Unconstrained Non-Linear Programs

Kaiwen Zhou

Contents

Topic: Primer

Lemma 0.1. *Sherman-Morrison formula:*

$$\left(\mathbf{A}+\mathbf{a}\mathbf{b}^{\top}\right)^{-1}=\mathbf{A}^{-1}-\frac{\mathbf{A}^{-1}\mathbf{a}\mathbf{b}^{\top}\mathbf{A}^{-1}}{1+\mathbf{b}^{\top}\mathbf{A}^{-1}\mathbf{a}}$$

Definition 0.2. *A sequence $\{\mathbf{x}_k\}_{k=0}^{\infty}\in\mathbb{R}^n$ converging to $\mathbf{x}^*$ is q-linearly convergent with order of convergence q and rate of convergence $\mu\geq 0$ if:*

$$\lim_{k\to\infty}\frac{\|\mathbf{x}_{k+1}-\mathbf{x}^*\|}{\|\mathbf{x}_k-\mathbf{x}^*\|^q}=\mu<\infty.$$

Important terminology for special cases:

- $q=1$ *linearly convergent, approximately one digit per iteration.*
- $q=2$ *quadratically convergent*
- $q=3$ *cubically convergent*
- $q=1,\mu=0$ *super linear convergent, quicker than vanilla linear convergent.*

Definition 0.3. *A function f is Lipschitz continuous if there exists some L s.t. $\forall x,y\in\mathbb{R}$, we have*

$$\|f(x)-f(y)\|=L\|x-y\|$$

Definition 0.4. ***Notation:*** *Suppose $\mathbf{F}:\mathbb{R}^n\to\mathbb{R}^m$, then we have*

$$\textit{Jacobian}(\mathbf{F})=\mathcal{D}(\mathbf{F})=\left(\frac{\partial\mathbf{F}}{\partial x_1},\ldots,\frac{\partial\mathbf{F}}{\partial x_n}\right)=\begin{pmatrix}\frac{\partial F_1}{\partial x_1}&\cdots&\frac{\partial F_1}{\partial x_n}\\ \vdots&\ddots&\vdots\\ \frac{\partial F_m}{\partial x_1}&\cdots&\frac{\partial F_m}{\partial x_n}\end{pmatrix}=\begin{bmatrix}\nabla^{\top}F_1\\ \vdots\\ \nabla^{\top}F_m\end{bmatrix}$$

Suppose $f:\mathbb{R}^n\to\mathbb{R}$, then we have

$$\textit{Hessian}(f)=\nabla\nabla^{\top}(f)=\nabla f\nabla^{\top}f\overset{\textit{notation}}{=\!=}\nabla^2 f=\begin{pmatrix}\frac{\partial f}{\partial x_1}\\ \vdots\\ \frac{\partial f}{\partial x_n}\end{pmatrix}\left(\frac{\partial f}{\partial x_1},\ldots,\frac{\partial f}{\partial x_n}\right)=\begin{pmatrix}\frac{\partial^2 f}{\partial x_1x_1}&\cdots&\frac{\partial^2 f}{\partial x_1x_n}\\ \vdots&\ddots&\vdots\\ \frac{\partial^2 f}{\partial x_nx_1}&\cdots&\frac{\partial^2 f}{\partial x_nx_n}\end{pmatrix}=\left(\frac{\partial\nabla f}{\partial x_1},\ldots,\frac{\partial\nabla f}{\partial x_n}\right)=\mathcal{D}\left(\nabla f\right)$$

Also

$$\textit{gradient}(f)=\nabla f=\begin{pmatrix}\frac{\partial f}{\partial x_1}\\ \vdots\\ \frac{\partial f}{\partial x_n}\end{pmatrix}=\left(\frac{\partial f}{\partial x_1},\ldots,\frac{\partial f}{\partial x_n}\right)^{\top}=[\mathcal{D}f]^{\top}=\textit{Jacobian}(f)^{\top}$$

1 Topic: Newton's Method

1.1 One Dimensional Newton's Method

The Newton's method only applies when $f(x)$ is a differentiable function. We start with an initial guess x_0, the closer to x^* the better. Then, we calculate the derivative $f'(x_0)$ at the guess point and have the following iteration scheme:

Iteration Scheme: Suppose we have x_k and $x^* = x_k + \delta_k$. Then, we have

$$0 = f(x^*) = f(x_k + \delta_k) = f(x_k) + f'(x_k)\delta_k + O(\|\delta_k\|^2) \quad \Longrightarrow \quad \delta_k = -\frac{f(x_k)}{f'(x_k)} + O(\|\delta_k\|^2)$$

We approximate δ_k with $-\frac{f(x_k)}{f'(x_k)}$ and obtain

$$x^* \approx x_{n+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

Repeat the process until the value of $f(x_{n+1})$ gets sufficiently close to zero or until a desired level of accuracy is reached.

Quadratic Convergence:

$$\delta_{n+1} = x^* - x_{n+1} = (x_k + \delta_k) - x_k + \frac{f(x_k)}{f'(x_k)} = O(\|\delta_k\|^2) = \frac{1}{2}\left|\frac{f''(\eta_k)}{f'(x_k)}\right|\delta_k^2 \quad \Longrightarrow \quad |\delta_{n+1}| \leq \frac{1}{2}\left|\frac{f''(\eta_k)}{f'(x_k)}\right||\delta_k|^2 \quad \longrightarrow \quad \text{quadratic convergence}$$

Conditions for Convergence:

1. f is C^2 on interval $[x^* - \delta, x^* + \delta], \delta > 0$ and that $f(x^*) = 0$ and $f''(x^*) \neq 0$.
2. There exists $A > 0$ such that $\left|\frac{f''(\eta_k)}{f'(x_k)}\right| \leq A$ for all $x, y \in [x^* - \delta, x^* + \delta]$.
3. If $|x^* - x_0| \leq h$ where $h \leq \delta, \frac{1}{A}$, then the sequence $\{x_k\}$ defined by Newton's method converges quadratically to x^*.

