Technical University of Chemnitz

Faculty of Mathematics

BFGS-Method On Riemannian Manifolds

Tom-Christian Riemer

for obtaining the academic degree
Bachelor of Science

Author:	Tom-Christian Riemer MatNr. 444019
Version from:	24th April 2020
1. Supervisor:	PD Dr. Ronny Bergmann
2. Supervisor:	Prof. Dr. Roland Herzog

Contents

1 The BFGS-Method For The Euclidean Case

1.1 Preliminaries

$$\min f(x), \quad x \in \mathbb{R}^n \tag{1.1}$$

Algorithm 1 General descent method

```
$x_0 \in \mathbb{R}^n,\ k = 0$
while $x_k$ does not satisfy any stopping criterion do
    Determine a descent direction $d_k$ of $f$ in $x_k$.
    Determine a step size $\alpha_k > 0$ with $f(x_k + \alpha_k d_k) < f(x_k)$.
    Set $x_{k+1} = x_k + \alpha_k d_k$ and $k = k + 1$.
end while
return $x_k$
```

Algorithm 2 Local Newton's method

```
$x_0 \in \mathbb{R}^n,\ 0 \le \epsilon < 1,\ k = 0$
while $\|\nabla f(x_k)\| > \epsilon$ do
    Determine $d_k \in \mathbb{R}^n$ by solving
        $\nabla^2 f(x_k) d = -\nabla f(x_k).$
    Set $x_{k+1} = x_k + d_k$ and $k = k + 1$.
end while
return $x_k$
```

Wolfe conditions:

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

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

strong Wolfe conditions:

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

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

Realization of the Wolfe-Powell conditions

In the following f is a continuously differentiable function and $c_1 \in (0, \frac{1}{1})$, $c_2 \in [c_1, 1)$ are fixed numbers. For $x_k \in \mathbb{R}^n$, $d_k \in \mathbb{R}^n$ with $\nabla f(x_k)^{\mathrm{T}} d_k < 0$ we define

$$\phi(\alpha) = f(x_k + \alpha d_k)$$

and

$$\psi(\alpha) = \phi(\alpha) - \phi(0) - \alpha c_1 \phi^{'}(0)$$

The Wolfe-Powell conditions read

$$\psi(\alpha) \leq 0,\ \phi^{'}(\alpha) \geq c_2 \phi^{'}(0)$$

Lemma 1 ([9, Lemma 3.1])**.** *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.*

Theorem 1 ([9, Theorem 3.6])**.** *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.*

Definition 1 ([6])**.** *A map T of $\mathbb{R}^n \times \mathbb{R}^n$ into the power set of positive real numbers $\mathcal{P}((0,\infty))$, i.e. a map that assigns a subset $T(x,d)$ of $\mathbb{R}$ to each pair (x,d), is called step size strategy or step size rule. We call such a step size rule (under certain conditions) well-defined, if (under these conditions) the set $T(x,d)$ for each pair (x,d) with $\nabla f(x)^{\mathrm{T}} d < 0$ is not empty.*

Definition 2 ([6, Definition 4.5])**.** *Let $f \colon \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable, $x \in \mathbb{R}^n$ and $d \in \mathbb{R}^n$ a descent direction from f in x. A step size strategy T is called efficient if there is a constant $\theta > 0$ independent of x and d with*

$$f(x + \alpha d) \leq -\theta \left(\frac{\nabla f(x)^{\mathrm{T}} d}{\|d\|} \right)^2$$

for all $\alpha \in T(x,d)$.

Corollary 1 (Superlinear convergence [6, Lemma 7.9]). *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 defnded by*

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

with the limit $\lim\limits_{n\to\infty} x_k = x^*$, $x_k \neq x^*$ 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. *$\|(\nabla^2 f(x^*) - H_k)(x_{k+1} - x_k)\| = o(\|x_{k+1} - x_k\|)$*

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 2 ([12, Theorem 1.2.15]). *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

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

Theorem 3 (Sherman-Morrison-Woodbury Theorem [12, Theorem 1.2.16]). *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*

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

Note: The Sherman-Morrison-Woodbury formula can be extended to rank R modifications [13].

Theorem 4 ([6, Theorem 4.6]). *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable and let $\{x_k\}_k$ be a sequence generated by the descent method (Algorithm 1) such that the following two conditions are satisfied:*

1. *There is a constant $c > 0$ such that*

$$-\frac{\nabla f(x_k)^{\mathrm{T}} d_k}{\|\nabla f(x_k)\|\|d_k\|} \geq c \tag{1.6}$$

for all $k \in \mathbb{N}$ (this is the so-called angle condition).

2. *The step sizes $\alpha_k > 0$ are efficient for all $k \in \mathbb{N}$.*

Then each limit point of the sequence $\{x_k\}_k$ is a stationary point of f.

Theorem 5 ([6, Theorem 4.7])**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable, the level set $\mathcal{L}(x_0) = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$ convex and f uniformly convex on $\mathcal{L}(x_0)$. Let $\{x_k\}_k$ be a sequence generated by the descent method (Algorithm 1) such that the following two conditions are satisfied:*

1. *It holds $\sum_{k=0}^{\infty} \delta_k = \infty$ where*

$$\delta_k = \left(\frac{\nabla f(x_k)^{\mathrm{T}} d_k}{\|\nabla f(x_k)\| \|d_k\|} \right)^2$$

(this is the so-called Zoutendijk condition).

2. *The step sizes $\alpha_k > 0$ are efficient for all $k \in \mathbb{N}$.*

Then the sequence $\{x_k\}_k$ converges towards the uniquely determined global minimum of (1.1).

Theorem 6 ([6, Theorem 5.3])**.**

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 Newton method, both the gradient and the Hessian are calculated in every 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 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.

What do we expect from this sequence $\{H_k\}_k$ now? The sequence should posses positive definiteness, $d_k = -H_k^{-1} \nabla f(x_k)$ should be a descent direction and the resulting method should behave like Newton's method in terms of convergence. Of course, the calculation

should cost less.
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}:

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

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

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

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.
Now one is interested in the fact that an approximation of the Hessian inverse B_{k+1} satisfies this relation for the quasi-Newton method, i.e.,

$$B_{k+1} y_k = s_k \tag{1.7}$$

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

$$H_{k+1}(x_{k+1} - x_k) = g_{k+1} - g_k \quad \Leftrightarrow \quad H_{k+1} s_k = y_k. \tag{1.8}$$

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 holds

$$H_k = B_k^{-1} \quad \text{for all } k \in \mathbb{N}_0 \tag{1.9}$$

and vice versa [12].
To the definition of an iteration. A model at the current iteration x_k is defined:

$$f(x) \approx m_k(x) = f(x_k) + g_k^{\mathrm{T}}(x - x_k) + \frac{1}{2}(x - x_k)^{\mathrm{T}} G_k (x - x_k)$$

Assuming that H_k is positive definite, we get a quadratic convex model m_k. The minimizer d_k of it, we can write explicitly as

$$d_k = -H_k^{-1} g_k = -B_k g_k \tag{1.10}$$

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

$$x_{k+1} = x_k + \alpha_k d_k \tag{1.11}$$

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.
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

$$s_k^{\mathrm{T}} y_k > 0. \tag{1.12}$$

This follows from 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:

$$y_k^{\mathrm{T}} s_k \geq (c_2 - 1)\alpha_k g_k^{\mathrm{T}} d_k. \tag{1.13}$$

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 [9].

