

Optimization for Data Science

ETH Zürich, FS 2023 261-5110-00L

Lecture 8: Newton's Method and Quasi-Newton methods

Bernd Gärtner
Niao He

<https://www.ti.inf.ethz.ch/ew/courses/ODS23/index.html>

April 3, 2023

1-dimensional case: Newton-Raphson method

Goal: find a zero of differentiable
 $f : \mathbb{R} \rightarrow \mathbb{R}$.

Method:

this is based on the definition of derivative

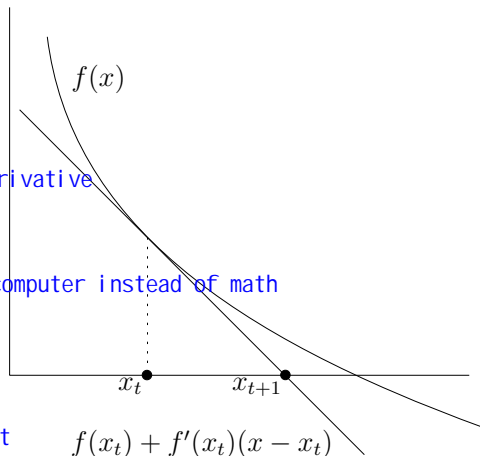
$$x_{t+1} := x_t - \frac{f(x_t)}{f'(x_t)}, \quad t \geq 0.$$

this is the updating method in the computer instead of math

x_{t+1} solves

$$f(x_t) + f'(x_t)(x - x_t) = 0,$$

in this way, we could find the x that
leads to 0 derivative.



The Babylonian method

Computing square roots: find a zero of $f(x) = x^2 - R, R \in \mathbb{R}_+$.

Newton-Raphson step:

$$x_{t+1} = x_t - \frac{f(x_t)}{f'(x_t)} = x_t - \frac{x_t^2 - R}{2x_t} = \frac{1}{2} \left(x_t + \frac{R}{x_t} \right).$$

Starting from $x_0 > 0$, we have

$$x_{t+1} = \frac{1}{2} \left(x_t + \frac{R}{x_t} \right) \geq \frac{x_t}{2}.$$

this is somehow a certain conclusion.

Starting from $x_0 = R \geq 1$, it takes

- ▶ at least $\log(R)/2$ steps to get $x_t < 2\sqrt{R} \dots$ why these steps
- ▶ ... but still only $O(\log R)$ steps to get $x_t - \sqrt{R} < 1/2$ (Exercise 51).

The Babylonian method - Takeoff this part is not very sure.

Suppose $x_0 - \sqrt{R} < 1/2$ (achievable after $O(\log R)$ steps).

$$x_{t+1} - \sqrt{R} = \frac{1}{2} \left(x_t + \frac{R}{x_t} \right) - \sqrt{R} = \frac{x_t}{2} + \frac{R}{2x_t} - \sqrt{R} = \frac{1}{2x_t} (x_t - \sqrt{R})^2.$$

Assume $R \geq 1/4$. Then all iterates have value at least $\sqrt{R} \geq 1/2$. Hence we get

$$x_{t+1} - \sqrt{R} \leq (x_t - \sqrt{R})^2.$$

$$x_T - \sqrt{R} \leq (x_0 - \sqrt{R})^{2^T} < \left(\frac{1}{2}\right)^{2^T}, \quad T \geq 0.$$

To get $x_T - \sqrt{R} < \varepsilon$, we only need $T = \log \log(\frac{1}{\varepsilon})$ steps!

The Babylonian method - Example

$R = 1000$, IEEE 754 double arithmetic

- ▶ 7 steps to get $x_7 - \sqrt{1000} < 1/2$
- ▶ 3 more steps to get x_{10} equal to $\sqrt{1000}$ up to machine precision (53 binary digits).
- ▶ First phase: \approx one more correct digit per iteration
- ▶ Last phase, \approx double the number of correct digits in each iteration!

Once you're close, you're there...

Newton's method for optimization

This part is very important.

1-dimensional case: Find a global minimum x^* of a differentiable convex function $f : \mathbb{R} \rightarrow \mathbb{R}$.

Can equivalently search for a zero of the derivative f' : Apply the Newton-Raphson method to f' . 同样的原理，我们得到的是一系列的一阶导数为0的值

Update step:

$$x_{t+1} := x_t - \frac{f'(x_t)}{f''(x_t)} = x_t - f''(x_t)^{-1} f'(x_t)$$

(needs f twice differentiable).

d -dimensional case: Newton's method for minimizing a convex function $f : \mathbb{R}^d \rightarrow \mathbb{R}$:

$$\mathbf{x}_{t+1} := \mathbf{x}_t - \nabla^2 f(\mathbf{x}_t)^{-1} \nabla f(\mathbf{x}_t)$$

这个里面是向量序列

Newton's method = adaptive gradient descent

General update scheme:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - H(\mathbf{x}_t) \nabla f(\mathbf{x}_t),$$

where $H(\mathbf{x}_t) \in \mathbb{R}^{d \times d}$ is some matrix.

Newton's method: $H(\mathbf{x}_t) = \nabla^2 f(\mathbf{x}_t)^{-1}$.

Gradient descent: $H(\mathbf{x}_t) = \gamma I$.

Newton's method: “adaptive gradient descent”, adaptation is w.r.t. the local geometry of the function at \mathbf{x}_t , as captured by the Hessian $\nabla^2 f(\mathbf{x}_t)$. [Hessian matrix](#)

Convergence in one step on quadratic functions

A **nondegenerate** quadratic function is a function of the form
the quadratic function
is not zero.

原理不变，只是把这些换成了矩阵和向量的形式。

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top M\mathbf{x} - \mathbf{q}^\top \mathbf{x} + c,$$

where $M \in \mathbb{R}^{d \times d}$ is an invertible symmetric matrix, $\mathbf{q} \in \mathbb{R}^d, c \in \mathbb{R}$. Let $\mathbf{x}^* = M^{-1}\mathbf{q}$ be the unique solution of $\nabla f(\mathbf{x}) = \mathbf{0}$.