Note:

By the conditions for convergence, it can be concluded that: for any function that is tangent to the x-axis, Newton's method fails. This is because it breaks the condition # 2, since $f'(x^*) = 0$.

Optimization Problem: For the optimization problem $f'(x) = 0$, the Newton's method yields the following iteration scheme:

$$x_{n+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$$

1.2 Multivariate Newton

Root-Finding Problem: Solve $\mathbf{F}(\mathbf{x}) = \mathbf{0}, \mathbf{F} : \mathbb{R}^n \to \mathbb{R}^m$.

Suppose $\mathbf{x}^* \approx \mathbf{x}_{n+1} = \mathbf{x}_k + \delta\mathbf{x}$ We have

$$0 = \mathbf{F}(\mathbf{x}^*) \approx \mathbf{F}(\mathbf{x}_k + \delta\mathbf{x}) = \mathbf{F}(\mathbf{x}_k) + \mathcal{D}\mathbf{F}(\mathbf{x}_k)\delta\mathbf{x} + O(\|\delta\mathbf{x}\|^2)$$

solve for $\delta\mathbf{x}$ and we obtain the Newton iteration scheme:

$$\mathbf{x}_{n+1} = \mathbf{x}_k - \left(\mathcal{D}\mathbf{F}(\mathbf{x}_k)^\top \mathcal{D}\mathbf{F}(\mathbf{x}_k)\right)^{-1} \mathcal{D}\mathbf{F}(\mathbf{x}_k)\mathbf{F}(\mathbf{x}_k) = [\mathcal{D}\mathbf{F}(\mathbf{x}_k)]^\dagger \mathbf{F}(\mathbf{x}_k)$$

where $[\mathcal{D}\mathbf{F}(\mathbf{x}_k)]^\dagger := \left(\mathcal{D}\mathbf{F}(\mathbf{x}_k)^\top \mathcal{D}\mathbf{F}(\mathbf{x}_k)\right)^{-1} \mathcal{D}\mathbf{F}(\mathbf{x}_k)$.

Optimization Problem: Find $\mathbf{x}$ such that $\nabla f = 0$, $f : \mathbb{R}^N \to \mathbb{R}$. Similarly, the Newton iteration scheme gives:

$$\mathbf{x}_{n+1} = \mathbf{x}_k - \left(\nabla^2 f(\mathbf{x}_k)\right)^{-1} \nabla f(\mathbf{x}_k) \quad \text{where} \quad \nabla^2 f = \left[\frac{\partial \nabla f}{\partial x_1}, \cdots \frac{\partial \nabla f}{\partial x_k}\right], \nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_k} \end{bmatrix}$$

and $\nabla^2 f(\mathbf{x}_k)$ is the corresponding Hessian matrix.

Modified Newton's Method:

We need the Newton's step $\mathbf{p}_k = -\left[\nabla^2 f(\mathbf{x}_k)\right]^{-1} \nabla f(\mathbf{x}_k)$ to points to the descent direction, i.e. to satisfy $\nabla f(\mathbf{x}_k) \cdot \mathbf{p}_k < 0$, which amount to the Hessian matrix in the Newton's step being positive definite. This is becasue

$$\nabla^\top f(\mathbf{x}_k)\mathbf{p}_k = -\nabla^\top f(\mathbf{x}_k)\left[\nabla^2 f(\mathbf{x}_k)\right]^{-1} \nabla f(\mathbf{x}_k) < 0 \quad \Longleftrightarrow \quad \left[\nabla^2 f(\mathbf{x}_k)\right]^{-1} \text{ is positive definite.}$$

However, this is not always the case, and we will try to modify our scheme to accomodate. The idea of modifying Newton's Method is simple and similar to that of the ridge regression: if $\nabla^2 f(\mathbf{x}_k)$ is not positive definite, we then add some diagonal matrix with nonnegative diagonal entries to it. Suppose $(\lambda, \mathbf{u})$ is an eigen-pair of a matrix $\mathbf{A} = \nabla^2 f(\mathbf{x}_k)$, then let $\alpha \in \mathbb{R}^+$, we obtain

$$(\mathbf{A} + \alpha\mathbf{I})\mathbf{u} = \mathbf{A}\mathbf{u} + \alpha\mathbf{u} = \lambda\mathbf{u} + \alpha\mathbf{u} = (\lambda + \alpha)\mathbf{u} \quad \Longrightarrow \quad \text{eventually modified eigenvalues } \lambda_{new} = \lambda + \alpha \text{ of } \nabla^2 f(\mathbf{x}_k) \text{ will all be positive}$$

Actually, if $\nabla^2 f(\mathbf{x}_k)$ is positive definite, then so is $\left[\nabla^2 f(\mathbf{x}_k)\right]^{-1}$ why? Let $(\lambda, \mathbf{u})$ be an eigen-pair of $\nabla^2 f(\mathbf{x}_k)$ where $\lambda > 0$:

$$\nabla^2 f(\mathbf{x}_k)\mathbf{u} = \lambda\mathbf{u} \quad \Longrightarrow \quad \lambda^{-1}\mathbf{u} = \left[\nabla^2 f(\mathbf{x}_k)\right]^{-1}\mathbf{u} \quad \Longrightarrow \quad \text{all eigenvalues } \lambda^{-1} \text{ of } \left[\nabla^2 f(\mathbf{x}_k)\right]^{-1} \text{ are positive}$$

Kantonovich theorem —- quadratic convergence of Newton's Method:

