Note on Optimization Methods

Chunpai Wang

October 2015

1 Quasi-Newton Methods

1.1 Objective Function

Given unconstrained, smooth convex optimization

$$\mathbf{min} f(x) \tag{1}$$

where f is convex, twice continuously differentiable, and $dom(f) = \mathbb{R}^n$.

1.2 Newton's Method

Recall the motivation for gradient descent step at x: we minimize the quadratic approximation

$$f(y) \approx f(x) + \nabla f(x)^T(y-x) + \frac{1}{2t}||y-x||_2^2 \tag{2}$$

over y, and this yields the update $x^+ = x - t\nabla f(x)$

Pure Newton's method uses in a sense a better quadratic approximation:

$$f(y) \approx f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(x)(y-x) \tag{3}$$

and minimizes over y to yield $x^+ = x - (\nabla^2 f(x))^{-1}\nabla f(x)$.

Newton Step (Direction):

$$\triangle x_{nt} = -\nabla^2 f(x)^{-1}\nabla f(x) \tag{4}$$

- $x + \triangle x_{nt}$ minimizes second order approximation
- $\triangle x_{nt}$ is the steepest descent direction at x in local Hessian norm:

$$||u||_{\nabla^2 f(x)} = (u^T\nabla^2 f(x)u)^{\frac{1}{2}} \tag{5}$$

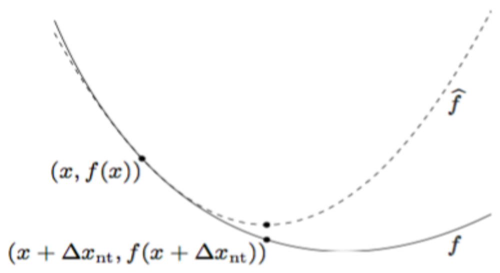

Figure 1: The function f (solid) and its second-order approximation $\hat{f}$ at x(dashed). The Newton step $\triangle x_{nt}$ is what must be added to x to give the minimizer of $\hat{f}$

The Newton step can be used in a line search method when $\nabla^2 f(x)$ is positive definite, for in this case we have (from formula(4))

$$\begin{aligned}
x_{k+1} - x_k = \triangle x_{nt} &= -\nabla^2 f(x)^{-1}\nabla f(x) \\
-\triangle x_{nt}\nabla^2 f(x) &= \nabla f(x) \\
\nabla f(x)\triangle x_{nt} &= -\triangle x_{nt}\nabla^2 f(x)\triangle x_{nt} \\
&\leq 0 \qquad \text{(as } \nabla^2 f(x) \text{ is positive definite)}
\end{aligned}$$

Unless the gradient $\nabla f(x)$ is zero, we have that $\nabla f(x)\triangle x_{nt} \leq 0$ (or $f(x^{(k+1)}) - f(x^{(k)}) \leq 0$), so the Newton step is a descent step.

Newton Method:

$$x^{+} = x - t\nabla^2 f(x)^{-1}\nabla f(x) \tag{6}$$

In pure newton method, t =1, while in practice we use backtracking line search to adaptively choose the step length t.

- advantages: fast convergence
- disadvantages: requires second derivatives, too expensive for large scale applications, and Hessian may be singular

1.3 Variable Metric Methods

Due to the difficulty of calculation on inverse of Hessian $\nabla^2 f(x)$, we update

$$x^{+} = x - tH^{-1}\nabla f(x) \tag{7}$$

with $H \succ 0$, approximation of the $\nabla^2 f(x)$.

- avoid calculation of second derivatives
- simplify computation of search direction ?

Variable Metric Interpretation:

$$\triangle x = -H^{-1}\nabla f(x) \tag{8}$$

is the steepest descent direction at x for quadratic norm (weighted Frobenius norm)

$$||z||_H = (z^T H z)^{\frac{1}{2}} \tag{9}$$

1.4 Quasi-Newton Methods

Quasi-Newton search directions provide an attractive alternative to Newton's method in that they do not require computation of the Hessian and yet still attain a superlinear rate of converangence.

Algorithm 1: Quasi-Newton Methods

```
1 Given starting point $x_0 \in dom(f), H_0 \succ 0$ **repeat**
2     1. compute quasi-Newton direction $\triangle x = -H_k^{-1}\nabla f(x_k)$ 2. determine step size $t$ (e.g. by
      backtracking line search) 3. compute $x_{k+1} = x_k + t\triangle x$ 4. compute $H_{k+1}$
3 **until** $k = 0, 1, 2, ...,$ *until a stopping criterion is satisfied*;
```

- different methods use different rules for updating H in step 4.
- can also propagate H_k^{-1} to simplify calculation of $\triangle x$.
- the updates make use of the fact that changes in the gradient provide information about the second derivative of f along the search direction.

