In blau Allgemeines und Formulierungen

Technical University of Chemnitz

Faculty of Mathematics

BFGS-Method On Riemannian Manifolds

Mathematical Optimization

Hier könnte auch dein Name hin,
Was soll die Zeile sonst ausdrücken?

for obtaining the academic degree
Bachelor of Science

Author: Tom-Christian Riemer
MatNr. 444019

Version from: 25th March 2020

1. Supervisor: PD Dr. Ronny Bergmann
2. Supervisor: Prof. Dr. Roland Herzog

Contents

1 The BFGS-Method For The Euclidean Case

1.1 Preliminary

Hier hast du bisher Themen gesammelt, dir du als Grundlagen/ Preliminaries einführen solltest?

Wolfe conditions:

$$f(x_k + \alpha_k d_k) \leq f(x_k) + c_1 \alpha_k \nabla f(x_k)^{\mathrm{T}} d_k \tag{1.1}$$

$$f(x_k + \alpha_k d_k)^{\mathrm{T}} d_k \geq c_2 \nabla f(x_k)^{\mathrm{T}} d_k \tag{1.2}$$

strong Wolfe conditions:

$$f(x_k + \alpha_k d_k) \leq f(x_k) + c_1 \alpha_k \nabla f(x_k)^{\mathrm{T}} d_k \tag{1.3}$$

$$|f(x_k + \alpha_k d_k)^{\mathrm{T}} d_k| \geq c_2 |\nabla f(x_k)^{\mathrm{T}} d_k| \tag{1.4}$$

Lemma 1. *Suppose that $f\colon \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable. Let d_k be a descent direction at x_k, and assume that f is bounded below along the ray $\{x_k + \alpha d_k | \alpha > 0\}$. Then if $0 < c_1 < c_2 < 1$, there exist intervals of step lengths satisfying the Wolfe conditions and the strong Wolfe conditions.*

[2]

Theorem 1. *Suppose that $f\colon \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable. Consider the iteration $x_{k+1} = x_k + \alpha_k d_k$, where d_k is a descent direction and α_k satisfies the Wolfe conditions with $c_1 \leq \frac{1}{2}$. If the sequence $\{x_k\}_k$ converges to a point x^* such that $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite, and if the search direction satisfies*

$$\lim_{n \to \infty} \frac{\|\nabla f(x_k) + \nabla^2 f(x_k) d_k\|}{\|d_k\|} = 0 \tag{1.5}$$

then

1. *the step length $\alpha_k = 1$ is admissible for all k greater than a certain index k_0 and*
2. *if $\alpha_k = 1$ for all $k > k_0$, $\{x_k\}_k$ converges to x^* superlinearly.*

[2]

Corollary 1. *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable, $\{H_k\}_k$ a sequence of regular matrices in $\mathbb{R}^{n\times n}$, $x_0 \in \mathbb{R}^n$ and $\{x_k\}_k \subseteq \mathbb{R}^n$ a sequence definded by*

$$x_{k+1} = x_k - H_k^{-1}\nabla f(x_k),\ k = 0,\ 1,\ \cdots$$

with the limit $\lim_{n\to\infty} x_k = x^*$, $x_k \neq x^*$ for all $k \in \mathbb{N}$ and $\nabla^2 f(x^*)$ regular. Then the following statements are equivalent

1. *$\{x_k\}_k \to x^*$ superlinear and $\nabla f(x^*) = 0$.*
2. *$\|(\nabla^2 f(x_k) - H_k)(x_{k+1} - x_k)\| = o(\|x_{k+1} - x_k\|)$*
3. *$\|(\nabla^2 f(x^*) - H_k)(x_{k+1} - x_k)\| = o(\|x_{k+1} - x_k\|)$*

[1]

These conditions are also called Dennis-Moré conditions. They show that for superlinear convergence it is only important that $\nabla^2 f(x_k)(x_{k+1} - x_k)$ and $H_k(x_{k+1} - x_k)$ match sufficiently well. It is therefore not necessary that H_k approximates the entire Hessian matrix $\nabla^2 f(x_k)$ well.

Theorem 1. *Let $A \in \mathbb{R}^{n\times n}$ be nonsingular and $u, v \in \mathbb{R}^n$ be arbitrary. If*

$$1 + v^{\mathrm{T}} A^{-1} u \neq 0,$$

then the rank-one update $A + uv^{\mathrm{T}}$ of A is nonsingular, and its inverse is represented by

$$(A + uv^{\mathrm{T}})^{-1} = A^{-1} - \frac{A^{-1}uv^{\mathrm{T}}A^{-1}}{1 + v^{\mathrm{T}}A^{-1}u}.$$

[3]

Sherman-Morrison-Woodburg Theorem 1. *Let $A \in \mathbb{K}^{n\times n}$ be a nonsingular matrix, $U, V \in \mathbb{K}^{n\times m}$. If $I + V^*A^{-1}U$ is invertible, then $A + UV^*$ is invertible and*

$$(A + UV^*)^{-1} = A^{-1} - A^{-1}U(I + V^*A^{-1}U)^{-1}V^*A^{-1}.$$

[3]

1.2 Quasi-Newton Methods

Quasi-Newton methods are a class of numerical methods for solving nonlinear minimization problems. As the name suggests, these are based on the Newton method, but attempt to minimize the computational effort. The class goes back to the physicist William Davidon of the Argonne National Laboratory, who developed the first algorithm in the mid 1950s.