We want to minimize $f : \mathbb{R}^n \to \mathbb{R}$ let $\mathbf{x}^*$ be a local minimum of f, and assume the sequence of Newton iterates $\{\mathbf{x}_k\}_{k=0}^{\infty} \in \mathbb{R}^n$ converges to $\mathbf{x}^*$, let $S \subset \mathbb{R}^n$ which is open and convex and st. $\mathbf{x}^* \in S$. Assume that f is C^2 on S, and for $\mathbf{x} \in S$, $\nabla^2 f(\mathbf{x})$ is Lipschitz-continuous with Lipchitz constant $L < \infty$ (i.e. $\forall x, y \in S$ we have $\left\|\nabla^2 f(x) - \nabla^2 f(y)\right\| \leq L\|x - y\|$). Assume $\nabla^2 f$ is positive definite on S . And finally, if $\mathbf{x}_0$ is "close enough" to $\mathbf{x}^*$, then $\mathbf{x}_k \to \mathbf{x}^*$ quadratically.

Note: S convex, $\nabla^2 f$ positive definite on S and $\mathbf{x}^*$ a local $\min \Rightarrow \mathbf{x}^*$ global min on S.

Another Theorem:

If instead of Newton's method, you use the iteration $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}_k$. where, as $k \to \infty, \mathbf{p}_k \longrightarrow -\left[\nabla^2 f(\mathbf{x}_k)\right]^{-1} \nabla f(\mathbf{x}_k)$ If plus some more conditions, then as $k \to \infty$, $\mathbf{x}_k \to \mathbf{x}^*$ and it will do so superlinearly.

Logic:

1. We know Newton's method sometimes fails.
2. Kantonovich theorem $\Longrightarrow$ **Condition*:** If f is C^2, $\nabla^2 f$ is Lipschitr-continuous (not too crazy) and $\mathbf{x}_0$ is close enough to $\mathbf{x}^*$ on S.
3. then Newton's method works and converges quadratically, Great!
4. How do we satisfy **Condition*** so we can use the best of NM? $\to$ We modify it to satisfy **Condition***.
5. Is the Newton's step always in a "descent direction"? $\to$ No always. Need $\nabla^2 f$ positive definite.
6. But $\nabla^2 f$ not always positive definite, what do we do? $\to$ modify $\nabla^2 f$ s.t. it's positive definite and $\nabla^2 f$ satisfies **Another Theorem**.
7. Use the **Another Theorem** above and the iteration converges superlinearly until we get to **Condition***, then we converge quadratically.

Newton's Method minimizes the quadratic function in one step:

1.3 Levenberg-Marquardt Algorithm

The Levenberg-Marquardt algorithm is an adaptive way to blend between Newton steps and steepest descent steps, and is widely used when solving nonlinear least squares problems.

In the Newton's method, we need the Newton's step $\mathbf{p}_k = -\left[\nabla^2 f(\mathbf{x}_k)\right]^{-1} \nabla f(\mathbf{x}_k)$ to points to the descent direction, i.e. to satisfy $\nabla f(\mathbf{x}_k) \cdot \mathbf{p}_k < 0$, which amount to the Hessian matrix in the Newton's step being positive definite. This is becasue

$$\nabla^\top f(\mathbf{x}_k)\mathbf{p}_k = -\nabla^\top f(\mathbf{x}_k)\left[\nabla^2 f(\mathbf{x}_k)\right]^{-1} \nabla f(\mathbf{x}_k) < 0 \quad \Longleftrightarrow \quad \left[\nabla^2 f(\mathbf{x}_k)\right]^{-1} \text{ is positive definite.}$$

However, this is not always the case, and we will try to modify our scheme to accomodate.

Different from the Modified Newton's Method, in this case, the Levenberg-Marquardt algorithm simply revert to the steepest descent (i.e. gradient descent) algorithm $\mathbf{p}_k = -\nabla f(\mathbf{x}_k)$.

1.4 Problems with the Newton's Method

1. If we assume each $\partial f/\partial x_i$ and $\partial^2 f/\partial x_i \partial x_j$ can be evaluated in $O(1)$ operations, then to process the n^{th} Newton step, i.e. to compute $p_k = -\left[\nabla f\left(x_k\right)\right]^{-1} \nabla f\left(x_k\right)$, we need to solve
$$\nabla^2 f\left(x_k\right) p_k = -\nabla f\left(x_k\right)$$
which costs $O\left(n^3\right)$ provided $\nabla^2 f\left(x_k\right)$ is symmetric AND if it's positive definite.
2. We know that if $\mathbf{x}_0$ is sufficiently close to $\mathbf{x}^*$, then Newton's method converges quadratically! However, the iteration is not guaranteed to converge. A consequence is that for $O(\varepsilon)$ accurary, we need $N = O\left(\log\log\frac{1}{\varepsilon}\right)$ iterations.

1.5 Non-Linear Least Square (NLS) Problem

Gauss-Newton: The Gauss-Newton Method is a Newton's Method with approximation of the corresponding Hessian Matrix. (Accuracy <—-> complexity) Suppose we have N parameters as $\mathbf{c}$ and M observations, then the Non-Linear Least Square problem is:

$$\text{minimize}_{\mathbf{c}\in\mathbb{R}^N} \|\mathbf{y} - \mathbf{f}(\mathbf{c})\|^2 \quad \mathbf{f} : \mathbb{R}^N \to \mathbb{R}^M$$

