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

Lecture 23: November 19

Lecturer: Ryan Tibshirani *Scribes: Charvi Rastogi, George Stoica, Shuo Li*

Charvi Rastogi: 23.1-23.4.2, George Stoica: 23.4.3-23.8, Shuo Li- 23.9-23.12
Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

23.1 Recap: ADMM

We use the ADMM method for the problem where the objective is split up as $f(x), g(z)$

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

. If the problem isn't originally in this form, recall that we can introduce auxiliary variables to get it in this form. we form augmented Lagrangian (scaled form):

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

if you expand the second term you get $\frac{\rho}{2}(2w^T(Ax + Bz - c) + \|Ax + Bz - c\|_2^2)$ ("augmented" lagrangian form). Alternation Direction methods of multipliers or ADMM has the following update step:

$$\begin{aligned} x^{(k)} &= \arg\min_x L_\rho(x, z^{(k-1)}, w^{(k-1)}) \\ z^{(k)} &= \arg\min_z L_\rho(x^{(k)}, z, w^{(k-1)}) \\ w^{(k)} &= w^{(k-1)} + Ax^{(k)} + Bz^{(k)} - c \end{aligned}$$

We saw that ADMM converges like a first-order method and has a very flexible framework. It can be applied to simple problems like LASSO along with more difficult problems like SDPs which cannot be solved easily using interior point methods.

23.2 Projected Gradient Descent

The motivation for Frank-Wolfe is projected gradient descent. Projected gradient descent is a special case of proximal gradient descent. Consider a constrained optimization problem, where the set the solution is constrained to belong to is defined as C,

$$\min_x f(x) \quad \text{subject to} \quad x \in C$$

where f is convex and smooth, and C is convex. Recall projected gradient descent uses an initial $x^{(0)}$, and then updates for $k = 1, 2, 3, \cdots$ by first performing gradient descent on the then current solution and then projecting it back onto the constraint set. This can be expressed as

$$x^{(k)} = P_c(x^{(k-1)} - t_k \nabla f(x^{(k-1)})$$

where P_c is the projection operator onto the set C. Special case of proximal gradient, motivated by local quadratic expansion of f:

$$x^{(k)} = P_c\Big(\underset{y}{\arg\min} \nabla f(x^{(k-1)})^T(y - x^{(k-1)}) + \frac{1}{2t}\|y - x^{(k-1)}\|_2^2\Big)$$

For this method we need to know the projection operator. The projection operator for a given constraint may or may not be known beforehand, it could be computationally too. In such a setup, moving from one constraint to another can change the difficulty of (projection operator) solution drastically. For ex, if your constraint set is a polyhedron, $C = \{x : Ax \leq b\}$, then projection onto it is generally very hard. There are special cases, such as if the polyhedra is a simplex $C = \{1^T x, x \geq 0\}$, then the projection can be computed in linear time.

23.3 Frank-Wolfe Method

The Frank-Wolfe method is an alternative to Projected Gradient Descent which doesn't involve projections. The Frank-Wolfe method is also known as conditional gradient method. Instead of using a quadratic expansion as shown above for the projected GD method, it uses a local linear expansion of f. Note that if you were to apply projected GD to a linear approximation of f, the minimum will be at $-\infty$ unless you are already at the solution ($\nabla f = 0$). In the Frank-Wolfe method they minimize the linear approximation over the constraint set, instead of projecting afterwards separately. Let s represent y, then

$$s^{(k-1)} \in \underset{s \in C}{\arg\min} \nabla f(x^{(k-1)})^T s$$
$$x^{(k)} = (1 - \gamma_k)x^{(k-1)} + \gamma s^{(k-1)}$$

This implies that we don't move fully in the direction of the projection, we take a convex combination of the new point $s^{(k-1)}$ and $x^{(k-1)}$ There is no projection involved, this method is solved directly over the constraint set C.
The default choice of step sizes is $\gamma_k = 2/(k+1), k = 1, 2, 3, \cdots$. This choice has been made to facilitate the convergence proof and to derive the convergence rate. Note for any $0 \leq \gamma_k \leq 1$, we have $x^{(k)} \in C$ by convexity. So we can rewrite the update as

$$x^{(k)} = x^{(k-1)} + \gamma_k(s^{(k-1)} - x^{(k-1)}).$$

Here, $(s^{(k-1)} - x^{(k-1)})$ represents the direction we are traveling in. This update states that we are moving less and less in the direction of the linearization minimizer as the algorithm proceeds (since γ_k is decreasing as $1/k$..