For the
In the ordinary Newton method, both the gradient and the Hessian are newly calculated in every the current iteration. Of course, we get useful information about curvature of our function from the Hessian, get local at least superlinear convergence and if we add a method for determining step sizes, we even get global convergence. But there are arguments against the ordinary Newton method, mainly related to the calculation of the Hessian. For example the calculation could be too costly or not possible at all (which includes the case that the Hessian does not exist). Quasi-Newton methods follow the strategy of not calculating and instead approximating it. Henceforth we call the approximation of the Hessian matrix $\nabla^2 f(x_k)$ used in each iteration H_k.
Muss noch eingeführt werden.
What do we expect from this sequence $\{H_k\}_k$ now? The sequence should posses positive defiteness, $d_k = -H_k^{-1}\nabla f(x_k)$ should be a descent direction and the resulting method should behave like the usual Newton method. Of course, the calculation should cost less.
Das ist aber sehr informell, bzgl Konvergenz?
Let $f\colon D \to \mathbb{R}$ be twice continously differentiable on an open subset $D \subset \mathbb{R}^n$. We consider the quadratic Taylor-approximation of f at x_{k+1}:

Hast du in der Intro einen Abschnitt zum allgemeinen Iterationsverfahren und Newton? Sonst ist das mit k+1 schwierig zu verstehen an dieser Stelle.

$$f(x) \approx m_{k+1}(x) = f(x_{k+1}) + g_{k+1}^{\mathrm{T}}(x - x_{k+1}) + \frac{1}{2}(x - x_{k+1})^{\mathrm{T}} G_{k+1}(x - x_{k+1})$$

where $g_{k+1} \triangleq \nabla f(x_{k+1})$ and $G_{k+1} \triangleq \nabla^2 f(x_{k+1})$. For the gradient we obtain With the derivative we get

$$\nabla f(x) \approx \nabla m_{k+1}(x) = g_{k+1} + G_{k+1}(x - x_{k+1})$$

and we want
Wieso?

Sollte diese 0 etwa gk sein?
Wodurch kommt der dazu?

$$\nabla m_{k+1}(x) \stackrel{!}{=} 0.$$

Setting $x = x_k$, $s_k = x_{k+1} - x_k$ and $y_k = \nabla f(x_{k+1}) - \nabla f(x_k) = g_{k+1} - g_k$, we get

$$G_{k+1}^{-1} y_k \approx s_k.$$

This holds with equality, if f is a quadratic function.
Kein schöner Übergang/kein schönes Englisch
Now we want the the approximation of the Hessian inverse B_{k+1} in the quasi-Newton method to satisfy this relation, i.e.,

$$B_{k+1} y_k = s_k \tag{1.6}$$

which is called the quasi-Newton equation, quasi-Newton condition or secant equation. A method that uses this condition to generate its symmetric Hessian (inverse) approximations is called a quasi-Newton method.
For quasi-Newton methods we replace the Hessian of our objective function $\nabla^2 f(x_{k+1})$ in the model by an approximation H_{k+1}:

$$m_{k+1}(x) = f(x_{k+1}) + g_{k+1}^{\mathrm{T}}(x - x_{k+1}) + \frac{1}{2}(x - x_{k+1})^{\mathrm{T}} H_{k+1}(x - x_{k+1})$$

which satisfies the interpolation conditions:

$$m_{k+1}(x_{k+1}) = f(x_{k+1}) \quad \text{and} \quad \nabla m_{k+1}(x_{k+1}) = \nabla f(x_{k+1}).$$

Unlike the normal Newton method, in which we require that $\nabla^2 m(x_{k+1}) = G_{k+1}$, we want the model to satisfy

$$\nabla m_{k+1}(x_k) = g_k$$

from which follows

$$g_k = g_{k+1} + H_{k+1}(x_k - x_{k+1}).$$

So we have

Kann man dazu nicht auch 1.6 einfach mit Hk multiplizieren? So wirkt diese Herleitung etwas doppelt.

$$H_{k+1}(x_{k+1} - x_k) = g_{k+1} - g_k \quad \Leftrightarrow \quad H_{k+1} s_k = y_k. \tag{1.7}$$

This is also called the quasi-Newton equation, quasi-Newton condition or secant equation but now expressed with the approximation of the Hessian.
Of course, we see immediately that the following relationship ~~is assumed~~ holds

$$H_k = B_k^{-1} \quad \forall k \in \mathbb{N}_0 \tag{1.8}$$

Bitte ausschreiben, kurzsymbole nur in Skripten von Tafelwerken

and vice versa. [3]
Assuming that H_k is positive definite, we get a quadratic convex model m_k. The minimizer d_k of it, which we can write explicitly as

Jetzt statt k+1 plötzlich k

$$d_k = -H_k^{-1} g_k = -B_k g_k \tag{1.9}$$

is used as the search direction, and the new iterate is

$$x_{k+1} = x_k + \alpha_k d_k \tag{1.10}$$

Ref.

where the step length α_k is chosen to satisfy the Wolfe conditions. This iteration is quite similar to the line search Newton method, also called globalized Newton method. The key difference is that the approximate Hessian H_k is used in place of the true Hessian $G_k = \nabla^2 f(x_k)$, as already mentioned.
We see that the quasi-Newton equation requires that the symmetric positive definite matrix H_{k+1} maps s_k, the difference between x_{k+1} and x_k, to y_k, the difference between g_{k+1} and g_k. This will be possible only if s_k and y_k satisfy the curvature condition

Why?

