TMA4180 Optimization: Quasi-Newton Methods

Elisabeth Köbis

NTNU, TMA4180, Optimering, Spring 2021

March 22nd, 2021

Remember to register your visit in S5!

Plan for Today

1. Recall Newton Method
2. Quasi-Newton Method
   - SR1-Method
   - DFP-Method
   - BFGS-Method

Today's lecture covers Chapters 6.1, 6.2 and 6.4 in N&W.

Unconstrained Programming

Now: Optimization without any constraints:

$$\boxed{\min_{x \in \mathbb{R}^n} f(x).}$$

Review: Gradient Descent

Gradient Descent

- iterative method for finding a local minimum of a real-valued, differentiable objective function $f(x)$

To find a local minimum, we start off at an initial point and iteratively take steps in the direction of the negative gradient of the function f at the current point. Since the gradient of a function points in the direction of steepest ascent, the negative gradient points in the direction of steepest descent, and thus at each step of gradient descent, we are moving in the direction where $f(x)$ decreases the fastest. An iteration of gradient descent is written as

$$x_{k+1} = x_k - \alpha \nabla f(x_k),$$

where $\alpha > 0$ is the step size. Too large of a step size and our model may diverge; if α is chosen too small, our model becomes highly inefficient, taking unnecessarily long to converge to the minimum.

Review: Gradient Descent

- gradient descent is a first-order optimization method, as it uses first-order information (ie. the gradient) to find the minimum. While this often reliably gets the job done, its main disadvantage lies in the fact that it is quite inefficient, even for a suitably chosen learning rate. By approximating our objective function linearly at each point, we are only working with very limited local information at each iteration, and we thus have to be cautious and restrain ourselves to small step sizes at each iteration. A natural next step would be to look at the second-order behavior of the objective function.

Review: Newton's Method

Let us consider the one-domensional case: Consulting Taylor, we know that the second order approximation of f at a point $x_k + \alpha$ is

$$f(x_k + p) = f(x_k) + f'(x_k)p + \frac{1}{2}f''(x_k)p^2.$$

Minimizing the Taylor approximation yields

$$\frac{d}{dp}[f(x_k+p) = f(x_k)+f'(x_k)p+\frac{1}{2}f''(x_k)p^2] = f'(x_k)+f''(x_k)+p = 0.$$

This yields the direction

$$p = -\frac{f'(x_k)}{f''(x_k)}.$$

So the new iteration scheme becomes (step size $\alpha = 1$)

$$\boxed{x_{k+1} = x_k - \frac{1}{f''(x_k)}f'(x_k)}.$$

Review: Newton's Method

Generalizing to n dimensions, we obtain

$$\boxed{x_{k+1} = x_k - (\nabla^2 f(x_k))^{-1} \nabla f(x_k)},$$

where $\nabla^2 f(x_k)$ is the *Hessian* of f at x_k.

Review: Newton's Method: Drawbacks

- sensitive to initial conditions: Unlike gradient descent, which ensures that we are always going downhill by always going in the direction opposite the gradient, Newton's method fits a paraboloid to the local curvature and proceeds to move to the stationary point of that paraboloid. Depending on the local behavior of our initial point, Newton's method could take us to a maximum or a saddle point instead of a minimum. We thus see that Newton's method is really only appropriate for minimizing convex objective functions.
- increased convergence rate comes with the cost of additional computation time

Newton's Method $\longrightarrow$ Quasi-Newton Method

Newton's methods makes the following update at each iteration:

$$\boxed{x_{k+1} = x_k - (\nabla^2 f(x_k))^{-1} \nabla f(x_k)},$$

where the Hessian is computed and inverted at each step. Now: Instead of computing the actual Hessian, we just **approximate** it with a positive definite matrix B, which is updated from iteration to iteration using information computed from previous steps. We immediately see that this scheme would yield a much less costly algorithm compared to Newton's method, because instead of computing a large amount of new quantities at each iteration, we're largely making use of previously computed quantities.

$$\longrightarrow \boxed{x_{k+1} = x_k - B^{-1} \nabla f(x_k)}.$$

Quasi-Newton Method

The first quasi-Newton algorithm was developed by **W.C. Davidon** (1927–2013, American professor of physics and mathematics) in the mid 1950s, but was published in 1991.

