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

Lecture 18: October 31

Lecturer: Zico Kolter *Scribes: Arjav Ashesh Desai, Yang Gao*

18.1 Review of Quasi-Newton methods

Consider the following minimization problem:

$$\underset{x}{\text{minimize}} \quad f(x)$$

where f is convex, twice differentiable and $\text{dom}(f) = \mathbb{R}^n$. According to the generic form of the quasi-Newton method,

- start with $x^{(0)} \in \mathbb{R}^n$, and
- repeat $x^{(k)} = x^{(k-1)} - t_k C^{(k-1)} x^{(k-1)}$ for $k = 1, 2, 3, \ldots$

where $C^{(k-1)} \approx (\nabla^2 f(x^{(k-1)}))^{-1}$ i.e. $C^{(k-1)}$ is an approximation to the inverse Hessian at $x^{(k-1)}$. Note that the step sizes are chosen by backtracking (for details, refer to the lecture on quasi-Newton methods). Some commonly used methods for computing the Hessian approximation, C, are:

- SR1: computes the rank 1 update for the Hessian. SR1 uses the Sherman-Morrison-Woodbury formula to compute the inverse Hessian.
- DFP: computes the rank 2 update for the *inverse* Hessian and uses the Sherman-Morrison-Woodbury formula for the Hessian.
- BFGS: reverse roles of the Hessian and the inverse Hessian in DFP.
- LBFGS: a popular, limited memory version of BFGS.

18.2 Review of Proximal Gradient Descent

Proximal gradient descent operators on the problem for the form

$$\underset{x}{\text{minimize}} \quad g(x) + h(x)$$

where g is convex and smooth and h is convex and "simple". For proximal gradient descent, we repeat the following steps:

- Choose an initial $x^{(0)}$.
- $x^{(k)} = \text{prox}_{t_k}(x^{(k-1)} - t_k \nabla g(x^{(k-1)}))$, for $k = 1, 2, 3, \ldots$

where $\text{prox}_t(\cdot)$ is the proximal operator associated with h i.e.

$$\text{prox}_t(x) = \underset{z}{\text{argmin}} \frac{1}{2t} ||x - z||_2^2 + h(z)$$

Assuming the ∇g is computable, the primary difficulty of iterations is in applying the prox, which depends only on h. Proximal gradient descent has the same convergence rate as its fully smooth version. Thus, it is useful when the prox computation is efficient.

Recall that the motivation for proximal gradient descent is that is iteratively minimizes the quadratic expansion in g, plus the original h i.e.

$$\begin{aligned} x^+ &= \underset{z}{\text{argmin}} \frac{1}{2t} ||x - t\nabla g(x) - z||_2^2 + h(z) \\ &= \underset{z}{\text{argmin}} \nabla g(x)^T (z - x) + \frac{1}{2t} ||z - x||_2^2 + h(z) \end{aligned}$$

This update is equivalent to applying the prox operator to the local quadratic approximation of the function where the curvature of the quadratic is given by $\frac{1}{t}I$ (spherical curvature). The fundamental difference between the gradient descent and the Newton's method is that for the latter, the the curvature of the quadratic approximation is given by the local Hessian of the function g i.e use $\nabla^2 g(x)$ instead of $\frac{1}{t}I$. This results in the proximal Newton method which is described in the subsequent section.

18.3 Proximal Newton method

In this section, we describe the proximal Newton method. Proximal Newton method can be described by the following steps - choose an initial starting point $x^{(0)}$ and repeat for $k = 1, 2, 3, \ldots$

- $v^{(k)} = \underset{v}{\text{argmin}}\ \nabla g(x^{(k-1)})^T v + \frac{1}{2} v^T H^{(k-1)} v + h(x^{(k-1)} + v)$.
- $x^{(k)} = x^{(k-1)} + t_k v^{(k)}$

Here, $H^{(k-1)} = \nabla^2 g(x^{(k-1)})$ is the Hessian at $x^{(k-1)}$, and t_k is a step size. An equivalent formulation is given by the following steps - choose an initial starting point $x^{(0)}$ and repeat for $k = 1, 2, 3, \ldots$

- $z^{(k)} = \underset{z}{\text{argmin}} \{\nabla g(x^{(k-1)})^T (z - x^{(k-1)}) + \frac{1}{2}(z - x^{(k-1)})^T H^{(k-1)} (z - x^{(k-1)}) + h(z)\}$
- $x^{(k)} = x^{(k-1)} + t_k (z^{(k)} - x^{(k-1)})$

In this equivalent formulation, we find a point z that minimizes the local quadratic approximation to g and take a small step (t_k is the step size) in the direction of z from $x^{(k-1)}$. The proximal Newton steps can be written in a form which resembles proximal gradient descent using the scaled proximal map which is described in the subsequent subsection.

18.3.1 Scaled Proximal Map

Given $H \succ 0$, we define the scaled proximal map as:

$$\text{prox}_H(x) = \underset{z}{\text{argmin}} \frac{1}{2}||x - z||_H^2 + h(z)$$

where $||x||_H^2 = x^T Hx$.

