10-725/36-725: Convex Optimization **Fall 2018**

Lecture 22: November 14

Lecturer: Ryan Tibshirani *Scribes: Darby Losey, Trevor Frisby, Ksenia Korovina*

Overview

In this lecture and accompanying notes, we focus on ADMM and its details, starting with the basic algorithm and a recap of related methods. Then we consider a few problems - lasso, group lasso, sparse subspace estimation and sparse+low rank decomposition - and practical issues of ADMM. Next we consider a variation on ADMM that is particularly adapted for distributed learning, and finish off with the Fused lasso example, illustrating the importance of choosing appropriate problem-specific decompositions for ADMM.

22.1 Dual Ascent, Method of multipliers, and ADMM

Last lecture, we discussed dual gradient descent, which is used to solve problems with strictly convex and closed objective f under linear equality constraints:

$$\min_x f(x) \text{ subject to } Ax = b$$

Its Lagrangian is written in familiar form as:

$$L(x,u) = f(x) + u^T(Ax - b)$$

Dual ascent alternates between updating primal and dual variables:

$$x^{(k)} = \operatorname{argmin} L(x, u^{(k-1)})$$

$$u^{(k)} = u^{(k-1)} + t_k(Ax^{(k)} - b)$$

The step size can be chosen in standard ways (e.g. using backtracking line search). Note that if f decomposes over x (or its blocks), then the primal updates decompose accordingly, which makes the minimization step can be done in a distributed setting. However, the downside of this algorithm is in strong requirements it imposes on f to ensure primal convergence (while dual convergence always holds), and these requirements are rarely satisfied by common problems we may want to solve.

One workaround is to strongly convexify the criterion, which is done by the **Augmented Lagrangian method** (also known as the method of multipliers, MM). For a fixed parameter $\rho > 0$, solves a modified problem:

$$\min_x f(x) + \frac{\rho}{2}\|Ax - b\|_2^2 \text{ subject to } Ax = b$$

Corresponding to it is the modified Lagrangian:

$$L_\rho(x, u) = f(x) + u^T(Ax - b) + \frac{\rho}{2}\|Ax - b\|_2^2$$

The updates are given by

$$x^{(k)} = \operatorname{argmin} L_\rho(x, u^{(k-1)})$$

$$u^{(k)} = u^{(k-1)} + \rho(Ax^{(k)} - b)$$

This now improves the properties of the objective - if A is full-rank, the modified objective is strongly convex - but at the same time decomposability of f no longer gives decomposable updates for the primal updates.

Finally, we arrive at the **Alternating direction method of multipliers** (or **ADMM**), which combines the best of Dual ascent and Augmented Lagrangian by assuming the problem takes the form

$$\min_{x,z} f(x) + g(z) \text{ subject to } Ax + bz = c \tag{22.1}$$

As we will see in the following sections, this is a fairly general form (many problems can be transformed into it). However, ADMM performance can vary depending on a particular transformation.

Defining the augmented Lagrangian akin to MM:

$$L_\rho(x, z, u) = f(x) + g(z) + u^T(Ax + Bz - c) + \frac{\rho}{2}\|Ax + Bz - c\|_2^2$$

The updates are performed as

$$\begin{aligned} x^{(k)} &= \operatorname*{argmin}_x L_\rho(x, z^{(k-1)}, u^{(k-1)}) \\ z^{(k)} &= \operatorname*{argmin}_z L_\rho(x^{(k)}, z, u^{(k-1)}) \\ u^{(k)} &= u^{(k-1)} + \rho(Ax^{(k)} + Bz^{(k)} - c) \end{aligned}$$

If we updated x and z together, that would be equivalent to MM. But now note that in the z update, we have to use the most recent x, $x^{(k)}$ (and if we switched the order of these updates, it would give a different algorithm, although it does not matter much in practice). Some other observations about this algorithm:

- In the limit, primal feasibility is achieved:

$$r^{(k)} = Ax^{(k)} + Bz^{(k)} - c \to 0 \text{ as } k \to \infty$$

- The objective converges to optimal:

$$f(x^{(k)}) + g(z^{(k)}) \to f^\star + g^\star$$

- Moreover, dual variable u converges to dual solution:

$$u^{(k)} \to u^\star$$

Convergence guarantees are almost the same as for other first order methods (see [Boyd10]).

It is important to realize that a specific parametrization chosen to take to original problem to the form (22.1).

