Fundamentals of Linear Algebra and Optimization

Solving SVM Using ADMM

Jean Gallier and Jocelyn Quaintance

CIS Department
University of Pennsylvania

jean@cis.upenn.edu

June 5, 2020

Alternating Direction Method of Multipliers

The *alternating direction method of multipliers*, for short ADMM, is the best method known for solving optimization problems for which the function J to be optimized can be split into two independent parts, as $J(x, z) = f(x) + g(z)$, and to consider the Minimization Problem (P_{admm}),

$$\begin{aligned} &\text{minimize} && f(x) + g(z) \\ &\text{subject to} && Ax + Bz = c, \end{aligned}$$

for some $p \times n$ matrix A, some $p \times m$ matrix B, and with $x \in \mathbb{R}^n$, $z \in \mathbb{R}^m$, and $c \in \mathbb{R}^p$. We also assume that f and g are *convex*.

Iterative Steps of ADMM

The above problem can be solved using an iterative process applying to the *augmented Lagrangian*

$$L_\rho(x, z, \lambda) = f(x) + g(z) + \lambda^\top (Ax + Bz - c) + (\rho/2) \|Ax + Bz - c\|_2^2 ,$$

with $\lambda \in \mathbb{R}^p$ and for some $\rho > 0$.

Iterative Steps of ADMM

The above problem can be solved using an iterative process applying to the *augmented Lagrangian*

$$L_\rho(x, z, \lambda) = f(x) + g(z) + \lambda^\top (Ax + Bz - c) + (\rho/2) \|Ax + Bz - c\|_2^2 ,$$

with $\lambda \in \mathbb{R}^p$ and for some $\rho > 0$.

Given some initial values (z^0, λ^0), the *ADMM method* consists of the following iterative steps:

$$x^{k+1} = \arg\min_x L_\rho(x, z^k, \lambda^k)$$

$$z^{k+1} = \arg\min_z L_\rho(x^{k+1}, z, \lambda^k)$$

$$\lambda^{k+1} = \lambda^k + \rho(Ax^{k+1} + Bz^{k+1} - c).$$

ADMM Methodology of Sequential Updates

Instead of performing a minimization step jointly over x and z, as the step

$$(x^{k+1}, z^{k+1}) = \arg\min_{x,z} L_\rho(x, z, \lambda^k),$$

ADMM first performs an x-minimization step, and then a z-minimization step. Thus x and z are updated in an alternating or sequential fashion, which accounts for the term *alternating direction*.

Specializing ADMM to Quadratic Programs

We specialize ADMM to quadratic programs of the following form:

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}x^\top Px + q^\top x + r \\ &\text{subject to} \quad Ax = b,\ x \geq 0, \end{aligned}$$

where P is an $n \times n$ *symmetric positive semidefinite* matrix, $q \in \mathbb{R}^n$, $r \in \mathbb{R}$, and A is an $m \times n$ matrix of rank m.

Specializing ADMM to Quadratic Programs

The above program is converted in ADMM form as follows:

$$\begin{array}{ll} \text{minimize} & f(x) + g(z) \\ \text{subject to} & x - z = 0, \end{array}$$

Specializing ADMM to Quadratic Programs

The above program is converted in ADMM form as follows:

$$\begin{array}{ll} \text{minimize} & f(x) + g(z) \\ \text{subject to} & x - z = 0, \end{array}$$

with

$$f(x) = \frac{1}{2}x^\top Px + q^\top x + r, \quad \mathrm{dom}(f) = \{x \in \mathbb{R}^n \mid Ax = b\},$$

Specializing ADMM to Quadratic Programs

The above program is converted in ADMM form as follows:

$$\begin{array}{ll} \text{minimize} & f(x) + g(z) \\ \text{subject to} & x - z = 0, \end{array}$$

with

$$f(x) = \frac{1}{2} x^\top P x + q^\top x + r, \quad \mathrm{dom}(f) = \{x \in \mathbb{R}^n \mid Ax = b\},$$

and

$$g = I_{\mathbb{R}^n_+},$$

the indicator function of the positive orthant $\mathbb{R}^n_+$.