With $H = \frac{1}{t}I$, we get back the usual, unscaled definition. The scaled prox shares many of the nice properties of the usual prox such as uniqueness and nonexpansiveness.

Now consider

$$\begin{aligned} z^+ &= \underset{z}{\text{argmin}} \nabla g(x)^T(z - x) + \frac{1}{2}(z - x)^T H(z - x) + h(z) \\ &= \underset{z}{\text{argmin}} \frac{1}{2}||x - H^{-1}\nabla g(x) - z||_H^2 + h(z) \end{aligned}$$

Thus, another equivalent form for the proximal Newton update is as follows - choose an initial $x^{(0)}$ and repeat for $k = 1, 2, 3, \ldots$

- $z^{(k)} = \text{prox}_{H^{(k-1)}}(x^{(k-1)} - (H^{(k-1)})^{-1}\nabla g(x^{(k-1)}))$
- $x^{(k)} = x^{(k-1)} + t_k(z^{(k)} - x^{(k-1)})$

This refers to taking a Newton step with respect to g, applying the scaled prox operator with respect to $H^{(k-1)}$, and then taking a small step (t_k is the step size) in that direction.

Some notes:

- When $h(z) = 0$, we get back the usual Newton update.
- If we replaced $H^{(k-1)}$ by $\frac{1}{r_k}I$, and set $t_k = 1$, we get proximal gradient update with step size r_k.
- The difficulty of computing the proximal operator depends strongly on h. However, now it also depends on the structure of the Hessian of g. For example, having a diagonal or banded Hessian generally makes a big difference compared to a dense Hessian.

In the subsequent subsection, we describe the backtracking line search for the proximal Newton method.

18.3.2 Backtracking line search

As with Newton's method in fully smooth problems, pure step sizes $t_k = 1$ for $k = 1, 2, 3, \ldots$ need not converge. The backtracking line search procedure for proximal Newton method is as follows:

- Fix $0 < \alpha \leq \frac{1}{2}$ and $0 < \beta < 1$ and let $v = \text{prox}_H(x - H^{-1}\nabla g(x)) - x$ be the proximal Newton direction at a given iteration.
- Start with $t = 1$.

- **while** $f(x+tv) > f(x) + \alpha t \nabla g(x)^T v + \alpha(h(x+tv) - h(x))$, **we shrink** $t = \beta t$. Note that $f = g + h$.

Note that this scheme is actually of a different spirit that the one we studied for proximal gradient descent, as it avoids recomputing the proximal operator at each inner backtarcking iteration.

18.3.3 Comparison between Proximal gradient and Proximal Newton

In this subsection, we note some differences between proximal gradient descent and proximal Newton method. For a problem $\min_x g(x) + h(x)$.

Proximal Gradient	**Proximal Newton Method**
Iteratively minimize $\|\|b - x\|\|_2^2 + h(z)$	Iteratively minimize $b^T x + x^T A x + h(x)$.
Often closed-form prox	Almost never closed-form prox update
Iterations are cheap	Iterations are very very expensive
Convergence of gradient descent	Convergence of Newton's method

Thus, proximal Newton method is useful when we have a **fast inner optimized for scaled proximal map** (for the quadratic + h) and expect only a few iterations.

18.3.4 Convergence Analysis

Following Lee et al. (2014), assume that:

- $f = g + h$, where g, h are convex and g is twice differentiable.
- $mI \preceq LI$, and $\nabla^2 g$ is Lipshitz with parameter M.
- $\text{prox}_H(\cdot)$ is exactly evaluable.

Theorem 18.1 *Proximal Newton method with backtracking line search converges globally/ Furthermore, for all* $k \geq k_0$*,*

$$||x^{(k)} - x^*||_2 \leq \frac{M}{2m} ||x^{(k-1)} - x^*||_2^2$$

Recall that this is called **local quadratic convergence**. After $k \geq k_0$, to get within $f(x^{(k)}) - f^* \leq \epsilon$, we need $\mathcal{O}(\log\log(\frac{1}{\epsilon}))$ iterations. Note that each iteration uses scaled prox evaluations.

Proof:

1. To prove global convergence, we can show that the at any step, the backtracking exit condition will be satisfied by $t \leq \min\{1, \frac{2m}{L}(1-\alpha)\}$, where t is the step size.
2. This can be used to show that the update direction converges to zero, which can only happen at the global minimum.
3. To prove local quadratic convergence, we can show that for large enough k, the pure step $t = 1$ eventually satisfies backtracking exit condition. Therefore,

$$\begin{aligned}
||x^+ - x^*||_2 &\leq \frac{1}{\sqrt{m}}||x^+ - x^*||_H \\
&= \frac{1}{\sqrt{m}}||\text{prox}_H(x - H^{-1}\nabla g(x)) - \text{prox}_H(x^* - H^{-1}\nabla g(x^*))||_H \\
&\leq \frac{M}{2m}||x^{(k-1)} - x^*||_2^2
\end{aligned}$$

Here, the first inequality results from the lowest eigenvalue bound and the final inequality results from nonexpansiveness, the Lipshitz assumption, and the largest eigenvalue bound.