We will discuss several different rules for updating H in step 4, such as DFP, BFGS, and SR1 in following sections.

1.5 Davidon-Fletcher-Powell(DFP) Algorithm

We know that we can use a quadratic function by Tayler theorem to approximate the objective function.

$$f(x_{k+1}) \approx f(x_k) + \nabla f(x_k)(x_{k+1} - x_k) + \frac{1}{2}(x_{k+1} - x_k)^T \nabla^2 f(x_k)(x_{k+1} - x_k) \tag{10}$$

Take the derivative for both sides with respect to x_{k+1}:

$$\nabla f(x_{k+1}) \approx \nabla f(x_k) + \nabla^2 f(x_k)(x_{k+1} - x_k) \tag{11}$$

$$\Rightarrow \nabla^2 f(x_k)(x_{k+1} - x_k) \approx \nabla f(x_{k+1}) - \nabla f(x_k) \tag{12}$$

We choose the new Hessian approximation H_{k+1} (like step 4 in algorithm) so that it mimics the property (formula(12)) of the true Hessian, that is, we use the following condition, known as the secant equation:

$$H_{k+1}s_k = y_k \tag{13}$$

to update the H, where

$$s_k = x_{k+1} - x_k = t\triangle x, \quad y_k = \nabla f(x_{k+1}) - \nabla f(x_k) \tag{14}$$

Why H should be positive definite ?

1. according to formula (4), positive definite of $\nabla^2 f(x)$ ensures newton step is descent step, therefore, the approximation H should be positive definite to make it descent.
2. according to monotonicity of gradient f and secant equation,

$$s_k^T y_k = (x_{k+1} - x_k)^T \nabla f(x_{k+1}) - \nabla f(x_k) > 0 \quad \text{with} \qquad x_k \neq x_{k+1}$$

$$s_k^T H s_k = s_k^T y_k > 0$$

Why H should be symmetric matrix?
Because the Hessian $\nabla^2 f(x)$ is the matrix containing the second derivatives of the function at that point. And the second-derivative is independent of the order in which derivatives are taken. Therefore, for better approximation of $\nabla^f(x)$, we H should be symmetric.

Hessian Matrix Correction

Understanding 1
It turns out there are infinitely many symmetric positive definite matrices H_{k+1} which satisfy the secant equation. To determine H_{k+1} uniquely, we must impose the additional condition that among all symmetric matrices satisfying the secant equation, H_{k+1} is, in some sense, closest to the current matrix H_k.
In other words, we need to solve the matrix minimization problem:

$$\min_H ||H - H_k|| \quad \text{subject to} \;\; H = H^T, \; Hs_k = y_k \tag{15}$$

where $s_k^T y_k > 0$ and H_k is symmetric and positive definite.
Different matrix norm can be used in $||H - H_k||$, and each norm gives a different quasi-Newton method. But, because of the easy computation and scale-invariant optimization, we use the weighted Frobenius norm here,

$$||A||_W = ||W^{1/2} A W^{1/2}||_F = ||C||_F = \sqrt{\sum_i^n \sum_j^n c_{ij}^2} \tag{16}$$

The weight matrix W can be chosen as any matrix satisfying the relation $W y_k = s_k$.With the weighting matrix and this norm, the unique solution is

$$H_{k+1} = (I - \gamma_k y_k s_k^T) H_k (I - \gamma_k s_k y_k^T) + \gamma_k y_k y_k^T, \quad \gamma_k = \frac{1}{y_k^T s_k} \tag{17}$$

Note that, γ_k is a scalar, and $y_k s_k^T, s_k y_k^T, y_k y_k^T$ are rank-1 matrices. The inverse of H_k is useful for the implementation of the method, since it allows the search direction $\triangle x_k$ to be computed using a simple matrix-vector product. We let

$$B_k = H_k^{-1} \tag{18}$$

and use
Sherman-Morrison-Woodbury Formula: If $P \in \mathbb{R}^{n \times n}$ is non-singular and $p, q \in \mathbb{R}^n$, and if

$$P = Q + pq^T \tag{19}$$

then

$$P^{-1} = Q^{-1} - \frac{Q^{-1} p q^T Q^{-1}}{1 + q^T Q^{-1} p} \tag{20}$$

and we end up with

$$B_{k+1} = B_k - \underbrace{\frac{B_k y_k y_k^T B_k}{y_k^T B_k y_k}}_{\text{correction 1}} + \underbrace{\frac{s_k s_k^T}{y_k^T s_k}}_{\text{correction 2}} \tag{21}$$

Both the correction terms are rank-1 matrices, so that B_k undergoes a rank-2 matrix correction in each iteration.