Quasi-Newton Method

SIAM J. OPTIMIZATION
Vol. 1, No. 1, pp. 1–17, February 1991

001

VARIABLE METRIC METHOD FOR MINIMIZATION*

WILLIAM C. DAVIDON†

Abstract. This is a method for determining numerically local minima of differentiable functions of several variables. In the process of locating each minimum, a matrix which characterizes the behavior of the function about the minimum is determined. For a region in which the function depends quadratically on the variables, no more than N iterations are required, where N is the number of variables. By suitable choice of starting values, and without modification of the procedure, linear constraints can be imposed upon the variables.

Key words. variable metric algorithms, quasi-Newton, optimization

AMS(MOS) subject classifications. primary, 65K10; secondary, 49D37, 65K05, 90C30

Quasi-Newton Method

$$\boxed{x_{k+1} = x_k - B^{-1}\nabla f(x_k)}.$$

Quasi-Newton methods (like steepest descent, unlike Newton method) require only the gradient of the objective function to be supplied at each iterate. By measuring the changes in gradients, they construct a model of the objective function that is good enough to produce superlinear convergence. Since second derivatives are not required, quasi-Newton methods are sometimes more efficient than Newton's method.

Quasi-Newton Method

Consider the following quadratic model of the objective function at the current iterate x_k, where B_k is an $(n \times n)$-**symmetric positive definite** matrix:

$$m_k(p) = f_k + \nabla f_k^T p + \frac{1}{2} p^T B_k p.$$

This model has the properties:

$$m_k(0) = f_k$$
$$\nabla m_k(0) = \nabla f_k.$$

Searching for the minimizer of m_k yields

$$\nabla m_k = \nabla f_k + p^T B_k = 0 \quad \longrightarrow \quad p_k = -B_k^{-1} \nabla f_k.$$

This minimizer p_k is used as the new search direction, and the new iterate is

$$x_{k+1} = x_k + \alpha p_k,$$

where the step length α_k is chosen to satisfy the Wolfe conditions. This iteration is quite similar to the line search Newton method; the key difference is that the approximate Hessian B_k is used in place of the true Hessian.

Quasi-Newton Method

Instead of computing B_k afresh at every iteration, Davidon proposed to update it in a simple manner to account for the curvature measured during the most recent step. Suppose that we have generated a new iterate x_{k+1} and wish to construct a new quadratic model, **an approximate quadratic model of** f of the form

$$m_{k+1}(p) = f_{k+1} + \nabla f_{k+1}^T p + \frac{1}{2} p^T B_{k+1} p.$$

What requirements should we impose on B_{k+1}, based on the knowledge gained during the latest step?

Quasi-Newton Method: Property of B

Reasonable requirement: The gradient of m_{k+1} should match the gradient of the objective function f at the latest two iterates x_k and x_{k+1}.

- $\nabla m_{k+1}(0) = \nabla f(x_{k+1})$ is fulfilled.
- $\nabla m_{k+1}(-\alpha_k p_k) = \nabla f(x_k)$:

$$\nabla m_{k+1}(\underbrace{x_k - x_{k+1}}_{=-\alpha_k p_k}) = \nabla f_{k+1} - \alpha_k B_{k+1} p_k = \nabla f_k,$$

which yields

$$B_{k+1}\alpha_k p_k = \nabla f_{k+1} - \nabla f_k.$$

Defining $s_k := x_{k+1} - x_k = \alpha_k p_k$ and $y_k = \nabla f_{k+1} - \nabla f_k$, we get

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

(1) is known as *quasi-Newton condition* or *secant equation*.

