Dual Methods and ADMM

Ryan Tibshirani
Convex Optimization 10-725/36-725

Last time: case studies of regularization problems

We studied fused lasso problems:

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

where $f : \mathbb{R}^n \to \mathbb{R}$ is a smooth, convex function and $D \in \mathbb{R}^{m \times n}$ is a difference operator. We also studied group lasso problems:

$$\min_{\beta} \; f(\beta) + \lambda \sum_{j=1}^{J} c_j \|\beta_{(j)}\|_2$$

where $\beta = (\beta_{(1)}, \ldots \beta_{(j)})$ is a block decomposition. (And briefly, group fused lasso problems)

We considered all algorithms we've learned so far applied to these problems (or in some cases, their duals), and saw these algorithms had different strengths, and were suitable for different situations

Reminder: conjugate functions

Recall that 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,

$$x \in \partial f^*(y) \iff y \in \partial f(x) \iff x \in \operatorname*{argmin}_z \; f(z) - y^T z$$

- If f is strictly convex f, then $\nabla f^*(y) = \operatorname*{argmin}_z \; f(z) - y^T z$

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 subradient or gradient methods

Example: consider the problem

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

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

$$x \in \partial f^*(-A^T u) \iff x \in \operatorname*{argmin}_z \ f(z) + u^T A z$$

Therefore the dual subgradient method (for maximizing the dual objective) starts with an initial dual guess $u^{(0)}$, and repeats for $k = 1, 2, 3, \ldots$

$$x^{(k)} \in \operatorname*{argmin}_x \; f(x) + (u^{(k-1)})^T Ax$$
$$u^{(k)} = u^{(k-1)} + t_k(Ax^{(k)} - b)$$

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 $k = 1, 2, 3, \ldots$

$$x^{(k)} = \operatorname*{argmin}_x \; f(x) + (u^{(k-1)})^T Ax$$
$$u^{(k)} = u^{(k-1)} + t_k(Ax^{(k)} - b)$$

(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 ∇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 ∇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: ∇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

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

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

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

$$\iff x_i^+ \in \operatorname*{argmin}_{x_i} f_i(x_i) + u^T A_i x_i, \quad i = 1, \ldots B$$

i.e., minimization decomposes into B separate problems

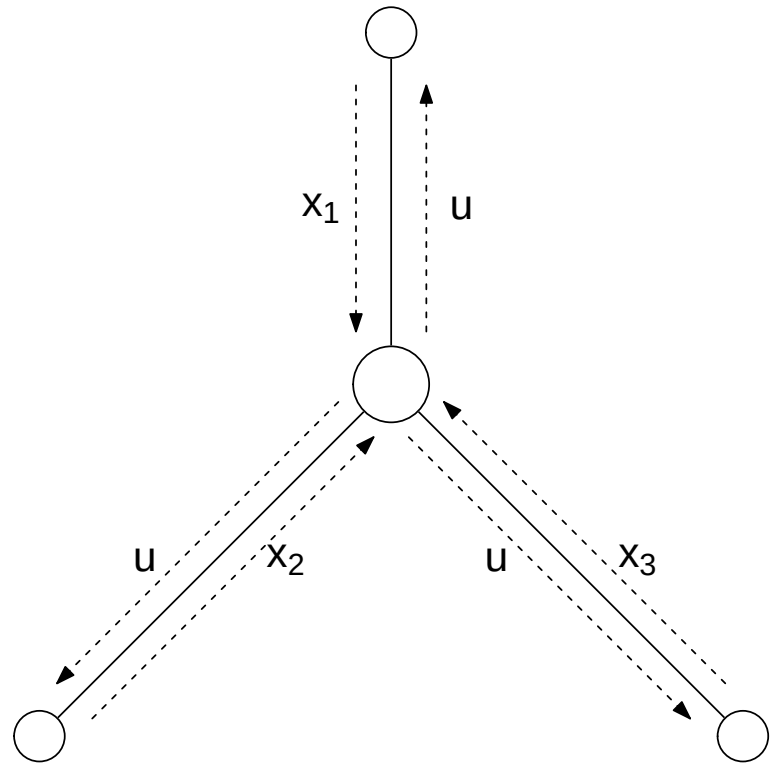
Dual decomposition algorithm: repeat for $k = 1, 2, 3, \ldots$

$$x_i^{(k)} \in \operatorname*{argmin}_{x_i} \; f_i(x_i) + (u^{(k-1)})^T A_i x_i, \quad i = 1, \ldots B$$

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

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 $k = 1, 2, 3, \ldots$

$$x_i^{(k)} \in \operatorname*{argmin}_{x_i} \; f_i(x_i) + (u^{(k-1)})^T A_i x_i, \quad i = 1, \ldots B$$