Understanding 2
From formula(21), we find that $B_{k+1} = H_{k+1}^{-1}$ can be approximated by adding a rank-2 matrix to $B_k = H_k^{-1}$. The intuition is, why not use matrix rank-2 correction to update B_{k+1}, and then compute its inverse to get Hessian approximation H_{k+1}. Note, we don't assume the closeness condition between H_{k+1} and H_k here.

First, we want to add a rank-2 matrix on B_k, and we define

$$B_{k+1} = B_k + \triangle B_k \tag{22}$$

where

$$\triangle B_k = \alpha vv^T + \beta uu^T, \text{where } \ v, u \in \mathbb{R}^n, \alpha, \beta \in \mathbb{R} \tag{23}$$

Outer product of vectors vv^T and uu^T give two symmetric rank-1 matrices, of which addition gives a rank-2 matrix correction. Since we are updating B_{k+1} rather than H_{k+1}, the secant equation can be modified to

$$B_{k+1}y_k = s_k \qquad (\text{same to } \ H_{k+1}s_k = y_k) \tag{24}$$

Plug in the formula (17) into secant equation,

$$(B_k + \triangle B_k)y_k = s_k \tag{25}$$

$$\Downarrow$$

$$\begin{aligned}(B_k + \alpha vv^T + \beta uu^T)y_k &= s_k \\ B_k y_k + \alpha vv^T y_k + \beta uu^T y_k &= s_k \\ B_k y_k + v(\alpha v^T y_k) + u(\beta u^T y_k) &= s_k\end{aligned}$$

Since $v, u, s_k, y_k \in \mathbb{R}^n$ and $\alpha, \beta \in \mathbb{R}$, $\alpha v^T y_k$ and $\beta u^T y_k$ are scalars. To satisfy above equation, we let $\alpha v^T y_k = -1$, and $\beta u^T y_k = 1$, that is

$$B_k y_k - v + u = s_k \tag{26}$$

And let $v = B_k y_k$ and $u = s_k$ to satisfy above equation. We get

$$\alpha = -\frac{1}{y_k^T B_k^T y_k} = -\frac{1}{y_k^T B_k y_k}, \quad \beta = \frac{1}{s_k^T y_k}, \quad v = B_k y_k, \quad u = s_k \tag{27}$$

then plug into formula (23),

$$\triangle B_k = -\frac{B_k y_k y_k^T B_k}{y_k^T B_k y_K} + \frac{s_k s_k^T}{s_k^T y_k} \tag{28}$$

$$B_{k+1} = B_k - \frac{B_k y_k y_k^T B_k}{y_k^T B_k y_k} + \frac{s_k s_k^T}{s_k^T y_k} \tag{29}$$

1.6 Broyden-Fletcher-Goldfarb-Shanno(BFGS) Algorithm

Understanding 1:
BFGS updating imposes conditions directly on the B_k(inverse of H_k) instead of imposing conditions on the Hessian approximation H_k. To do like this, we can fasten the algorithm by skipping a step to compute the inverse of Hessian approximation in each iteration.
There are several similar constraints on B_k:

1. B_k should be symmetric and positive definite
2. should satisfy the secant equation

$$B_{k+1}y_k = s_k \tag{30}$$

The condition of closeness to B_k is now specified by the following:

$$\min_B ||B - B_k|| \quad \text{subject to} \;\; B = B^T,\; By_k = s_k \tag{31}$$

With the weighed Frobenius norm and any weight matrices $W s.t. W s_k = y_k$, the unique solution B_{k+1} is

$$B_{k+1} = (I - \delta_k s_k y_k^T) B_k (I - \delta_k y_k s_k^T) + \delta_k s_k s_k^T, \quad \delta_k = \frac{1}{y_k s_k} \tag{32}$$

which translated back to the Hessian approximation H_{k+1} yields

$$H_{k+1} = H_k - \frac{H_k s_k s_k^T H_k}{s_k^T H_k s_k} + \frac{y_k y_k^T}{y_k^T s_k} \tag{33}$$