Let's define $F(\mathbf{c}) = \|\mathbf{y} - \mathbf{f}(\mathbf{c})\|_2^2$, $F : \mathbb{R}^N \to \mathbb{R}$, we have by Taylor Expansion:

$$\begin{aligned} F(\mathbf{c}+\delta\mathbf{c}) &= \|\mathbf{y} - \mathbf{f}(\mathbf{c}+\delta\mathbf{c})\|_2^2 = (\mathbf{y} - \mathbf{f}(\mathbf{c}+\delta\mathbf{c}))^T(\mathbf{y} - \mathbf{f}(\mathbf{c}+\delta\mathbf{c})) = \left(\mathbf{y} - \mathbf{f}(\mathbf{c}) - \mathcal{D}\mathbf{f}(\mathbf{c})\delta\mathbf{c} - O\left(\|\delta\mathbf{c}\|_2^2\right)\right)^T \left(\mathbf{y} - \mathbf{f}(\mathbf{c}) - \mathcal{D}\mathbf{f}(\mathbf{c})\delta\mathbf{c} - O\left(\|\delta\mathbf{c}\|_2^2\right)\right) \\ &= \mathbf{y}^\top\mathbf{y} + \mathbf{f}(\mathbf{c})^\top\mathbf{f}(\mathbf{c}) - \mathbf{y}^\top\mathbf{f}(\mathbf{c}) - \mathbf{f}(\mathbf{c})^\top\mathbf{y} - \mathbf{y}^\top\mathcal{D}\mathbf{f}(\mathbf{c})\delta\mathbf{c} - (\mathcal{D}\mathbf{f}(\mathbf{c})\delta\mathbf{c})^\top\mathbf{y} + \mathbf{f}(\mathbf{c})^\top\mathcal{D}\mathbf{f}(\mathbf{c})\delta\mathbf{c} + (\mathcal{D}\mathbf{f}(\mathbf{c})\delta\mathbf{c})^\top\mathbf{f}(\mathbf{c}) + O\left(\|\delta\mathbf{c}\|_2^2\right) \\ &= F(\mathbf{c}) + 2\left(\mathbf{f}(\mathbf{c})^\top\mathcal{D}\mathbf{f}(\mathbf{c}) - \mathbf{y}^\top\mathcal{D}\mathbf{f}(\mathbf{c})\right)\delta\mathbf{c} + O\left(\|\delta\mathbf{c}\|_2^2\right) \end{aligned}$$

Since $F(\mathbf{c}+\delta\mathbf{c}) = F(\mathbf{c}) + \mathcal{D}F(\mathbf{c})\delta\mathbf{c} + O\left(\|\delta\mathbf{c}\|_2^2\right)$, comparing with the above equation, we have

$$\mathcal{D}F(\mathbf{c}) = 2(\mathbf{f}(\mathbf{c}) - \mathbf{y})^\top \cdot \mathcal{D}\mathbf{f}(\mathbf{c}) \quad \Longrightarrow \quad \nabla F(\mathbf{c}) = [\mathcal{D}F(\mathbf{c})]^\top = 2\mathcal{D}\mathbf{f}(\mathbf{c})^\top \cdot (\mathbf{f}(\mathbf{c}) - \mathbf{y})$$

Apply the Newton Method, we have the iteration:

$$\mathbf{c}_{k+1} = \mathbf{c}_k - \left[\nabla^2 F(\mathbf{c}_k)\right]^{-1} \cdot \nabla F(\mathbf{c}_k)$$

Using $\mathcal{D}\left[\mathbf{f}^\top \mathbf{g}\right] = \mathcal{D}\mathbf{f}^\top \cdot \mathbf{g} + \mathbf{f}^\top \cdot \mathcal{D}\mathbf{g}$ and $\nabla F(\mathbf{c}) = 2\mathcal{D}\mathbf{f}(\mathbf{c})^\top(\mathbf{f}(\mathbf{c}) - \mathbf{y})$, we obtain

$$\nabla^2 F(\mathbf{c}) = \mathcal{D}\left(\nabla F(\mathbf{c})\right) = 2\mathcal{D}\left(\mathcal{D}\mathbf{f}(\mathbf{c})^\top \cdot (\mathbf{f}(\mathbf{c}) - \mathbf{y})\right) = 2\left[\mathcal{D}^2\mathbf{f}(\mathbf{c})^\top \cdot (\mathbf{f}(\mathbf{c}) - \mathbf{y}) + \mathcal{D}\mathbf{f}(\mathbf{c})^\top \mathcal{D}\mathbf{f}(\mathbf{c})\right] \approx 2\mathcal{D}\mathbf{f}(\mathbf{c})^\top \cdot \mathcal{D}\mathbf{f}(\mathbf{c})$$

Such approximation is valid since near optimum, $\mathbf{y} - \mathbf{f}(\mathbf{c})$ should be very small, so we can safely ignore this term. Now plug this approximation in the Newton's iteration scheme, we get:

$$\mathbf{c}_{k+1} \approx \mathbf{c}_k - \left(2 \cdot \mathcal{D}\mathbf{f}(\mathbf{c}_k)^\top \cdot \mathcal{D}\mathbf{f}(\mathbf{c}_k)\right)^{-1} \cdot 2\mathcal{D}\mathbf{f}(\mathbf{c}_k)^\top \cdot (\mathbf{f}(\mathbf{c}_k) - \mathbf{y})$$

Simply it, we obtain

$$\mathbf{c}_{n+1} \approx \mathbf{c}_k - \left(\mathcal{D}\mathbf{f}(\mathbf{c}_k)^\top \mathcal{D}\mathbf{f}(\mathbf{c}_k)\right)^{-1} \mathcal{D}\mathbf{f}(\mathbf{c}_k)^\top \left(\mathbf{f}\left(\mathbf{c}_k\right) - \mathbf{y}\right)$$

2 Topic: Quasi-Newton's Method

The Secant Method:

In one-dimensional root-finding problem, if the objective function $f(x) = 0$ is C^1, the Newton's method gives: $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$ We approximate the derivative with $f'(x_k) \approx \frac{f(x_k)-f(x_{k-1})}{x_k - x_{k-1}}$ and obtain the secant iteration:

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

In one-dimensional optimization problem, if the objective function $f'(x) = 0$ is C^1, the Newton's method gives: $x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$ We approximate the derivative with $f''(x_k) \approx \frac{f'(x_k)-f'(x_{k-1})}{x_k - x_{k-1}}$ and obtain the secant iteration:

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

We have that, under appropriate assumptions, the secant method converges superlinearly with order $q = \frac{1+\sqrt{5}}{2}$.