► \mathbf{x}^* is the unique global minimum if f is convex.

Lemma 8.1

On nondegenerate quadratic functions, with any starting point $\mathbf{x}_0 \in \mathbb{R}^d$, Newton's method yields $\mathbf{x}_1 = \mathbf{x}^*$.

Proof.

We have $\nabla f(\mathbf{x}) = M\mathbf{x} - \mathbf{q}$ (this implies $\mathbf{x}^* = M^{-1}\mathbf{q}$) and $\nabla^2 f(\mathbf{x}) = M$. Hence,

$$\mathbf{x}_1 = \mathbf{x}_0 - \nabla^2 f(\mathbf{x}_0)^{-1} \nabla f(\mathbf{x}_0) = \mathbf{x}_0 - M^{-1}(M\mathbf{x}_0 - \mathbf{q}) = M^{-1}\mathbf{q} = \mathbf{x}^*.$$



Minimizing the second-order Taylor approximation

Alternative interpretation of Newton's method:

Each step minimizes the local **second-order Taylor approximation**.

Lemma 8.3 (Exercise 55) We need to minimize since we want the smallest one.

Let f be convex and twice differentiable at $\mathbf{x}_t \in \text{dom}(f)$, with $\nabla^2 f(\mathbf{x}_t) \succ 0$ being invertible. The vector \mathbf{x}_{t+1} resulting from the Newton step satisfies

$$\mathbf{x}_{t+1} = \underset{\mathbf{x} \in \mathbb{R}^d}{\operatorname{argmin}} f(\mathbf{x}_t) + \nabla f(\mathbf{x}_t)^\top (\mathbf{x} - \mathbf{x}_t) + \frac{1}{2} (\mathbf{x} - \mathbf{x}_t)^\top \nabla^2 f(\mathbf{x}_t) (\mathbf{x} - \mathbf{x}_t).$$

二阶导数的泰勒级数展开, 这个地方不是很清楚, 如何利用泰勒级数进行函数的近似。

A quadratic function coincides with its second order Taylor approximation.

\Rightarrow The lemma yields another proof for convergence of Newton's method in one step, for nondegenerate convex quadratic functions.

Local Convergence (à la Babylonian method)

We will prove: under suitable conditions, and starting close to the global minimum, Newton's method will reach distance at most ε to the minimum within $\log \log(1/\varepsilon)$ steps.

- ▶ much faster than anything we have seen so far. . .
- ▶ . . . but we need to start close to the minimum already.

This is a **local convergence** result.

General **global convergence** results that hold for every starting point are unknown for Newton's method.

Under a stability assumption on the Hessian, global convergence holds [KSJ18].

Local Convergence (à la Babylonian method) II

stationary point where f' is 0.

We will **actually** prove: under suitable conditions, and starting close to a **critical point**, Newton's method will reach distance at most ε to the critical point within $\log \log(1/\varepsilon)$ steps.

Convexity is not needed!

The “suitable conditions” ensure that we are close to only one critical point.

Once you're close, you're there...

Theorem 8.4

Let $f : \text{dom}(f) \rightarrow \mathbb{R}$ be twice differentiable with a critical point \mathbf{x}^* . Suppose there is a ball $X \subseteq \text{dom}(f)$ with center \mathbf{x}^* , s.t.

(i) **Bounded inverse Hessians:** There exists a real number $\mu > 0$ such that

Hessians are bounded.
$$\|\nabla^2 f(\mathbf{x})^{-1}\| \leq \frac{1}{\mu}, \quad \forall \mathbf{x} \in X.$$

(ii) **Lipschitz continuous Hessians:** There exists a real number $B > 0$ such that

$$\|\nabla^2 f(\mathbf{x}) - \nabla^2 f(\mathbf{y})\| \leq B\|\mathbf{x} - \mathbf{y}\| \quad \forall \mathbf{x}, \mathbf{y} \in X.$$

Then, for $\mathbf{x}_t \in X$ and \mathbf{x}_{t+1} resulting from the Newton step, we have

This could be the updating rules.

$$\|\mathbf{x}_{t+1} - \mathbf{x}^*\| \leq \frac{B}{2\mu} \|\mathbf{x}_t - \mathbf{x}^*\|^2.$$

Super-exponentially fast

Corollary 8.5 (Exercise 53)

With the assumptions and terminology of the convergence theorem, and if

We will use this rule in the exam. $\|\mathbf{x}_0 - \mathbf{x}^*\| \leq \frac{\mu}{B},$

then Newton's method yields

$$\|\mathbf{x}_T - \mathbf{x}^*\| \leq \frac{\mu}{B} \left(\frac{1}{2}\right)^{2^T - 1}, \quad T \geq 0.$$

Starting close to the critical point, we will reach distance at most ε to it within $\mathcal{O}(\log \log(1/\varepsilon))$ steps.

Bound as for the last phase of the Babylonian method.

Super-exponentially fast — intuitive reason

Almost constant Hessians within the ball X of interest. . .

. . . so f behaves almost like a quadratic function which has truly constant Hessians and allows Newton's method to converge in one step.

Lemma 8.6 (Exercise 54)

With the assumptions and terminology of the convergence theorem, and if $\mathbf{x}_0 \in X$ satisfies

$$\|\mathbf{x}_0 - \mathbf{x}^*\| \leq \frac{\mu}{B},$$

then the Hessians in Newton's method satisfy the relative error bound

$$\frac{\|\nabla^2 f(\mathbf{x}_t) - \nabla^2 f(\mathbf{x}^*)\|}{\|\nabla^2 f(\mathbf{x}^*)\|} \leq \left(\frac{1}{2}\right)^{2^t - 1}, \quad t \geq 0.$$