Specializing ADMM to Quadratic Programs

Then ADMM consists of the following steps:

$$
\begin{aligned}
x^{k+1} &= \arg\min_x \left(f(x) + (\rho/2) \left\| x - z^k + u^k \right\|_2^2 \right) \\
z^{k+1} &= (x^{k+1} + u^k)_+ \\
u^{k+1} &= u^k + x^{k+1} - z^{k+1},
\end{aligned}
$$

where $u^k = \lambda^k/\rho$ (this is the scaled version of ADMM). Here, v_+ is the vector obtained by setting the negative components of v to zero.

Specializing ADMM to Quadratic Programs

Then ADMM consists of the following steps:

$$
\begin{aligned}
x^{k+1} &= \operatorname*{arg\,min}_x \left(f(x) + (\rho/2) \left\| x - z^k + u^k \right\|_2^2 \right) \\
z^{k+1} &= (x^{k+1} + u^k)_+ \\
u^{k+1} &= u^k + x^{k+1} - z^{k+1},
\end{aligned}
$$

where $u^k = \lambda^k/\rho$ (this is the scaled version of ADMM). Here, v_+ is the vector obtained by setting the negative components of v to zero. The x-update involves solving the KKT equations

$$
\begin{pmatrix} P + \rho I & A^\top \\ A & 0 \end{pmatrix} \begin{pmatrix} x^{k+1} \\ y \end{pmatrix} = \begin{pmatrix} -q + \rho(z^k - u^k) \\ b \end{pmatrix}.
$$

Solving $(\mathrm{SVM}_{s2'})$ *Using ADMM*

In order to solve $(\mathrm{SVM}_{s2'})$ using ADMM we need to write the matrix corresponding to the constraints in equational form,

$$\begin{aligned}
\sum_{i=1}^{p} \lambda_i - \sum_{j=1}^{q} \mu_j &= 0 \\
\sum_{i=1}^{p} \lambda_i + \sum_{j=1}^{q} \mu_j - \gamma &= K_m \\
\lambda_i + \alpha_i &= K_s, \quad i = 1, \ldots, p \\
\mu_j + \beta_j &= K_s, \quad j = 1, \ldots, q,
\end{aligned}$$

with $K_m = (p+q)K_s\nu$.

Constraint Matrix for the Dual of $(\mathrm{SVM}_{s2'})$

This is the $(p+q+2)\times(2(p+q)+1)$ matrix A given by

$$A = \begin{pmatrix} \mathbf{1}_p^\top & -\mathbf{1}_q^\top & 0_p^\top & 0_q^\top & 0 \\ \mathbf{1}_p^\top & \mathbf{1}_q^\top & 0_p^\top & 0_q^\top & -1 \\ I_p & 0_{p,q} & I_p & 0_{p,q} & 0_p \\ 0_{q,p} & I_q & 0_{q,p} & I_q & 0_q \end{pmatrix}.$$

Constraint Matrix for the Dual of $(\mathrm{SVM}_{s2'})$

This is the $(p+q+2)\times(2(p+q)+1)$ matrix A given by

$$A = \begin{pmatrix} \mathbf{1}_p^\top & -\mathbf{1}_q^\top & 0_p^\top & 0_q^\top & 0 \\ \mathbf{1}_p^\top & \mathbf{1}_q^\top & 0_p^\top & 0_q^\top & -1 \\ I_p & 0_{p,q} & I_p & 0_{p,q} & 0_p \\ 0_{q,p} & I_q & 0_{q,p} & I_q & 0_q \end{pmatrix}.$$