Remark 2.1. Newton converges quadratically $\rightarrow$ fewer iterations. Secant converges superlineally $\rightarrow$ more iterations. However each Newton iteration cost more than that of secant iteration. $\Rightarrow$ In total, secant method converges faster in TIME. $\triangle$

Why do we need Quasi-Newton:

Consider $f : \mathbb{R}^n \rightarrow \mathbb{R}$. In a Newton iteration, we set $\mathbf{p}_k$ to be the solution of

$$\nabla^2 f(\mathbf{x}_k)\, \mathbf{p}_k = -\nabla f(\mathbf{x}_k)$$

However, forming the Hessian matrix $\nabla^2 f(x_k)$ cost $O(n^2)$ and solving $\mathbf{p}_k$ cost $O(n^3)$. The Quasi-Newton wants to improve this by replacing $\nabla^2 f(\mathbf{x}_k)$ with an approximation $\mathbf{B}_k$.

From Newton to Quasi-Newton:

Similar to the approximation $f'(x_k) \approx \frac{f(x_k)-f(x_{k-1})}{x_k - x_{k-1}}$ used in the secant method, the Quasi-Newton method uses an approximation $\mathbf{B}_k$ of the Hessian matrix $\mathbf{H} = \nabla^2 f(\mathbf{x}_k)$ where $\mathbf{B}_k$ is built-up incrementally as a running sum:

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \mathbf{C}_k$$

Setting $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ and $\mathbf{y}_k = \nabla f(\mathbf{x}_{k-1}) - \nabla f(\mathbf{x}_k)$, we get the secant condition for the Quasi-Newton method:

$$\mathbf{B}_{k+1}\mathbf{s}_k = \mathbf{y}_k$$

Having only the secant condition is not enough, since $\mathbf{B}_k$ has n^2 parameters but secant condition only provides n equations. Hence, in addition to this, we also want $\mathbf{B}_k$ to satisfy some nice properties.

There are three competing concerns to guide what props we want:

1. We want our iteration to be cheap. $\longrightarrow$ One way to make our iteration cheap is to force $\mathbf{C}_k$ to be low-ranked.
2. $\mathbf{B}_k$ to be symmetric.
3. $\mathbf{B}_k$ to be positive (semi) definite.

2.1 The Broyden's Method:

Broyden suggests using the current estimate of the Jacobian matrix $\mathbf{B}_{k-1}$ and improving upon it by taking the solution to the secant equation that is a minimal modification to $\mathbf{B}_k$:

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k}{\|\mathbf{s}_k\|^2}\mathbf{s}_k^{\mathrm{T}} \quad \text{where} \quad \mathbf{B}_{k+1} = \arg\min_{\mathbf{B}_{k+1}} \|\mathbf{B}_{k+1} - \mathbf{B}_k\|_{\mathrm{F}} \quad \text{subject to} \quad \mathbf{B}_{k+1} = \mathbf{B}_{k+1}^{\top}, \quad \mathbf{B}_{k+1}\mathbf{s}_k = \mathbf{y}_k$$

By the Sherman-Morrison formula, We eventually have $\mathbf{B}_k\mathbf{p}_k = -\nabla f(\mathbf{x}_k)$, $\mathbf{B}_k^{-1} = \mathbf{H}_k$:

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \frac{\mathbf{s}_k - \mathbf{H}_k\mathbf{y}_k}{\mathbf{s}_k^{\top}\mathbf{H}_k\mathbf{y}_k}\mathbf{s}_k^{\top}\mathbf{H}_k$$

2.2 The Symmetric Rank-1 Update (SR1):

Let $\mathbf{B}_{k+1}$, $\mathbf{B}_k$, and $\mathbf{C}_k$ be symmetric where $\mathbf{C}_k = \gamma\mathbf{w}\mathbf{w}^{\top}$ is rank 1 and $(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^{\top}\mathbf{s}_k \neq 0$. Then, the symmetric rank-1 update is given as:

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \mathbf{C}_k, \quad \mathbf{C}_k = \frac{(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^{\top}}{(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^{\top}\mathbf{s}_k}$$

By the Sherman-Morrison formula, We eventually have $\mathbf{B}_k\mathbf{p}_k = -\nabla f(\mathbf{x}_k)$, $\mathbf{B}_k^{-1} = \mathbf{H}_k$:

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \frac{(\mathbf{s}_k - \mathbf{H}_k\mathbf{y}_k)(\mathbf{s}_k - \mathbf{H}_k\mathbf{y}_k)^{\top}}{(\mathbf{s}_k - \mathbf{H}_k\mathbf{y}_k)^{\top}\mathbf{y}_k}, \quad \mathbf{p}_k = -\mathbf{H}_k\nabla f(\mathbf{x}_k)$$

2.3 The DFP Update (SR2):

We use the current estimate of the Jacobian matrix $\mathbf{B}_k$ and improving upon it by taking the solution to the secant equation that is a minimal modification to $\mathbf{B}_k$. That is,

$$\mathbf{B}_{k+1} = \arg\min_{\mathbf{B}_{k+1}} \|\mathbf{B}_{k+1} - \mathbf{B}_k\|_{\mathrm{F}} \quad \text{subject to} \quad \mathbf{B}_{k+1} = \mathbf{B}_{k+1}^{\top}, \quad \mathbf{B}_{k+1}\mathbf{s}_k = \mathbf{y}_k$$