We can re-write ADMM to take the **Scaled ADMM** form, which has easier-to-write updates: namely, we take a new dual variable $w = u/\rho$; rewriting the augmented Lagrangian gives

$$L_\rho(x, z, w) = f(x) + g(z) + \frac{\rho}{2}\|Ax + Bz - c + w\|_2^2 - \frac{\rho}{2}\|w\|_2^2$$

Updates now take the following explicit form:

$$\begin{aligned} x^{(k)} &= \underset{x}{\operatorname{argmin}}\, f(x) + \frac{\rho}{2}\|Ax + Bz^{(k-1)} - c + w^{(k-1)}\| \\ z^{(k)} &= \underset{z}{\operatorname{argmin}}\, g(z) + \frac{\rho}{2}\|Ax^{(k)} + Bz - c + w^{(k-1)}\| \\ w^{(k)} &= w^{(k-1)} + Ax^{(k)} + Bz^{(k)} - c \end{aligned}$$

In the next section, we will build on this form of ADMM and work through several example applications of ADMM, as well as practical details one should know before deploying this algorithm in practice.

22.2 Examples and practicalities of ADMM

ADMM performs similarly to first order methods. That is, it usually finds a fairly accurate solution in only a handful of iterations, but a highly accurate solution requires a large number of iterations. The convergence of ADMM is strongly influenced by choice for ρ. A choice that is too large may not emphasize enough minimizing $f + g$, whereas a choice that is too small may not adequately emphasize feasibility. Strategies exist that vary the choice for ρ, but these do not have convergence guarantees (Boyd et. al. 2010).

The following examples show how we can massage some examples we have seen previously in the course into ones that we can apply ADMM (note that this can be subtle).

22.2.1 Group lasso regression

Recall that the group lasso problem is given as follows:

$$\min_\beta \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \sum_{g=1}^{G} c_g \|\beta_g\|_2$$

where $y \in \mathbb{R}^n, X \in \mathbb{R}^{n\times p}$, and G indexes the groups. We can rewrite this for ADMM by introducing the variable α as follows:

$$\min_{\beta,\alpha} \frac{1}{2} + \lambda \sum_{g=1}^{G} c_g \|\alpha_g\|_2 \quad \text{Subject to} \quad \beta - \alpha = 0$$

With this reformulation, we can now apply the familiar ADMM updates:

$$\beta^+ = (X^TX + \rho I)^{-1}(X^Ty + \rho(\alpha - w)) \qquad (X^TX + \rho I \text{ is always invertible, regardless of X})$$
$$\alpha_g^+ = R_{c_g\lambda/\rho}(\beta_g^+ + w_g), \quad g = 1, \cdots, G \qquad \left(R_t(x) = \left(1 - \frac{t}{\|x\|_2}\right)_+ x\right)$$
$$w^+ = w + \beta^+ - \alpha^+$$

Note that if we compute a factorization using Cholesky in $O(p^3)$ flops, then we update β in $O(p^2)$ flops. Additionally, $R_t(x)$ refers to the group soft-thresholding operator, and $(\cdot)_+$ is the positive parts function. This process can be used for an arbitrary sum of norms, so long as we know the proximal operator.

22.2.2 Sparse subspace estimation

Let $S \in \mathbb{S}^p$ (in practice, we typically have $S \succeq 0$ is a covariance matrix). The sparse subspace estimation problem is given as follows:

$$\max_Y \operatorname{tr}(SY) - \lambda\|Y\|_1 \quad \text{subject to} \quad Y \in \mathcal{F}_k$$

Recall from homework 1 that $\mathcal{F}_k$ is the Fantope of order k, which is given by:

$$\mathcal{F}_k = \{Y \in \mathcal{S}^p : 0 \preceq Y \preceq I, \operatorname{tr}(Y) = k\}$$

Equivalently, we can think of $\mathcal{F}_k$ as the convex hull of all rank k projection matrices. In the case where $\lambda = 0$, the above is simply equivalent to ordinary PCA. Since this problem is an SDP, it is solvable by interior point methods. However, these tend to be slow for large problems, and generally cumbersome to implement, highlighting the usefulness of recasting into an ADMM form. We now introduce the variable Z and rewrite as follows:

$$\min_{Y,Z} -\operatorname{tr}(SY) + I_{\mathcal{F}_k}(Y) + \lambda\|Z\|_1 \quad \text{subject to} \quad Y = Z$$