23.4 Norm Constraints

Now, we look at how the Frank-Wolfe method deals with optimization problems where the constraint is on the norm of the solution, ie, what happens when $C = \{x : \|x\| \leq t\}$ for a norm $\|\cdot\|$. By definition of the

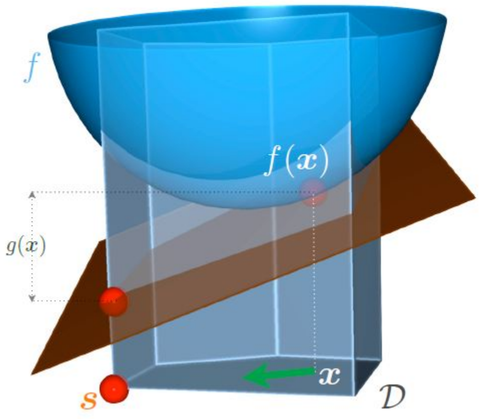

Figure 23.1: (from Jaggi 2011) $f(x)$ is a differentiable convex function, D is the constraint set. The brown plane is a linear approximation of the function at x. s is the point that minimizes the linear approximation over the constraint set D. Instead of moving to the point s, the update is a convex combination of x, s.

problem, we have

$$s \in \underset{\|x\| \le t}{\arg\min} \nabla f(x^{(k-1)})^T s$$

We see that,

$$\begin{aligned}\min_{\|s\| \le t} \nabla f(x)^T s &= -\max_{\|s\| \le t} -\nabla f(x)^T s \\ &= -t \cdot \max_{\|z\| \le 1} -\nabla f(x)^T z \\ &= -t \cdot \max_{\|z\| \le 1} \nabla f(x)^T z.\end{aligned}$$

Therefore, $\arg\min_{\|s\| \le t} \nabla f(x)^T s = -t \arg\max_{\|s\| \le 1} \nabla f(x)^T s$ which is equal to the subdifferential of the dual norm. The dual norm is written as

$$\|z\|_* = \max_{\|z\| \le 1} z^T x$$

Hence, following the rule of the subgradients of the max function, we have,

$$\begin{aligned}s &= -t \cdot \left(\underset{\|s\| \le t}{\arg\max} \nabla f(x^{(k-1)})^T s \right) \\ &= -t \cdot \delta \|\nabla f(x^{(k-1)})\|_*\end{aligned}$$

where $\| \cdot \|_*$ denotes the corresponding dual norm. That is, if we know how to compute the subgradients of the dual norm, then we can easily perform the Frank-Wolfe steps. Since we have a closed form update now, this can often be much cheaper or simpler than projection onto $C = \{x : \|x\| \le t\}$. We, now, look at some examples of norm based constraints and how to derive the Frank-Wolfe method in these special cases.

23.4.1 ℓ_1 Regularization

We look at the updates for some special cases of norm constraints and compare it to the projection gradient descent method for these cases. If we were to solve the constrained form of LASSO or logistic LASSO, we'd have a ℓ_1 ball as the constraint set C. To use the Frank-Wolfe method we'd need to know what the dual norm is and what its subgradient looks like, for the ℓ_1 norm.