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

$$u^{(k)} = (v^{(k)})_+$$

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:

$$\begin{array}{ll} \min\limits_{x} & f(x) + \dfrac{\rho}{2}\|Ax - b\|_2^2 \\ \text{subject to} & Ax = b \end{array}$$

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

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

(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

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

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

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

$$\min_x \ f_1(x_1) + f_2(x_2) \ \text{ subject to } \ A_1x_1 + A_2x_2 = b$$

As before, we augment the objective

$$\begin{aligned} \min_x \quad & f_1(x_1) + f_2(x_2) + \frac{\rho}{2}\|A_1x_1 + A_2x_2 - b\|_2^2 \\ \text{subject to} \quad & A_1x_1 + A_2x_2 = b \end{aligned}$$

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 $k = 1, 2, 3, \ldots$

$$\begin{aligned}
x_1^{(k)} &= \operatorname*{argmin}_{x_1} \; L_\rho(x_1, x_2^{(k-1)}, u^{(k-1)}) \\
x_2^{(k)} &= \operatorname*{argmin}_{x_2} \; L_\rho(x_1^{(k)}, x_2, u^{(k-1)}) \\
u^{(k)} &= u^{(k-1)} + \rho(A_1x_1^{(k)} + A_2x_2^{(k)} - b)
\end{aligned}$$

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

$$(x_1^{(k)}, x_2^{(k)}) = \operatorname*{argmin}_{x_1, x_2} \; L_\rho(x_1, x_2, u^{(k-1)})$$

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: $f_1(x_1^{(k)}) + f_2(x_2^{(k)}) \to f^\star$, where $f^\star$ is the optimal primal criterion value
- Dual convergence: $u^{(k)} \to u^\star$, where $u^\star$ is a dual solution

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

Convergence rate: not known in general, theory is currently being developed, e.g., in Hong and Luo (2012), Nishihara et al. (2015). Roughly, it behaves like a first-order method (or a bit faster)

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

$$
\begin{aligned}
x_1^{(k)} &= \underset{x_1}{\operatorname{argmin}} \; f_1(x_1) + \frac{\rho}{2}\|A_1 x_1 + A_2 x_2^{(k-1)} - b + w^{(k-1)}\|_2^2 \\
x_2^{(k)} &= \underset{x_2}{\operatorname{argmin}} \; f_2(x_2) + \frac{\rho}{2}\|A_1 x_1^{(k)} + A_2 x_2 - b + w^{(k-1)}\|_2^2 \\
w^{(k)} &= w^{(k-1)} + A_1 x_1^{(k)} + A_2 x_2^{(k)} - b
\end{aligned}
$$

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

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

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 ρ can greatly influence practical convergence of ADMM:

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

Boyd et al. (2010) give a strategy for varying ρ 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:

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

This is like the classical alternating projections method, but now with a dual variable w. It is more efficient

Example: fused lasso regression

Given $y \in \mathbb{R}^n$, $X \in \mathbb{R}^{n \times p}$, and a difference operator $D \in \mathbb{R}^{m \times p}$, the fused lasso regression 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

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

and ADMM gives us a simple algorithm for this problem:

$$\begin{aligned}
\beta^{(k)} &= (X^T X + \rho D^T D)^{-1}\big(X^T y + \rho D^T(\alpha^{(k-1)} - w^{(k-1)})\big) \\
\alpha^{(k)} &= S_{\lambda/\rho}(D\beta^{(k)} + w^{(k-1)}) \\
w^{(k)} &= w^{(k-1)} + D\beta^{(k)} - \alpha^{(k)}
\end{aligned}$$

Notes:

- The matrix $X^T X + \rho D^T D$ is assumed here to be invertible; if not, replace the inverse by a pseudoinverse
- If we take its factorization (say QR), in $O(p^3)$ flops, then each subsequent solve takes $O(p^2)$ flops
- The soft-thresolding operator S_t recall is defined as

$$[S_t(x)]_j = \begin{cases} x_j - t & x > t \\ 0 & -t \leq x \leq t \\ x_j + t & x < -t \end{cases}, \quad j = 1, \ldots p$$

- A different initial reparametrization, rather than $D\beta = \alpha$, will lead to a different ADMM algorithm
- Sometimes it is more efficient to make the substitution $\beta = \alpha$ in the penalty term; this is favorable when $h(x) = \|Dx\|_1$ has a fast proximal operator

Example: group lasso regression

Now consider the group lasso regression problem

$$\min_{\beta \in \mathbb{R}^p} \ \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \sum_{j=1}^{J} c_j \|\beta_{(j)}\|_2$$