Quasi-Newton Method: Property of B

Quasi-Newton Condition / Secant Equation

$$\boxed{B_{k+1}(x_{k+1} - x_k) = \nabla f(x_{k+1}) - \nabla f(x_k)}$$

$\longrightarrow$ all quasi-Newton methods must share this condition.

Quasi-Newton Method: Property of B

As B_{k+1} is a symmetric positive definite matrix, we have from (1) that

$$\boxed{s_k^T B_{k+1} s_k = s_k^T y_k > 0.} \tag{2}$$

(2) is called *curvature condition*.

Quasi-Newton Method: Property of B

If f is strongly convex, then (2) holds for any vector x_k and x_{k+1}.

Proof: A function $f(x)$ is strongly convex if all eigenvalues of $\nabla^2 f(x)$ are positive and bounded away from zero. This implies that there exists $\sigma > 0$ such that

$$p^T \nabla^2 f(x) p \geq \sigma ||p||^2 \quad \text{for any } p. \tag{3}$$

By Taylor's theorem, if $x_{k+1} = x_k + \alpha_k p_k$, we have

$$\nabla f(x_{k+1}) = \nabla f(x_k) + \int_0^1 [\nabla^2 f(x_k + z\alpha_k p_k)\alpha_k p_k] dz.$$

We get from (3) that

$$\begin{aligned} \underbrace{\alpha_k p_k^T}_{=s_k} y_k &= \alpha_k p_k^T [\nabla f(x_{x+1}) - \nabla f(x_k)] \\ &= \alpha_k^2 \int_0^1 [p_k^T \nabla^2 f(x_k + z\alpha_k p_k) p_k] dz \\ &\geq \sigma ||p_k||^2 \alpha_k^2 > 0. \quad \square \end{aligned}$$

Quasi-Newton Method: Property of B

(2)[1] will not always hold for nonconvex functions, and in this case we need to enforce (2) explicitly, by imposing restrictions on the line search procedure that chooses the step length α. In fact, the condition (2) is guaranteed to hold if we impose the Wolfe or strong Wolfe conditions on the line search. Recall the Wolfe conditions:

$$\begin{aligned} f(x_k + \alpha_k p_k) &\leq f(x_k) + c_1 \alpha_k \nabla f_k^T p_k, \\ \nabla f(x_k + \alpha_k p_k)^T \underbrace{p_k}_{=s_k/\alpha_k} &\geq c_2 \nabla f_k^T p_k, \end{aligned} \tag{4}$$

and notice that (4) implies $\nabla f_{k+1}^T s_k \geq c_2 \nabla f_k^T s_k$ and thus

$$y_k^T s_k \geq (c_2 - 1)\alpha_k \nabla f_k^T p_k \overset{p_k \text{ descent direction}}{>} 0,$$

and so the curvature condition (2) is satisfied.

[1] $s_k^T B_{k+1} s_k = s_k^T y_k > 0$

Symmetric Rank One Updates (SR1): Dyadic Product of Two Vectors

Recall that

$$ab^T := \begin{pmatrix} a_1b_1 & a_1b_2 & \dots & a_1b_n \\ a_2b_1 & a_2b_2 & \dots & a_2b_n \\ \vdots & \vdots & \vdots & \vdots \\ & & \vdots & \\ a_nb_1 & a_nb_2 & \vdots & a_nb_n \end{pmatrix}.$$

Note that the rank of $ab^T = 1$ if $a \neq 0$ and $b \neq 0$.

Symmetric Rank One Updates (SR1)

Define

$$B_{k+1} = B_k + \sigma \cdot \underbrace{vv^T}_{\text{dyadic product}},$$

where $\sigma \in \{\pm 1\}$ and $v \in \mathbb{R}^n$ are chosen s.t. the secant equation is satisfied. We obtain:

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k} \tag{5}$$

(5) is a generalization of the secant method for a multidimensional problem.