Proof of convergence theorem

We abbreviate $H := \nabla^2 f$, $\mathbf{x} = \mathbf{x}_t$, $\mathbf{x}' = \mathbf{x}_{t+1}$. Subtracting \mathbf{x}^* from both sides of the Newton step definition:

$$\begin{aligned}\mathbf{x}' - \mathbf{x}^* &= \mathbf{x} - \mathbf{x}^* - H(\mathbf{x})^{-1} \nabla f(\mathbf{x}) \\ &= \mathbf{x} - \mathbf{x}^* + H(\mathbf{x})^{-1} (\nabla f(\mathbf{x}^*) - \nabla f(\mathbf{x})) \\ &= \mathbf{x} - \mathbf{x}^* + H(\mathbf{x})^{-1} \int_0^1 H(\mathbf{x} + t(\mathbf{x}^* - \mathbf{x})) (\mathbf{x}^* - \mathbf{x}) dt,\end{aligned}$$

using the fundamental theorem of calculus

$$\int_0^1 h'(t) dt = h(1) - h(0)$$

componentwise, with the vector-valued functions

$$\begin{aligned}h(t) &= \nabla f(\mathbf{x} + t(\mathbf{x}^* - \mathbf{x})), \\ h'(t) &= \nabla^2 f(\mathbf{x} + t(\mathbf{x}^* - \mathbf{x})) (\mathbf{x}^* - \mathbf{x}).\end{aligned}$$

Note that h' is continuous by Lipschitz continuity of the Hessian.

Proof of convergence theorem, II

So far we have

$$\mathbf{x}' - \mathbf{x}^* = \mathbf{x} - \mathbf{x}^* + H(\mathbf{x})^{-1} \int_0^1 H(\mathbf{x} + t(\mathbf{x}^* - \mathbf{x}))(\mathbf{x}^* - \mathbf{x}) dt.$$

With

$$\mathbf{x} - \mathbf{x}^* = H(\mathbf{x})^{-1} H(\mathbf{x})(\mathbf{x} - \mathbf{x}^*) = H(\mathbf{x})^{-1} \int_0^1 -H(\mathbf{x})(\mathbf{x}^* - \mathbf{x}) dt,$$

we further get

$$\mathbf{x}' - \mathbf{x}^* = H(\mathbf{x})^{-1} \int_0^1 (H(\mathbf{x} + t(\mathbf{x}^* - \mathbf{x})) - H(\mathbf{x}))(\mathbf{x}^* - \mathbf{x}) dt.$$

Taking norms, we have

$$\|\mathbf{x}' - \mathbf{x}^*\| \leq \|H(\mathbf{x})^{-1}\| \cdot \left\| \int_0^1 (H(\mathbf{x} + t(\mathbf{x}^* - \mathbf{x})) - H(\mathbf{x}))(\mathbf{x}^* - \mathbf{x}) dt \right\|,$$

because $\|A\mathbf{y}\| \leq \|A\| \cdot \|\mathbf{y}\|$ for any A, \mathbf{y} (by def. of spectral norm).

Proof of convergence theorem, III

So far we have

$$\begin{aligned}\|\mathbf{x}' - \mathbf{x}^*\| &\leq \|H(\mathbf{x})^{-1}\| \cdot \left\| \int_0^1 (H(\mathbf{x} + t(\mathbf{x}^* - \mathbf{x})) - H(\mathbf{x}))(\mathbf{x}^* - \mathbf{x}) dt \right\| \\ &\leq \|H(\mathbf{x})^{-1}\| \int_0^1 \|(H(\mathbf{x} + t(\mathbf{x}^* - \mathbf{x})) - H(\mathbf{x}))(\mathbf{x}^* - \mathbf{x})\| dt \quad (\text{Ex. 57}) \\ &\leq \|H(\mathbf{x})^{-1}\| \int_0^1 \|H(\mathbf{x} + t(\mathbf{x}^* - \mathbf{x})) - H(\mathbf{x})\| \cdot \|\mathbf{x}^* - \mathbf{x}\| dt \\ &= \|H(\mathbf{x})^{-1}\| \cdot \|\mathbf{x}^* - \mathbf{x}\| \int_0^1 \|H(\mathbf{x} + t(\mathbf{x}^* - \mathbf{x})) - H(\mathbf{x})\| dt.\end{aligned}$$

We can now use the properties (i) and (ii) (bounded inverse Hessians, Lipschitz continuous Hessians) to conclude that

$$\|\mathbf{x}' - \mathbf{x}^*\| \leq \frac{1}{\mu} \|\mathbf{x}^* - \mathbf{x}\| \int_0^1 B \|t(\mathbf{x}^* - \mathbf{x})\| dt = \frac{B}{\mu} \|\mathbf{x}^* - \mathbf{x}\|^2 \underbrace{\int_0^1 t dt}_{1/2} = \frac{B}{2\mu} \|\mathbf{x} - \mathbf{x}^*\|^2.$$

Strong convexity \Rightarrow Bounded inverse Hessians

One way to ensure bounded inverse Hessians is to **require strong convexity over X** .

Lemma 8.7 (Exercise 58)

Let $f : \text{dom}(f) \rightarrow \mathbb{R}$ be twice differentiable and strongly convex with parameter μ over an open convex subset $X \subseteq \text{dom}(f)$ meaning that

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}) + \frac{\mu}{2} \|\mathbf{x} - \mathbf{y}\|^2, \quad \forall \mathbf{x}, \mathbf{y} \in X.$$

Then $\nabla^2 f(\mathbf{x})$ is invertible and $\|\nabla^2 f(\mathbf{x})^{-1}\| \leq 1/\mu$ for all $\mathbf{x} \in X$, where $\|\cdot\|$ is the spectral norm.

Downside of Newton's method

Computing the Hessian matrix is difficult.

Computational bottleneck in each step:

- ▶ compute and invert the **Hessian matrix**
- ▶ or solve the linear system $\nabla^2 f(\mathbf{x}_t) \Delta \mathbf{x} = -\nabla f(\mathbf{x}_t)$ for the next step $\Delta \mathbf{x}$.

Matrix / system has size $d \times d$, taking up to $\mathcal{O}(d^3)$ time to invert / solve.

In many applications, d is large. . .