That it makes sense that the matrices H_{k+1} satisfy the quasi-Newton equation is indicated by the Corollary 1 of Dennis and Moré. Necessary and sufficient for the superlinear convergence of the sequence $\{x_k\}_k$ to a minimizer x^* is the condition:

$$\|(G_k - H_k)(x_{k+1} - x_k)\| = o(\|x_{k+1} - x_k\|). \tag{1.14}$$

It can be shown that (1.14) 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}:

$$H_{k+1}(x_{k+1} - x_k) = g_{k+1} - g_k \tag{1.15}$$

which we see immediately that it is the quasi-Newton equation (1.8) [6].

The current theory is nevertheless sufficient to formulate a general algorithm.

Algorithm 3 General Quasi-Newton Method

```
$x_0 \in \mathbb{R}^n,\ B_0 \in \mathbb{R}^{n\times n}$ approximation of $\nabla^2 f(x_0)^{-1},\ 0 \le \epsilon < 1,\ k = 0$
while $\|\nabla f(x_k)\| > \epsilon$ do
    Compute $d_k = -B_k \nabla f(x_k)$.
    Determine the step size $\alpha_k > 0$ by line search.
    Set $x_{k+1} = x_k + \alpha_k d_k$.
    Update $B_k$ into $B_{k+1}$ such that the quasi-Newton equation holds.
    Set $k = k + 1$.
end while
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 the solution of the minimization problem

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

$$\begin{aligned} \min \quad & g_k^{\mathrm{T}} d \\ \text{s.t.} \quad & \|d\|_{H_k} \leq 1. \end{aligned} \tag{1.16}$$

It follows from

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

that the solution of (1.16) is

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

and $g_k^{\mathrm{T}} d_k$ is the smallest vallue. 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 [12].

1.3 The Broyden-Fletcher–Goldfarb-Shanno Formula

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

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

and the new iterate is

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

This iteration is quite similar to the one of Newton's 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 [9]. 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 Broyden-Fletcher–Goldfarb-Shanno formula or short BFGS formula, which has proven to be the most efficient quasi-Newton method in practice [13]. However, all approaches follow the following three important guidelines to create H_{k+1}:

1. H_{k+1} satisfies the quasi-Newton equation.
2. H_{k+1} is symmetric and positive definite.
3. H_{k+1} is “near” H_k.

Of course these three characteristics should also hold for the approximation of the inverse B_{k+1}. In the previous subsection was shown that H_{k+1} should satisfy the quasi-Newton equation (1.8). 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 will be related to the rate of convergence of the resulting method and the uniqueness of the formula. It’s obvious that the matrix H_{k+1} should be symmetric, 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 [12].
Before we get to the BFGS formula, also called BFGS update, let us first look at another one. By exchanging variables, we then get the BFGS formula. It’s the so called DFP update, proposed by Davidon [3] and developed later by Fletcher and Powell [5]. We assume that the matrix B_k approximates $\nabla^2 f(x_k)^{-1}$ sufficiently well. 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 $u, v \in \mathbb{R}^n$, $a, b \in \mathbb{R}$ are to be determined. From the quasi-Newton equation follows

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

Clearly, u and v can not uniquely be determined. One possible choice is

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

Hence we obtain

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

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 every iteration [12].

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 Hessian (or inverse Hessian) 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 [9].

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. The result is

$$H_{k+1}^{BFGS} = H_k^{BFGS} + \frac{y_k y_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k} - \frac{H_k^{BFGS} s_k s_k^{\mathrm{T}} H_k^{BFGS}}{s_k^{\mathrm{T}} H_k^{BFGS} s_k}. \tag{1.17}$$

[12]

This formula was discovered independently by Broyden [1], Fletcher [4], Goldfarb [7] and Shanno [11], which is the reason for the name. All four authors derive the BFGS formula in a slightly different way, which can be seen as a reason why it is superior to the other updating formulae in practice [6]. It is presently considered to be the most effective of all quasi-Newton updating formulae [9]. The DFP update is quite effective, but it was soon superseded by the BFGS formula, which has all good properties of the DFP update [12].

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 applying the the Sherman–Morrison–Woodbury formula twice to (1.17), we obtain

$$
\begin{aligned}
B_{k+1}^{BFGS} &= B_k^{BFGS} + \frac{(s_k - B_k^{BFGS} y_k)s_k^{\mathrm{T}} + s_k(s_k - B_k^{BFGS} y_k)^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k} - \frac{(s_k - B_k^{BFGS} y_k)^{\mathrm{T}} y_k s_k s_k^{\mathrm{T}}}{(s_k^{\mathrm{T}} y_k)^2} \\
&= B_k^{BFGS} + \left(1 + \frac{y_k^{\mathrm{T}} B_k^{BFGS} y_k}{s_k^{\mathrm{T}} y_k}\right)\frac{s_k s_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k} - \frac{s_k y_k^{\mathrm{T}} B_k^{BFGS} + B_k^{BFGS} y_k s_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k} \\
&= \left(I - \frac{s_k y_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k}\right) B_k^{BFGS} \left(I - \frac{y_k s_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k}\right) + \frac{s_k s_k^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k}.
\end{aligned}
\tag{1.18}
$$

These are different formulae for the approximation of the Hessian inverse. Furthermore, reference is also made to this with BFGS formula. It is easy to see that (1.18) is also a rank-two modification of B_k^{BFGS}. One can easily show that

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

Replacing the triple (B_k^{BFGS}, s_k, y_k) in (1.18) 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. For this reason, the DFP and BFGS formulae are sometimes referred to as "dual" updating formulae [12].

Now it is checked that H_{k+1}^{BFGS} meets the given characteristics. For 1. and 2. there is the following statement:

Theorem 7 ([13, Theorem 13.4])**.** *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 $H_{k+1}^{BFGS} \in \mathbb{R}^{n \times n}$ are well defined, symmetric and satisfy the quasi-Newton equation (1.8).*

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

Such an update is also called positive definite update. The same holds of course for the approximation of the inverse B_{k+1}^{BFGS}. In the previous subsection was shown that the curvature condition (1.12) must hold. This was achieved by imposing restrictions on the line search method (1.13). So the positive definiteness can be guaranteed just by a Wolfe line search strategy. The statement was actually made for Broyden class matrices (the matrices are a convex combination of DFP and BFGS matrices) in [13], this means that it can be transferred one-to-one to the DFP update H_{k+1}^{DFP}.