$$\min_x f(x) \quad \text{subject to} \quad \|x\|_1 \le t$$

The dual norm of the ℓ_1 norm is the ℓ_∞ norm. So, we have $s^{(k-1)} \in -t\delta\|\nabla f(x^{(k-1)})\|_\infty$. The Frank-Wolfe update requires the subgradient of the ℓ_∞ norm. If $|\nabla f(x^{(k-1)})|$ has its component-wise maximum at i, then the subgradient is the standard basis vector e_i. This can be written as

$$\begin{aligned} i_{k-1} &\in \operatorname*{arg\,max}_{i=1,\cdots,p} |\nabla_i f(x^{(k-1)})| \\ x^{(k)} &= (1-\gamma_k)x^{(k-1)} - \gamma_k t \cdot \operatorname{sign}\big(\nabla_{i_{k-1}} x^{(k-1)})\big) \cdot e_{i_{k-1}} \end{aligned}$$

This looks like greedy coordinate descent since coordinate descent goes through the vector cyclically whereas Frank-Wolfe here picks the largest component at each iteration. Note that this update is simpler than projecting onto a ℓ_1 ball, although they both have the same computational complexity, ie, $\mathcal{O}(n)$.

23.4.2 ℓ_p Regularization

More generally, for the ℓ_p regularized problem

$$\min_x f(x) \quad \text{subject to} \quad \|x\|_p \le t$$

for $1 \le p \le \infty$, we have $s^{(k-1)} \in -t\delta\|\nabla f(x^{(k-1)})\|_q$, where p, q are such that $\|\cdot\|_p$ is the dual of $\|\cdot\|_q$. Recall that this is true if and only if $1/p + 1/q = 1$. Itis interesting to note that the subgradient of a given dual norm can be computed efficiently using the following function, where α is a scaling factor and the rest is the scaled form of the subgradient of max over ℓ_q norm.

$$s_i^{(k-1)} = -\alpha \cdot \operatorname{sign}\big(\nabla f_i(x^{(k-1)})\big) \cdot |\nabla f_i(x^{(k-1)})|^{p/q}, \quad i = 1, \cdots, n$$

where α is such that the constraint is satisfied, ie, $\|s^{(k-1)}\|_q = t$.
This is followed by the main update with a convex combination (γ_k) of $(s^{(k-1)}, x^{(k-1)})$.
Note that these update rules are a lot simpler than projection onto the ℓ_p ball for any general p since there exists no general projection rule. Aside from special cases ($p = 1, 2, \infty$), these projections cannot be computed directly, the projection step must be treated as an optimization.

23.4.3 Trace Norm Regularization

Here we discuss an example of Frank-Wolfe on a matrix valued problem, where you have a much cheaper linear optimization oracle vs a projection (very big difference comparatively). The trace-regularized problem takes the following form,

$$\min_X f(X) \qquad\qquad \text{subject to } \|X\|_{tr} \le t$$

Recall that the trace norm is the sum of the singular values of X. Now, to derive our update S we need to (1) compute the dual norm, (2) find the subgradients of the resultant dual norm. We first note that the dual of the trace norm is the operator norm (largest singular value of X), and obtain the following general update,

$$S^{(k-1)} \in -t\partial \left\|\nabla f(X^{(k-1)})\right\|_{op}$$

We now claim that the update $S^{(k-1)}$ can be explicitly written as, $S^{(k-1)} = -tuv^T$, where u, v are the leading left and right singular vectors of $\nabla f(X^{(k-1)})$ (recall proof from previous lectures). Hence, this means that $\partial \left\|\nabla f(X^{(k-1)})\right\|_{op} = uv^T$. Further, note that we can compute u, v using the power method on $\nabla f(X^{(k-1)})$, which is very cheap if the matrix is sparse.

We next consider, the alternative or projection onto the norm ball. This would require computing the full SVD, which is more complex and expensive than Frank-Wolfe.

23.5 Constrained and Lagrange Norms

Recall that the solution of the constrained problem,

$$\min_x f(x) \qquad \text{subject to } \|x\| \le t$$

are equivalent to those of the Lagrange problem

$$\min_x f(x) + \lambda \|x\|$$

From a stats/ML point of view there is no strong preference for constrained form vs penalized form. Typically, you'd do whichever is cheaper and more convenient, then solve over a range of tuning parameter values (t or λ for the first or second expression respectively) over $[0, \infty]$, and choose best via something like Cross Validation. Furthermore, the two forms are not exactly equivalent. Solving for the best t or λ actually results in different estimators and don't have the same operator characteristics. Equivalency between t and λ is instead data dependent. So if for instance the data is random, then the mapping between t and λ is random.

