Dual Methods and ADMM

Ryan Tibshirani
Convex Optimization 10-725/36-725
Last time: case study of generalized lasso

We studied generalized lasso problems:

$$\min_{\beta} \ f(\beta) + \lambda \|D\beta\|_1$$

where $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a smooth, convex function and $D \in \mathbb{R}^{m \times n}$ is a penalty matrix

- We derived its dual problem, and considered applying all of the algorithms we’ve learned so far to both its primal and its dual
- We saw that different algorithms had different strengths, and were suitable for different situations

For the remainder of the course, we will study advanced methods, which go beyond the first- and second-order paradigms
Conjugate functions

Reminder: given $f : \mathbb{R}^n \to \mathbb{R}$, the function

$$f^*(y) = \max_x y^T x - f(x)$$

is called its conjugate

- Conjugates appear frequently in dual programs, since

$$-f^*(y) = \min_x f(x) - y^T x$$

- If $f$ is closed and convex, then $f^{**} = f$. Also,

$$x \in \partial f^*(y) \iff y \in \partial f(x) \iff x \in \arg\min_z f(z) - y^T z$$

- If $f$ is strictly convex, then $\nabla f^*(y) = \arg\min_z f(z) - y^T z$
Outline

Today:

- Dual (sub)gradient methods
- Dual decomposition
- Augmented Lagrangians
- ADMM
Dual (sub)gradient methods

What if we can’t derive dual (conjugate) in closed form, but want to utilize dual relationship? Turns out we can still use dual-based subgradient or gradient methods

Example: consider the problem

\[
\min_x f(x) \quad \text{subject to} \quad Ax = b
\]

Its dual problem is

\[
\max_u -f^*(-A^T u) - b^T u
\]

where \( f^* \) is conjugate of \( f \). Defining \( g(u) = f^*(-A^T u) \), note that \( \partial g(u) = -A \partial f^*(-A^T u) \), and recall

\[
x \in \partial f^*(-A^T u) \iff x \in \arg\min_z f(z) + u^T A z
\]
Therefore the dual subgradient method (for maximizing the dual objective) starts with an initial dual guess $u^{(0)}$, and repeats for $k = 1, 2, 3, \ldots$

$$x^{(k)} \in \arg\min_x f(x) + (u^{(k-1)})^T Ax$$

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

where $t_k$ are step sizes, chosen in standard ways.

Recall that if $f$ is strictly convex, then $f^*$ is differentiable, and so we get dual gradient ascent, which repeats for $k = 1, 2, 3, \ldots$

$$x^{(k)} = \arg\min_x f(x) + (u^{(k-1)})^T Ax$$

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

(difference is that each $x^{(k)}$ is unique, here). Proximal gradients and acceleration carry through in similar manner.
Covergence analysis

First recall that if $f$ strongly convex with parameter $d$, then $\nabla f^*$ Lipschitz with parameter $1/d$

Proof: if $f$ strongly convex and $x$ is its minimizer, then

\[
    f(y) \geq f(x) + \frac{d}{2} \|y - x\|_2, \quad \text{for all } y
\]

Hence defining $x_u = \nabla f^*(u)$, $x_v = \nabla f^*(v)$,

\[
    f(x_v) - u^T x_v \geq f(x_u) - u^T x_u + \frac{d}{2} \|x_u - x_v\|_2^2
\]

\[
    f(x_u) - v^T x_u \geq f(x_v) - v^T x_v + \frac{d}{2} \|x_u - x_v\|_2^2
\]

Adding these together, using Cauchy-Schwartz, and rearranging shows that

\[
    \|x_u - x_v\|_2 \leq \frac{1}{d} \cdot \|u - v\|_2
\]
Applying what we know about gradient descent: if $f$ is strongly convex with parameter $d$, then dual gradient ascent with constant step size $t_k \leq d$ converges at rate $O(1/\epsilon)$

Is this a slow or fast rate, compared to what we would get out of primal gradient descent? It’s actually essentially the same

- When $f$ is strongly convex, primal gradient descent converges at rate $O(1/\epsilon)$. But if we further assume that $\nabla f$ is Lipschitz, then we get the linear rate $O(\log(1/\epsilon))$