The last characteristic, that H_{k+1} should be "near" H_k, has a far more powerful meaning than the other two. Many authors use only this to define the BFGS formula which of course is perfectly legitimate. As already mentioned, this property leads to the fact that the formula can be considered as unique and it has something to do with the rate of convergence. The two go hand in hand.

One wants the BFGS method to be similar to Newton's method in terms of convergence. This means that it should converge superlinearly. This can be proven by the Dennis-

Moré condition. The connection between this condition and the characteristic is shown in [13] by

Lemma 2 ([13, Lemma 13.2])**.** *x^* fulfils the sufficient condition of second order. Algorithm 3 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^*.

Therefore one looks for quasi-Newton updates for which H_{k+1} is close to H_k in each iteration, so that the distance between them converges towards zero.
We would now like to consider the third property (H_{k+1} is "near" H_k) from the point of view of the uniqueness of the formula. One obtains this by considering the formula for B_{k+1}^{BFGS} as the solution to an optimization problem. More information can be found in [6] subsection 11.1 and [9], subsection 6.1. 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 3 ([6, Lemma 11.7])**.** *Let be $s \in \mathbb{R}^n$, $y \in \mathbb{R}^n$ with $y \neq 0$ and a symmetric matrix $B \in \mathbb{R}^{n\times n}$ given. Furthermore let $W \in \mathbb{R}^{n\times n}$ be symmetric and positive definit. Then the unique solution of the inverse weighted problem*

$$\begin{aligned} \min_{B_+} \quad & \|W(B_+ - B)W\|_{\mathrm{F}}^2 \\ s.t. \quad & B_+ = B_+^{\mathrm{T}}, \qquad\qquad B_+ y = s \end{aligned} \tag{1.19}$$

is given by

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

In order to be able to convert this to the BFGS formula one is looking for the so-called weighting matrix W:

Theorem 8 ([6, Theorem 11.8])**.** *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 (1.19) with the weighted W is given by*

$$B_+^{BFGS} = B + \frac{(s - By)s^{\mathrm{T}} + s(s - By)^{\mathrm{T}}}{y^{\mathrm{T}}s} - \frac{(s - By)^{\mathrm{T}}yss^{\mathrm{T}}}{(y^{\mathrm{T}}s)^2}. \tag{1.20}$$

One can choose $Q = W^2 = \bar{G_k}$ and $\bar{G_k}$ is the average Hessian, i.e.

$$\bar{G_k} = \int_0^1 \nabla^2 f(x_k + \tau s_k) d\tau,$$

which is positive definite for a strong convex function. The property

$$y_k = \nabla f(x_{k+1}) - \nabla f(x_k) = \int_0^1 \nabla^2 f(x_k + \tau s_k) s_k d\tau = \bar{G_k} s_k$$

follows from Taylor's formula [12]. With this choice of weighting matrix W, the norm

$$\|A\|_{W^2} = \|WAW\|_{\mathrm{F}}$$

is non-dimensional, which is a desirable property, since we do not wish the solution of (1.19) to depend on the units of the problem [9]. The existence of Q follows from Lemma 11.5. in [6] and since it is symmetric and positiv definite, the existence of a spd matrix W follows from it, see Theorem B.6. in [6]. 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 [13].

The initial approximation B_0^{BFGS} must still be discussed. Unfortunately, there is no perfect strategy for this yet. One possibility is to use information about the problem and approximate the Hessian inverse by finite differences at x_0. One could also use a multiple of the identity matrix βI, where β is a scaling factor for the variables. But to determine this factor is problematic. 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. A quite effective heuristic is to scale the starting matrix after the first step, but before the first BFGS update is performed. The provisional value $B_0^{BFGS} = I$ is changed 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 $\nabla^2 f(x_0)^{-1}$ [9].

1.4 Local Convergence of The BFGS-Method

In this subsection a local BFGS method is introduced. 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":

The algorithm can be formulated with the approximation of the actual Hessian H_k^{BFGS},

Algorithm 4 Inverse Local BFGS-Method

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

but that would increase the effort again to $\mathcal{O}(n^3)$ (for solving a linear system or matrix–matrix operations), which is not desirable [9]. It is noticeable that the step size in this algorithm is constant, namely $\alpha_k = 1$ for all $k \in \mathbb{N}_0$, throughout this subsection. In the following we assume that the algorithm does not abort after a finite number of steps, in particular $\nabla f(x_k) \neq 0$ holds for all $k \in \mathbb{N}$.

The convergence analysis of the BFGS-method is very complex and would go beyond the scope of this work. Nevertheless, some statements about the convergence rate should be mentioned. The derivation of these results can be found in [6], subsection 11.3.

We will investigate the convergence rate in the following. We start with a lemma that contains a representation of the error.

Lemma 4 ([6, Lemma 11.23])**.** *Let $B_k^{BFGS}, A \in \mathbb{R}^{n \times n}$ be symmetric matrices, $W \in \mathbb{R}^{n \times n}$ symmetric and regular and $s_k, y_k \in \mathbb{R}^n$ with $s_k^{\mathrm{T}} y_k > 0$ given. Then the following holds:*

$$E_{k+1} = P_k^{\mathrm{T}} E_k P_k + \frac{W(s_k - Ay_k)(Ws_k)^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k} + \frac{Ws_k(s_k - Ay_k)^{\mathrm{T}} W P_k}{s_k^{\mathrm{T}} y_k}$$

where

$$P_k = I - \frac{(W^{-1} y_k)(W s_k)^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k},$$

$$E_k = W(B_k^{BFGS} - A)W$$

and

$$E_{k+1} = W(B_{k+1}^{BFGS} - A)W.$$

Later, the Hessian inverse $\nabla^2 f(x^*)^{-1}$ at the minimum x^* is used for the Matrix A. This lemma is used in the following to estimate the error $\|W(B_{k+1}^{BFGS} - A)W\|_{\mathrm{F}}$ in the $(k+1)$-th iteration approximately by the error $\|W(B_k^{BFGS} - A)W\|_{\mathrm{F}}$ in the k-th iteration upwards. Finding a suitable estimation is indeed the essential part of the convergence analysis and will require a number of technical results. Note, that we are using a Frobenius norm weighted with a matrix $W \in \mathbb{R}^{n \times n}$ here. Such a weighting matrix W also appeared in

the derivation of the inverse BFGS formula in Theorem 8.
After a lot of preparation, an estimation for

$$\|E_{k+1}\|_{\mathrm{F}} = \|W(B_{k+1}^{BFGS} - A)W\|_{\mathrm{F}}$$

can be given. This will not be discussed further. The following requirements are important for this estimation to work: $B_k^{BFGS}, A \in \mathbb{R}^{n\times n}$ symmetric matrices, $W \in \mathbb{R}^{n\times n}$ symmetric and regular and $s_k, y_k \in \mathbb{R}^n$ with $y_k \neq 0$ and

$$\|Ws_k - W^{-1}y_k\| \leq \beta\|W^{-1}y_k\| \tag{1.21}$$

for a given $\beta \in [0, \frac{1}{3}]$. The next step would be then to show that the condition (1.21) is always fulfilled in an area of a "solution" x^* of the minimization problem if the matrix W is chosen suitably. W is then chosen as the symetric and positve definite square root of the matrix $\nabla^2 f(x^*)$ where $f\colon \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable, $\nabla^2 f$ is locally Lipschitz continuous, $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite.