We leave it as an exercise to prove that A has rank $p+q+2$. The right-hand side is

$$c = \begin{pmatrix} 0 \\ K_m \\ K_s \mathbf{1}_{p+q} \end{pmatrix}.$$

Solving $(\mathrm{SVM}_{s2'})$ *Using ADMM*

The symmetric positive semidefinite $(p+q) \times (p+q)$ matrix P defining the quadratic functional is

$$P = X^{\top} X, \quad \text{with} \quad X = \begin{pmatrix} -u_1 & \cdots & -u_p & v_1 & \cdots & v_q \end{pmatrix},$$

and

$$q = 0_{p+q}.$$

Solving $(\mathrm{SVM}_{s2'})$ *Using ADMM*

Since there are $2(p+q)+1$ Lagrange multipliers $(\lambda, \mu, \alpha, \beta, \gamma)$, the $(p+q) \times (p+q)$ matrix $X^\top X$ must be augmented with zero's to make it a $(2(p+q)+1) \times (2(p+q)+1)$ matrix P_a given by

$$P_a = \begin{pmatrix} X^\top X & 0_{p+q,p+q+1} \\ 0_{p+q+1,p+q} & 0_{p+q+1,p+q+1} \end{pmatrix},$$

and similarly q is augmented with zeros as the vector $q_a = 0_{2(p+q)+1}$.

Simplification of the Dual Constraints

Using the fact that the duality gap is zero it can be shown that if the primal problem $(\mathrm{SVM}_{s2'})$ has an optimal solution with $w \neq 0$, then $\eta \geq 0$.

Simplification of the Dual Constraints

Using the fact that the duality gap is zero it can be shown that if the primal problem $(\mathrm{SVM}_{s2'})$ has an optimal solution with $w \neq 0$, then $\eta \geq 0$.

Consequently we can drop the constraint $\eta \geq 0$ from the primal problem.

Simplification of the Dual Constraints

In this case there are $2(p + q)$ Lagrange multipliers $(\lambda, \mu, \alpha, \beta)$. It is easy to see that the objective function of the dual is unchanged and the set of constraints is

Simplification of the Dual Constraints

In this case there are $2(p+q)$ Lagrange multipliers $(\lambda, \mu, \alpha, \beta)$. It is easy to see that the objective function of the dual is unchanged and the set of constraints is

$$\sum_{i=1}^{p} \lambda_i - \sum_{j=1}^{q} \mu_j = 0$$

$$\sum_{i=1}^{p} \lambda_i + \sum_{j=1}^{q} \mu_j = K_m$$

$$\lambda_i + \alpha_i = K_s, \quad i = 1, \ldots, p$$

$$\mu_j + \beta_j = K_s, \quad j = 1, \ldots, q,$$

with $K_m = (p+q)K_s\nu$.

Simplifying the Constraint Matrix

The constraint matrix corresponding to this system of equations is the $(p+q+2) \times 2(p+q)$ matrix A_2 given by

$$A_2 = \begin{pmatrix} \mathbf{1}_p^\top & -\mathbf{1}_q^\top & 0_p^\top & 0_q^\top \\ \mathbf{1}_p^\top & \mathbf{1}_q^\top & 0_p^\top & 0_q^\top \\ I_p & 0_{p,q} & I_p & 0_{p,q} \\ 0_{q,p} & I_q & 0_{q,p} & I_q \end{pmatrix}.$$

Simplifying the Constraint Matrix

The constraint matrix corresponding to this system of equations is the $(p+q+2) \times 2(p+q)$ matrix A_2 given by

$$A_2 = \begin{pmatrix} \mathbf{1}_p^\top & -\mathbf{1}_q^\top & 0_p^\top & 0_q^\top \\ \mathbf{1}_p^\top & \mathbf{1}_q^\top & 0_p^\top & 0_q^\top \\ I_p & 0_{p,q} & I_p & 0_{p,q} \\ 0_{q,p} & I_q & 0_{q,p} & I_q \end{pmatrix}.$$

We leave it as an exercise to prove that A_2 has rank $p+q+2$. The right-hand side is

$$c_2 = \begin{pmatrix} 0 \\ K_m \\ K_s \mathbf{1}_{p+q} \end{pmatrix}.$$

Solving $(\mathrm{SVM}_{s2'})$ *Using ADMM*

The symmetric positive semidefinite $(p+q) \times (p+q)$ matrix P is

$$P = X^\top X, \quad \text{with} \quad X = \begin{pmatrix} -u_1 & \cdots & -u_p & v_1 & \cdots & v_q \end{pmatrix},$$

and $q = 0_{p+q}$.