- Note: the converse of the statement on the last slide is also true: $\nabla f^*$ being Lipschitz with parameter $1/d$ implies that $f$ is strongly convex with parameter $d$

- Hence assume $f^{**} = f$. When $f$ has Lipschitz gradient and is strongly convex, the same is true about $f^*$, and dual gradient ascent also converges at the linear rate $O(\log(1/\epsilon))$
Dual decomposition

Consider

\[
\min_x \sum_{i=1}^B f_i(x_i) \quad \text{subject to} \quad Ax = b
\]

Here \( x = (x_1, \ldots, x_B) \in \mathbb{R}^n \) divides into \( B \) blocks of variables, with each \( x_i \in \mathbb{R}^{n_i} \). We can also partition \( A \) accordingly

\[
A = [A_1, \ldots, A_B], \quad \text{where} \quad A_i \in \mathbb{R}^{m \times n_i}
\]

Simple but powerful observation, in calculation of (sub)gradient:

\[
x^+ \in \arg\min_x \sum_{i=1}^B f_i(x_i) + u^T Ax
\]

\[
\iff x_i^+ \in \arg\min_{x_i} f_i(x_i) + u^T A_i x_i, \quad i = 1, \ldots, B
\]

i.e., minimization decomposes into \( B \) separate problems
Dual decomposition algorithm: repeat for $k = 1, 2, 3, \ldots$

$$x_i^{(k)} \in \arg\min_{x_i} f_i(x_i) + (u^{(k-1)})^T A_i x_i, \quad i = 1, \ldots B$$

$$u^{(k)} = u^{(k-1)} + t_k \left( \sum_{i=1}^{B} A_i x_i^{(k-1)} - b \right)$$

Can think of these steps as:

- **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 processor, update the global dual variable $u$
Example with inequality constraints:

\[
\min_x \sum_{i=1}^{B} f_i(x_i) \quad \text{subject to} \quad \sum_{i=1}^{B} A_i x_i \leq b
\]

Dual decomposition (projected subgradient method) repeats for \( k = 1, 2, 3, \ldots \)

\[
x_i^{(k)} \in \arg\min_{x_i} f_i(x_i) + (u^{(k-1)})^T A_i x_i, \quad i = 1, \ldots, B
\]

\[
v^{(k)} = u^{(k-1)} + t_k \left( \sum_{i=1}^{B} A_i x_i^{(k-1)} - b \right)
\]

\[
u^{(k)} = (v^{(k)})_+
\]

where \((\cdot)_+\) is componentwise thresholding, \((u_+)_i = \max\{0, u_i\}\)
Price coordination interpretation (from Vandenberghe’s lecture notes):

- 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:

$$u_j^+ = (u_j - ts_j)_+, \quad j = 1, \ldots, m$$

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
Augmented Lagrangian

Disadvantage of dual methods: require strong conditions to ensure primal iterates converge to solutions. Convergence properties can be improved by utilizing augmented Lagrangian. Transform primal:

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

Clearly extra term $(\rho/2) \cdot \|Ax - b\|_2^2$ does not change problem. Use dual gradient ascent: repeat for $k = 1, 2, 3, \ldots$

$$x^{(k)} = \arg\min_x f(x) + (u^{(k-1)})^T Ax + \frac{\rho}{2} \|Ax - b\|_2^2$$
$$u^{(k)} = u^{(k-1)} + \rho (Ax^{(k-1)} - b)$$

(When, e.g., $A$ has full column rank, primal is guaranteed strongly convex)
Notice step size choice $t_k = \rho$, for all $k$, in dual gradient ascent. Why? Since $x^{(k)}$ minimizes $f(x) + (u^{(k-1)})^T Ax + \frac{\rho}{2} \|Ax - b\|^2_2$ over $x$, we have

$$0 \in \partial f(x^{(k)}) + A^T \left( u^{(k-1)} + \rho(Ax^{(k)} - b) \right)$$

$$= \partial f(x^{(k)}) + A^T u^{(k)}$$

This is the stationarity condition for the original primal problem; can show under mild conditions that $Ax^{(k)} - b$ approaches zero (i.e., primal iterates approach feasibility), hence in the limit KKT conditions are satisfied and $x^{(k)}, u^{(k)}$ approach optimality

Advantage: much better convergence properties. Disadvantage: lose decomposability! (Separability is compromised by augmented Lagrangian ...)
Alternating direction method of multipliers or ADMM: the best of both worlds!