Further, we should compare the Frank-Wolfe updates under $\|\cdot\|$ to projected gradient or proximal gradient where we use the proximal operator of $\|\cdot\|$ depending on which is easier.

23.5.1 Method Comparisons over Various norms

- ℓ_1 norm: Frank-Wolfe update scans for maximum of gradient; proximal operator soft-thresholds the gradient step; both use O(n) flops
- ℓ_p norm:Frank-Wolfe update raises each entry of gradient to power and sums, in O(n) flops; proximal operator not generally directly computable
- Trace norm: Frank-Wolfe update computes top left and right singular vectors of gradient; proximal operator soft-thresholds the gradient step, requiring a singular value decomposition

Various other constraints yield efficient Frank-Wolfe updates, e.g., special polyhedra or cone constraints, sum-of-norms (group-based) regularization, atomic norms, See Jaggi (2011).

23.6 Example: Lasso comparison

Comparing projected and conditional gradient for the constrained lasso problem, with n=100, p=500: Note that both projected gradient and Frank-Wolfe are both O(n), and both are just as cheap as one another so there is no reason to use our presented version of Frank-Wolfe over projected gradient for this problem.

Further, we notice two things:

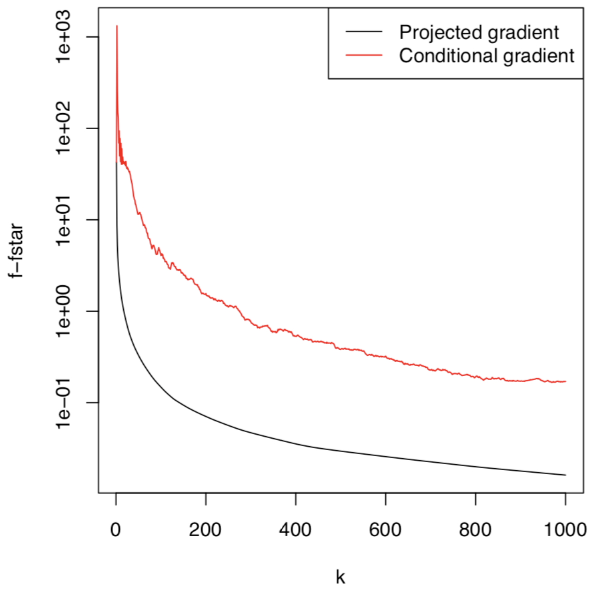

Figure 23.2: Convergence of Projected Gradient (black) vs Frank-Wolfe (red)

- Frank-Wolfe is not a descent estimator as the objective is not monotonically decreasing over each k
- Frank-Wolfe converges slower than projected gradient

Lastly, Frank-Wolfe in this problem uses standard step sizes, and a different step size method such as line search would probably help in terms of convergence.

23.7 Duality Gap

Frank-Wolfe iterations admit a very natural duality gap:

$$f(x^{(k)})^T(x^{(k)} - s^{(k)})$$

Claim: this upper bounds on $f(x^{(k)}) - f^*$
Proof: by the first-order convexity condition,

$$f(s) \geq f(x^{(k)}) + \nabla f(x^{(k)})^T(s - x^{(k)})$$

Minimizing both sides over all $s \in C$ yields,

$$\begin{aligned} f^* &\geq f(x^{(k)}) + \min_{s \in C} \nabla f(x^{(k)})^T(s - x^{(k)}) \\ &= f(x^{(k)}) + \nabla f(x^{(k)})^T(s - x^{(k)}) \end{aligned}$$

Which can then be re-written as,

$$\begin{aligned} f^* \geq f(x^{(k)}) + \nabla f(x^{(k)})^T(s - x^{(k)}) \\ -\nabla f(x^{(k)})^T(s^{(k)} - x^{(k)}) \geq f(x^{(k)}) - f^* \\ \nabla f(x^{(k)})^T(x^{(k)} - s^{(k)}) \geq f(x^{(k)}) - f^* \end{aligned}$$

Hence, we have shown the upper bound.