Using all these results obtained so far, one can prove the estimation for the (weighted) error $\|W(B_{k+1}^{BFGS} - \nabla^2 f(x^*)^{-1})W\|_{\mathrm{F}}$ of the inverse BFGS matrices. This estimation is very technical and can be looked up in [6]. The most important thing is that it exists, as it is needed to prove linear convergence:

Theorem 9 ([6, Theorem 11.30])**.** *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, such that the inverse local BFGS-method (Algorithm 4) is well defined for all $x_0 \in \mathbb{R}^n$ with $\|x_0 - x^*\| < \epsilon$ and for all symmetrical and positive definite $B_0^{BFGS} \in \mathbb{R}^{n\times n}$ with $\|B_0^{BFGS} - \nabla^2 f(x^*)^{-1}\|_{\mathrm{F}} < \delta$ and produces a sequence $\{x_k\}_k$ which converges linear to x^*.*

The proof can be looked up in [6]. Nevertheless, the following things should be noted, which are shown there:

- The sequence $\{\|W(B_k^{BFGS} - \nabla^2 f(x^*)^{-1})W\|_{\mathrm{F}}\}_k$ (where $W = \nabla^2 f(x^*)^{\frac{1}{2}}$) remains restricted.
- All matrices B_k^{BFGS} are regular with $\|(B_k^{BFGS})^{-1}\| \leq c$ for all $k \in \mathbb{N}$ and for a suitable constant $c > 0$.

The proof shows that the latter implies in particular that we can assume that $y_k \neq 0$ holds for all $k \in \mathbb{N}$. For the constant is used: $c = \frac{\sigma}{1-r}$ where $\sigma \geq \|\nabla^2 f(x^*)\|$ and $r \in (0, 1)$ are arbitrarily given.
Next, we show that the sequences $\{x_k\}_k$ and $\{B_k^{BFGS}\}_k$ generated by the Algorithm 4 satisfy a kind of "dual" Dennis-Moré condition which we know is sufficient for superlinear convergence. This condition is then needed to show the superlinear convergence of the BFGS-method.

Lemma 5 ([6, Lemma 11.32])**.** *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 (Algorithm 4) with*

$$\sum_{k=0}^{\infty} \|x_k - x^*\| < \infty \tag{1.22}$$

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

$$\rho_k = \|W(B_k^{BFGS} - \nabla^2 f(x^*)^{-1})W\|_{\mathrm{F}} \tag{1.23}$$

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

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

holds.

In the proof it is actually shown:

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

which is nothing other than the “dual” Dennis-Moré condition due to the equivalence of all norms in $\mathbb{R}^n$

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

Note that the rule $\|x\|_W = \|Wx\|$ defines a vector norm in $\mathbb{R}^n$ for each regular W. Using this statement the following theorem can be proved:

Theorem 10 ([6, Theorem 11.33])**.** *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, such that the inverse local BFGS-method (Algorithm 4) is well defined for all $x_0 \in \mathbb{R}^n$ with $\|x_0 - x^*\| < \epsilon$ and for all symmetrical and positive definite $B_0^{BFGS} \in \mathbb{R}^{n\times n}$ with $\|B_0^{BFGS} - \nabla^2 f(x^*)^{-1}\|_{\mathrm{F}} < \delta$ and produces a sequence $\{x_k\}_k$ which converges superlinear to x^*.*

In the proof is also shown: Whenever the BFGS method generates a sequence $\{x_k\}_k$ under the assumptions of the Theorem 10, for which (1.22) holds and the sequence $\{\rho_k\}_k$ generated by (1.23) remains restricted, then the sequence $\{x_k\}_k$ already converges superlinear against x^*. The proof of the Theorem 10 uses at two places explicitly the linear convergence of the sequence $\{x_k\}_k$ already known from the Theorem 9, but in both cases the linear convergence can be replaced by the weaker condition (1.22).
Very often authors use this condition to prove superlinear convergence and formulate a much easier to remember statement. Of course, we also look at this statement:

Theorem 11 ([12, Theorem 5.4.16]). *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable and $\nabla^2 f$ be Lipschitz continuous at x^*. Suppose that the sequence $\{x_k\}_k$ generated by the inverse local BFGS-method (Algorithm 4) converges to a minimizer x^* and that the condition (1.22) holds. Then $\{x_k\}_k$ converges to x^* at a superlinear rate.*

In conclusion, it can be said that under certain conditions the method converges locally superlinear with constant step size $\alpha_k = 1$. This is remarkable, considering that only information about the first derivative is used, and is advantageous in practice.

In this subsection a local BFGS method is introduced. For this we define certain parameters of the method more detailed, so that we get a unique algorithm:

Algorithm 1. *With the following changes of the General Quasi-Newton Method (Algorithm 3) we get the Inverse Local BFGS-Method:*

- *Let $B_0^{BFGS} \in \mathbb{R}^{n\times n}$ be symmetric and positive definite.*
- $\alpha_k = 1$ *for all* $k \in \mathbb{N}_0$.
- *The inverse BFGS formula (1.18) is used.*

We call it "Inverse Local BFGS-Method" because we 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. 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 [9]. It is noticeable that the step size in this algorithm is constant, this holds throughout this chapter; otherwise the statements made here would not hold. In the following we assume that the algorithm does not abort after a finite number of steps, in particular $\nabla f(x_k) \neq 0$ should hold for all $k \in \mathbb{N}$.

1.5 A Globalized BFGS-Method

Now the global variant of the BFGS-Method is studied. One will immediately notice that not much is changing. In the globalized method, it must be ensured that the curvature condition (1.12) is fulfilled. It is therefore important that a step size strategy is chosen to ensure that. This means that now the determination of the step size gains importance, of which was assumed for the local BFGS-Method (Algorithm 4) that it is equal to 1 in every iteration. The globalization of the BFGS-Method is similar to the globalization of Newtons method. But in contrast to it, not the Armijo rule is chosen here, but the Wolfe-Powell step size strategy, wich ensures $s_k^{\mathrm{T}} y_k > 0$ for all $k \in \mathbb{N}$ and was presented in [10]. This line search is called inexact line search, or approximate line search, or acceptable line search. The step size $\alpha_k > 0$ is choosen such that the objective function has an acceptable descent amount, i.e., such that the descent $f(x_k) - f(x_k + \alpha_k d_k) > 0$ is acceptable by users. In contrast to this, there is the so called exact line search or optimal line search where a $\alpha_k > 0$ is choosen such that the objective function in the direction d_k is minimized, i.e., $f(x_k+\alpha_k d_k) = \min_{\alpha>0} f(x_k+\alpha d_k)$ [12]. Since, in practical computation, theoretically an exact optimal step size generally cannot be found, and it is also expensive to find almost an exact step size, therefore the inexact line search with less computation load is highly popular [12]. Also the convergence rate does not depend on the exact line search. Therefore, as long as there is an acceptable steplength rule which ensures that the objective function has sufficient descent, the exact line search can be avoided and the computing efforts will be decreased greatly [12].

Algorithm 5 Inverse Global BFGS-Method