$$s_k^{\mathrm{T}} y_k > 0. \tag{1.11}$$

This follows by ~~premultiplying~~ Multiplying the quasi-Newton equation by s_k^{T} From the left, because we assume that H_{k+1} is positive definite. If the function f is strictly convex, then this inequality will be satisfied for any two points x_k and x_{k+1}. For nonconvex functions will this condition not always hold. In this scenario we have to impose restrictions on the line search procedure that chooses the step length α_k. The curvature condition holds if we impose the Wolfe or strong Wolfe conditions on the line search:

$$y_k^{\mathrm{T}} s_k \geq (c_2 - 1)\alpha_k g_k^{\mathrm{T}} d_k. \tag{1.12}$$

Since $c_2 < 1$ and d_k is a descent direction, the right side is positive and the curvature condition holds. When the curvature condition is satisfied, the quasi-Newton equation has always a solution H_{k+1}. In fact, it admits an infinite number of solutions, since the $n(n+1)/2$ degrees of freedom in a symmetric positive definite matrix exceed the conditions imposed by the quasi-Newton equation. The requirement of positive definiteness imposes n additional inequalities - all principal minors must be positive - but these conditions do not absorb the remaining degrees of freedom [2].

A further indication of how the matrices H_k can be constructed is given by the Corollary of Dennis and Moré. Ref Necessary and sufficient for the superlinear convergence of the sequence $\{x_k\}_k$ is, under the conditions mentioned there, the condition: Where, there? Bitte selfcontained arbeiten

$$\|(G_k - H_k)(x_{k+1} - x_k)\| = o(\|x_{k+1} - x_k\|). \tag{1.13}$$

Due to the inequality

$$\begin{aligned}\|g_{k+1} - g_k - G_k(x_{k+1} - x_k)\| \leq\ & \|g_{k+1} - g^* - G^*(x_{k+1} - x^*)\| \\ & + \|g_k - g^* - G^*(x_k - x^*)\| + \|(G^* - G_k)(x_{k+1} - x_k)\|\end{aligned}$$

Wieso gilt das?

where $g^* = \nabla f(x^*)$ and $G^* = \nabla^2 f(x^*)$, one can easily show that Und x* ist?

$$\|g_{k+1} - g_k - G_k(x_{k+1} - x_k)\| = o(\|x_{k+1} - x_k\|)$$

which implicates that the formula (1.13) is equivalent to

$$\|g_{k+1} - g_k - H_k(x_{k+1} - x_k)\| = o(\|x_{k+1} - x_k\|).$$

This motivates the following requirement on H_{k+1}:

$$g_{k+1} - g_k = H_{k+1}(x_{k+1} - x_k)$$

which we see immediately that it is the quasi-Newton equation. [1]

The key point of the quasi-Newton method is to produce H_{k+1} (or B_{k+1}) by use of some convenient methods such that the quasi-Newton condition holds. We will see that we

Ich würde hier erst den Algorithmus erwähnen (und auch auf Algorithmus 1 verweisen), und dann darauf eingehen, dass als nächstes Methoden zur Berechnung von Hk hermüssen.

quasi-Newton method	Newton method
Only need the function values and gradients	Need the function values, gradients and Hessians
$\{H_k\}_k$ maintains positive definite for several updates	$\{G_k\}_k$ is not sure to be positive definite
Need $\mathcal{O}(n^2)$ multiplications in each iteration	Need $\mathcal{O}(n^3)$ multiplications in each iteration

Table 1: Comparison

will get H_{k+1} by updating H_k. The current theory is nevertheless sufficient to formulate a general algorithm.

Algorithm 1 A general quasi-Newton algorithm

1: $x_0 \in \mathbb{R}^n,\ B_0 \in \mathbb{R}^{n\times n},\ 0 \leq \epsilon < 1,\ k = 0$
2: **while** $\|\nabla f(x_k)\| > \epsilon$ **do**
3: Compute $d_k = -B_k \nabla f(x_k)$
4: Determine the step size $\alpha_k > 0$ by line search, and set $x_{k+1} = x_k + \alpha_k d_k$
5: Update B_k into B_{k+1} such that the quasi-Newton equation holds
6: Set $k = k + 1$
7: **end while**
8: **return** x_k

One commonly starts the algorithm with $B_0 = I$, the identity matrix or set B_0 to be a finite-difference approximation to the inverse Hessian $\nabla^2 f(x_0)^{-1}$. If $B_0 = I$, the first iteration is just a steepest descent iteration. In some cases one uses the direct approximation H_k of the Hessian. In this case we need to solve a system of equations in step 3 to get d_k and we need to update H_k instead of B_k. However, since one generally wants to do without solving a system of equations, this variant is not recommended.
The resulting advantages of the quasi-Newton method over the ordinary Newton method are shown in the table 1.

As Newton's method is a steepest descent method under the norm $\|\cdot\|_{G_k}$ the quasi-Newton method is a steepest descent method under the norm $\|\cdot\|_{H_k}$. In fact, d_k is ~~now~~ the solution of the minimization problem

$$\begin{aligned} &\min \quad g_k^{\mathrm{T}} d \\ &\text{s.t.} \quad \|d\|_{H_k} \leq 1. \end{aligned} \tag{1.14}$$

From the inequality

alignment

Which? Nicht due letzte vom Constraint.