The secant method

Another iterative method for finding zeros in dimension 1

Start from Newton-Raphson step

$$x_{t+1} := x_t - \frac{f(x_t)}{f'(x_t)},$$

Use **finite difference approximation** of $f'(x_t)$:

$$f'(x_t) \approx \frac{f(x_t) - f(x_{t-1})}{x_t - x_{t-1}}.$$

将导数替换掉

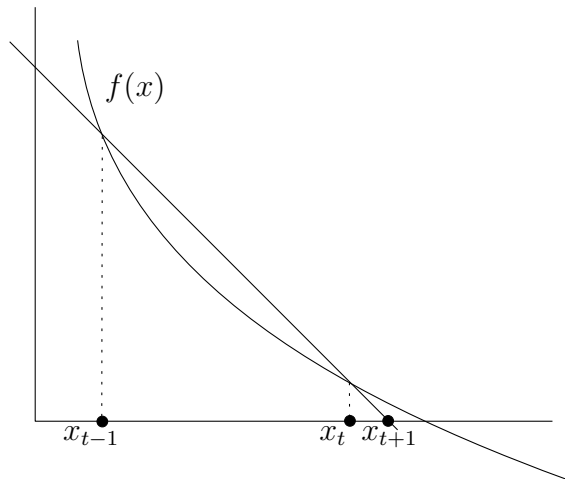
(for $|x_t - x_{t-1}|$ small)

Obtain the **secant method**:

正割的方法

$$x_{t+1} := x_t - f(x_t) \frac{x_t - x_{t-1}}{f(x_t) - f(x_{t-1})}$$

The secant method II



- ▶ construct the line through the two points $(x_{t-1}, f(x_{t-1}))$ and $(x_t, f(x_t))$;
- ▶ next iterate x_{t+1} is where this line intersects the x -axis (Exercise 59)

The secant method III

不用直接计算导数，直接就能进行函数的更新。

We now have a **derivative-free** version of the Newton-Raphson method.

Secant method for optimization: Can we also **optimize** a differentiable univariate function f ?— Yes, apply the secant method to f' :

$$x_{t+1} := x_t - f'(x_t) \frac{x_t - x_{t-1}}{f'(x_t) - f'(x_{t-1})}$$

- a **second-derivative-free** version of Newton's method for optimization.

Can we generalize this to higher dimensions to obtain a **Hessian-free** version of Newton's method in \mathbb{R}^d ?

The secant condition

Apply finite difference approximation to f'' (still 1-dim),

apply the secant method
to the Hessian.

$$H_t := \frac{f'(x_t) - f'(x_{t-1})}{x_t - x_{t-1}} \approx f''(x_t)$$

\Leftrightarrow

$$f'(x_t) - f'(x_{t-1}) = H_t(x_t - x_{t-1}),$$

the secant condition.

- ▶ Newton's method: $x_{t+1} := x_t - f''(x_t)^{-1} f'(x_t)$
- ▶ Secant method: $x_{t+1} := x_t - H_t^{-1} f'(x_t)$

In higher dimensions: Let $H_t \in \mathbb{R}^{d \times d}$ be an invertible symmetric matrix satisfying the d -dimensional secant condition

$$\nabla f(\mathbf{x}_t) - \nabla f(\mathbf{x}_{t-1}) = H_t(\mathbf{x}_t - \mathbf{x}_{t-1}).$$

The secant method step then becomes

$$\mathbf{x}_{t+1} := \mathbf{x}_t - H_t^{-1} \nabla f(\mathbf{x}_t). \quad (1)$$

Quasi-Newton methods

Newton: $\mathbf{x}_{t+1} := \mathbf{x}_t - \nabla^2 f(\mathbf{x}_t)^{-1} \nabla f(\mathbf{x}_t)$

Secant $\mathbf{x}_{t+1} := \mathbf{x}_t - H_t^{-1} \nabla f(\mathbf{x}_t)$, where $\nabla f(\mathbf{x}_t) - \nabla f(\mathbf{x}_{t-1}) = H_t(\mathbf{x}_t - \mathbf{x}_{t-1})$

If f is twice differentiable, secant condition and first-order approximation of $\nabla f(\mathbf{x})$ at \mathbf{x}_t yield:

$$\nabla f(\mathbf{x}_t) - \nabla f(\mathbf{x}_{t-1}) = H_t(\mathbf{x}_t - \mathbf{x}_{t-1}) \approx \nabla^2 f(\mathbf{x}_t)(\mathbf{x}_t - \mathbf{x}_{t-1}).$$

Might therefore hope that $H_t \approx \nabla^2 f(\mathbf{x}_t) \dots$

... meaning that the secant method approximates Newton's method.

- ▶ $d = 1$: unique number H_t satisfying the secant condition
- ▶ $d > 1$: Secant condition $\nabla f(\mathbf{x}_t) - \nabla f(\mathbf{x}_{t-1}) = H_t(\mathbf{x}_t - \mathbf{x}_{t-1})$ has infinitely many symmetric solutions H_t (underdetermined linear system).

Any scheme of choosing in each step of the secant method a **symmetric H_t** that satisfies the secant condition defines a **Quasi Newton method**.

Quasi-Newton methods II

- ▶ Exercise 60: Newton's method is a Quasi-Newton method if and only if f is a nondegenerate quadratic function.
- ▶ Hence, Quasi-Newton methods do not generalize Newton's method but form a family of related algorithms.
- ▶ The first Quasi-Newton method was developed by William C. Davidon in 1956; he desperately needed iterations that were faster than those of Newton's method in order obtain results in the short time spans between expected failures of the room-sized computer that he used to run his computations on.
- ▶ But the paper he wrote about his new method got rejected for lacking a convergence analysis, and for allegedly dubious notation. It became a very influential Technical Report in 1959 [Dav59] and was finally officially published in 1991, with a foreword giving the historical context [Dav91]. Ironically, Quasi-Newton methods are today the methods of choice in a number of relevant machine learning applications.
- ▶ Here: no convergence analysis (for a change), we focus on the development of algorithms from first principles.

Developing a Quasi-Newton method

For efficiency reasons (want to **avoid matrix inversions!**), directly deal with the inverse matrices H_t^{-1} .