Here, $I_{\mathcal{F}_k}(Y)$ is an indicator of whether or not Y belongs to the Fantope. We now have the following ADMM steps:

$$Y^+ = P_{\mathcal{F}_k}(Z - W + S/\rho)$$
$$Z^+ = S_{\lambda/\rho}(Y^+ + W)$$
$$W^+ = W + Y^+ - Z^+$$

Here, $S_{\lambda/\rho}(\cdot)$ is the familiar soft-thresholding, and $P_{\mathcal{F}_k}(\cdot)$ it the Fantope projection operator. This is computed by clipping the eigendecomposition $A = U\Sigma U^T$ with $\Sigma = \operatorname{diag}(\sigma_1, \cdots, \sigma_p)$:

$$P_{\mathcal{F}_k}(A) = U\Sigma_\theta U^T, \quad \Sigma = \operatorname{diag}(\sigma_1(\theta), \cdots, \sigma_p(\theta))$$

Note that each $\sigma_i(\theta) = \min\{\max\{\sigma_i - \theta, 0\}, 1\}$ and $\sum_{i=1}^p \sigma_i(\theta) = k$.

22.2.3 Sparse and low rank decomposition

The final example from this section is the sparse plus low rank decomposition problem. For $M \in \mathbb{R}^{n\times m}$, this is given as:

$$\min_{L,S} \|L\|_{\text{tr}} + \lambda\|S\|_1 \quad \text{subject to} \quad L + S = M$$

The ADMM steps are then:

$$
\begin{aligned}
L^+ &= S^{\text{tr}}_{1/\rho}(M - S + W) \\
S^+ &= S^{\ell_1}_{\lambda/\rho}(M - L^+ + W) \\
W^+ &= W + M - L^+ - S^+
\end{aligned}
$$

Here, $S^{\text{tr}}_{1/\rho}(\cdot)$ refers to matrix soft-thresholding, and $S^{\ell_1}_{\lambda/\rho}(\cdot)$ refers to elementwise soft-thresholding. The example from the lecture slides shows hows this method can be applied to individual frames taken from a video. The original frames are decomposed into low rank and sparse components. The low rank component captures the static elements of the image, that is, those parts that are not moving in the video. The sparse component captures only the parts that are moving.

22.3 Consensus ADMM

Consensus ADMM considers problems of the form $\min_x \sum_{i=1}^B f_i(x)$. This is then rewritten in the following way:

$$
\min_{x_1,\cdots,x_B,x} \sum_{i=1}^B f_i(x_i) \quad \text{subject to} \quad x_i = x, i = 1,\cdots,B
$$

This gives us the following decomposable ADMM steps:

$$
\begin{aligned}
x_i^+ &= \underset{x_i}{\operatorname{argmin}}\, f_i(x_i) + \frac{\rho}{2}\|x_i - x + w_i\|_2^2, \quad i = 1,\cdots,B \\
x^+ &= \frac{1}{B}\sum_{i=1}^B (x_i^+ + w_i) \\
w_i^+ &= w_i + x_i^+ - x^+, i = 1,\cdots,B
\end{aligned}
$$

If we let $\bar{x} = \frac{1}{B}\sum_{i=1}^B x_i$ and do similarly for other variables, we see that $\bar{w}^+ = 0$ for all iterations $k \geq 1$. We can thus let $x^+ = \bar{x}^+$ and obtain the following simplified ADMM steps:

$$
\begin{aligned}
x_i^+ &= \underset{x_i}{\operatorname{argmin}}\, f_i(x_i) + \frac{\rho}{2}\|x_i - \bar{x} + w_i\|_2^2, \quad i = 1,\cdots,B \\
w_i^+ &= w_i + x_i^+ - \bar{x}^+ \quad i = 1,\cdots,B
\end{aligned}
$$

The x_i updates are done in parallel. Here, we are trying to minimize each $f_i(x_i)$, and use squared ℓ_2 regularization to pull each x_i towards the average $\bar{x}$. When x_i is bigger than the average, then w_i is increased. Thus, the regularization in the next step pulls x_i even closer.