Das ist mir zu ungenau formuliert

$$(g_k^{\mathrm{T}} d)^2 \leq (g_k^{\mathrm{T}} H_k^{-1} g_k)(d^{\mathrm{T}} H_k d)$$

it follows that when

$$d_k = -H_k^{-1} g_k = -B_k g_k,$$

$g_k^{\mathrm{T}} d_k$ is the smallest. By the way, since the metric matrices H_k are positive definite and always changed from iteration to iteration, the method is also called the variable metric method. [3]

1.3 The BFGS Formula

We have seen that the search direction in a quasi-Newton method is

$$d_k = -B_k g_k = -H_k^{-1} g_k$$

and the new iterate is

$$x_{k+1} = x_k + \alpha_k d_k.$$

This iteration is quite similar to the Newton method. The key difference is that the approximate Hessian H_k is used in place of the true Hessian $\nabla^2 f(x_k)$. Instead of computing H_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. [2] The question now is how the matrix H_{k+1} (or B_{k+1}) should be constructed from H_k (or B_k) and other information. Various formulae have been developed for this, some of which are interrelated. In this thesis the main focus is on the BFGS formula, which has proven to be superior in practice. However, all approaches follow the following three important guidelines to create H_{k+1}:

1. H_{k+1} should satisfy the quasi-Newton equation.
2. H_{k+1} should be symmetric and positive definite.
3. H_{k+1} should be "near" H_k.

Of course these three characteristics should also hold for the approximation of the inverse B_{k+1}. But where do these characteristics come from? Number 1 and number 3 are requirements that we place on H_{k+1}. We have seen why it should satisfy the quasi-Newton equation, but the strongest motivation comes from the fact that we approximate our objective function local by a quadratic model and the Hessian of a quadratic function always satisfies the quasi-Newton equation. The fact that the distance between H_{k+1} and H_k should not be too large is related to the rate of convergence of the resulting method and the uniqueness of the formula, we will discuss this at the end of this subsection. That the matrix H_{k+1} should be symmetric is obvious, since we want to approximate

the Hessian and the Hessian is always symmetric in the case of a twice continuously differentiable function $f \in C^2$. We need positive definiteness for efficiency, numerical stability and global convergence. If the Hessian $\nabla^2 f(x^*)$ is positive definite, the stationary point x^* is a strong minimizer. Hence, we hope the Hessian approximations $\{H_k\}_k$ (or inverse Hessian approximations $\{B_k\}_k$) are positive definite. In addition, if H_k (or B_k) is positive definite, the local quadratic model of f has a unique local minimizer, and the direction d_k is a descent direction. [3]
Before we introduce the BFGS formula, also called BFGS update, let us first consider another. Because from this we get the BFGS formula by exchanging the variables. It's the so called DFP update, proposed by Davidon and developed later by Fletcher and Powell. This approach is quite intuitive, it deals with the approximation of the inverse. Let us consider a symmetric rank-two update, that means we add two symmetric rank-one matrices to the current matrix

$$B_{k+1} = B_k + auu^{\mathrm{T}} + bvv^{\mathrm{T}}$$

where $u, v \in \mathbb{R}^n$, a and b are scalars to be determined. From the quasi-Newton equation follows

$$B_k y_k + auu^{\mathrm{T}} y_k + bvv^{\mathrm{T}} y_k = s_k.$$

Clearly, u and v are not uniquely determined, but their obvious choices are

$$u = s_k,\ v = B_k y_k.$$

Then we have

$$a = \frac{1}{u^{\mathrm{T}} y_k} = \frac{1}{s_k^{\mathrm{T}} y_k},\ b = -\frac{1}{v^{\mathrm{T}} y_k} = \frac{1}{y_k^{\mathrm{T}} B_k y_k}.$$

Therefore

$$B_{k+1}^{DFP} = B_k^{DFP} + \frac{s_k s_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k} - \frac{B_k^{DFP} y_k y_k^{\mathrm{T}} B_k^{DFP}}{y_k^{\mathrm{T}} B_k^{DFP} y_k}.$$

This is the DFP update, which approximates the inverse of the Hessian $\nabla^2 f(x_k)^{-1}$ in each iteration. [3]

We see, that the last two terms in the right-hand-side are symmetric rank-one matrices. This is the fundamental idea of quasi-Newton updating: Instead of recomputing the approximate Hessians (or inverse Hessians) from scratch at every iteration, we apply a simple modification that combines the most recently observed information about the objective function with the existing knowledge embedded in our current Hessian approximation. The DFP updating formula is quite effective, but it was soon superseded by the BFGS formula,which is presently considered to be the most effective of all quasi-Newton updating formulae. [2]

The BFGS formula can be obtained by simple trick: for H_{k+1}^{BFGS} replace the triple (B_k^{DFP}, s_k, y_k) in B_{k+1}^{DFP} by (H_k^{BFGS}, y_k, s_k). Thus, BFGS update is also said to be a complement DFP update. This means that

$$H_{k+1}^{BFGS} = H_k^{BFGS} + \frac{y_k y_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k} - \frac{H_k^{BFGS} s_k s_k^{\mathrm{T}} H_k^{BFGS}}{s_k^{\mathrm{T}} H_k^{BFGS} s_k}. \tag{1.15}$$

[3]

This formula was discovered practically simultaneously and independently of each other by Broyden, Fletcher, Goldfarb and Shanno. It is interesting that all four authors derive this formula in a slightly different way. The fact that so many different methods lead to the BFGS formula may already be seen here as a reason why the BFGS formula is superior to the other updating formulae in practice. [1]