Given: iterates $\mathbf{x}_{t-1}, \mathbf{x}_t$ as well as the matrix H_{t-1}^{-1} .

Wanted: next matrix H_t^{-1} needed in next Quasi-Newton step

$$\mathbf{x}_{t+1} := \mathbf{x}_t - H_t^{-1} \nabla f(\mathbf{x}_t).$$

How should we choose H_t^{-1} ?

Newton's method: $\nabla^2 f(\mathbf{x}_t)$ fluctuates only very little in the region of extremely fast convergence.

Hence, in a Quasi-Newton method, it also makes sense to have that $H_t \approx H_{t-1}$, or $H_t^{-1} \approx H_{t-1}^{-1}$.

Greenstadt's family of Quasi-Newton methods

Given: iterates $\mathbf{x}_{t-1}, \mathbf{x}_t$ as well as the matrix H_{t-1}^{-1} .

Wanted: next matrix H_t^{-1} needed in next Quasi-Newton step

$$\mathbf{x}_{t+1} := \mathbf{x}_t - H_t^{-1} \nabla f(\mathbf{x}_t).$$

Greenstadt [Gre70]: Update

$$H_t^{-1} := H_{t-1}^{-1} + E_t,$$

E_t an error matrix.

Try to minimize the error subject to H_t satisfying the secant condition!

Simple error measure: squared Frobenius norm

$$\|E\|_F^2 := \sum_{i=1}^d \sum_{j=1}^d E_{ij}^2.$$

Greenstadt's family of Quasi-Newton methods II

Greenstadt: minimizing $\|E\|_F$ gives just one method, this is “too specialized”.

Greenstadt searched for a compromise between variability in the method and simplicity of the resulting formulas.

More general error measure

$$\|AEA^\top\|_F^2,$$

where $A \in \mathbb{R}^{d \times d}$ is some fixed invertible transformation matrix.

$A = I$: squared Frobenius norm of E , the “specialized” method.

The Greenstadt Update $H_{t-1}^{-1} \rightarrow H_t^{-1}$

Secant condition in terms of H_t^{-1} :

$$H_t^{-1}(\nabla f(\mathbf{x}_t) - \nabla f(\mathbf{x}_{t-1})) = (\mathbf{x}_t - \mathbf{x}_{t-1}).$$

Fix t and simplify notation:

H	$:= H_{t-1}^{-1}$	(old inverse)
H'	$:= H_t^{-1}$	(new inverse)
E	$:= E_t,$	(error matrix)
σ	$:= \mathbf{x}_t - \mathbf{x}_{t-1}$	(step in solutions)
\mathbf{y}	$= \nabla f(\mathbf{x}_t) - \nabla f(\mathbf{x}_{t-1})$	(step in gradients)
\mathbf{r}	$= \sigma - H\mathbf{y}$	(error of old inverse in new secant condition)

The update formula is

$$H' = H + E,$$

Secant condition becomes

$$H'\mathbf{y} = \sigma \quad \Leftrightarrow \quad (H + E)\mathbf{y} = \sigma \quad \Leftrightarrow \quad E\mathbf{y} = \sigma - H\mathbf{y} \quad \Leftrightarrow \quad E\mathbf{y} = \mathbf{r}.$$

The Greenstadt Update $H_{t-1}^{-1} \rightarrow H_t^{-1}$ II

Minimizing the error becomes a convex constrained minimization problem in the d^2 variables E_{ij} :

$$\begin{array}{ll} \text{minimize} & \frac{1}{2} \|AEA^\top\|_F^2 \quad (\text{error function}) \\ \text{subject to} & E\mathbf{y} = \mathbf{r} \quad (\text{secant condition}) \\ & E^\top - E = 0 \quad (\text{symmetry}) \end{array}$$

Don't need to solve it computationally (for numbers E_{ij}) ...

... but mathematically (formula for E)

Minimize **convex quadratic** function subject to **linear equations** \rightarrow analytic formula for the minimizer from the **method of Lagrange multipliers**.

The method of Lagrange multipliers

Theorem 9.1

Let $f : \mathbb{R}^d \rightarrow \mathbb{R}$ be convex and differentiable, $C \in \mathbb{R}^{m \times d}$ for some $m \in \mathbb{N}$, $\mathbf{e} \in \mathbb{R}^m$, $\mathbf{x}^* \in \mathbb{R}^d$ such that $C\mathbf{x}^* = \mathbf{e}$. Then the following two statements are equivalent.

- (i) $\mathbf{x}^* = \operatorname{argmin}\{f(\mathbf{x}) : \mathbf{x} \in \mathbb{R}^d, C\mathbf{x} = \mathbf{e}\}$
- (ii) There exists a vector $\boldsymbol{\lambda} \in \mathbb{R}^m$ such that

$$\nabla f(\mathbf{x}^*)^\top = \boldsymbol{\lambda}^\top C.$$

The entries of $\boldsymbol{\lambda}$ are known as the Lagrange multipliers.

Proof 1 (Consequence of previous material).

Theorem 2.48: a Slater point implies strong Lagrange duality. If, as in the (i), there are only affine equality constraints, the Slater point condition is void, and we obtain strong Lagrange duality “for free”. In this case, the equivalence of (i) and (ii) follows from the Karush-Kuhn-Tucker necessary and sufficient conditions (Theorems 2.52 and 2.53). \square

The method of Lagrange multipliers II

Proof 2 (direct).

(ii) \Rightarrow (i): if $\nabla f(\mathbf{x}^\star)^\top = \boldsymbol{\lambda}^\top C$ and $\mathbf{x} \in \mathbb{R}^d$ satisfies $C\mathbf{x} = \mathbf{e}$, we get

$$\nabla f(\mathbf{x}^\star)^\top (\mathbf{x} - \mathbf{x}^\star) = \boldsymbol{\lambda}^\top C(\mathbf{x} - \mathbf{x}^\star) = \boldsymbol{\lambda}^\top (\mathbf{e} - \mathbf{e}) = 0.$$

Hence, \mathbf{x}^\star is a minimizer of f over $\{\mathbf{x} \in \mathbb{R}^d : C\mathbf{x} = \mathbf{e}\}$ by the optimality condition of Lemma 2.28.