■

18.3.5 Examples

Two notable examples of proximal Newton methods are:

- **glmnet**: prox Newton for ℓ_1 penalized penalized generalized linear models. The inner problems are solved using coordinate descent.
- **QIUC**: prox Newton for graphical lasso.

At the proper scale, both these methods are considered to be the state-of-the-art. Note that proximal Newton method with use far less evaluations of the gradient of g than proximal gradient. When, these evaluations are expensive, proximal Newton can win. For example, in the lasso logistic regression, proximal Newton does better than FISTA (accelerated prox gradient) and spaRSA (spectral projected gradient method).

18.3.6 Inexact proximal Newton

In proximal Newton, the prox evaluations are always performed inexactly (not so with proximal gradient). Lee (2014) propose a stopping rule for tis inner problem that maintains global convergence and local superlinear convergence.

In the regular Newton method, the inner problem is to minimize $\tilde{g}_{k-1}$ quadratic approximation to g about $x^{(k-1)}$ and stops when

$$||\nabla\tilde{g}_{k-1}(x^{(k)})||_2 \leq \eta_k|| \leq ||\nabla g(x^{(k-1)})||_2$$

where $\tilde{f}_{k-1} = \tilde{g}_{k-1} + h$, and η_k is a "forcing" sequence. Recall that $m \preceq \nabla^2 g \preceq MI$. To get superlinear convergence, the forcing sequence is defined as:

$$\eta_k = \min\left\{\frac{m}{2}, \frac{||G_{\tilde{f}_{k-2}/M}(x^{(k-1)}) - G_{f/M}(x^{(k-1)})||_2}{||G_{f/M}(x^{(k-2)})||_2}\right\}$$

18.3.7 Proximal quasi-Newton

The proximal quasi-Newton methods follow the proximal Newton methods and it takes an approximation of Hessian to take place of the true H, which sometimes make your proximal map easier to solve. It can also help when the Hessian is ill-conditioned: (near) singular.

18.4 Projected Newton Method

18.4.1 Relationship between projected and proximal methods

Proximal gradient descent reduces to projected gradient descent when $h(x) = I_C(x)$. In other words, projected gradient descent is a special case of proximal gradient descent.

$$\begin{aligned} \text{prox}(x) &= \underset{z}{\text{argmin}} ||x - z||_2^2 + h(z) \\ &= \underset{z \in C}{\text{argmin}} \frac{1}{2} ||x - z||_2^2 \end{aligned}$$

However, it doesn't work for Newton Method. The projected Newton method is not a special case of proximal Newton method.

$$z^+ = \underset{z \in C}{\text{argmin}} \nabla g(x)^T (z - x) + \frac{1}{2} (z - x)^T H (z - x) + h(z)$$

It is not a projection onto the set C. It is a minimization of quadratic subject to the constraint of being in that set, which is a hard problem to solve. There are certain types of problems that projected Newton can work but it is still not easy to solve and only apply to certain types of C.

18.4.2 Projected Newton for Box Constraints

Particular, one type of algorithm that Projected Newton method can be made to work is for box constraints (Bertsekas, 1982; Kim et al., 2010; Schmidt et al., 2011). We will use a method called active set method to solve this kind of problem. Given

$$\min_x g(x), l \leq q \leq u \tag{18.1}$$

We will define binding variables. $B^{(k-1)} = i : x_i \leq l_i + \epsilon$. So the value is at the lower bound and the gradients is greater than the zero. Basically, this it will push even further to the lower bound.
Meanwhile we have $i : x_i u_i - \epsilon$ and $\nabla g(x^{(k-1)}) < 0$. These are the variables that are at (close to) boundary, and moving them inward would increase the criterion.
Then we define the free set contains the variables that are not at the boundary: $F_{(k1)}$: complement of the binding set $1, ..., n$ \B$^{(k-1)}$.
Then we define the inverse of the principal submatrix of the Hessian.
along the free variables. Take a Newton step along the free variables only, then projec.
So that the update leaves binding set effectively untouched.

There are many problems that can use projected Newton methods for box constrains such as nonnegative least squares, support vector machine dual, graphical lasso dual and fused lasso (TV denoising) dual. In these problems, we can use projected Newton method.

18.4.3 Convergence for projected Newton for Box constraints

This section is not discussed in the class. According to the slide: Bertsekas (1982) shows that, under appropriate assumptions, projected Newton identifies the proper binding constraints in a finite number of iterations. Then it is just the usual Newtons method on the free variables. Bertsekas (1982) also proves superlinear convergence. And Kim et al. (2010), Schmidt et al. (2011) describe a projected quasi-Newton method, using BFGS-style updates.

References

[1] J. Lee and Y. Sun and M. Saunders, "Proximal Newton-type methods for minimizing composite functions", 2014

[2] D. Bertsekas , "Projected Newton methods for optimization problems with simple constraints", 1982

[3] D. Kim and S. Sra and I. Dhillon , "Tackling box-constrained optimization via a new projected quasi-Newton approach", 2010

[4] M. Schmidt and D. Kim and S. Sra , " Projected Newton-type methods in machine learning", 2011