Since $H_k s_k = -\alpha_k g_k$ and $H_k d_k = -g_k$, this formula can also be written as

$$H_{k+1}^{BFGS} = H_k^{BFGS} + \frac{g_k g_k^{\mathrm{T}}}{g_k^{\mathrm{T}} d_k} + \frac{y_k y_k^{\mathrm{T}}}{\alpha_k y_k^{\mathrm{T}} d_k}.$$

By using twice the Sherman-Morrison formula, we get

$$\begin{aligned} B_{k+1}^{BFGS} &= B_k^{BFGS} + \frac{(s_k - B_k^{BFGS} y_k) s_k^{\mathrm{T}} + s_k (s_k - B_k^{BFGS} y_k)^{\mathrm{T}}}{y_k^{\mathrm{T}} s_k} - \frac{(s_k - B_k^{BFGS} y_k)^{\mathrm{T}} y_k s_k s_k^{\mathrm{T}}}{(y_k^{\mathrm{T}} s_k)^2} \\ &= B_k^{BFGS} + \left(1 + \frac{y_k^{\mathrm{T}} B_k^{BFGS} y_k}{s_k^{\mathrm{T}} y_k}\right) \frac{s_k s_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k} - \frac{s_k y_k^{\mathrm{T}} B_k^{BFGS} + B_k^{BFGS} y_k s_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k} \\ &= \left(I - \frac{s_k y_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k}\right) B_k^{BFGS} \left(I - \frac{y_k s_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k}\right) + \frac{s_k s_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k} \end{aligned} \tag{1.16}$$

three different BFGS formulae for the approximation of the inverse of the Hessian. It is to see that (1.16) is also a rank-two modification of B_k^{BFGS}. One can simply show that

$$H_{k+1}^{BFGS} B_{k+1}^{BFGS} = B_{k+1}^{BFGS} H_{k+1}^{BFGS} = I.$$

If one now replaces the triple (B_k^{BFGS}, s_k, y_k) in 1.16 by (H_k^{DFP}, y_k, s_k) one would get a formula for H_{k+1}^{DFP}, the direct DFP update. This describes a method for finding its dual update from a given update. Given a quasi-Newton update B_{k+1} about B-form, by replacing the triple (B_k, s_k, y_k) by $(H_k^{(D)}, y_k, s_k)$, we can get its dual update $H_{k+1}^{(D)}$ about H-form. Then, applying the Sherman-Morrison formula to $H_{k+1}^{(D)}$, we will produce the dual update $B_{k+1}^{(D)}$ of B_{k+1} about the B-form. Similarly, if we employ the same operations to the dual update $B_{k+1}^{(D)}$, the original update B_{k+1} will be restored. Notice that, for an B-form, the dual update of B_{k+1} is $B_{k+1}^{(D)}$. In addition, the dual operation maintains the

quasi-Newton equation. For this reason, the DFP and BFGS formulae are sometimes referred to as "dual" updating formulae. [3]
It now remains to be clarified whether the constructed formula for H_{k+1}^{BFGS} has the desired characteristics. For 1. and 2. we have the following

Theorem 1. *1. If $y_k^{\mathrm{T}} s_k \neq 0$ and $s_k^{\mathrm{T}} H_k^{BFGS} s_k \neq 0$ holds, the matrices $H_{k+1}^{BFGS} \in \mathbb{R}^n$ are well defined, symmetric and satisfy the quasi-Newton equation.*

2. If H_k^{BFGS} is positive definite and $y_k^{\mathrm{T}} s_k > 0$, the H_{k+1}^{BFGS} is positive definite.

[4]

Such an update is also called positive definite update. The same holds of course for the approximation of the inverse B_{k+1}^{BFGS}. This theorem is a special case of a theorem from Ulbrich, Ulbrich "Nichtlineare Optimierung" in which it was applied to Broyden class matrices. This means that the statement of the theorem can be transferred one-to-one to the DFP update H_{k+1}^{DFP}.

The third characteristic has a far more powerful meaning than the other two. Many authors use it to define the BFGS formula which of course is perfectly legitimate. As already mentioned, this property leads to the fact that we can consider the formula as unique and it has something to do with the rate of convergence. The two go hand in hand.
An actual statement on the rate of convergence will only be dealt with in the next subsection. Nevertheless, we will already give a motivation for this here. We want the BFGS method to be similar to the usual Newton method in terms of convergence. This means that we want local superlinear convergence. We remember that we can prove superlinear convergence by the Dennis-Moré condition. Now we must show that this condition is fulfilled. This works through the following

Lemma 1. *x^* fulfils the sufficient condition of second order. Algorithm 1 with $\alpha_k = 1$ for all $k \in \mathbb{N}$ generates a sequence $\{x_k\}_k$ convergent to x^* and also holds*

$$\lim_{k\to\infty} \|H_{k+1} - H_k\| = 0,$$

then H_k satisfies the Dennis-Moré condition and $\{x_k\}_k$ converges q-superlinear to x^*.

We are therefore looking for quasi-Newton updates for which H_{k+1} is close to H_k. As we see, this motivation aims at solving an optimization problem. We want to keep the distance from H_{k+1} to H_k small in each iteration, so that it converges towards zero. Later we will also see that this method has superlinear convergence rate under further conditions.
We would now like to consider the last property from the point of view of the uniqueness of the formula. If we were to pursue the motivation of convergence further, we would come to the same result. The terms uniqueness and distance suggest that the formula is the solution to an optimization problem, which it is. For a more detailed treatment of this

topic I recommend Geiger, Kanzow, "Numerische Verfahren zur Lösung unrestringierter Optimierungsaufgaben" (1999) or Nocedal, Wright "Numerical Optimization" (2006). The derivation with the optimization problem is again closely related to the DFP update, but as said before, more about this can be found in the mentioned sources. The following two statements provide us with the uniqueness of the BFGS formula.