Algorithm

- Initialize B_0, x_0, $k = 0$
- Compute $p_k = -B_k^{-1} \nabla f(x_k)$; Find step length $\alpha_k > 0$ and set $x_{k+1} = x_k + \alpha_k p_k$
- Compute B_{k+1} according to (5); set $k := k + 1$ and repeat

Symmetric Rank One Updates (SR1)

Summary of content covered so far

Symmetric Rank One Updates (SR1): Convergence

For general nonlinear functions, the SR1 update generates good Hessian approximations under certain conditions.

Theorem

Suppose that f is twice continuously differentiable, and that its Hessian is bounded and Lipschitz continuous in a neighborhood of a point x^*. Let $\{x_k\}$ be any sequence of iterates such that $x_k \to x^*$. Suppose in addition that the inequality

$$|s_k^T(y_k - B_k s_k)| \geq r||s_k|| \cdot ||y_k - B_k s_k||$$

holds for all k, for some $r \in (0, 1)$, and that the steps s_k are uniformly linearly independent[2]. Then the matrices B_k generated by the SR1 updating formula satisfy

$$\lim_{k \to \infty} ||B_k - \nabla^2 f(x^*)|| = 0.$$

[2] The term "uniformly linearly independent steps"means, roughly speaking, that the steps do not tend to fall in a subspace of dimension less than n. This assumption is usually, but not always, satisfied in practice

Symmetric Rank One Updates (SR1) of the Inverse

Instead of working with the matrices B_k, work with their inverses $H_k := B_k^{-1}$ instead. We obtain:

$$H_{k+1} = H_k + \frac{(s_k - H_k y_k)(s_k - H_k y_k)^T}{(s_k - H_k y_k)^T y_k} \tag{6}$$

Algorithm

- Initialize H_0, x_0, $k = 0$
- Compute $p_k = -H_k \nabla f(x_k)$; Find step length $\alpha_k > 0$ and set $x_{k+1} = x_k + \alpha_k p_k$
- Compute H_{k+1} according to (6); set $k := k + 1$ and repeat

Symmetric Rank One Updates (SR1) of the Inverse: Convergence

For quadratic functions, the choice of step length does not affect the update, therefore we consider now:

$$p_k = -H_k \nabla f_k, \quad x_{k+1} = x_k + p_k.$$

Theorem (Theorem 6.1 in N&W)

Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is the strongly convex quadratic function $f(x) = b^T x + \frac{1}{2} x^T A x$, where A is symmetric positive definite. Then for any starting point x_0 and any symmetric starting matrix H_0, the iterates $\{x_k\}$ generated by the SR1 method converge to the minimizer in at most n steps, provided that $(s_k - H_k y_k)^T y_k \neq 0$ for all k. Moreover, if n steps are performed, and if the search directions p_i are linearly independent, then $H_n = A^{-1}$.

Proof:

Symmetric Rank One Updates (SR1) of the Inverse: Convergence

Theorem (Theorem 6.7 in N&W)

Suppose that the following conditions hold:

1. The sequence of iterates does not terminate, but remains in a closed, bounded, convex set D, on which the function f is twice continuously differentiable, and in which f has a unique stationary point x^*;
2. the Hessian $\nabla^2 f(x^*)$ is positive definite, and $\nabla^2 f(x)$ is Lipschitz continuous in a neighborhood of x^*;
3. the sequence of matrices $\{B_k\}$ is bounded in norm;
4. condition

$$|s_k^T(y_k - B_k s_k)| > r||s_k|| \cdot ||y_k - B_k s_k||$$

holds at every iteration, where r is some constant in $(0, 1)$.

Then $\lim_{k\to\infty} x_k = x^*$, and we have that

$$\lim_{k\to\infty} \frac{||x_{k+n+1} - x^*||}{||x_k - x^*||} = 0.$$

Properties of SR1 Updates

- Instabilities if $(s_k - H_k y_k)^T y_k \approx 0 \Longrightarrow$ Do not update H in this case (set $H_{k+1} = H_k$).
- Cannot guarantee that H_k remains positive definite and that p_k is a descent direction $\Longrightarrow$ Modified updates, e.g. self-scaling SR1 (SSR1). One can also use SR1 updates together with trust-region methods to solve this issue.
- If the updates are fine, the matrix B_k tends to be a good approximation of the Hessian of f. Typically, this leads to superlinear convergence of the iteration.