I.e., good convergence properties of augmented Lagrangians, along with decomposability

Consider minimization problem

$$\min_x f_1(x_1) + f_2(x_2) \quad \text{subject to} \quad A_1 x_1 + A_2 x_2 = b$$

As before, we augment the objective

$$\min_x f_1(x_1) + f_2(x_2) + \frac{\rho}{2} \| A_1 x_1 + A_2 x_2 - b \|^2_2$$

subject to \( A_1 x_1 + A_2 x_2 = b \)
Write the augmented Lagrangian as

\[ L_\rho(x_1, x_2, u) = f_1(x_1) + f_2(x_2) + u^T(A_1x_1 + A_2x_2 - b) + \]
\[ \frac{\rho}{2} \|A_1x_1 + A_2x_2 - b\|^2_2 \]

Now ADMM repeats the steps, for \( k = 1, 2, 3, \ldots \)

\[ x_1^{(k)} = \arg\min_{x_1} L_\rho(x_1, x_2^{(k-1)}, u^{(k-1)}) \]
\[ x_2^{(k)} = \arg\min_{x_2} L_\rho(x_1^{(k)}, x_2, u^{(k-1)}) \]
\[ u^{(k)} = u^{(k-1)} + \rho(A_1x_1^{(k)} + A_2x_2^{(k)} - b) \]

Note that the usual method of multipliers would have replaced the first two steps by

\[ (x_1^{(k)}, x_2^{(k)}) = \arg\min_{x_1, x_2} L_\rho(x_1, x_2, u^{(k-1)}) \]
Convergence guarantees

Under modest assumptions on $f_1, f_2$ (these do not require $A_1, A_2$ to be full rank), the ADMM iterates satisfy, for any $\rho > 0$:

- **Residual convergence**: $r^{(k)} = A_1 x_1^{(k)} - A_2 x_2^{(k)} - b \to 0$ as $k \to \infty$, i.e., primal iterates approach feasibility
- **Objective convergence**: $f_1(x_1^{(k)}) + f_2(x_2^{(k)}) \to f^*$, where $f^*$ is the optimal primal criterion value
- **Dual convergence**: $u^{(k)} \to u^*$, where $u^*$ is a dual solution

For details, see Boyd et al. (2010). Note that we do not generically get primal convergence, but this can be shown under more assumptions

Convergence rate: not known in general, but known in few special cases. Rough consensus seems to be that it behave like first-order methods
Scaled form

It is often easier to express the ADMM algorithm in a scaled form, where we replace the dual variable $u$ by a scaled variable $w = u / \rho$. In this parametrization, the ADMM steps are

$$
x_1^{(k)} = \arg\min_{x_1} f_1(x_1) + \frac{\rho}{2} \| A_1 x_1 + A_2 x_2^{(k-1)} - b + w^{(k-1)} \|_2^2
$$

$$
x_2^{(k)} = \arg\min_{x_2} f_2(x_2) + \frac{\rho}{2} \| A_1 x_1^{(k)} + A_2 x_2 - b + w^{(k-1)} \|_2^2
$$

$$
w^{(k)} = w^{(k-1)} + A_1 x_1^{(k)} + A_2 x_2^{(k)} - b
$$

Note that here the $k$th iterate $w^{(k)}$ is just given by a running sum of residuals:

$$
w^{(k)} = w^{(0)} + \sum_{i=1}^{k} (A_1 x_1^{(i)} + A_2 x_2^{(i)} - b)
$$
Practicalities and tricks

Practical experience shows that ADMM usually obtains a relatively accurate solution in a handful of iterations, but requires a very large number of iterations for a highly accurate solution. This is more evidence that it behaves like a first-order method.

