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

Lecture 21: November 12

Lecturer: Ryan Tibshirani *Scribes: Maria Jahja, Yuting Shi, Yu Chen*

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

21.1 Last time: Coordinate Descent

Consider function

$$\min_x f(x)$$

where $f(x) = g(x) + \sum_{i=1}^n h_i(x_i)$, with g convex and differentiable and each h_i is convex.

Algorithm 1 Coordinate Descent Algorithm

1: Initialize with guess $x = [x_1, x_2, ..., x_n]^T$
2: **repeat**
3: **for all** j in $1, 2, ..., n$ **do**
4: $x_j \leftarrow \text{argmin}_{x_j} f(x)$
5: **end for**
6: **until** convergence

21.1.1 Conjugate functions and their properties

$$f^*(x) = \max_y y^T x - f(y) \tag{21.1}$$

This definition can be interpreted as the maximum gap between linear function y^T and $f(y)$. Obviously, this is equivalent to,

$$-f^*(x) = \min_y f(y) - y^T x \tag{21.2}$$

Fenchel's inequality,

$$f^{**}(x) + f(y) \geq x^T y \tag{21.3}$$

If f is convex and closed, then

$$f^{**} = f \tag{21.4}$$

In general, we have

$$f^{**} \leq f \tag{21.5}$$

$$x \in \partial f^*(y) \iff y \in \partial f(x) \iff x \in \underset{z}{\text{argmin}}\, f(z) - z^T y \tag{21.6}$$

If f is strictly convex,

$$\nabla f^*(y) \in \operatorname*{argmin}_z f(z) - z^T y \tag{21.7}$$

21.1.2 Dual descent

The purpose of using dual descent is that we can optimize the dual problem without calculate the dual function's gradient directly. We start with convex optimization problem with affine constraints.

$$\min_x \ f(x) \tag{21.8}$$

$$\text{subject to } Ax = b \tag{21.9}$$

By the Lagrangian is,

$$L(x,u) = f(x) + u^T(Ax - b) = f(x) - (A^T u)^T x - u^T b \tag{21.10}$$

The dual function is,

$$g(u) = \min_x \ L(x,u) \tag{21.11}$$

$$= \min_x \ f(x) + u^T(Ax - b) \tag{21.12}$$

$$= \min_x f(x) - (-A^T u)^T x - u^T b \tag{21.13}$$

$$= -f^*(-A^T u) - u^T b \tag{21.14}$$

So in order to maximize the dual objective function,

$$\partial g(u) = A\partial f^*(-A^T u) - b \tag{21.15}$$

$$:= Ax - b \tag{21.16}$$

where $x = \partial f^*(-A^T u)$. Therefore, from the basic properties of the conjugate function we know,

$$x \in \operatorname*{argmin}_z f(z) + z^T A^T u \tag{21.17}$$

The dual ascent algorithm is given below, Note that the u is updated by adding the subgradient, since this

Algorithm 2 Dual Ascent Algorithm.

1: **repeat for k**
2: $x^{(k)} \in \operatorname{argmin}_z f(z) + z^T A^T u^{(k-1)}$
3: $u^{(k)} = u^{(k-1)} + t_k(Ax^{(k)} - b)$.

is for the ascent update.

Theorem 21.1 *Assume f is a closed and convex function.*
Then f is strongly convex with m $\iff$ f^* is strongly convex with parameter $\frac{1}{m}$

