Numerical Nonlinear Optimization Part I

Andreas Wächter

Center for Nonlinear Studies

June 22, 2020

Goal of this Lecture Mini-Series

- Accessible to broad audience.
  - No prior knowledge of optimization required.
  - Assume basic knowledge of multi-dimensional calculus.

- Give overview of practical optimization algorithms for nonlinear constrained optimization.

- Concentrate on intuition of algorithmic ideas.
  - No complicated proofs.
  - Some "cheating" (ignoring some subtleties).

- 90 min reserved, but roughly targeting 60 min.

- I will make slides available after the lectures.

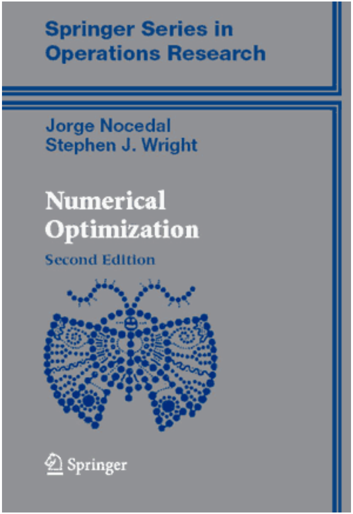
Constrained Nonlinear Optimization Problems

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \ & f(x) \\ \text{s.t. } & c_E(x) = 0 \\ & c_I(x) \leq 0 \end{aligned}$$

$$\begin{aligned} f &: \mathbb{R}^n \longrightarrow \mathbb{R} \\ c_E &: \mathbb{R}^n \longrightarrow \mathbb{R}^p \\ c_I &: \mathbb{R}^n \longrightarrow \mathbb{R}^q \end{aligned}$$

- We assume that all functions are twice continuously differentiable.
- Example applications:
  - Optimal operation of electricity or gas networks.
  - Optimal control of a chemical plant.
  - Transistor sizing in digital circuits.
  - Inverse problems (fit coefficients in PDEs).

Book Recommendation



Part 1 (Today+): Unconstrained Optimization

- Optimality conditions for unconstrained optimization.
- Basic algorithms:
  - Gradient method
  - Newton's method
  - Quasi-Newton methods
- Strategies ensuring convergence:
  - Line-search method
  - Trust-region method
- Will not cover stochastic gradient method (for machine learning problems with large data sets).

Later: Constrained Optimization

- Optimality conditions for constrained optimization.
- Solving quadratic programs
  - with equality constraints
  - with inequality constraints
- Sequential Quadratic Programming (SQP) methods
- Interior-point methods

Unconstrained Optimization Problems

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

- We assume that f is (twice) continuously differentiable.
- We deal with continuous variables in finite-dimensional space.

Examples:

- Nonlinear regression
  - Fit model parameters to data.
- Inverse problems
  - Fit PDE coefficients to observations.
  - Determine initial conditions for weather prediction.

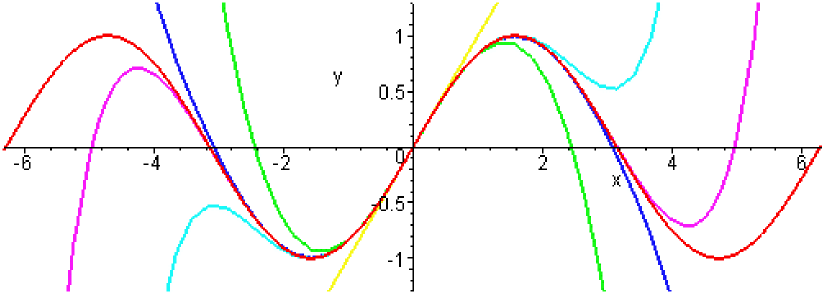
Types of Minimizers

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

- A point $x^* \in \mathbb{R}^n$ is a global minimizer of f, if $f(x) \geq f(x^*)$ for all $x \in \mathbb{R}^n$.
- A point $x^* \in \mathbb{R}^n$ is a local minimizer of f, if $f(x) \geq f(x^*)$ for all $x \in N_\epsilon(x^*) = \{x \in \mathbb{R}^n : \|x - x^*\| \leq \epsilon\}$ for some $\epsilon > 0$.
- The methods we will discuss try to find local minimizers.

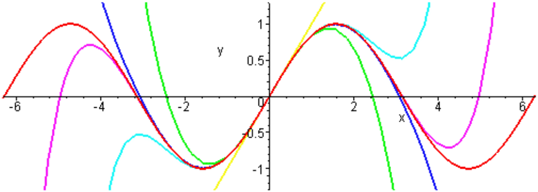
Main Tool: Taylor Expansions

$$\boxed{f(\bar{x}+d) \approx f(\bar{x}) + f'(\bar{x})\cdot d + \frac{1}{2}f''(\bar{x})\cdot d^2 + \frac{1}{3!}f'''(\bar{x})\cdot d^3 + \dots}$$



Example: $f(x) = \sin(x)$ with $\bar{x} = 0$.

Main Tool: Taylor Expansions



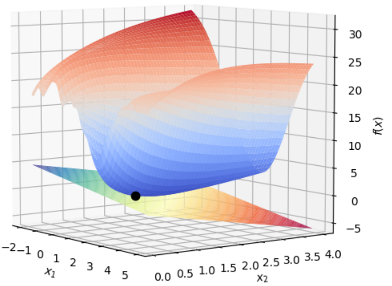
- Provide local models of functions around a reference point.
- Algorithms use them to figure out where to go next.