Lemma 1. *Let be $s \in \mathbb{R}^n$, $y \in \mathbb{R}^n$ with $y \neq 0$ and a symmetrical matrix $B \in \mathbb{R}^{n\times n}$ given. Furthermore $W \in \mathbb{R}^{n\times n}$ should be symmetrical and positive definit. Then the clearly determined solution of the inverse weighted problem*

$$\begin{aligned} \min_{B_+} \quad & \|W(B_+ - B)W\|_{\mathrm{F}}^2 \\ s.t. \quad & B_+ = B_+^{\mathrm{T}}, \qquad\qquad B_+ y = s \end{aligned} \tag{1.17}$$

is given by

$$B_+^W = B + \frac{(s-By)(W^{-2}y)^{\mathrm{T}} + W^{-2}y(s-By)^{\mathrm{T}}}{(W^{-2}y)^{\mathrm{T}}y} - \frac{y^{\mathrm{T}}(s-By)W^{-2}y(W^{-2}y)^{\mathrm{T}}}{((W^{-2}y)^{\mathrm{T}}y)^2}.$$

By a suitable selection of the weighting matrix W one obtains now the

Theorem 1. *Let $B \in \mathbb{R}^{n\times n}$ be symmetric and positive definite and $s, y \in \mathbb{R}^n$ with $s^{\mathrm{T}}y > 0$. Let $Q \in \mathbb{R}^{n\times n}$ be a symmetric and positive definite matrix with $Qs = y$, and let $W = Q^{\frac{1}{2}}$ be a square root of Q. Then the unique solution of the inverse weighted problem with the thus weighted W is given by*

$$B_+^{BFGS} = B + \frac{(s-By)s^{\mathrm{T}} + s(s-By)^{\mathrm{T}}}{y^{\mathrm{T}}s} - \frac{(s-By)^{\mathrm{T}}yss^{\mathrm{T}}}{(y^{\mathrm{T}}s)^2}. \tag{1.18}$$

For corectness, we can assume that $W = \bar{G}_k$ and $\bar{G}_k$ is the average Hessian, i.e.

$$\bar{G}_k = \int_0^1 \nabla^2 f(x_k + \tau s_k) d\tau.$$

The specified minimum characteristic with respect to the weighted norms mentioned in the theorem automatically ensures the invariance of the BFGS method under affin-linear variable transformations. This important characteristic is also present in the Newton method. [4]

Just one issue has to be resolved before we can define a complete BFGS algorithm: How should we choose the initial approximation B_0^{BFGS}? Unfortunately, there is no magic formula that works well in all cases. We can use specific information about the problem, for instance by setting it to the inverse of an approximate Hessian calculated by finite differences at x_0. Otherwise, we can simply set it to be the identity matrix, or a multiple of the identity matrix βI, where the multiple β is chosen to reflect the scaling of the variables. There is no good general strategy for choosing the multiple β. If β is

too large, so that the first step $d_0 = -\beta g_0$ is too long, many function evaluations may be required to find a suitable value for the step length α_0. Some software asks the user to prescribe a value δ for the norm of the first step, and then set $B_0^{BFGS} = \delta \|g_0\|^{-1} I$ to achieve this norm. A heuristic that is often quite effective is to scale the starting matrix after the first step has been computed but before the first BFGS update is performed. We change the provisional value $B_0^{BFGS} = I$ by setting

$$B_0^{BFGS} = \frac{y_1^{\mathrm{T}} s_1}{y_1^{\mathrm{T}} y_1} I$$

before applying the update to obtain B_1^{BFGS}. This formula attempts to make the size of B_0^{BFGS} similar to that of $\nabla^2 f(x_0)^{-1}$. [2]

1.4 A Local BFGS-Method

In this subsection we introduce a local BFGS method. For this we will use the updating formula for the approximation of the inverse of the Hessian ($B_k^{BFGS} \mapsto B_{k+1}^{BFGS}$), since we are spared the solving of a system of equations and we only have to work with matrix-vector-multiplications. Therefore we call the following algorithm "Inverse Local BFGS-Method":

Algorithm 2 Inverse Local BFGS-Method

1: $x_0 \in \mathbb{R}^n,\ B_0^{BFGS} \in \mathbb{R}^{n \times n}$ spd, $0 \leq \epsilon < 1,$ set $k = 0$
2: **while** $\|\nabla f(x_k)\| > \epsilon$ **do**
3: Compute $d_k = -B_k^{BFGS} \nabla f(x_k)$
4: Set $x_{k+1} = x_k + d_k,\ s_k = x_{k+1} - x_k,\ y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$
5: Set $B_{k+1}^{BFGS} = B_k^{BFGS} + \frac{(s_k - B_k^{BFGS} y_k) s_k^{\mathrm{T}} + s_k (s_k - B_k^{BFGS} y_k)^{\mathrm{T}}}{y_k^{\mathrm{T}} s_k} - \frac{(s_k - B_k^{BFGS} y_k)^{\mathrm{T}} y_k s_k s_k^{\mathrm{T}}}{(y_k^{\mathrm{T}} s_k)^2}$
6: Set $k = k + 1$
7: **end while**
8: **return** x_k