Proof: Proof of "$\Longrightarrow$"
If x is the minimizer,

$$g(y) \geq g(x) + \frac{m}{2}\|y - x\|^2 \tag{21.18}$$

Define $g_u(x) = f(x) - u^T x$, by using the properties of conjugate function in section 21.1, we define

$$x_u = \operatorname*{argmin}_x g_u(x) = \nabla f^*(u)$$

$$x_v = \operatorname*{argmin}_x g_v(x) = \nabla f^*(v)$$

So by using the inequality of the minimizer, we have

$$g_u(x_v) \geq g_u(x_u) + \frac{m}{2}\|x_v - x_u\|^2 \tag{21.19}$$

$$\iff f(x_v) - u^T x_v \geq f(x_u) - u^T x_u + \frac{m}{2}\|x_v - x_u\|^2 \tag{21.20}$$

$$\tag{21.21}$$

Similarly, we have,

$$g_v(x_u) \geq g_v(x_v) + \frac{m}{2}\|x_u - x_v\|^2 \tag{21.22}$$

$$\iff f(x_u) - v^T x_u \geq f(x_v) - v^T x_v + \frac{m}{2}\|x_u - x_v\|^2 \tag{21.23}$$

$$\tag{21.24}$$

By adding these two inequalities together, we have,

$$f(x_v) - u^T x_v + f(x_u) - v^T x_u \geq f(x_u) - u^T x_u + f(x_v) - v^T x_v + m\|x_u - x_v\|^2 \tag{21.25}$$

$$\iff (u-v)^T(x_u - x_v) \geq m\|x_u - x_v\|_2^2 \tag{21.26}$$

$$\implies \|x_u - x_v\|_2 \cdot \|u - v\|_2 \geq (u-v)^T(x_u - x_v) \quad \text{By Cauchy-Schwartz} \tag{21.27}$$

$$\geq m\|x_u - x_v\|_2^2 \tag{21.28}$$

$$\implies \|x_u - x_v\|_2 \leq \frac{1}{m}\|u - v\|_2 \tag{21.29}$$

Proof of "$\Longleftarrow$"
Assume f^* has Lipschitz property, with constant $L = \frac{1}{m}$. Define the follow about $g_x(z)$

$$g_x(z) = f^*(z) - \nabla f^*(x)^T z$$

$$\nabla g_x(z) = \nabla f^*(z) - \nabla f^*(x)$$

$$\nabla^2 g_x(z) = \nabla^2 f^*(z)$$

so by using the Taylor expansion we can derive the following inequality,

$$g_x(z) \leq g_x(y) + \nabla g_x(y)^T(z-y) + \frac{L}{2}\|z-y\|_2^2 \tag{21.30}$$

$$\tag{21.31}$$

Now we minimize the right hand side over z, then we have the minimizer of the quadratic equation with

$$z^* = -\frac{\nabla g_x(y)}{L} + y = -\frac{\nabla f^*(y) - \nabla f^*(x)}{L} + y$$

Take this back to the inequality we have,

$$g_x(-\frac{\nabla g_x(y)}{L} + y) \leq g_x(y) - \nabla g_x(y)^T \frac{\nabla g_x(y)}{L} + \frac{L}{2}\left\| -\frac{\nabla g_x(y)}{L}\right\|_2^2 \tag{21.32}$$

$$\implies \frac{1}{2L}\left\|\nabla f^*(y) - \nabla f^*(x)\right\|_2^2 \leq f^*(y) - f^*(x) + \nabla f^*(x)^T(x-y) \tag{21.33}$$

Similarly we also have,

$$\frac{1}{2L}\left\|\nabla f^*(x) - \nabla f^*(y)\right\|_2^2 \leq f^*(x) - f^*(y) + \nabla f^*(y)^T(y-x) \tag{21.34}$$

Finally we get,

$$\frac{1}{L}\left\|\nabla f^*(x) - \nabla f^*(y)\right\|_2^2 \leq \nabla(f^*(y) - f^*(x))^T(y-x) \tag{21.35}$$