```
$x_0 \in \mathbb{R}^n,\ B_0^{BFGS} \in \mathbb{R}^{n\times n}$ spd, $0 \le \epsilon < 1,\ \sigma \in (0, \frac{1}{2}),\ \rho \in (\sigma, 1),$ 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 Powell-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 $B_{k+1}^{BFGS} = B_k^{BFGS} + \frac{(s_k - B_k^{BFGS} y_k)s_k^{\mathrm{T}} + s_k(s_k - B_k^{BFGS} y_k)^{\mathrm{T}}}{s_k^{\mathrm{T}} y_k} - \frac{(s_k - B_k^{BFGS} y_k)^{\mathrm{T}} y_k s_k s_k^{\mathrm{T}}}{(s_k^{\mathrm{T}} y_k)^2}$.
    Set $k = k + 1$.
end while
return $x_k$
```

As in the previous chapter it is assumed that the algorithm does not abort after a finite number of steps, in particular $\nabla f(x_k) \neq 0$ should hold for all $k \in \mathbb{N}$. One could also use the direct approximation H_k^{BFGS} here. But this would increase the cost again and is therefore omitted.

First it is shown that this method is well-defined. This is summarised in the following:

Theorem 12 ([6, Theorem 11.37])**.** *Be $f\colon \mathbb{R}^n \to \mathbb{R}$ continuously differentiable and bounded below. Then for the globalized BFGS-Method (Algorithm 5):*

1. *$s_k^{\mathrm{T}} y_k > 0$ for all $k \in \mathbb{N}$.*
2. *The matrices H_{k+1}^{BFGS} are symmetric and positive definite for all $k \in \mathbb{N}$.*
3. *The method is well defined.*

So it turns out that the Wolfe-Powell step size rule is crucial. In (1.13) is shown, that it ensures the curvature condition $s_k^{\mathrm{T}} y_k > 0$. This in turn ensures that the matrices B_k^{BFGS} (or H_k^{BFGS}) remain symmetric and positive definite, see theorem 7.

Now the local and global convergence of this Method is studied. It is desirable that each limit point of a sequence $\{x_k\}_k$ generated by the globalized BFGS-Method (Algorithm 5) is a stationary point of f and that we get locally superlinear convergence. Unfortunately, neither of these statements is in general true.
Little is known about global convergence for nonconvex minimization problems. Indeed, so far, no one has proven global convergence of the globalized BFGS-Method (Algorithm 5) for nonconvex minimization problems or has given a counter example that shows nonconvergence of this method. Whether it converges globally for a nonconvex function remains unanswered. This open problem has been mentioned many times and is currently regarded as one of the most fundamental open problems in the theory of quasi-Newton methods [8].

It can be shown that the inverse global BFGS-method (Algorithm 5) converges globally towards the unique determined minimum x^* of a twice continuously differentiable and uniformly convex objective function f, with arbitrary start vector $x_0 \in \mathbb{R}^n$ and arbitrary choice of the symmetrical and positive definite start matrix $B_0^{BFGS} \in \mathbb{R}^{n\times n}$. This is a very strong convergence result for the BFGS-Method, and it is currently not known whether this also applies to the DFP method.
The first basic statements on this appeared in [10]. These are presented, the proofs can be looked up in the original source.

Lemma 6 ([10, Lemma 3.1])**.** *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a convex function that is twice continuously differentiable, and whose second derivatives are bounded by the inequality*

$$\|\nabla^2 f(x)\| \leq M \tag{1.24}$$

for all x in the level set $\mathcal{L}(x_0) = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$. Then the inequality

$$\frac{\|y_k\|^2}{s_k^{\mathrm{T}} y_k} \leq M \tag{1.25}$$

is satisfied.

Lemma 7 ([10, Lemma 3.2])**.** *If the globalized BFGS-Method (Algorithm 5) is applied to a differentiable function $f\colon \mathbb{R}^n \to \mathbb{R}$ that is bounded below, if $x_0 \in \mathbb{R}^n$ is any starting point and $B_0^{BFGS} \in \mathbb{R}^{n\times n}$ is any positive definite matrix, and if inequality (1.25) is satisfied for all $k \in \mathbb{N}_0$, where M is a constant, then the limit*

$$\liminf_{k\to\infty} \|\nabla f(x_k)\| = 0 \tag{1.26}$$

is obtained.

Combining the statements from the previous two lemmas leads to the highlight of this subsection, the mentioned global convergence of twice continuously differentiable and uniformly convex functions. Powell showed this first in [10]. Here a modified version from [6] is presented, since it fits better to to the previous theory. The proof of this statement requires much preparation and is technical, which is why only the theorem is mentioned here.

Theorem 13 ([6, Theorem 11.42]). *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable, the level set $\mathcal{L}(x_0) = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$ convex and f uniformly convex on $\mathcal{L}(x_0)$. Let $\{x_k\}_k$ be a sequence generated by the globalized BFGS-Method (Algorithm 5) with arbitrary starting point $x_0 \in \mathbb{R}^n$ and arbitrary (symmetric and positive definite) starting matrix $B_0^{BFGS} \in \mathbb{R}^{n \times n}$. Then the whole sequence $\{x_k\}_k$ converges towards the uniquely determined minimum x^* of f.*