Rewrite as

$$\min_{\beta \in \mathbb{R}^p,\, \alpha \in \mathbb{R}^p} \ \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \sum_{j=1}^{J} c_j \|\alpha_{(j)}\|_2 \ \text{ subject to } \ \beta - \alpha = 0$$

andf ADMM updates become:

$$\begin{aligned}
\beta^{(k)} &= (X^T X + \rho I)^{-1}\big(X^T y + \rho(\alpha^{(k-1)} - w^{(k-1)})\big) \\
\alpha_{(j)}^{(k)} &= R_{c_j \lambda/\rho}\big(\beta_{(j)}^{(k)} + w_{(j)}^{(k-1)}\big), \quad j = 1, \ldots J \\
w^{(k)} &= w^{(k-1)} + \beta^{(k)} - \alpha^{(k)}
\end{aligned}$$

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 arbitrary norms of subblocks, 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)

Consensus ADMM

Consider a problem of the form

$$\min_x \sum_{i=1}^{B} f_i(x)$$

The consensus ADMM approach begins by reparametrizing:

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

and this yields the decomposable ADMM updates:

$$x_i^{(k)} = \operatorname*{argmin}_{x_i}\ f_i(x_i) + \frac{\rho}{2}\|x_i - x^{(k-1)} + w_i^{(k-1)}\|_2^2, \quad i = 1,\dots B$$

$$x^{(k)} = \frac{1}{B}\sum_{i=1}^{B}\left(x_i^{(k)} + w_i^{(k-1)}\right)$$

$$w_i^{(k)} = w_i^{(k-1)} + x_i^{(k)} - x^{(k)}, \quad i = 1,\dots B$$

Write $\bar{x} = \frac{1}{B}\sum_{i=1}^B x_i$ and similarly for other variables. Not hard to see that $\bar{w}^{(k)} = 0$ for all iterations $k \geq 1$

Hence ADMM steps can be simplified, by taking $x^{(k)} = \bar{x}^{(k)}$:

$$x_i^{(k)} = \underset{x_i}{\operatorname{argmin}} \; f_i(x_i) + \frac{\rho}{2}\|x_i - \bar{x}^{(k-1)} + w_i^{(k-1)}\|_2^2, \quad i = 1, \ldots B$$

$$w_i^{(k)} = w_i^{(k-1)} + x_i^{(k)} - \bar{x}^{(k)}, \quad i = 1, \ldots B$$

To reiterate, the x_i, $i = 1, \ldots B$ updates here are done in parallel

Intuition:

- We try to minimize each $f_i(x_i)$, and use ridge regularization to pull each x_i towards the average $\bar{x}$
- If a variable x_i is bigger than the average, then w_i is increased
- So the ridge regularization in the next step pulls x_i even closer

General consensus ADMM with regularization

Consider 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 again reparametrize:

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

and this yields the decomposable ADMM updates:

$$x_i^{(k)} = \operatorname*{argmin}_{x_i} f_i(a_i^T x_i + b_i) + \frac{\rho}{2}\|x_i - x^{(k-1)} + w_i^{(k-1)}\|_2^2,$$

$$i = 1,\dots B$$

$$x^{(k)} = \operatorname*{argmin}_{x} \frac{B\rho}{2}\|x - \bar{x}^{(k)} - \bar{w}^{(k-1)}\|_2^2 + g(x)$$

$$w_i^{(k)} = w_i^{(k-1)} + x_i^{(k)} - x^{(k)}, \quad i = 1,\dots B$$

Notes:

- It is no longer true that $w^{(k)} = 0$, so ADMM steps do not simplify as before
- To reiterate, the x_i, $i = 1, \ldots B$ are done in parallel
- Each x_i, $i = 1, \ldots B$ can be thought of as a loss minimization on part of the data, with ridge regularization
- The x update is a proximal operation in regularizer g
- The w update drives the individual variables into consensus
- A different initial reparametrization (i.e., changing $x_i = x$, $i = 1, \ldots B$ to say, $a_i^T x_i + b_i = x$, $i = 1, \ldots B$) will lead to a different ADMM algorithm

See Boyd et al. (2010) for more details about consensus ADMM, implementation tips, and more advanced strategies for splitting up into different subproblems

References

- 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"
- M. Hong and Z. Luo (2012), "On the linear convergence of the alternating direction method of multipliers"
- R. Nishihara and L. Lessard and B. Recht and A. Packard and M. Jordan (2015), "A general analysis of the convergence of ADMM"
- L. Vandenberghe, Lecture Notes for EE 236C, UCLA, Spring 2011-2012