Davidon-Fletcher-Powell method (DFP)

When the curvature condition is satisfied, the secant equation always has a solution B_{k+1}. In fact, there are infinitely many solutions $\longrightarrow$ goal: Determine B_{k+1} uniquely. Additional condition: Among all symmetric matrices satisfying the secant equation, B_{k+1} is, in some sense, closest to the current matrix B_k:

$$\begin{aligned} &\min_{B} ||B - B_k|| \\ \text{subject to } &\underbrace{B = B^T}_{\text{symm.}}, \quad Bs_k = y_k, \end{aligned} \tag{7}$$

where s_k and y_k satisfy the curvature condition (2) and B_k is symmetric and positive definite. Different matrix norms can be used (7), and each norm gives rise to a different quasi-Newton method

Davidon-Fletcher-Powell method (DFP)

Weighted Frobenius norm:

$$||A||_W := ||W^{1/2} A W^{1/2}||_F$$

with$||\cdot||_F$ defined by $||C||_F = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij}^2$. The weight matrix W can be chosen as *any* matrix satisfying the relation $W y_k = s_k$. Here, we use the following W:

$$W = \bar{G}_k^{-1}, \quad \bar{G}_k = \underbrace{\int_0^1 \nabla^2 f(x_k + \tau \alpha_k p_k) d\tau}_{\text{average Hessian}}$$

It can be shown by means of Taylor's Theorem that

$$y_k = \bar{G}_k \alpha_k p_k = \bar{G}_k s_k.$$

The unique solution of B is

$$\underbrace{B_{k+1} = (1 - \rho_k y_k s_k^T) B_k (1 - \rho_k s_k y_k^T) + \rho_k y_k y_k^T,}_{\text{DFP updating formula: Davidon, Fletcher, Powell}} \tag{8}$$

with $\rho_k = \frac{1}{y_k^T s_k}$.

Davidon-Fletcher-Powell method (DFP)

In the quasi-Newton method, we are actually not computing B_k itself, but the inverse B_k^{-1}. Therefore, let $H_k = B_k^{-1}$. Then (8) becomes

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

BFGS-Method

- The most popular quasi-Newton algorithm is the BFGS method, named for its discoverers Charles George Broyden, Roger Fletcher, Donald Goldfarb and David Shanno.

BFGS Method

Instead of imposing conditions on the Hessian approximations B_k, we impose similar conditions on their inverses H_k. The updated approximation H_{k+1} must be symmetric and positive definite, and must satisfy the secant equation $H_{k+1}y_k = s_k$.

$$\begin{aligned} &\min_H ||H - H_k|| \\ &\text{subject to } \underbrace{H = H^T}_{\text{symm.}}, \quad Hy_k = s_k, \end{aligned} \tag{9}$$

where s_k and y_k satisfy the curvature condition and H_k is symmetric and positive definite.
BFGS update:

$$H_{k+1} = (I - \rho_k s_k y_k^T) H_k (I - \rho_k s_k y_k^T) + \rho_k s_k s_k^T \tag{10}$$

with $\rho_k = \frac{1}{y_k^T s_k}$ (as before).

BFGS Method

Algorithm: BFGS Method

Input: starting points x_0^s, convergence tolerance $\epsilon > 0$ and inverse Hessian approximation H_0

$k := 0$;

while $||\nabla f_k|| > \epsilon$

Compute search direction

$$p_k = -H_k \nabla f_k;$$

Set $x_{k+1} = x_k + \alpha_k p_k$, where α_k is computed from a line search procedure to satisfy the Wolfe conditions

Define $s_k = x_{k+1} - x_k$ and $y_k = \nabla f_{k+1} - \nabla f_k$;

Compute H_{k+1} by means of (10)

$k := k + 1$;

end while