Let $u = \nabla f(x)$, $v = \nabla f(y)$, then we have $x \in \partial f^*(u)$, $y \in \partial f^*(v)$ so,

$$\frac{1}{L}\|v-u\|_2^2 \leq \nabla(\nabla f(y) - \nabla f(x))^T(y-x) \leq \|\nabla f(y) - \nabla f(x)\|_2\|y-x\|_2 \tag{21.36}$$

$$\Longrightarrow \|\nabla f(y) - \nabla f(x)\|_2 \leq \|y-x\|_2 \tag{21.37}$$

■

21.2 Dual Decomposition

21.2.1 Dual Decomposition with Equality

Consider

$$\min_x \sum_{i=1}^{B} f_i(x_i) \text{ subject to } Ax = b \tag{21.38}$$

where $x = (x_1, ..., x_B) \in R^n$ is divided into B blocks of variables, with each $x_i \in R^{n_i}$. Similarly, we can partition A accordingly:

$$A = [A_1, ..., A_B], \text{ where } A_i \in R^{m \times n_i} \tag{21.39}$$

This helps in calculation of sub-gradient where the minimization process decomposes into B separate problems:

$$x^+ \in \text{ argmin }_x \sum_{i=1}^{B} f_i(x_i) + u^T Ax \tag{21.40}$$

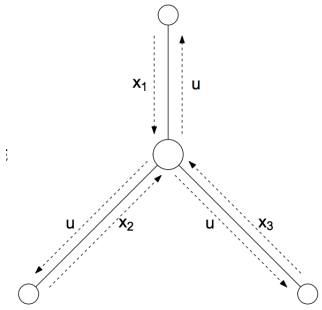
$$\Longleftrightarrow x_i^+ \in \text{ argmin }_{x_i} f_i(x_i) + u^T A_i x_i, \ i = 1, ..., B \tag{21.41}$$

Algorithm 3 Dual Decomposition Algorithm. Note that the order of update for each coordinate can be changed.

repeat for k
  1. Solve the dual sub-problems (in parallel)
  Find x_1 that minimize $f_1(x_1) + (u^{(k-1)})^T A_1 x_1$
  Find x_2 that minimize $f_2(x_2) + (u^{(k-1)})^T A_2 x_2$
  ...
  Find x_B that minimize $f_B(x_B) + (u^{(k-1)})^T A_B x_B$
  2. Update dual variables. (account together)
  $u^{(k)} = u^{(k-1)} + t_k(\sum_{i=1}^{B} A_i x_i^{(k)} - b)$

We can think of these steps as:

Figure 21.1: Visualization of the steps



- **Broadcast**: send u to each of the B processors, each optimizes in parallel to find x_i
- **Gather**: collect $A_i x_i$ from each processors, update the global dual variable u.

21.2.2 Dual Decomposition with Inequality Constraints

Consider

$$\min_x \sum_{i=1}^{B} f_i(x_i) \text{ subject to } \sum_{i=1}^{B} A_i x_i \leq b \tag{21.42}$$

where the dual decomposition, i.e., projected sub-gradient method is:

$$x_i^{(k)} \in \text{ argmin }_{x_i} f_i(x_i) + (u^{(k-1)})^T A_i x_i, \ i = 1, ..., B \tag{21.43}$$

$$\text{where } u^{(k)} = (u^{(k-1)} + t_k(\sum_{i=1}^{B} A_i x_i^{(k)} - b))_+ \tag{21.44}$$

where u_+ denotes the positive part of u, i.e., $(u_+)_i = \max\{0, u_i\}$, $i = 1, ..., m$

Interpretation:
According to Vandenberghe, we can use price coordination interpretation:

- Have B units in a system, each unit chooses its own decision variable x_i (how to allocate its goods).
- Constraints are limits on shared resources (rows of A), each component of dual variable u_j is price of resource j.
- Dual update:

$$u_j^+ = (u_j - t s_j)_+, \ j = 1, ..., m \tag{21.45}$$

  where $s = b - \sum_{i=1}^{B} A_i x_i$ are slacks

  - Increase price u_j if resource j is over-utilized, $s_j < 0$
  - Decrease price u_j if resource j is under-utilized, $s_j > 0$
  - Never let prices get negative

21.3 Augmented Lagrangian Method

One disadvantage of dual ascent is that it requires strong conditions to ensure convergence. The augmented Lagrangian method, also known as the method of multipliers, gains better convergence properties by transforming the primal problem