Solving $(\mathrm{SVM}_{s2'})$ *Using ADMM*

The symmetric positive semidefinite $(p+q) \times (p+q)$ matrix P is

$$P = X^\top X, \quad \text{with} \quad X = \begin{pmatrix} -u_1 & \cdots & -u_p & v_1 & \cdots & v_q \end{pmatrix},$$

and $q = 0_{p+q}$.

Since there are $2(p+q)$ Lagrange multipliers, the $(p+q) \times (p+q)$ matrix $X^\top X$ must be augmented with zero's to make it a $2(p+q) \times 2(p+q)$ matrix P_{2a} given by

$$P_{2a} = \begin{pmatrix} X^\top X & 0_{p+q,p+q} \\ 0_{p+q,p+q} & 0_{p+q,p+q} \end{pmatrix},$$

and similarly q is augmented with zeros as the vector $q_{2a} = 0_{2(p+q)}$.

Matlab Illustrations of ADMM Solutions

The above method was implemented in Matlab with $\rho = 10$.

We ran our program on two sets of 30 points each generated at random using the following code which calls the function runSVMs2pbv3:

```
rho = 10;
u16 = 10.1*randn(2,30)+7 ;
v16 = -10.1*randn(2,30)-7;
[~,~,~,~,~,~,w3] = runSVMs2pbv3(0.37,rho,u16,v16,1/60)
```

Matlab Illustrations of ADMM Solutions

We picked $K_s = 1/60$ and various values of ν starting with $\nu = 0.37$, which appears to be the smallest value for which the method converges; see Figure 1.

Matlab Illustrations of ADMM Solutions

We picked $K_s = 1/60$ and various values of ν starting with $\nu = 0.37$, which appears to be the smallest value for which the method converges; see Figure 1.

Reducing ν below $\nu = 0.37$ has the effect that p_f, q_f, p_m, q_m decrease but the following situation arises. Shrinking η a little bit has the effect that $p_f = 9, q_f = 10, p_m = 10, q_m = 11$.

Matlab Illustrations of ADMM Solutions

Then $\max\{p_f, q_f\} = \min\{p_m, q_m\} = 10$, so the only possible value for ν is $\nu = 20/60 = 1/3 = 0.3333333\cdots$.

Matlab Illustrations of ADMM Solutions

Then $\max\{p_f, q_f\} = \min\{p_m, q_m\} = 10$, so the only possible value for ν is $\nu = 20/60 = 1/3 = 0.3333333\cdots$.

When we run our program with $\nu = 1/3$, it returns a value of η less than 10^{-13} and a value of w whose components are also less than 10^{-13}. This is probably due to numerical precision. Values of ν less than $1/3$ cause the same problem. It appears that the geometry of the problem constrains the values of p_f, q_f, p_m, q_m in such a way that it has no solution other than $w = 0$ and $\eta = 0$.

Solving $(\mathrm{SVM}_{s2'})$ *Using ADMM*

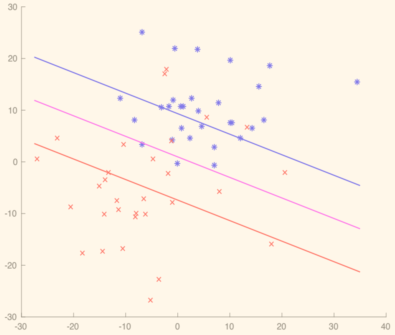

Figure 1: Running $(\mathrm{SVM}_{s2'})$ on two sets of 30 points; $\nu = 0.37$.

Matlab Illustrations of ADMM Solutions

Figure 2 shows the result of running the program with $\nu = 0.51$. We have $p_f = 15, q_f = 16, p_m = 16, q_m = 16$. Interestingly, for $\nu = 0.5$, we run into the singular situation where there is only one support vector and $\nu = 2p_f/(p+q)$.