The other direction is Exercise 61.



Application to Greenstadt Update

If f is quadratic, $\nabla f(\mathbf{x})$ is linear.

Hence, the minimizer \mathbf{x}^* and the Lagrange multipliers $\boldsymbol{\lambda}$ can be obtained by solving a [system of linear equations](#):

$$\begin{aligned} C\mathbf{x} &= \mathbf{e} \\ \nabla f(\mathbf{x})^\top &= \boldsymbol{\lambda}^\top C \end{aligned}$$

Greenstadt:

$$f(E) = \frac{1}{2} \|AEA^\top\|_F^2.$$

Exercise 62: $\nabla f(E)$ (written as a matrix) is

$$\nabla f(E) = A^\top AEA^\top A =: WEW =: M^{-1}EM^{-1}.$$

Now solve the system of linear equations for the optimal $E^* \rightarrow$ Section 9.4.2

The Greenstadt family

Here is what we get:

Definition 9.4

Let $M \in \mathbb{R}^{d \times d}$ be a symmetric and invertible matrix. Consider the Quasi-Newton method

$$\mathbf{x}_{t+1} = \mathbf{x}_t - H_t^{-1} \nabla f(\mathbf{x}_t), \quad t \geq 1,$$

where $H_0 = I$ (or any positive definite matrix), and $H_t^{-1} = H_{t-1}^{-1} + E_t$ for all $t \geq 1$.

For any fixed t , set $H := H_{t-1}^{-1}$, $H' := H_t^{-1}$, $\boldsymbol{\sigma} := \mathbf{x}_t - \mathbf{x}_{t-1}$, $\mathbf{y} := \nabla f(\mathbf{x}_t) - \nabla f(\mathbf{x}_{t-1})$. Define

$$\begin{aligned} E^* = \frac{1}{\mathbf{y}^\top M \mathbf{y}} & \left(\boldsymbol{\sigma} \mathbf{y}^\top M + M \mathbf{y} \boldsymbol{\sigma}^\top - H \mathbf{y} \mathbf{y}^\top M - M \mathbf{y} \mathbf{y}^\top H \right. \\ & \left. - \frac{1}{\mathbf{y}^\top M \mathbf{y}} (\mathbf{y}^\top \boldsymbol{\sigma} - \mathbf{y}^\top H \mathbf{y}) M \mathbf{y} \mathbf{y}^\top M \right). \end{aligned}$$

If the update matrix $E_t = E^*$ is used, the method is called the **Greenstadt method** with parameter M .

Choosing M

$$E^* = \frac{1}{\mathbf{y}^\top M \mathbf{y}} \left(\boldsymbol{\sigma} \mathbf{y}^\top M + M \mathbf{y} \boldsymbol{\sigma}^\top - H \mathbf{y} \mathbf{y}^\top M - M \mathbf{y} \mathbf{y}^\top H \right. \\ \left. - \frac{1}{\mathbf{y}^\top M \mathbf{y}} (\mathbf{y}^\top \boldsymbol{\sigma} - \mathbf{y}^\top H \mathbf{y}) M \mathbf{y} \mathbf{y}^\top M \right).$$

Greenstadt suggested

$$M = I \quad (\text{default choice})$$

$$M = H \quad (\text{previous inverse } H_{t-1}^{-1})$$

Goldfarb [Gol70] suggested

$$M = H' \quad (\text{next inverse } H_t^{-1})$$

But wait, we don't even know H_t yet. . .

. . . doesn't matter, this is math, not computation!

The BFGS method

$$E^* = \frac{1}{\mathbf{y}^\top M \mathbf{y}} \left(\boldsymbol{\sigma} \mathbf{y}^\top M + M \mathbf{y} \boldsymbol{\sigma}^\top - H \mathbf{y} \mathbf{y}^\top M - M \mathbf{y} \mathbf{y}^\top H \right. \\ \left. - \frac{1}{\mathbf{y}^\top M \mathbf{y}} (\mathbf{y}^\top \boldsymbol{\sigma} - \mathbf{y}^\top H \mathbf{y}) M \mathbf{y} \mathbf{y}^\top M \right).$$

Chose $M = H'$. Because of secant condition: $M \mathbf{y} = H' \mathbf{y} = \boldsymbol{\sigma}$. M cancels.

Definition 9.5

The **BFGS method** is the Greenstadt method with parameter $M := H' = H_t^{-1}$ in step t , in which case the update matrix E^* assumes the form

$$E^* = \frac{1}{\mathbf{y}^\top \boldsymbol{\sigma}} \left(2 \boldsymbol{\sigma} \boldsymbol{\sigma}^\top - H \mathbf{y} \boldsymbol{\sigma}^\top - \boldsymbol{\sigma} \mathbf{y}^\top H - \frac{1}{\boldsymbol{\sigma}^\top \mathbf{y}} (\mathbf{y}^\top \boldsymbol{\sigma} - \mathbf{y}^\top H \mathbf{y}) \boldsymbol{\sigma} \boldsymbol{\sigma}^\top \right) \\ = \frac{1}{\mathbf{y}^\top \boldsymbol{\sigma}} \left(-H \mathbf{y} \boldsymbol{\sigma}^\top - \boldsymbol{\sigma} \mathbf{y}^\top H + \left(1 + \frac{\mathbf{y}^\top H \mathbf{y}}{\mathbf{y}^\top \boldsymbol{\sigma}} \right) \boldsymbol{\sigma} \boldsymbol{\sigma}^\top \right), \quad (2)$$

where $H = H_{t-1}^{-1}$, $\boldsymbol{\sigma} = \mathbf{x}_t - \mathbf{x}_{t-1}$, $\mathbf{y} = \nabla f(\mathbf{x}_t) - \nabla f(\mathbf{x}_{t-1})$.