The result is, that the BFGS-Method with the Wolfe-Powell step size strategy applied to a smooth convex function f from an arbitrary starting point $x_0 \in \mathbb{R}^n$ and from any initial Hessian approximation $B_0^{BFGS} \in \mathbb{R}^{n \times n}$, that is symmetric and positive definite, is globally convergent. From the very beginning it was assumed that our objective function is twice continuously differentiable, so this assumption does not cause any additional excitement. But the convexity is new here, which is why it should be studied more closely. We assume for this theorem that the objective f is uniformly convex on the convex level set $\mathcal{L}(x_0)$. This means, there exist positive constants $m, M > 0$ such that

$$m\|z\|^2 \leq z^{\mathrm{T}} \nabla^2 f(x) z \leq M\|z\|^2$$

for all $z \in \mathbb{R}^n$ and all $x \in \mathcal{L}(x_0)$. This implies that $\nabla^2 f(x)$ is positive definite on $\mathcal{L}(x_0)$ and as a consequence has f an unique minimizer $x^* \in \mathcal{L}(x_0)$ [9].

One can even show that the globalized BFGS-Method for twice continuously differentiable and uniformly convex functions also converges locally superlinear, still with arbitrary start vector $x_0 \in \mathbb{R}^n$ and arbitrary symetric and positve definite start matrix $B_0^{BFGS} \in \mathbb{R}^{n \times n}$. For this, however, it must be assumed that $\nabla^2 f$ is locally Lipschitz-continuous and that the step size $\alpha_k = 1$ is always taken, provided it satisfies the Wolfe-Powell conditions. It can be shown that this is always the case locally, so that the globalized BFGS-Method does not need a step size strategy locally. The trick to prove superlinear convergence now is to show that $\{x_k\}_k$ already converges sufficiently fast against x^* so that 1.22 holds. An approach to proof and references to other useful sources can be found in: Geiger, Kanzow, “Numerische Verfahren zur Lösung unrestringierter Optimierungsaufgaben” (1999).

In general, it is not possible to show that the condition of the matrix sequence $\{B_k^{BFGS}\}_k$ is bounded.

Does the BFGS-Method with the Wolfe line search converge globally for general functions? Recent studies provide a negative answer to the convergence problem of the BFGS-Method for nonconvex functions [2].

In addition, when inexact line search (2.5.3) and (2.5.7) are used, BFGS-Method is globally convergent. [12]

It can be shown that the globalized BFGS-Method (Algorithm 5) not only converges globally by using the Powell step size rule but also by using a great number of inexact, efficient step size strategies used in practice for uniformly convex objective functions, which can be seen as an indication of the numerical stability of this method [14]. Finally, the rate of convergence is discussed. For this, further conditions must be made, with respect to the step size, so that the Dennis-Moré conditions are fulfilled.

Theorem 14 ([14, Satz 3.5])**.** *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable, the level set $\mathcal{L}(x_0) = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$ compact and convex, f uniformly convex on $\mathcal{L}(x_0)$. Let ∇f be Lipschitz continuous on $\mathcal{L}(x_0)$ and $\nabla^2 f$ be Lipschitz continuous at x^*.Let $\lim_{k\to\infty} \alpha_k = 1$ hold. Then the sequence $\{x_k\}_k$ converges superlinear against x^*.*

Interessant sind jetzt nattirlich Aussagen dartiber, ftir welche effizienten Schrittweitenfunktionen auf lim tk= 1 geschlossen werden kann. Diese Aussage ist richtig (s. Dennis u. More) ftir Die Powell-Schrittweitenfunktion T, falls fl E (0, 89 Es ist sogar le T e (x k, pk) ftir alle hinreichend grogen k

1.6 Limited-Memory BFGS-Method

It should also be computationally challenging to update B

One of the disadvantages of the Quasi-Newton methods is that a $n \times n$ matrix (namely B_{k+1}^{BFGS}) must be stored in each iteration. Even when using the symmetry of this matrix, a memory requirement of $n(n+1)/2$ matrix entries remains. For large-scale optimization problems is this far too much [6].

Zu wertend, besser: not feasible

Die Quellenangabe muss genauer

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 [9].

Ist das vorher eingeführt, was das heisst? Ist ja sehr informell

Der Satz ist auch etwas informell

Due to the outstanding importance of the BFGS-method in the class of quasi-Newton methods [6], 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 [9]. In subsection 1.3 three different formulae for the approximation of the Hessian inverse were introduced, see (1.18). ~~The last one is now needed~~

Ref zu ungenau

Auch hier ist ein ganzes Buch zitieren zu ungenau. Und: Welche?

We start from the last one

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

For given vectors $s_k, y_k \in \mathbb{R}^n$ with $s_k^{\mathrm{T}} y_k > 0$ one sets

$$\rho_k = \frac{1}{s_k^{\mathrm{T}} y_k}, \; V_k = I - \rho_k y_k s_k^{\mathrm{T}}, \tag{1.27}$$

obtaining

$$B_{k+1}^{BFGS} = V_k^{\mathrm{T}} B_k^{BFGS} V_k + \rho_k s_k s_k^{\mathrm{T}}. \tag{1.28}$$

Bitte auch hier genauer