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

where $\rho > 0$ is a parameter. This is clearly the same problem as the original, and when A has full column rank, the objective function is strongly convex. We can iterate using dual gradient ascent

$$\begin{aligned} x^{(k)} &= \operatorname*{argmin}_x \ f(x) + (u^{(k-1)})^T Ax + \frac{\rho}{2}\|Ax - b\|_2^2 \\ u^{(k)} &= u^{(k-1)} + \rho(Ax^{(k)} - b). \end{aligned}$$

Note that we have the step size as $t_k = \rho$ in the dual algorithm. Due to the stationarity condition in the original primal problem

$$\begin{aligned} 0 &\in \partial f(x^{(k)}) + A^T(u^{(k-1)} + \rho(Ax^{(k)} - b)) \\ &= \partial f(x^{(k)}) + A^T u^{(k)} \end{aligned}$$

we have that, under mild conditions $Ax^{(k)} - b \to 0$ as $k \to \infty$ (primal iterates approach feasibility), the KKT conditions are satisfied in the limit. Hence, $x^{(k)}, u^{(k)}$ converge to solutions.

While we see much better convergence properties with the augmented Lagrangian method, we have lost the property of decomposability.

21.4 Alternating direction method of multipliers (ADMM)

Alternating direction method of multipliers strives to get the best of both words: retrain better convergence properties of the augmented Lagragian method and keep decomposability.

ADMM applies to problems of the form

$$\min_{x,z} \ f(x) + g(z) \quad \text{subject to} \quad Ax + Bz = c.$$

We can often manipulate problems to fit this form by introducing auxiliary variables. Just as we did before, we can augment the objective function

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

where $\rho > 0$ is some parameter. With this formulation, we can write the augmented Lagrangian

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

So far, everything looks the same as the augmented Lagrangian method, with the key difference being the decomposition of the primal variables into x and z. The augmented Lagrangian method would have jointly minimized the first two steps by

$$(x^{(k)}, z^{(k)}) = \operatorname*{argmin}_{x,z} \ L_p(x, z, u^{(k-1)}).$$

ADMM splits the minimization into two (hence the name "alternating"), and repeat the following steps for $k = 1, 2, 3, \ldots$

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

21.4.1 Convergence guarantees

Without requiring A, B to be full rank, and under modest assumptions on f, g (closed and convex is enough), ADMM iterates satisfy, for any $\rho > 0$

- *Residual convergence*: $r^{(k)} = Ax^{(k)} - Bz^{(k)} - x \to 0$ as $k \to \infty$
  This implies that as we continue iterating, primal iterates approach feasibility.
- *Objective convergence*: $f(x^{(k)}) + g(z^{(k)}) \to f^\star + g^\star$, where $f^\star + g^\star$ is the optimal objective value for the primal.
- *Dual convergence*: $u^{(k)} \to u^\star$, where $u^\star$ is a dual solution.

For details, see Boyd et al. (2010), brought attention to ADMM methods. Roughly, ADMM behaves like a first-order method, but it much more flexible and allows problems to be solved in parallel—even when it is not obvious from the problem's structure. Theory on convergence rates are still being worked out: see Hong and Luo (2012), Deng and Yin (2012), Iutzeler et al. (2014), Nishihara et al. (2015).

21.4.2 Scaled form ADMM

Typically, ADMM is used in scaled form for convenience. Let $w = u/\rho$. Then the augmented Lagrangian is