Why do we call this a "duality gap"? Re-write original problem as

$$min_x f(x) + I_C(x)$$

Where I_C is the indicator function of C. The dual problem is then (by conjugate and dual of indicator function),

$$\max_u -f^*(u) - I_C^*(-u)$$

Where I_C^* is the support function pf C. If we then choose and x, u that are feasible in the primal and dual respectively, then $I_C(x) = 0$ and we obtain the following duality gap,

$$f(x) - (-f^*(u) - I_C^*(-u)) = f(x) + f^*(u) + I_C^*(-u)$$

Then, by Fenchel's inequality,

$$f(x) + f^*(u) + I_C^*(-u) \geq x^T u + I_C^*(-u)$$

Evaluating this at $x = x^{(k)}, u = \nabla f(x^{(k)})$, we obtain:

$$\nabla f(x^{(k)})^T x^{(k)} + \max_{s \in C} -\nabla f(^{(k)})^T s = \nabla f(^{(k)})^T (x^{(k)} - s^{(k)})$$

Which is our duality gap.

23.8 Convergence Analysis

Following Jaggi (2011), define the curvature constant of f over C:

$$M = \max_{\substack{\gamma \in [0,1] \\ x,s,y \in C \\ y = (1-\gamma)x + \gamma s}} \frac{2}{\gamma^2} \big(f(y) - f(x) - \nabla f(x)^T (y - x) \big)$$

Note that $M = 0$ for linear f, big if f is highly curved over the set C, and $f(y) - f(x) - \nabla f(x)^T (y - x)$ is called the Bregman divergence from x to y, defined by f

Theorem 1 *The Frank-Wolfe method using standard step sizes* $\gamma = \frac{2}{k+1}, k = 1, 2, 3...$ *satisfies,*

$$f(x^{(k)}) - f^* \leq \frac{2M}{k+2}$$

Then, the number of iterations needed for $f(x^{(k)}) - f^* \leq \epsilon$ is O$(1/\epsilon)$.

This matches the sublinear rate for projected gradient descent for Lipschitz ∇f, but how do the assumptions compare?

For Lipschitz ∇f with constant L, recall,

$$f(y) - f(x) - \nabla f(x)^T (y - x) \leq \frac{L}{2} \|y - x\|_2^2$$

Maximizing over all $y = (1-\gamma)x + \gamma s$, and multiplying by $2/\gamma^2$,

$$\begin{aligned} M &\le \max_{\substack{\gamma\in[0,1]\\ x,s,y\in C\\ y=(1-\gamma)x+\gamma s}} \frac{2}{\gamma^2}\cdot\frac{L}{2}\|y-x\|_2^2 \\ &= \max_{x,s\in C} L\|x-s\|_2^2 = L\cdot \operatorname{diam}^2(C) \end{aligned}$$

Where $\operatorname{diam}^2(C)$ is the squared diameter of the set C. So, if f has a gradient that is Lipschitz, and C is compact, then it immediately has a curvature that is finite and that is at most $L\cdot\operatorname{diam}^2(C)$.

Hence assuming a bounded curvature is basically no stronger than what we assumed for projected gradient.