The BFGS method II

$$H' = H + \frac{1}{\mathbf{y}^\top \boldsymbol{\sigma}} \left(-H\mathbf{y}\boldsymbol{\sigma}^\top - \boldsymbol{\sigma}\mathbf{y}^\top H + \left(1 + \frac{\mathbf{y}^\top H\mathbf{y}}{\mathbf{y}^\top \boldsymbol{\sigma}}\right) \boldsymbol{\sigma}\boldsymbol{\sigma}^\top \right).$$

- ▶ Exercise 63 (i): $\mathbf{y}^\top \boldsymbol{\sigma} > 0$ unless f is flat between \mathbf{x}_{t-1} and \mathbf{x}_t .
- ▶ Exercise 63 (ii): If $\mathbf{y}^\top \boldsymbol{\sigma} > 0$ and H is positive definite, then also H' is positive definite. In this respect, the matrices in the BFGS method behave like proper inverse Hessians.
- ▶ Method is named after Broyden, Fletcher, Goldfarb and Shanno who all came up with it independently around 1970. Greenstadt's name is mostly forgotten.
- ▶ Cost per update step: $O(d^2)$, no Hessians and no inversions required.
- ▶ Newton and Quasi-Newton methods are often performed with **scaled steps**. This means that the iteration becomes

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \alpha_t H_t^{-1} \nabla f(\mathbf{x}_t), \quad t \geq 1, \quad (3)$$

for some $\alpha_t \in \mathbb{R}_+$.

Quasi-Quasi-Newton methods?

Newton's method: $O(d^3)$ per iteration.

BFGS: $O(d^2)$ per iteration. This is still prohibitive if d is large.

Can we get down to $O(d)$?

Limited-memory variant L-BFGS comes to the rescue.

The L-BFGS method approximates the BFGS method.

The BFGS method revisited

Rewrite Update step in product form:

Observation 9.6

With E^* as in the BFGS method and $H' = H + E^*$, we have

$$H' = \left(I - \frac{\sigma \mathbf{y}^\top}{\mathbf{y}^\top \sigma} \right) H \left(I - \frac{\mathbf{y} \sigma^\top}{\mathbf{y}^\top \sigma} \right) + \frac{\sigma \sigma^\top}{\mathbf{y}^\top \sigma}.$$

Proof.

Elementary calculations. (But still $O(d^2)$ time; we need to compute a matrix H' .) □

Recall Quasi-Newton step:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - H_t^{-1} \nabla f(\mathbf{x}_t), \quad t \geq 1.$$

Note: We don't need the $(d \times d)$ -matrix H_t^{-1} but only the d -vector $H_t^{-1} \nabla f(\mathbf{x}_t)$.

Idea: Don't update the matrix, but only the vector!

Efficiently computing matrix vector-products $H_t^{-1} \nabla f(\mathbf{x}_t)$

Lemma 9.7

Let $H = H_{t-1}^{-1}$, $H' = H_t^{-1}$ as in the BFGS method, i.e.

$$H' = \left(I - \frac{\boldsymbol{\sigma} \mathbf{y}^\top}{\mathbf{y}^\top \boldsymbol{\sigma}} \right) H \left(I - \frac{\mathbf{y} \boldsymbol{\sigma}^\top}{\mathbf{y}^\top \boldsymbol{\sigma}} \right) + \frac{\boldsymbol{\sigma} \boldsymbol{\sigma}^\top}{\mathbf{y}^\top \boldsymbol{\sigma}}.$$

Let $\mathbf{g}' \in \mathbb{R}^d$. Suppose that we have an oracle to compute $\mathbf{s} = H\mathbf{g}$ for any vector \mathbf{g} . Then $\mathbf{s}' = H'\mathbf{g}'$ can be computed with one oracle call and $O(d)$ additional arithmetic operations, assuming that $\boldsymbol{\sigma}$ and \mathbf{y} are known.

Efficiently computing matrix vector-products $H_t^{-1} \nabla f(\mathbf{x}_t)$ - Proof

$$H' \mathbf{g}' = \underbrace{\left(I - \frac{\boldsymbol{\sigma} \mathbf{y}^\top}{\mathbf{y}^\top \boldsymbol{\sigma}} \right) H \underbrace{\left(I - \frac{\mathbf{y} \boldsymbol{\sigma}^\top}{\mathbf{y}^\top \boldsymbol{\sigma}} \right) \mathbf{g}'}_{\mathbf{g}}}_{\mathbf{s}} + \underbrace{\frac{\boldsymbol{\sigma} \boldsymbol{\sigma}^\top}{\mathbf{y}^\top \boldsymbol{\sigma}} \mathbf{g}'}_{\mathbf{h}}.$$

$\underbrace{\hspace{10em}}_{\mathbf{w}}$
 $\underbrace{\hspace{15em}}_{\mathbf{z}}$

$$\mathbf{h} = \frac{\boldsymbol{\sigma} \boldsymbol{\sigma}^\top}{\mathbf{y}^\top \boldsymbol{\sigma}} \mathbf{g}' = \boldsymbol{\sigma} \frac{\boldsymbol{\sigma}^\top \mathbf{g}'}{\mathbf{y}^\top \boldsymbol{\sigma}}$$

(two inner products, a real division, and a multiplication of $\boldsymbol{\sigma}$ with a scalar)

$$\mathbf{g} = \left(I - \frac{\mathbf{y} \boldsymbol{\sigma}^\top}{\mathbf{y}^\top \boldsymbol{\sigma}} \right) \mathbf{g}' = \mathbf{g}' - \mathbf{y} \frac{\boldsymbol{\sigma}^\top \mathbf{g}'}{\mathbf{y}^\top \boldsymbol{\sigma}}$$

(multiplication of \mathbf{y} with a scalar that we already know, followed by a vector addition)