Of course the algorithm could be formulated with the approximation of the actual Hessian H_k^{BFGS}, but that would increase the effort again to $\mathcal{O}(n^3)$, which is not desirable. [2] It is noticeable that the step size in this algorithm is constant, which is $\alpha_k = 1$ for all $k \in \mathbb{N}_0$. The convergence analysis of this method is very complex and would go beyond the scope of this work. Nevertheless, a statement regarding the convergence rate should be mentioned, as it characterizes the method well. The derivation of the next theorem and its proof can be found in the book "Numerische Verfahren zur Lösung unrestringierter Optimierungsaufgaben" by C. Geiger and C. Kanzow. In this book the analysis of local convergence is examined in more detail. In the following we assume that the algorithm does not abort after a finite number of steps, in particular f = 0 should hold.

Theorem 1. *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable, $\nabla^2 f$ locally Lipschitz continuous and $x^* \in \mathbb{R}^n$ with $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ positive definite. Then a $\epsilon > 0$ and a $\delta > 0$ exist, so the inverse local BFGS-method (Algorithm 2) is well defined for each start vector $x_0 \in \mathbb{R}^n$ with $\|x_0 - x^*\| < \epsilon$ and each symmetrical and positive definite start matrix $B_0^{BFGS} \in \mathbb{R}^{n\times n}$ with $\|B_0^{BFGS} - \nabla^2 f(x^*)^{-1}\|_{\mathrm{F}} < \delta$ and produces a sequence $\{x_k\}_k$ which converges superlinear to x^*.*

Lemma 1. *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable, $\nabla^2 f$ locally Lipschitz continuous and $x^* \in \mathbb{R}^n$ with $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ positive definite. Furthermore, let $\{x_k\}_k$ be a sequence generated by the inverse local BFGS-method 2 with*

$$\sum_{k=0}^{\infty} \|x_k - x^*\| < \infty$$

(in particular, the sequence $\{x_k\}_k$ converges to x^*). Furthermore, if the sequence $\{\rho_k\}_k$ defined by

$$\rho_k = \|W(B_k^{BFGS} - \nabla^2 f(x^*)^{-1})W\|_{\mathrm{F}}$$

(with $W = \nabla^2 f(x^*)^{\frac{1}{2}}$) is restricted,

$$\|(B_k^{BFGS} - \nabla^2 f(x^*)^{-1})(\nabla f(x_{k+1}) - \nabla f(x_k))\| = o(\|\nabla f(x_{k+1}) - \nabla f(x_k)\|)$$

holds.

Theorem 1. *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable, $\nabla^2 f$ locally Lipschitz continuous and $x^* \in \mathbb{R}^n$ with $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ positive definite. Then a $\epsilon > 0$ and a $\delta > 0$ exist, so the inverse local BFGS-method (Algorithm 2) is well defined for each start vector $x_0 \in \mathbb{R}^n$ with $\|x_0 - x^*\| < \epsilon$ and each symmetrical and positive definite start matrix $B_0^{BFGS} \in \mathbb{R}^{n\times n}$ with $\|B_0^{BFGS} - \nabla^2 f(x^*)^{-1}\|_{\mathrm{F}} < \delta$ and produces a sequence $\{x_k\}_k$ which converges superlinear to x^*.*

1.5 A Globalized BFGS-Method

The algorithm is robust, and its rate of convergence is superlinear,whichisfastenoughformostpracticalpur convergesmorerapidly(thatis,quadratically),itscostperiterationusuallyishigher,because of its need for second derivatives and solution of a linear system. [2]

1.6 Limited-Memory BFGS-Method

One of the disadvantages of the Quasi-Newton methods is that an $n \times n$ matrix must be stored in each iteration step. We have seen, even when using the symmetry of this matrix, a memory requirement of $n(n + 1)/2$ matrix entries remains. For large-scale optimization problems, however, this is far too much.

Algorithm 3 Inverse Global BFGS-Method

$x_0 \in \mathbb{R}^n,\ B_0^{BFGS} \in \mathbb{R}^{n\times n}$ spd, $0 \leq \epsilon < 1,\ \sigma \in (0, \frac{1}{2}),\ \rho \in (\sigma, 1),$ set $k = 0$
while $\|\nabla f(x_k)\| > \epsilon$ **do**
    Compute $d_k = -B_k^{BFGS}\nabla f(x_k)$
    Find a step length $\alpha_k = \alpha(\sigma, \rho)$ that satisfies the Wolfe conditions.
    Set $x_{k+1} = x_k + \alpha_k d_k,\ s_k = x_{k+1} - x_k,\ y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$
    Set $B_{k+1}^{BFGS} = B_k^{BFGS} + \frac{(s_k - B_k^{BFGS}y_k)s_k^{\mathrm{T}} + s_k(s_k - B_k^{BFGS}y_k)^{\mathrm{T}}}{y_k^{\mathrm{T}}s_k} - \frac{(s_k - B_k^{BFGS}y_k)^{\mathrm{T}}y_k s_k s_k^{\mathrm{T}}}{(y_k^{\mathrm{T}}s_k)^2}$
    Set $k = k + 1$
end while
return x_k

Limited-memory quasi-Newton methods, also called variable-storage quasi-Newton methods, are useful for solving large problems whose Hessian matrices cannot be computed at a reasonable cost or are not sparse. The methods save only a few n-dimensional vectors, instead of storing and computing fully dense $n \times n$ approximations of the Hessian. The main idea is to use the curvature information from only the most recent iterations to construct the Hessian approximation. Curvature information from earlier iterations, which is less likely to be relevant to the actual behavior of the Hessian at the current iteration, is discarded in the interest of saving storage. Despite these modest storage requirements, they often yield an acceptable rate of convergence.
Due to the outstanding importance of the BFGS method in the class of quasi-Newton methods, it is also predominantly used as a limited memory variant, called L-BFGS. But there are also limited-memory versions of other quasi-Newton methods such as SR1. but also to limited-memory versions of other quasi-Newton procedures such as SR1.