22.3.1 General consensus ADMM

Suppose we have a problem of the form $\min_x \sum_{i=1}^B f_i(a_i^T x + b_i) + g(x)$. For consensus ADMM, we reparameterize as follows:

$$
\min_{x_1,\cdots,x_B,x} \sum_{i=1}^B f_i(a_i^T x_i + b_i) + g(x) \quad \text{subject to} \quad x_i = x, \quad i = 1,\cdots,B
$$

This gives us decomposable ADMM updates:

$$\begin{aligned}
x_i^+ &= \underset{x_i}{\operatorname{argmin}}\, f_i(a_i^T x_i + b_i) + \frac{\rho}{2}\|x_i - x + w_i\|_2^2 \quad i = 1, \cdots, B \\
x^+ &= \underset{x}{\operatorname{argmin}}\, \frac{B\rho}{2}\|x - \bar{x}^+ - \bar{w}\|_2^2 + g(x) \\
w_i^+ &= x_i^+ - x^+, \quad i = 1, \cdot, B
\end{aligned}$$

In general consensus ADMM, it is no longer the case that $\bar{w}^+ = 0$ for general iteration k, meaning that the ADMM steps do not simplify. The updates for each x_i are still done in parallel. For some intuition into each of the updates, each x_i update can be thought as a loss minimization on part of the data using ℓ_2 regularization. The update on x is a proximal operation in regularizer g. The w update drives individual variables into consensus. Importantly, note that a different initial reparameterization will give rise to a different ADMM algorithm.

22.4 Special decompositions for ADMM

ADMM is reminiscent of block coordinate descent in that the minimization is done over one variable, then another variable etc. Coordinate descent doesn't make much progress when the blocks of coordinates are highly correlated. This motivates how we may want to bring a given problem into ADMM form. Ideally, this can be done such that the ADMM steps make progress in the criterion in as uncorrelated directions as possible. Because of this, different parameterizations of ADMM can lead to different convergence rates.

An example of this is the 2D fused lasso problem, where ADMM converges relatively fast. For image Y such that $Y \in \mathbb{R}^{d\times d}$ and the vectorized form $y \in \mathbb{R}^n$, the 2D lasso problem is given by

$$\min_{\Theta} ||\frac{1}{2}Y - \Theta||_F^2 + \lambda \sum_{i,j} (|\Theta_{i,j} - \Theta_{i+1,j}| + |\Theta_{i,j} - \Theta_{i,j+1}|) \iff \min_{\theta} \frac{1}{2}||y - \theta||_2^2 + \lambda ||D\theta||_1$$

Where $D \in \mathbb{R}^{d\times d}$ gives the appropriate differences across vertically and horizontally adjacent pixels.

Because we know the prox operator of the L1 norm is, we replace $z = D\theta$. Thus we can now rewrite the problem as

$$\min_{\theta} \frac{1}{2}||y - \theta||_2^2 + \lambda ||z||_1$$
$$\text{subject to } \theta = Dz$$

This gives ADMM steps:

$$\begin{aligned}
\theta^k &= (I + \rho D^T D)^{-1}(y + \rho D^T(z^{(k-1)} + w^{(k-1)})) \\
z^{(k)} &= S_{\lambda/\rho}(D\theta^{(k) - w^{(k-1)}}) \\
w^{(k)} &= w^{(k-1)} - z^{(k-1)} - D\theta^{(k)}
\end{aligned}$$

Solving the linear system in the θ update can be done quickly due to the sparse structure. The z update is soft thesholding, so it can also be solved quickly. Therefore, each ADMM cycle can be done in time O(n). However, this tends not to converge very quickly.

A second way to write this problem that converges more quickly is:

$$min_{H,V} \frac{1}{2}||Y - H||_F^2 + \lambda \sum_{i,j} + \lambda \sum_{i,j} (|H_{i,j} - H_{i+1,j}| + |V_{i,j} - V_{i,j+1}|)$$

$$\text{subject to } H = V$$

This gives ADMM steps:

$$H^{(k)}_{\cdot,j} = FL^{1d}_{\lambda/(1+p)}\left(\frac{Y + \rho(V^{(k-1)}_{\cdot,j} - W^{(k-1)}_{\cdot,j})}{1+\rho}\right), \quad j = 1, ..., d$$

$$V^{(k)}_{i,\cdot} = FL^{1d}_{\lambda/(p)}(H^{(k)}_{i,\cdot} + W^{(k-1)}_{i,\cdot}), \quad i = 1, ..., d$$

$$W^{(k)} = W^{(k-1)} + H^{(k)} - V^{(k)}$$

V only appears in the vertical distances and H only appears in the horizontal differences so we can think of these variables as addressing different parts of the problem. Now the problem splits up into a number of 1D total variation problems. Thus this problem can be solved more quickly.

References

[1] S. Boyd and N. Parikh and E. Chu and B. Peleato and J. Eckstein (2010), Distributed optimization and statistical learning via the alternating direction method of multipliers