The DFP update scheme is then given by:

$$\mathbf{B}_{k+1} = \left(\mathbf{I} - \rho_k \mathbf{y}_k \mathbf{s}_k^{\top}\right) \mathbf{B}_k \left(\mathbf{I} - \rho_k \mathbf{s}_k \mathbf{y}_k^{\top}\right) + \rho_k \mathbf{y}_k \mathbf{y}_k^{\top} \quad \text{where} \quad \rho_k = \frac{1}{\mathbf{y}_k^T \mathbf{s}_k}$$

We eventually have $\mathbf{B}_k \mathbf{p}_k = -\nabla f(\mathbf{x}_k)$, $\mathbf{B}_k^{-1} = \mathbf{H}_k$:

$$\mathbf{H}_{k+1} = \mathbf{H}_k - \frac{\mathbf{H}_k \mathbf{y}_k \mathbf{y}_k^{\top} \mathbf{H}_k}{\mathbf{y}_k^{\top} \mathbf{H}_k \mathbf{y}_k} + \frac{\mathbf{s}_k \mathbf{s}_k^{\top}}{\mathbf{y}_k^{\top} \mathbf{s}_k}$$

It is also known that the DFP method is less effective in correcting bad Hessian approximations; this property is believed to be the reason for its poorer practical performance.

It is interesting to note that the DFP and BFGS updating formulae are duals of each other, in the sense that one can be obtained from the other by the interchanges $\mathbf{s} \longleftrightarrow \mathbf{y}$, $\mathbf{B} \longleftrightarrow \mathbf{H}$. This symmetry is not surprising, given the manner in which we derived these methods above.

2.4 The BFGS Update (SR2):

There is no rank-one update formula that maintains both symmetry and positive definiteness of the Hessian approximations. However, there are infinitely many rank-two formulas that do this. The most popular and the most effective, is the BFGS update formula.

In order to maintain the symmetry and positive definiteness of $\mathbf{B}_{k+1}$, the update form can be chosen as $\mathbf{B}_{k+1} = \mathbf{B}_k + \alpha \mathbf{u}\mathbf{u}^{\top} + \beta \mathbf{v}\mathbf{v}^{\top}$ where $\mathbf{B}_k$ is a symmetric positive-definite matrix. Imposing the secant condition, $\mathbf{B}_{k+1}\mathbf{s}_k = \mathbf{y}_k$. Choosing $\mathbf{u} = \mathbf{y}_k$ and $\mathbf{v} = \mathbf{B}_k \mathbf{s}_k$, we can obtain $\alpha = \frac{1}{\mathbf{y}_k^{\top} \mathbf{s}_k}$ and $\beta = -\frac{1}{\mathbf{s}_k^{\top} \mathbf{B}_k \mathbf{s}_k}$. Then, we have the following updating formula:

$$\mathbf{B}_{k+1} = \mathbf{B}_k - \frac{(\mathbf{B}_k s_k)(\mathbf{B}_k \mathbf{s}_k)^T}{\mathbf{s}_k^T \mathbf{B}_k \mathbf{s}_k} + \frac{\mathbf{y}_k \mathbf{y}_k^T}{\mathbf{y}_k^T \mathbf{s}_k}$$

and $\mathbf{B}_{k+1}$ is positive definite if and only if $\mathbf{y}_k^T \mathbf{s}_k > 0$ (curvature condition).

A naive implementation of this variant is not efficient for unconstrained minimization, because it requires the system $\mathbf{B}_k \mathbf{p}_k = -\nabla f_k$ to be solved for the step $\mathbf{p}_k$, thereby increasing the cost of the step computation to $O\left(n^3\right)$. We discuss later, however, that less expensive implementations of this variant are possible by updating Cholesky factors of $\mathbf{B}_k$.

Setting $\rho_k = \frac{1}{\mathbf{y}_k^T \mathbf{s}_k}$, and by the Sherman-Morrison formula, we eventually have $\mathbf{B}_k \mathbf{p}_k = -\nabla f(\mathbf{x}_k)$, $\mathbf{B}_k^{-1} = \mathbf{H}_k$:

$$\mathbf{H}_{k+1} = \left(\mathbf{I} - \rho_k \mathbf{s}_k \mathbf{y}_k^T\right) \mathbf{H}_k \left(\mathbf{I} - \rho_k \mathbf{y}_k \mathbf{s}_k^T\right) + \rho_k \mathbf{s}_k \mathbf{s}_k^T$$

Each iteration can be performed at a cost of $O(n^2)$ arithmetic operations (plus the cost of function and gradient evaluations); there are no $O(n^3)$ operations such as linear system solves or matrix-matrix operations. The algorithm is robust, and its rate of convergence is superlinear, which is fast enough for most practical purposes.

Remark 2.2. Even though Newton's method converges more rapidly (that is, quadratically), its cost per iteration usually is higher, because of its need for second derivatives and solution of a linear system. △

2.5 The Broyden Family

So far, we have described the BFGS, DFP, and SR1 quasi-Newton updating formulae, but there are many others. Of particular interest is the Broyden class, a family of updates specified by the following general formula:

$$\mathbf{B}_{k+1} = \mathbf{B}_k - \frac{\mathbf{B}_k \mathbf{s}_k \mathbf{s}_k^T \mathbf{B}_k}{\mathbf{s}_k^T \mathbf{B}_k \mathbf{s}_k} + \frac{\mathbf{y}_k \mathbf{y}_k^T}{\mathbf{y}_k^T \mathbf{s}_k} + \phi_k \left(\mathbf{s}_k^T \mathbf{B}_k \mathbf{s}_k\right) \mathbf{v}_k \mathbf{v}_k^T \quad \text{where} \quad \mathbf{v}_k = \left[\frac{\mathbf{y}_k}{\mathbf{y}_k^T \mathbf{s}_k} - \frac{\mathbf{B}_k \mathbf{s}_k}{\mathbf{s}_k^T \mathbf{B}_k \mathbf{s}_k}\right]$$

and ϕ_k is a scalar parameter.