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

with corresponding ADMM updates

$$\begin{aligned} x^{(k)} &= \operatorname*{argmin}_x \; f(x) + \frac{\rho}{2}\|Ax + Bz^{(k-1)} - c + w^{(k-1)}\|_2^2 \\ z^{(k)} &= \operatorname*{argmin}_z \; g(z) + \frac{\rho}{2}\|Ax^{(k)} + Bz - c + w^{(k-1)}\|_2^2 \\ w^{(k)} &= w^{(k-1)} + Bz^{(k)} - c \end{aligned}$$

where the iterate $w^{(k}$ is a running sum of residuals

$$w^{(k)} = w^{(0)} + \sum_{i=1}^{k}(Ax^{(i)} + Bz^{(i)} - c).$$

21.4.3 Example: alternating projections

Consider the problem of finding a point in the intersection of two convex sets $C, D \subseteq \mathbb{R}^n$, which we write as

$$\min_x \; I_C(x) + I_D(x).$$

Recall we can reframe this as the maximum distance to each of the two sets, and apply the subgradient method with a chosen step-size. Alternatively, to get this into ADMM form, we can introduce an auxiliary variable z so that we express the problem as

$$\min_{x,z} \; I_C(x) + I_D(z) \quad \text{subject to} \quad x - z = 0.$$

Now each ADMM iteration involves two projections

$$\begin{aligned} x^{(k)} &= \operatorname*{argmin}_x \; P_C(z^{(k-1)} - w^{(k-1)}) \\ z^{(k)} &= \operatorname*{argmin}_z \; P_D(x^{(k)} - w^{(k-1)}) \\ w^{(k)} &= w^{(k-1)} + x^{(k)} - z^{(k)}. \end{aligned}$$

It is similar to the classic von Neumann alternating projections algorithm

$$\begin{aligned} x^{(k)} &= \operatorname*{argmin}_x \; P_C(z^{(k-1)}) \\ z^{(k)} &= \operatorname*{argmin}_z \; P_D(x^{(k)}) \end{aligned}$$

but we now have a dual variable, w, which is often called the "offset" variable. In this setting, the ADMM algorithm converges much quicker than standard alternating projections. To see this, consider one of the sets, say C, is a linear subspace (D is some generic closed convex set). Then the ADMM updates are

$$\begin{aligned} x^{(k)} &= \operatorname*{argmin}_x \; P_C(z^{(k-1)}) \\ z^{(k)} &= \operatorname*{argmin}_z \; P_D(x^{(k)} - w^{(k-1)}) \\ w^{(k)} &= w^{(k-1)} + x^{(k)} - z^{(k)}, \end{aligned}$$

since, due to linearity, w does not matter for the first projection. This is equivalent to Dykstra's algorithm (which came after von Neumann's designed for better convergence properties) when $z^{(0)} = y$ for finding the closest point in the intersection $C \cap D$ to y.

21.5 Extra discussion: scope

Let's consider first-order methods, second-order methods, coordinate descent, and ADMM. First-order methods and second-order methods are heavy machinery we can apply to optimization problems. Interior point methods can be applied very widely and generically (after some reparameterization), but can be computationally slower. Interior point methods start being competitive when we know the Hessian has good structure (e.g. banded, sparse) that we can leverage.

When comparing first-order methods to coordinate descent to ADMM, we see that first-order methods have similar advantages from generality. Coordinate descent applies to a more specific subgroup of optimization problems, but can be chosen depending on how quickly the coordinate updates can be computed. ADMM converges very differently depending on how the decomposition is done, roughly on the order of first-order methods, slower than coordinate descent. A major strength of ADMM lies in its parallelizable structure, which will be discussed in the following lecture.

References

[Boyd 10] S. Boyd and N. Parikh and E. Chu and B. Peleato and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," 2010.

[Deng 12] W. Deng and W. Yin, "On the global and linear convergence of the generalized alternating direction method of multipliers," 2012.

[Hong 12] M. Hong and Z. Luo, "On the linear convergence of the alternating direction method of multipliers," 2012.

[Iutz 14] F. Iutzeler and P. Bianchi and Ph. Ciblat and W. Hachem, "Linear convergence rate for distributed optimization with the alternating direction method of multipliers," 2014.

[Nish 15] R. Nishihara and L. Lessard and B. Recht and A. Packard and M. Jordan, "A general analysis of the convergence of ADMM," 2015.