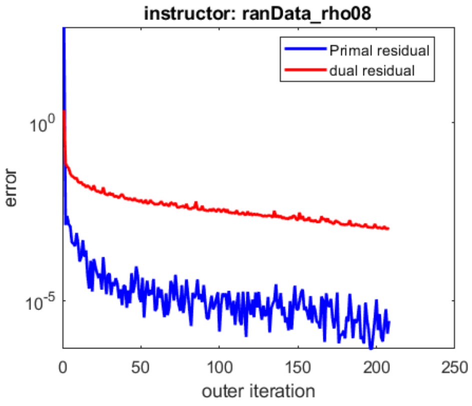

Figure 8: Instructor ALM

Test	Max Iterations	r_p	r_d	Time (s)	Accuracy
ADMM	None (111)	8.6791e-04	1.8072e+01	0.0619	78.00%
ADMM	1000	4.7111e-05	3.6761e+00	0.3345	89.50%
ADMM	100000	2.1242e-13	8.9186e-08	18.6859	88.00%
ALM_I	207 Out * 100 In	7.0812e-06	9.8157e-04	4.9799	87.00%

Unlike in the rho_02 dataset, my ADMM algorithm satisfied the stopping conditions very quickly, with only 111 iterations required. However, it did not have the best accuracy.

Removal of the stopping conditions and only using a max iteration lead to much better accuracy, with only 1000 iterations required to surpass instructor accuracy. Additionally, my algorithm ran much faster than the instructor for this dataset. It is possible that the provided data might just be better conditioned for the variable change I introduced, since the opposite was true for rho_02.

7.3 SPAM Dataset

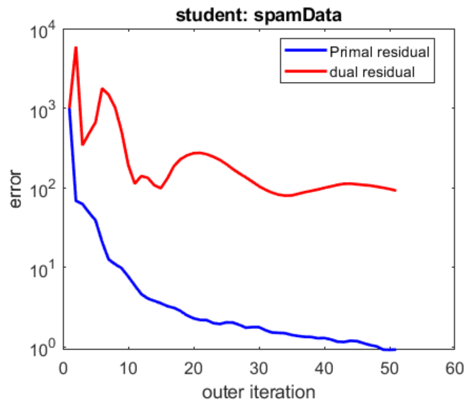

Figure 9: 50 Iterations

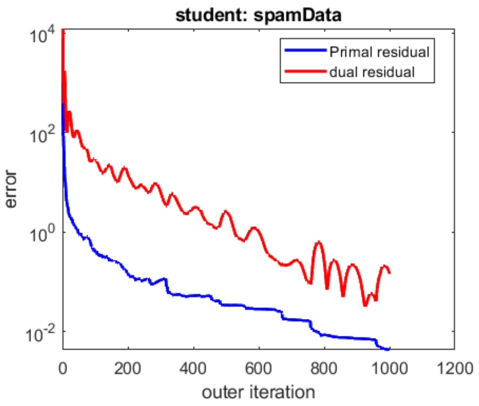

Figure 10: 1000 Iterations

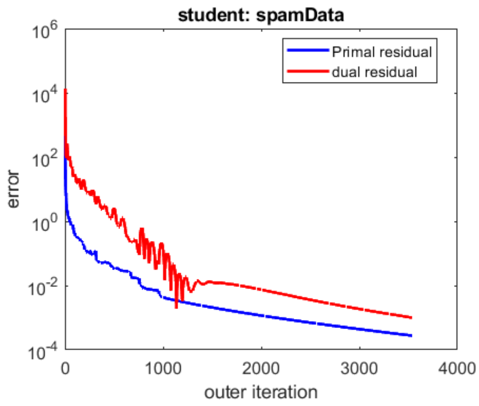

Figure 11: Convergence Iterations

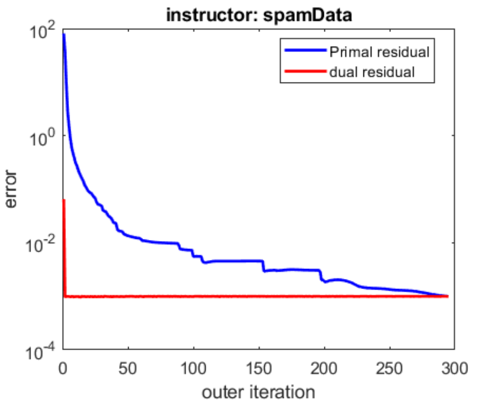

Figure 12: Instructor ALM

Test	Max Iterations	r_p	r_d	Time (s)	Accuracy
ADMM	50	9.3061e-01	9.2192e+01	0.1008	78.87%
ADMM	1000	4.3231e-03	1.3938e-01	1.3254	79.37%
ADMM	None (3542)	2.7555e-04	9.9910e-04	4.3058	79.37%
ALM_I	294 Out*276 In	9.9206e-04	9.9021e-04	111.11	82.00%

For my ADMM implementation, the SPAM dataset seemed to follow the same pattern as rho_08. Although, ADMM never reached the exact accuracy that ALM did, it converged to a close value very quickly, taking only a tenth of a second to be within 3.5% of the over 100 second ALM accuracy. My hypothesis is that the constraints in the spam and rho_08 data sets are difficult to use proximal gradient descent on, and as such take a large amount of sub-iterations for ALM. Thus since ADMM has only closed form solutions we see a major speedup. UNlike the previous two data sets, the dual residual was many times more volatile, and the primal was smoother. The gap between dual and primal was smaller on this data set as well. I do not know why that is but I believe it has something to do with the data's structure.

7.4 Discussion

ADMM is often to applied to difficult datasets which do not need exact solutions. It is noted that ADMM tends to experience rapid convergence in the beginning iterations, and then almost asymptotically converge the rest over a large number of iterations. Thus it is seen that ADMM would be best applied in computationally difficult data sets which do not need perfect classification. I believe the SPAM and Rho_08 datasets are examples of these. The benefit of closed form solutions allows us to circumvent the load of subsolver methods.

In analysis of my two residuals, I noticed that they are nearly inversely proportional. The primal residual drops below the predefined tolerance nearly instantly, while the dual floats above the limit for a long time. This seems to point towards a relationship between the two. Most major decreases in dual residual coincide almost exactly with massive spikes in primal residual. For instance, on the rho_08 dataset, the primal residual drops from a magnitude of 10^3 to 10^1 exactly when the primal jumps from 10^{-3} to 10^0. Many other examples of this can be seen in the above figures.

My implementation of this algorithm was crude, and there exists numerous paths for optimization. While I did calculate all constants before hand, the closed forms could certainly be optimized. For instance, the first term of the update step for $\mathbf{W}_b$ involves a matrix inversion. The matrix being inverted has a very similar form to that of the Woodbury matrix inversion identity. I believe it could be rearranged such that the identity could be applied. However, since the first term is constant, I did not see a need to undergo that effort as I only calculate the value once.