We remember the inverse BFGS formulas (1.16). We now need the latter

$$B_{k+1}^{BFGS} = \Big(I - \frac{s_k y_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k}\Big) B_k^{BFGS} \Big(I - \frac{y_k s_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k}\Big) + \frac{s_k s_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k}$$

Set

$$\rho_k = \frac{1}{s_k^{\mathrm{T}} y_k},\ V_k = I - \rho_k y_k s_k^{\mathrm{T}}, \tag{1.19}$$

then we get

$$B_{k+1}^{BFGS} = V_k^{\mathrm{T}} B_k^{BFGS} V_k + \rho_k s_k s_k^{\mathrm{T}}. \tag{1.20}$$

The above equation says that the matrix B_{k+1}^{BFGS} is obtained by updating B_k^{BFGS} using the pair $\{s_k, y_k\}$. [3] Since the inverse Hessian approximation B_k^{BFGS} will generally be dense, the cost of storing and manipulating it is prohibitive when the number of variables is large. To circumvent this problem, we store a modified version of B_k^{BFGS} implicitly,

by storing a certain number (say, m) of the vector pairs $\{s_i, y_i\}$. The product $B_k^{BFGS} g_k$ can be obtained by performing a sequence of inner products and vector summations involving g_k and the pairs $\{s_i, y_i\}$. After the new iterate is computed, the oldest vector pair in the set of pairs $\{s_i, y_i\}$ is replaced by the new pair $\{s_k, y_k\}$ obtained from the current step. In this way, the set of vector pairs includes curvature information from the m most recent iterations. Practical experience has shown that modest values of m (between 3 and 20, say) often produce satisfactory results.

$$B_k^{(j+1)} = V_{k-m+j}^{\mathrm{T}} B_k^{(j)} V_{k-m+j} + \rho_{k-m+j} s_{k-m+j} s_{k-m+j}^{\mathrm{T}},\ j = 0, 1, \cdots, m-1 \tag{1.21}$$

Algorithm 4 L-BFGS two-loop recursion for $B_k^{BFGS} g_k$

$q = g_k$
for $i = k-1, k-2, \cdots, k-m$ **do**
    $\alpha_i = \rho_i s_i^{\mathrm{T}} q$
    $q = q - \alpha_i y_i$
end for
$r = B_k^{(0)} q$
for $i = k-m, k-m+1, \cdots, k-1$ **do**
    $\beta = \rho_i y_i^{\mathrm{T}} r$
    $r = r + si(\alpha_i - \beta)$
end for
stop with result $B_k^{BFGS} g_k = r$

Algorithm 5 L-BFGS Method

Given a starting point $x_0 \in \mathbb{R}^n$, an initial symmetric and positive definite matrix $B_0 \in \mathbb{R}^{n \times n}$, a nonnegative integer $m \geq 0$, an error tolerance $\epsilon > 0$, $k = 0$.
Compute g_k. If $\|g_k\| \leq \epsilon$, we take $x^* = x_k$, stop. Otherwise, compute $d_k = -B_k g_k$ from Algorithm 4.
Find a step size $\alpha_k > 0$ by using Wolfe rule.
Set $x_{k+1} = x_k + \alpha_k d_k$.
If $k > m$, discard the vector pairs $\{s_{k-m}, y_{k-m}\}$ from storage. Set $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$. Take $B_k^{(0)} = \frac{s_k^{\mathrm{T}} y_k}{\|y_k\|^2} I$.

2 Riemannian Manifolds

3 The BFGS-Method For Riemannian Manifolds

3.1 Preliminary

3.2 Quasi-Newton Methods For Riemannian Manifolds

3.3 The BFGS Formula For Riemannian Manifolds

3.4 A Local BFGS-Method Fo rRiemannian Manifolds

3.5 A Globalized BFGS-Method For Riemannian Manifolds

3.6 Limited-Memory BFGS-Method For Riemannia nManifolds

4 Numerics

4.1 Implementation Of A Local BFGS-Method For Riemannian Manifolds

4.2 Implementation Of A Globalized BFGS-Method For Riemannian Manifolds

4.3 Implementation Of A Limited-Memory BFGS-Method For Riemannian Manifolds

5 Conclusion

Literatur

References

[1] Carl Geiger and Christian Kanzow. *Numerische Verfahren zur Lösung unrestringierter Optimierungsaufgaben*. Springer, 1999. DOI: 10.1007/978-3-642-58582-1.

[2] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Second Edition. Springer, 2006. DOI: 10.1007/978-0-387-40065-5.

[3] Wenyu Sun and Ya-Xiang Yuan. *Optimization Theory and Methods: Nonlinear Programming*. Vol. 1. Springer, 2006. DOI: 10.1007/b106451.

[4] Michael Ulbrich and Stefan Ulbrich. *Nichtlineare Optimierung*. Springer, 2012. DOI: 10.1007/978-3-0346-0654-7.