23.9 Affine invariance

One of important properties that are not shared with the projected gradient method is affine invariance. For nonsingular matrix A, define $x = Ax'$ and $F(x') = f(x) = f(Ax')$. Consider Frank-Wolfe on F and $x\in C, x = Ax' \iff x' \in A^{-1}C$, we have

$$\begin{aligned} s' &= \arg\min_{z\in A^{-1}C} \nabla F(x')^T z \\ (x')^+ &= (1-\gamma)x' + \gamma s' \end{aligned}$$

By multiplying A on both sides,

$$A(x')^+ = (1-\gamma)Ax' + \gamma As'$$

and then by applying $\nabla F(x') = A^T\nabla f(Ax')$,

$$\begin{aligned} As' &= A\cdot\arg\min_{z\in A^{-1}C}\nabla F(x')^T z \\ &= A\cdot\arg\min_{Az\in C}\nabla f(Ax')^T Az \\ &= AA^{-1}\arg\min_{w\in C}\nabla f(Ax')^T w \\ &= \arg\min_{w\in C}\nabla f(Ax')^T w \end{aligned}$$

which produces the same Frank-Wolfe update as that from f.

Convergence analysis is also affine invariant. The curvature constant M of F:

$$\max_{\substack{\gamma\in[0,1]\\ x',s',y'\in A^{-1}C\\ y'=(1-\gamma x'+\gamma s')}} \frac{2}{\gamma^2}\big(F(y') - F(x') - \nabla F(x')^T(y'-x')\big)$$

matches that of fm because $\nabla F(x')^T(y'-x') = \nabla f(x)^T(y-x)$.

23.10 Inexact updates

Suppose we choose $s^{(k-1)}$ so that

$$\nabla f(x^(k-1))^T s^{(k-1)} \le \min_{s\in C}\nabla f(x^{(k-1)})^T s + \frac{M\gamma_k}{2}\cdot\delta$$

where $\delta \geq 0$ is an inaccuracy parameter. Then we attain the same rate.

Theorem: Frank-Wolfe using step sizes $\gamma_k = \frac{2}{k+1}, k = 1, 2, 3, \dots$, and inaccuracy parameter $\delta \geq 0$, satisfies

$$f(x^(k)) - f^* \leq \frac{2M}{k+1}(1+\delta)$$

23.11 Two variants

There are two important variants of Frank-Wolfe method.

23.11.1 Line search

Instead of using fixed step size $\gamma_k = \frac{2}{k+1}$, we can use

$$\gamma_k = \arg \min_{\gamma \in [0,1]} f\big(x^{(k-1)} + \gamma(s^{(k-1)} - x^{(k-1)})\big)$$

at each $k = 1, 2, 3, \dots$, or we could use backtracking.

23.11.2 Fully corrective

We update x directly according to

$$x^{(k)} = \arg \min_{y} f(y) \quad \text{subject to} \quad y \in conv\{x^{(0)}, s^{(0)}, s^{(k-1)}\}$$

Fully corrective is fairly expensive, but can lead to much faster convergence.

23.12 Path following

Given the norm constrained problem

$$\min_{x} f(x) \quad \text{subject to} \quad ||x|| \leq 1$$

Frank-Wolfe can be used for path following, i.e. we can produce an approximate solution path $\hat{x}(t)$ that is ϵ-suboptimal for every $t \geq 0$. Beginning at $t_0 = 0$ and the only feasible point $x^*(0) = 0$, we fix parameters $\epsilon, m > 0$, then repeat for $k = 1, 2, 3, \dots$
Calculate

$$t_k = t_{k-1} + \frac{(1 - \frac{1}{m})\epsilon}{||\nabla f(\hat{x}(t_{k-1}))||_*}$$

and set $\hat{x}(t) = \hat{x}(t_{k-1})$ for all $t \in (t_{k-1}, t_k)$.

Compute $\hat{x}(t_k)$ by running Frank-Wolfe at $t = t_k$, terminating when the duality gay is $\leq \frac{\epsilon}{m}$.

References

K. Clarkson (2010), ”Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm”

J. Giesen and M. Jaggi and S. Laue, S. (2012), ”Approximating parametrized convex optimization problems”

M. Jaggi (2011), ”Sparse convex optimization methods for machine learning”

M. Jaggi (2011), ”Revisiting Frank-Wolfe: projection-free sparse convex optimization”

M. Frank and P. Wolfe (2011), ”An algorithm for quadratic programming”

R. J. Tibshirani (2015), ”A general framework for fast stagewise algorithms”