We can rewrite as a "linear combination" of these two methods:

$$\mathbf{B}_{k+1} = (1 - \phi_k)\, \mathbf{B}_{k+1}^{\mathrm{BFGS}} + \phi_k \mathbf{B}_{k+1}^{\mathrm{DFP}}$$

Specifically, the BFGS and DFP methods are members of the Broyden class—we recover BFGS by setting $\phi_k = 0$ and DFP by setting $\phi_k = 1$.

1. All members of the Broyden class satisfy the secant condition, since the BFGS and DFP matrices themselves satisfy this equation.
2. The parameter ϕ is, in general, allowed to vary from one iteration to another.
3. A Broyden family is defined by a sequence $\phi_1, \phi_2, \cdots$, of parameter values.
4. For $0 \leqslant \phi \leqslant 1$, the Hessian approximations when $\mathbf{s}_k^T \mathbf{y}_k > 0$ are positive definite since BFGS and DFP updating preserve positive definiteness of the Hessian approximations when $\mathbf{s}_k^T \mathbf{y}_k > 0$.
5. For $\phi < 0$ and $\phi > 1$ there is the possibility that the Hessian approximations may become singular.
6. In practice, $0 \leqslant \phi \leqslant 1$ is usually imposed to avoid difficulties.

2.6 Comparing SR1, Broyden, DFP and BFGS:

1. The SR1 update does NOT maintain the positive definteness of the Hessian approximation $\mathbf{B}_k$ (also $\mathbf{H}_k$).
2. The SR1 update generated $\mathbf{B}_k$ is a better approximations to the true Hessian matrix than the BFGS approximations.
3. The condition $(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^\top \mathbf{s}_k \neq 0$ might not hold which is required in the SR1 update. In fact, even when the objective function is a convex quadratic, there may be steps on which there is no symmetric rank-1 update that satisfies the secant equation.
4. In Quasi-Newton methods for constrained problems, it may not be possible to impose the curvature condition $\mathbf{y}_k^T\mathbf{s}_k >> 0$ required in the BFGS framework.
5. The BFGS update maintains the positive definteness of the Hessian approximation $\mathbf{B}_k$ (also $\mathbf{H}_k$)

Remark 2.3.

1. Since storing the Hessian takes $O\left(n^2\right)$ space, for very large problems, one can use limited memory BFGS, or L-BFGS, where $\mathbf{H}_k$ or $\mathbf{H}_k^{-1}$ is approximated by a diagonal plus low rank matrix and the product $\mathbf{H}_k^{-1}\nabla f(\mathbf{x}_k)$ can be obtained by performing a sequence of inner products.
2. There is also a generalization of L-BFGS which handles bound constraints, ie. constraints of the form

$$\ell_i \leq x_i \leq u_i$$

where $\ell_i < u_i$ are real-valued bounds.

These are sometimes referred to as box constraints, and arise often in statistical parameter estimation problems. For example, one might want to enforce positivity for a variance parameter.

△

3 Topic: Descent

Descent direction:

We call $\mathbf{p}_k$ a descent direction if

$$\mathcal{D}f(\mathbf{x}_k)\,\mathbf{p}_k < 0$$

If $\mathbf{p}_k$ is a descent direction then you shall be able to choose $\alpha_k > 0$ s.t. $f(\mathbf{x}_k + \alpha_k \mathbf{p}_k) < f(\mathbf{x}_k)$ (FYI: this is for $\mathbf{x}_{n+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$ such as:

$$\begin{aligned} Gradient Descent: &-\nabla f(\mathbf{x}_k) \\ \text{Newton:} &-\nabla^2 f(\mathbf{x}_k)\,\nabla f(\mathbf{x}_k) \\ Gauss-Newton: &-\mathcal{D}f(\mathbf{c}_k)^{\dagger}(\mathbf{y} - f(\mathbf{c}_k)) \end{aligned}$$

Line Search:

1. **Exact Line Search:** Idea: Find α_k that will minimize f at every step (not necessarily the ultimate minimum). This is because the gradient only points "down" but not directly towards the minimum.

   $$\alpha_k = \arg\min_{\alpha>0} f(\mathbf{x}_k + \alpha \mathbf{p}_k)$$

   However, exact line search can be very COSTLY computationally. Further more, the steepest descent path with exact line-search exhibits a characteristic zig-zag behavior. This is because the necessay condition for α_k is

   $$\phi(\alpha_k) = f(\mathbf{x}_k + \alpha_k \mathbf{p}_k) \Longrightarrow \phi'(\alpha_k) = (\nabla f)(\mathbf{x}_k + \alpha_k \mathbf{p}_k) \cdot \mathbf{p}_k = 0$$

   which means that ∇f at the end of the step is orthogonal to $\mathbf{p}_k$, the descent direction, whenever $\mathbf{p}_k \neq 0$.

2. **Heuristic Line Search:** $\alpha_k^{(0)} := 1$; check if $f\left(\mathbf{x}_k + \alpha_k^{(0)} \mathbf{p}_k\right) < f(\mathbf{x}_k)$, if not, set $\alpha_k^{(1)} = \frac{1}{2}\alpha_k^{(0)}$ and so on.

   e.g. First try $\alpha_k = 1$ —> Failure? —> try $\alpha_k = \frac{1}{2}\alpha_k = \frac{1}{2}$ and so on.