The matrix B_{k+1}^{BFGS} is obtained by updating B_k^{BFGS} using the pair $\{s_k, y_k\}$ [12]. 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, one stores a modified version of B_k^{BFGS} implicitly, by storing a certain number (say, m) of the vector pairs $\{s_i, y_i\}$ [9]. After the new iterate is x_{k+1} computed, the oldest vector pair in the set $\{s_i, y_i\}_{i=k-m}^{k-1}$ (namely $\{s_{k-m}, y_{k-m}\}$) 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) often produce satisfactory results. The strategy of keeping the m most recent pairs $\{s_i, y_i\}_{i=k-m}^{k-1}$ works well in practice. Indeed no other strategy has yet proved to be consistently better [9]. Normally, for large-scale problems, one takes $m \ll n$. In practice, the choice of m depends on the dimension of the problem and the storage of the employed computer [12].

Aber nicht an der Stelle, sondern am anderen Ende der Folge?

S.o.,

Kein Satz.

Describing the updating process more in detail: at iteration k, the current iterate is x_k and the set of vector pairs is given by $\{s_i, y_i\}_{i=k-m}^{k-1}$. At first some initial Hessian approximation $B_k^{(0)}$ is choosen for the k-th iteration (in contrast to the standard BFGS iteration, this initial approximation is allowed to vary from iteration to iteration). The formula (1.28) is applied m times repeatedly, i.e.

$$B_k^{(j+1)} = V_{k-m+j}^{\mathrm{T}} B_k^{(j)} V_{k-m+j} + \rho_{k-m+j} s_{k-m+j} s_{k-m+j}^{\mathrm{T}}, \; j = 0, 1, \cdots, m-1. \tag{1.29}$$

The L-BFGS approximation, called B_k^{L-BFGS}, reads the following:

$$\begin{aligned}
B_k^{L-BFGS} = B_k^{(m)} =& V_{k-1}^{\mathrm{T}} B_k^{(m-1)} V_{k-1} + \rho_{k-1} s_{k-1} s_{k-1}^{\mathrm{T}} = \\
=& \cdots = \\
=& (V_{k-1}^{\mathrm{T}} \cdots V_{k-m}^{\mathrm{T}}) B_k^{(0)} (V_{k-m} V_{k-m+1} \cdots V_{k-1}) + \\
& + \rho_{k-m} (V_{k-1}^{\mathrm{T}} \cdots V_{k-m+1}^{\mathrm{T}}) s_{k-m} s_{k-m}^{\mathrm{T}} (V_{k-m+1} \cdots V_{k-1}) \\
& + \rho_{k-m+1} (V_{k-1}^{\mathrm{T}} \cdots V_{k-m+2}^{\mathrm{T}}) s_{k-m+1} s_{k-m+1}^{\mathrm{T}} (V_{k-m+2} \cdots V_{k-1}) \\
& + \cdots \\
& + \rho_{k-1} s_{k-1} s_{k-1}^{\mathrm{T}}.
\end{aligned}$$

That means B_k^{L-BFGS} can be calculated completely from $B_k^{(0)}$ and the vector pairs

$\{s_i, y_i\}_{i=k-m}^{k-1}$. B_k^{L-BFGS} must be considered as an approximation of B_k^{BFGS}. Nevertheless this matrix fulfils the Quasi-Newton equation (1.8) [6].
In fact, there is no need to compute and save B_k^{L-BFGS} explicitly. Instead, one only saves the pairs $\{s_i, y_i\}_{i=k-m}^{k-1}$ and computes $B_k^{L-BFGS} g_k = B_k^{L-BFGS} \nabla f(x_k)$ [12]. The product can be obtained by performing a sequence of inner products and vector summations involving g_k and the pairs $\{s_i, y_i\}_{i=k-m}^{k-1}$ [9]. So, we have

$$\begin{aligned} B_k^{L-BFGS} g_k = {} & (V_{k-1}^{\mathrm{T}} \cdots V_{k-m}^{\mathrm{T}}) B_k^{(0)} (V_{k-m} V_{k-m+1} \cdots V_{k-1}) g_k + \\ & + \rho_{k-m} (V_{k-1}^{\mathrm{T}} \cdots V_{k-m+1}^{\mathrm{T}}) s_{k-m} s_{k-m}^{\mathrm{T}} (V_{k-m+1} \cdots V_{k-1}) g_k \\ & + \rho_{k-m+1} (V_{k-1}^{\mathrm{T}} \cdots V_{k-m+2}^{\mathrm{T}}) s_{k-m+1} s_{k-m+1}^{\mathrm{T}} (V_{k-m+2} \cdots V_{k-1}) g_k \\ & + \cdots \\ & + \rho_{k-1} s_{k-1} s_{k-1}^{\mathrm{T}} g_k. \end{aligned}$$

✓

Since $V_i g_k = (I - \rho_i y_i s_i^{\mathrm{T}}) g_k$ for $i = k-1, \cdots, k-m$, one can derive a recursive method to compute the product efficiently, see the L-BFGS two-loop recursion for $B_k^{L-BFGS} g_k$, Algorithm 6 [12]. Welcher ist es da?

Algorithm 6 L-BFGS two-loop recursion for $B_k^{L-BFGS} g_k$

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

$q_{k-1} = g_k$
$\alpha_i = \rho_i s_i q_i$
$q_{i+1} = q_i - \alpha_i y_i$

Sieht evtl. so besser aus

Dieser Teil, , dass das für kleine m deutlich weniger als n^2 Multiplikationen sind, sollte hier deutlicher hervorgehoben werden... und zu Beginn des Abschnitts auch genannt werden.

Without considering the multiplication $B_k^{(0)} q$, the L-BFGS two-loop recursion requires $4mn$ multiplications. If $B_k^{(0)}$ is diagonal, then n additional multiplications are needed. Apart from being inexpensive, this recursion has the advantage that the multiplication by the initial matrix $B_k^{(0)}$ is isolated from the rest of the computations, allowing this matrix to be chosen freely and to vary between iterations. One may even use an implicit choice of $B_k^{(0)}$ by defining some initial approximation $H_k^{(0)}$ to the Hessian (not its inverse) and obtaining r by solving the system $H_k^{(0)} r = q$ [9].
$B_k^{(0)}$ can be an arbitrarily, symmetrical and positive definite matrix. In general $B_k^{(0)}$ will be a multiple of the identity matrix, so that it can be stored very easily [6]. A method

for choosing $B_k^{(0)}$ that has proven effective in practice is to set $B_k^{(0)} = \gamma_k I$, where

$$\gamma_k = \frac{s_{k-1}^{\mathrm{T}} y_{k-1}}{y_{k-1}^{\mathrm{T}} y_{k-1}}. \tag{1.30}$$

γ_k is the scaling factor that attempts to estimate the size of the true Hessian matrix along the most recent search direction. This choice helps to ensure that the search direction d_k is well scaled, and as a result the step length $\alpha_k = 1$ is accepted in most iterations. It is important that the line search is based on the (strict) Powell-Wolfe conditions, so that the BFGS updating is stable [9].
With the previous theoery, the following algorithm can be created for the L-BFGS-method:

Algorithm 7 L-BFGS-Method