Solving $(\mathrm{SVM}_{s2'})$ *Using ADMM*

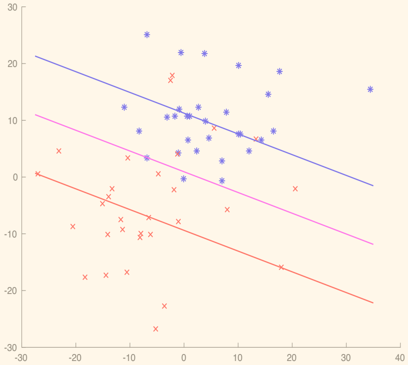

Figure 2: Running $(\mathrm{SVM}_{s2'})$ on two sets of 30 points; $\nu = 0.51$.

Solving $(\text{SVM}_{s2'})$ *Using ADMM*

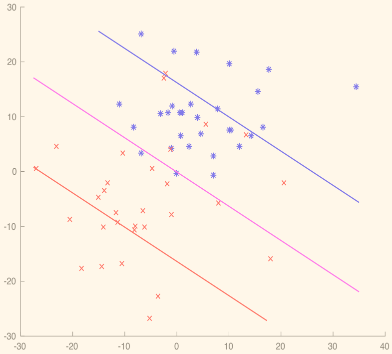

Figure 3: Running $(\text{SVM}_{s2'})$ on two sets of 30 points; $\nu = 0.71$.

Matlab Illustrations of ADMM Solutions

Next Figure 3 shows the result of running the program with $\nu = 0.71$. We have $p_f = 21, q_f = 21, p_m = 22, q_m = 23$. Interestingly, for $\nu = 0.7$, we run into the singular situation where there are no support vectors.

Matlab Illustrations of ADMM Solutions

Next Figure 3 shows the result of running the program with $\nu = 0.71$. We have $p_f = 21, q_f = 21, p_m = 22, q_m = 23$. Interestingly, for $\nu = 0.7$, we run into the singular situation where there are no support vectors.

For our next to the last run, Figure 4 shows the result of running the program with $\nu = 0.95$. We have $p_f = 28, q_f = 28, p_m = 29, q_m = 29$.

Solving $(\mathrm{SVM}_{s2'})$ *Using ADMM*

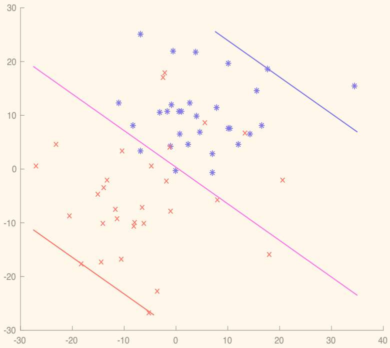

Figure 4: Running $(\mathrm{SVM}_{s2'})$ on two sets of 30 points; $\nu = 0.95$.

Matlab Illustrations of ADMM Solutions

Figure 5 shows the result of running the program with $\nu = 0.97$. We have $p_f = 29, q_f = 29, p_m = 30, q_m = 30$, which shows that the largest margin has been achieved.

Matlab Illustrations of ADMM Solutions

Figure 5 shows the result of running the program with $\nu = 0.97$. We have $p_f = 29, q_f = 29, p_m = 30, q_m = 30$, which shows that the largest margin has been achieved.

However, after 80000 iterations the dual residual is less than 10^{-12} but the primal residual is approximately 10^{-4} (our tolerance for convergence is 10^{-10}, which is quite high). Nevertheless the result is visually very good.

Solving $(\mathrm{SVM}_{s2'})$ *Using ADMM*

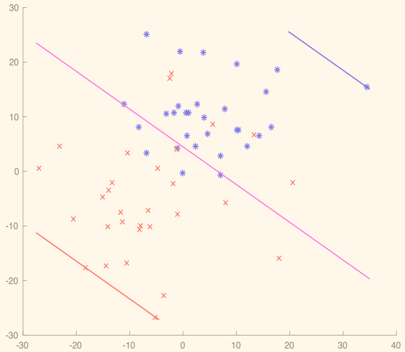

Figure 5: Running $(\mathrm{SVM}_{s2'})$ on two sets of 30 points; $\nu = 0.97$.