Efficiently computing matrix vector-products $H_t^{-1} \nabla f(\mathbf{x}_t)$ - Proof II

$$\begin{aligned} H' \mathbf{g}' &= \left(I - \frac{\boldsymbol{\sigma} \mathbf{y}^\top}{\mathbf{y}^\top \boldsymbol{\sigma}} \right) H \underbrace{\left(I - \frac{\mathbf{y} \boldsymbol{\sigma}^\top}{\mathbf{y}^\top \boldsymbol{\sigma}} \right) \mathbf{g}'}_{\mathbf{g}} + \underbrace{\frac{\boldsymbol{\sigma} \boldsymbol{\sigma}^\top}{\mathbf{y}^\top \boldsymbol{\sigma}} \mathbf{g}'}_{\mathbf{h}} \\ &\quad \underbrace{\hspace{10em}}_{\mathbf{s} \leftarrow \text{oracle call}} \\ &\quad \underbrace{\hspace{10em}}_{\mathbf{w}} \\ &\quad \underbrace{\hspace{10em}}_{\mathbf{z}} \\ \mathbf{w} &= \left(I - \frac{\boldsymbol{\sigma} \mathbf{y}^\top}{\mathbf{y}^\top \boldsymbol{\sigma}} \right) \mathbf{s} = \mathbf{s} - \boldsymbol{\sigma} \frac{\mathbf{y}^\top \mathbf{s}}{\mathbf{y}^\top \boldsymbol{\sigma}} \end{aligned}$$

(one new inner product, a real division, a multiplication of $\boldsymbol{\sigma}$ with a scalar, and a vector addition)

$$H' \mathbf{g}' = \mathbf{z} = \mathbf{w} + \mathbf{h}$$

(a vector addition). In total, three inner product computations, three scalar multiplications, three vector additions, two real divisions, and one oracle call.

The oracle

With $O(d)$ arithmetic operations, we can reduce the computation of matrix-vector products $H_t^{-1} \mathbf{g}'$ to the computation of matrix vectors products $H_{t-1}^{-1} \mathbf{g}$ (the oracle).

How do we implement the oracle?

Let

$$\begin{aligned}\boldsymbol{\sigma}_k &= \mathbf{x}_k - \mathbf{x}_{k-1}, \\ \mathbf{y}_k &= \nabla f(\mathbf{x}_k) - \nabla f(\mathbf{x}_{k-1})\end{aligned}$$

be the values of $\boldsymbol{\sigma}$ and \mathbf{y} in iteration $k \leq t$ (we know them all)

This lets us compute the BFGS-step $H_t^{-1} \nabla f(\mathbf{x}_t)$ **recursively**.

The recursive BFGS-step (update step in one iteration)

function BFGS-STEP(k, \mathbf{g}')

▷ returns $H_k^{-1} \mathbf{g}'$

if $k = 0$ **then**

return $H_0^{-1} \mathbf{g}'$

else

▷ apply Lemma 9.7

$$\mathbf{h} = \sigma \frac{\sigma_k^\top \mathbf{g}'}{\mathbf{y}_k^\top \sigma_k}$$

$$\mathbf{g} = \mathbf{g}' - \mathbf{y} \frac{\sigma_k^\top \mathbf{g}'}{\mathbf{y}_k^\top \sigma_k}$$

$\mathbf{s} = \text{BFGS-STEP}(k-1, \mathbf{g})$

$$\mathbf{w} = \mathbf{s} - \sigma_k \frac{\mathbf{y}_k^\top \mathbf{s}}{\mathbf{y}_k^\top \sigma_k}$$

$$\mathbf{z} = \mathbf{w} + \mathbf{h}$$

return \mathbf{z}

end if

end function

The recursive BFGS-step II (update step in one iteration)

In iteration t , call $\text{BFGS-STEP}(t, \nabla f(\mathbf{x}_t))$ to get $H_t^{-1} \nabla f(\mathbf{x}_t)$.

Runtime $O(td)$.

Worse than before if $t > d$.

Idea: only go down m levels of recursion for some small m .

The L-BFGS method (update step in one iteration)

function L-BFGS-STEP(k, ℓ, \mathbf{g}')

▷ $\ell \leq k$; returns $\mathbf{s}' \approx H_k^{-1} \mathbf{g}'$

if $\ell = 0$ **then**

return $H_0^{-1} \mathbf{g}'$

▷ correct value would be $H_k^{-1} \mathbf{g}'$, but we don't have it...

else

▷ apply Lemma 9.7

$$\mathbf{h} = \sigma \frac{\sigma_k^\top \mathbf{g}'}{\mathbf{y}_k^\top \sigma_k}$$

$$\mathbf{g} = \mathbf{g}' - \mathbf{y} \frac{\sigma_k^\top \mathbf{g}'}{\mathbf{y}_k^\top \sigma_k}$$

$\mathbf{s} = \text{L-BFGS-STEP}(k-1, \ell-1, \mathbf{g})$

$$\mathbf{w} = \mathbf{s} - \sigma_k \frac{\mathbf{y}_k^\top \mathbf{s}}{\mathbf{y}_k^\top \sigma_k}$$

$$\mathbf{z} = \mathbf{w} + \mathbf{h}$$

return \mathbf{z}

end if

end function

The L-BFGS method II (update step in one iteration)






In iteration t , call $\text{L-BFGS-STEP}(t, m, \nabla f(\mathbf{x}_t))$ to get an approximation of $H_t^{-1} \nabla f(\mathbf{x}_t)$ based on the previous m iterations.

Runtime per update $O(dm) = O(d)$ if m is constant.

A Quasi-Quasi-Newton method?

- ▶ L-BFGS is still a proper Quasi-Newton method.
- ▶ The matrix H' will satisfy the secant condition by design, irrespective of H .

Bibliography

-  William C. Davidon.
Variable metric method for minimization.
Technical Report ANL-5990, AEC Research and Development, 1959.
-  William C. Davidon.
Variable metric method for minimization.
SIAM J. Optimization, 1(1):1–17, 1991.
-  D. Goldfarb.
A family of variable-metric methods derived by variational means.
Mathematics of Computation, 24(109):23–26, 1970.
-  J. Greenstadt.
Variations on variable-metric methods.
Mathematics of Computation, 24(109):1–22, 1970.
-  Sai Praneeth Karimireddy, Sebastian U Stich, and Martin Jaggi.
Global linear convergence of Newton's method without strong-convexity or Lipschitz gradients.
arXiv, 2018.