Understanding 2:
Here, we do not assume the closeness condition on B_k. According to formula(33), we want to make a small tweak on H_k, such that

$$H_{k+1} = H_k + \triangle H_k \tag{34}$$

where

$$\triangle H_k = \alpha v v^T + \beta u u^T, \text{where} \;\; v, u \in \mathbb{R}^n, \alpha, \beta \in \mathbb{R} \tag{35}$$

Outer product of vectors vv^T and uu^T give two symmetric rank-1 matrices, of which addition gives a rank-2 matrix correction. By adding a rank-2 matrix, it seems actually assume that the closeness between H_{k+1} and H_k, which reflect the real closeness of true Hessian $\nabla^2 f(x_k)$

Plug in the formula (35) into secant equation (formula(13)),

$$(H_k + \triangle H_k) s_k = y_k \tag{36}$$

$$\Downarrow$$

$$\begin{aligned} (H_k + \alpha v v^T + \beta u u^T) s_k &= y_k \\ H_k s_k + \alpha v v^T s_k + \beta u u^T s_k &= y_k \\ H_k s_k + v(\alpha v^T s_k) + u(\beta u^T s_k) &= y_k \end{aligned}$$

Since $v, u, s_k \in \mathbb{R}^n$ and $\alpha, \beta \in \mathbb{R}$, $\alpha v^T s_k$ and $\beta u^T s_k$ are scalars. To satisfy above equation, we let $\alpha v^T s_k = -1$, and $\beta u^T s_k = 1$, that is

$$H_k s_k - v + u = y_k \tag{37}$$

And let $v = H_k s_k$ and $u = y_k$ to satisfy above equation. We get

$$\alpha = -\frac{1}{s_k^T H_k^T s_k} = -\frac{1}{s_k^T H_k s_k}, \quad \beta = \frac{1}{y_k^T s_k}, \quad v = H_k s_k, \quad u = y_k \tag{38}$$

then plug into formula (35),

$$\triangle H_k = -\frac{H_k s_k s_k^T H_k}{s_k^T H_k s_K} + \frac{y_k y_k^T}{y_k^T s_k} \tag{39}$$

$$H_{k+1} = H_k - \frac{H_k s_k s_k^T H_k}{s_k^T H_k s_k} + \frac{y_k y_k^T}{y_k^T s_k} \tag{40}$$

1.7 Comparison Between DFP and BFGS

Same: Both keep positive definite during updating.
Why BFGS is duall of DFP ?
Why BFGS is better than DFP ?
What is the self-correcting properties for BFGS ?
Can we use SVD in the method ?

1.8 Limited Memory BFGS (L-BFGS)

BFGS is a very effective optimization algorithm that does not require computing the exact Hessian, or finding any matrix inverses. However, it is not possible to use BFGS on problems with high dimensional variable, $dom(f) \in R^n, n$ is very large (say millions), because in that case it is impossible to store or manipulate the approximiate inverse Hessian H, which is of size n^2.
L-BFGS solves this problem by storing the approximate Hessian in a compressed form that requires storing only a constant multiple of vectors n. In particular, L-BFGS only remembers updates from the last m iterations, so information about iterates before that is lost. Furthermore, the search direction can be computed in a number of operation that is also linear in $nandm$.

- instead we store the m (e.g., m = 30) most recent values of

$$s_j = x_{j+1} - x(j), \quad y_j = \nabla f(x_{j+1}) - \nabla f(x_j) \tag{41}$$

- we evaluate $\triangle x = H_k^{-1} \nabla f(x_k)$ recursively, using

$$H_j^{-1} = \Big(I - \frac{s_j y_j^T}{y_j^T s_j}\Big) H_{j-1}^{-1} \Big(I - \frac{y_j s_j^T}{y_j^T s_j}\Big) + \frac{s_j s_j^T}{y_j^T s_j} \tag{42}$$

  for $j = k, k-1, ..., k-m+1$, assuming, for example, $H_{k-m}^{-1} = I$

- time cost per iteration is $O(nm)$, space cost is $O(nm)$

1.9 SR1

Why not use a rank-1 correction rather than a rank-2 correction on DFP and BFGS is because by adding a rank-1 matrix to H_k will not always derive a positive definite H_{k+1}. SR1 algorithm does not guarantee the update matrix to maintain positive definiteness.

1.10 The Broyden's Method

The Broyden's Method does not require the update matrix to be symmetric and it is used to find the root of a general system of equations (rather than the greadient) by updating the Jacobian (rather than the Hessian).