Choice of $\rho$ can greatly influence practical convergence of ADMM:

- $\rho$ too large $\rightarrow$ not enough emphasis on minimizing $f_1 + f_2$
- $\rho$ too small $\rightarrow$ not enough emphasis on feasibility

Boyd et al. (2010) give a strategy for varying $\rho$ that can be useful in practice (but does not have convergence guarantees)

Like deriving duals, transforming a problem into that ADMM can handle often requires a bit of trickery (and different forms can lead to different algorithms)
Example: alternating projections

Consider finding a point in intersection of convex sets $C, D \subseteq \mathbb{R}^n$, i.e., solving

$$\min_x 1_C(x) + 1_D(x)$$

To get this into ADMM form, we express it as

$$\min_{x,z} 1_C(x) + 1_D(z) \text{ subject to } x - z = 0$$

Each ADMM cycle involves two projections:

$$x^{(k)} = \arg\min_x P_C(z^{(k-1)} - w^{(k-1)})$$

$$z^{(k)} = \arg\min_z P_D(x^{(k)} + w^{(k-1)})$$

$$w^{(k)} = w^{(k-1)} + x^{(k)} - z^{(k)}$$

This is like the classical alternating projections method, but now with a dual variable $w$. It is much more efficient
Example: generalized lasso regression

Given the usual $y \in \mathbb{R}^n$, $X \in \mathbb{R}^{n \times p}$, and an additional $D \in \mathbb{R}^{m \times p}$, the generalized lasso problem solves

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{2} \| y - X \beta \|_2^2 + \lambda \| D \beta \|_1$$

This computationally harder than the lasso problem (with $D = I$); recall our study on algorithms for this problem. We can rewrite as

$$\min_{\beta \in \mathbb{R}^p, \alpha \in \mathbb{R}^m} \frac{1}{2} \| y - X \beta \|_2^2 + \lambda \| \alpha \|_1 \quad \text{subject to} \quad D \beta - \alpha = 0$$

and ADMM gives us a simple algorithm for the generalized lasso:

$$\beta^{(k)} = (X^T X + \rho D^T D)^+ (X^T y + \rho D^T (\alpha^{(k-1)} - w^{(k-1)}))$$
$$\alpha^{(k)} = S_{\lambda/\rho} (D \beta^{(k)} + w^{(k-1)})$$
$$w^{(k)} = w^{(k-1)} + D \beta^{(k)} - \alpha^{(k)}$$
Example: sum-of-norms regularization

Now consider

\[
\min_{\beta \in \mathbb{R}^p} \frac{1}{2} \| y - X \beta \|^2_2 + \lambda \sum_{g=1}^{G} \| \beta_{I_g} \|^2_2
\]

where each \( \beta_{I_g} \in \mathbb{R}^{|I_g|} \) is a sub-block of the full coefficient vector \( \beta \). Called a group lasso problem, or a sum-of-norms regularization problem when we generalize the \( \ell_2 \) norm above. Rewrite as

\[
\min_{\beta \in \mathbb{R}^p, \alpha \in \mathbb{R}^p} \frac{1}{2} \| y - X \beta \|^2_2 + \lambda \sum_{g=1}^{G} \| \alpha_{I_g} \|^2_2 \quad \text{subject to} \quad \beta - \alpha = 0
\]

and if ADMM updates become:

\[
\beta^{(k)} = (X^T X + \rho I)^{-1} (X^T y + \rho (\alpha^{(k-1)} - w^{(k-1)}))
\]

\[
\alpha_{I_g}^{(k)} = R_{\lambda/\rho} \left( \beta_{I_g}^{(k)} + w_{I_g}^{(k-1)} \right), \quad g = 1, \ldots, G
\]

\[
w^{(k)} = w^{(k-1)} + \beta^{(k)} - \alpha^{(k)}
\]
Notes:

- The matrix $X^T X + \rho I$ is always invertible, regardless of $X$
- If we take its factorization (say QR), in $O(p^3)$ flops, then each subsequent solve takes $O(p^2)$ flops
- The shrinkage operator $R_t$ is defined as
  \[ R_t(x) = \left(1 - \frac{t}{\|x\|_2}\right)_+ x \]
- Similar steps can be performed for a sum-of-norms problem, as long as can solve for the prox operator of the individual norms
- An ADMM algorithm can also be developed for the case of overlapping groups (which is otherwise quite a hard problem to optimize!). See Boyd et al. (2010)
References

- L. Vandenberghe, Lecture Notes for EE 236C, UCLA, Spring 2011-2012