```
$x_0 \in \mathbb{R}^n,\ B_0^{L-BFGS} \in \mathbb{R}^{n \times n}$ spd, $0 \le \epsilon < 1,\ \sigma \in (0, \frac{1}{2}),\ \rho \in (\sigma, 1),\ m \in \mathbb{N}$, set $k = 0$
while $\|\nabla f(x_k)\| > \epsilon$ do
    Choose $B_k^{(0)}$ (e.g. $B_k^{(0)} = \gamma_k I$ with $\gamma$ from (1.30)). $B^{L-BFGS}$ from
    Compute $d_k = -B_k^{L-BFGS} \nabla f(x_k)$ with Algorithm 6.
    Find a step length $\alpha_k = \alpha(\sigma, \rho)$ that satisfies the (strict) Powell-Wolfe conditions.   Formel? Ref.
    Set $x_{k+1} = x_k + \alpha_k d_k$.
    if $k > m$ then
        Discard the vector pairs $\{s_{k-m}, y_{k-m}\}$ from storage, mention that the iters are shifted!
        Compute and save $s_k = x_{k+1} - x_k,\ y_k = g_{k+1} - g_k$.
    end if                 Oh, wo kommen die Differenzen her!
    Set $k = k + 1$        In diesem Abschnitt noch nicht genannt.
end while
return $x_k$
```

Zu informell, streichen

It's easy to notice: U unlike the conventional quasi-Newton methods, this one follows a different algorithmic sequence. Here, first the approximation of the Hessian inverse B_k^{L-BFGS} is calculated for the current iteration x_k and then the next iteration x_{k+1} is calculated. For example with the (globalized) BFGS-method, the new iteration x_{k+1} is calculated first and then the approximation of the Hessian inverse for the new iteration B_{k+1}^{BFGS} is calculated. This is the core idea of this method. Instead of passing the completely calculated matrix, only the vector pairs $\{s_i, y_i\}_{i=k-m}^{k-1}$ are passed in each iteration and at the beginning the approximation B_k^{L-BFGS} is created from these. The matrices B_k^{L-BFGS} are not explicitly stored, but only the vector pairs needed for the calculation and the start matrix $B_k^{(0)}$ and, if necessary, the real numbers $\{\rho_j\}_{j=k-m}^{k-1}$. For small values of m and larger dimensions n, the memory requirement for the L-BFGS-method is thus considerably lower than for the (globalized) BFGS-method itself, namely $\mathcal{O}(mn)$ instead of $\mathcal{O}(n^2)$, which is due to the fact that the step size d_k can be obtained with $\mathcal{O}(mn)$ operations [6]. Zu ungenaue Ref.

Wann ist das nicht notwendig?!

sonst unklar, wieso da ein if nec. steht.

In practical applications of the L-BFGS-method, the strict Powell-Wolfe stepsize strategy is ~~usually~~ used. This is because the L-BFGS-method seems to depend more on the choice of a “good” stepsize $\alpha_k > 0$ than, for example, the (globalized) BFGS-method, and because the “optimal” stepsize can be better approximated by means of the strict Powell-Wolfe rule [6].
During its first $m-1$ iterations, the L-BFGS-method (Algorithm 7) is equivalent to the inverse global BFGS-method (Algorithm 5) if the initial matrix is the same in both methods ($B_0^{L-BFGS} = B_0^{BFGS}$), and if L-BFGS chooses $B_k^{(0)} = B_0^{L-BFGS}$ at each iteration [9]. ~~In the first few iterations, you can only use the vector pairs already existing until then, and these can be less than m.~~
Before discussing the convergence properties, it must first be ensured that the L-BFGS-method (Algorithm 7) is well defined:

Theorem 15 ([6, Note 12.3])**.** *If $f\colon \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and bounded below, then $s_k^{\mathrm{T}} y_k > 0$ holds for the sequences $\{s_k\}_k$ and $\{y_k\}_k$ generated by the L-BFGS-method (Algorithm 7). Furthermore, the sequence of the matrices $\{B_k^{L-BFGS}\}_k$ is symmetric and positive definite and the L-BFGS-method (Algorithm 7) is well defined.*

Wie ist eine Sequenz symmetrisch und pos def? Genauer formulieren.

The following statement can be made about global convergence:

Theorem 16 ([12, Theorem 5.7.4])**.** *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a twice continuously differentiable and uniformly convex function. Then the iterative sequence $\{x_k\}_k$ generated by the L-BFGS-method (Algorithm 7) converges to the unique minimizer x^* of f.*

Thus sequences $\{x_k\}_k$ generated by the L-BFGS-method, like by the inverse global BFGS-method, converge globally for twice continuously differentiable, uniformly convex functions. The following can be said about their rate of convergence:

Theorem 17 ([12, Theorem 5.7.7])**.** *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a twice continuously differentiable and uniformly convex function. Assume that the iterative sequence $\{x_k\}_k$ generated by the L-BFGS-method (Algorithm 7) converges to the unique minimizer x^* of f. Then the rate of convergence is at least R-linear.*

This theorem indicates that the L-BFGS-method often converges slowly, which leads to a relatively large number of function evaluations. Also, it is inefficient on highly ill-conditioned optimization problems. Though there are some weaknesses, L-BFGS-method is a main choice for large-scale problems in which the true Hessian is not sparse, because, in this case, it may outperform other rival algorithms [12]. Ref zu ungenau.

The memoryless BFGS-method should also be mentioned. Here, $B_k^{BFGS} = I$ is inserted into the formula (1.28). It reads

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

This formula satisfies the quasi-Newton equation (1.8), is positive definite and is called the memoryless BFGS formula. Obviously, if $m = 1$ and $B_k^{(0)} = I$ for all $k \in \mathbb{N}$, the limited-memory BFGS-method is just the memoryless BFGS-method. The idea is to use only the information from the previous iteration [12].

2 Riemannian Manifolds

3 The BFGS-Method For Riemannian Manifolds

3.1 Preliminaries

3.2 Quasi-Newton Methods For Riemannian Manifolds

3.3 The BFGS Formula For Riemannian Manifolds

3.4 A Local BFGS-Method On Riemannian Manifolds

3.5 A Globalized BFGS-Method On Riemannian Manifolds

3.6 Limited-Memory BFGS-Method On Riemannian Manifolds

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] C. G. Broyden. “Quasi-Newton Methods and their Application to Function Minimisation”. In: *Mathematics of Computation* 21.99 (1967), pp. 368–381.

[2] Yu-Hong Dai. “A Perfect Example for The BFGS Method”. In: ().

[3] William Cooper Davidon. “VARIABLE METRIC METHOD FOR MINIMIZATION”. In: (1959). DOI: 10.2172/4252678.

[4] R. Fletcher. “A new approach to variable metric algorithms”. In: *The Computer Journal* 13.3 (Jan. 1970), pp. 317–322. DOI: 10.1093/comjnl/13.3.317. eprint: https://academic.oup.com/comjnl/article-pdf/13/3/317/988678/130317.pdf. URL: https://doi.org/10.1093/comjnl/13.3.317.

[5] R. Fletcher and M. J. D. Powell. “A Rapidly Convergent Descent Method for Minimization”. In: (1963). DOI: 10.1093/comjnl/6.2.163.

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

[7] Donald Goldfarb. “A family of variable-metric methods derived by variational means”. In: 1970.

[8] DONG-HUI LI and MASAO FUKUSHIMA. “ON THE GLOBAL CONVERGENCE OF THE BFGS METHOD FOR NONCONVEX UNCONSTRAINED OPTIMIZATION PROBLEMS”. In: ().

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

[10] Michael David Powell. “Some global convergence properties of a variable metric algorithm for minimization without exact lin”. In: 1976.

[11] D. F. Shanno. “Conditioning of Quasi-Newton Methods for Function Minimization”. In: *Mathematics of Computation* 24.111 (1970), pp. 647–656. URL: http://www.jstor.org/stable/2004840.

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

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

[14] J. Werner. “Über die globale Konvergenz von Variable-Metrik-Verfahren mit nichtexakter Schrittweitenbestimmung.” In: *Numerische Mathematik* 31 (1978/79), pp. 321–334. URL: http://eudml.org/doc/132583.