3. **Backtracking Line Search:** This is just the Heuristic Line Search + the Armijo condition.

   Setting $\mu \in (0, 1/2), \beta \in (0,1)$, and given a descent direction $\mathbf{p}_k$, then starting at $\alpha_k^{(0)} = 1$, repeat $\alpha_k^{(k)} := \beta \alpha_k^{(k-1)}$ until

   $$f(\mathbf{x}_k + \alpha_k^{(k)} \mathbf{p}_k) < f(\mathbf{x}_k) + \mu \alpha_k^{(k)} \nabla^{\top} f(\mathbf{x}_k)\,\mathbf{p}_k \quad \text{(Armijo condition)}$$

   Since $\mathbf{p}_k$ is assumed to be a descent direction, we have $\nabla^{\top} f(\mathbf{x}_k)\,\mathbf{p}_k < 0$, so for small enough $\alpha_k^{(k)}$ we have

   $$f(\mathbf{x}_k + \alpha_k^{(k)} \mathbf{p}_k) \approx f(\mathbf{x}_k) + \alpha_k^{(k)} \nabla^{\top} f(\mathbf{x}_k)\,\mathbf{p}_k < f(\mathbf{x}_k) + \mu \alpha_k^{(k)} \nabla^{\top} f(\mathbf{x}_k)\,\mathbf{p}_k$$

   This shows that the backtracking line search eventually terminates.

   The constant μ can be interpreted as the fraction of the decrease in f predicted by linear extrapolation that we will accept.

General conditions for descent algorithm $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}_k \alpha_k$ to converge:

Suppose $\alpha_k \in [0,1]$ is chosen using backtracking line search and $\mathbf{p}_k$ is the "search direction", we need

1. **Armijo condition on α_k (Assures descent):** For

   $$f(\mathbf{x}_k + \alpha_k \mathbf{p}_k) \leqslant f(\mathbf{x}_k) + \mu \alpha_k \nabla f(\mathbf{x}_k)^{\top} \mathbf{p}_k \text{ where } \mu_k \in (0,1) \text{ w.r.t. every } k$$

2. **Angle condition (Avoids running in circles – very slow):**

   $$-\frac{\mathbf{p}_k^{\top} \nabla f(\mathbf{x}_k)}{\|\mathbf{p}_k\|_2 \|\nabla f(\mathbf{x}_k)\|_2} = \cos\theta \geqslant \varepsilon \longrightarrow \text{ constant w.r.t. } k$$

3. **Gradient condition (Assures sufficient descent step's magnitude):**

   $$\exists m > 0, \text{ s.t. } \|\mathbf{p}_k\| \geqslant m \|\nabla f(\mathbf{x}_k)\|_2 .$$

   This condition specifies that the magnitude of $\mathbf{p}_k$ should NOT be too much smaller than that of the gradient $\nabla f(\mathbf{x}_k)$.

Theorem (Griva 11.7)

Let $S : \mathbb{R}^n \to \mathbb{R}$ set. ∇f is Lipchitz with constant L (i.e. $\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\|_2 \leq L \|\mathbf{x}_1 - \mathbf{x}_2\|_2$). Let $\mathbf{x}_0 \in \mathbb{R}^n$, and consider the iteration $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$ where α_k is chosen using the Backtracking line search with $\alpha_k = 1, \frac{1}{2}, \frac{1}{4}, \ldots$ satisfying the following properties:

1. The set $S = \{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}) \leqslant f(\mathbf{x}_k)\}$ is bounded.
2. The $\mathbf{p}_k$'s satisfy the angle condition with parameter $\varepsilon > 0$.
3. The $\mathbf{p}_k$'s are gradient-related with constant $m > 0$.
4. $\|\mathbf{p}_k\|_2 \leq M < \infty \quad \forall k \geq 0$.
5. $\alpha_k \in (0,1]$ is the first $\alpha_k \in \left\{1, \frac{1}{2}, \frac{1}{4}, \cdots\right\}$ such that the Armijo condition holds.

Then, we have $\lim_{k\to\infty} \|\nabla f(\mathbf{x}_k)\|_z = 0$.

3.1 Gradient Descent

$$\mathbf{x}_{n+1} = \mathbf{x}_k - \alpha \nabla f(\mathbf{x}_k)$$

Note: The negative gradient is always a descent direction unless we are already at a point where the gradient vanishes. This is because if $\nabla f(\mathbf{x}_k) \neq 0$, then

$$\mathcal{D} f(\mathbf{x}_k) \cdot (-\nabla f(\mathbf{x}_k)) = -\nabla^\top f(\mathbf{x}_k) \nabla f(\mathbf{x}_k) = -\|\nabla f(\mathbf{x}_k)\|^2 < 0$$

satisfies the condition for $\nabla f(\mathbf{x}_k)$ being a descent direction. If $\nabla f(\mathbf{x}_k) = 0$, this implies that we have arrived at a critical point. In other words,

Why do we need learning rate? Ans: Because the iteration might bounce back and forth, think $f(x) = x^2$.

4 Comparing the Newton's Method and Gradient Descent

Newton's Method is useful, but it's costly: $O(N^3)$. Newton's Method might not converge if it's not sufficiently close to the optimal.

What are the advantages and disadvantages of Newton's Method compared to Gradient Descent?

1. Gradient Descent is parametric and requires choosing the step size, while Newton's Method does not have hyperparameters.
2. Gradient Descent needs more iterations, but each iteration has smaller complexity, $O(n)$ compared to $O(n^3)$ for Newton's Method.
3. Newton's Method requires second-order differentiability.
4. They exhibit different behavior around stationary points.