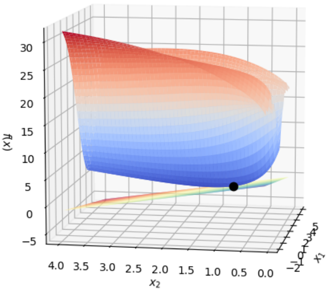
- Methods only need values and derivatives at specific points $\bar{x}$.
- Do not need to assume particular representation of objective f.
  - No analytical expression required.
  - Could be result of complicated computational procedure.

First-Order Taylor Expansion in Multiple Dimensions

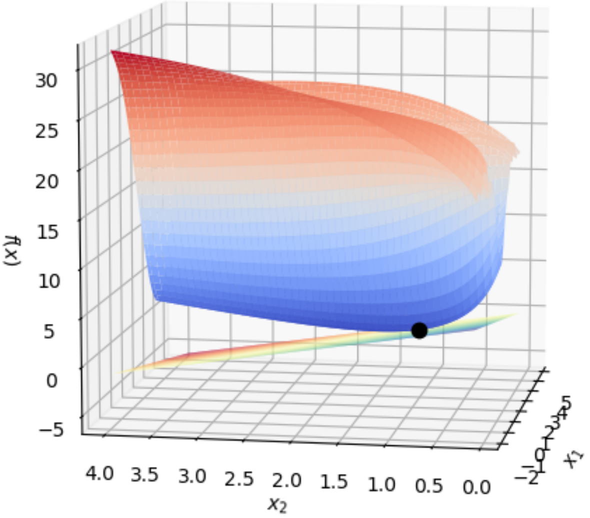
$$\boxed{f(\bar{x}+d) \approx f(\bar{x}) + \nabla f(\bar{x})^T d}$$

Gradient: $\nabla f(x) = \begin{pmatrix} \frac{\partial f}{\partial x_1}(x) \\ \frac{\partial f}{\partial x_2}(x) \\ \vdots \\ \frac{\partial f}{\partial x_n}(x) \end{pmatrix}$





First-Order Optimality Conditions



First-Order Optimality Conditions

$$\boxed{f(x^* + d) \approx f(x^*) + \nabla f(x^*)^T d}$$

- Suppose x^* is a local minimizer of f.
- x^* must be a minimizer along any direction $d \in \mathbb{R}^n$:

$$f(x^* + t \cdot d) \approx g(t) := f(x^*) + \nabla f(x^*)^T d \cdot t$$

- So, $t^* = 0$ must be a local minimizer of $g(t)$.
- From 1-dim calculus:

$$0 = g'(t) = \nabla f(x^*)^T d$$

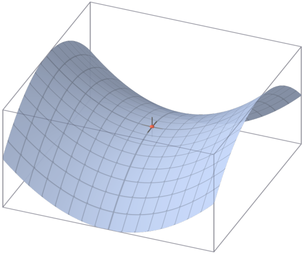
- Since this is $\underline{\text{true for every } d \in \mathbb{R}^n}$, we must have $\nabla f(x^*) = 0$.

First-Order Optimality Conditions

Theorem (First-Order Necessary Condition)

Let $f \in C^1$ and $x^* \in \mathbb{R}^n$ be a local minimizer of f. Then

$$\boxed{\nabla f(x^*) = 0.}$$



Comments

- We call such a point a stationary point.
- This is not a sufficient condition.
- Also maximizers and saddle points are stationary points.

Second-Order Optimality Conditions (1-dim)

$$\boxed{\min_{x \in \mathbb{R}} \; f(x)}$$

Theorem (Second Order Necessary Condition)

Let $f \in C^2$ and $x^* \in \mathbb{R}$ be a local minimizer. Then

$$\boxed{f'(x^*) = 0 \;\; \textit{and} \;\; f''(x^*) \geq 0.}$$

Theorem (Second Order Sufficient Condition)

Let $f \in C^2$ and $x^* \in \mathbb{R}$ be such that

$$\boxed{f'(x^*) = 0 \;\; \textit{and} \;\; f''(x^*) > 0.}$$

Then x^* is a strict local minimizer.

Second-Order Taylor Model in Higher Dimensions

$$\boxed{f(\bar{x}+d) \approx f(\bar{x}) + \nabla f(\bar{x})^T d + \tfrac{1}{2} d^T \nabla^2 f(\bar{x}) d}$$

Hessian matrix:

$$\nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2}(x) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(x) & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n}(x) \\ \frac{\partial^2 f}{\partial x_2 \partial x_1}(x) & \frac{\partial^2 f}{\partial x_2^2}(x) & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n}(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1}(x) & \frac{\partial^2 f}{\partial x_n \partial x_2}(x) & \cdots & \frac{\partial^2 f}{\partial x_n^2}(x) \end{bmatrix}$$

- If $f \in C^2$, then $\nabla^2 f(x)$ is symmetric.

Second-Order Optimality Conditions

$$\boxed{f(x^* + d) \approx f(x^*) + \nabla f(x^*)^T d + \tfrac{1}{2} d^T \nabla^2 f(x^*) d}$$

- If x^* is a local minimizer of f, then $t^* = 0$ is a local minimizer of

  $$f(x^* + t \cdot d) \approx g(t) = f(x^*) + \nabla f(x^*)^T d \cdot t + \tfrac{1}{2} d^T \nabla^2 f(x^*) d \cdot t^2$$

  for any $d \in \mathbb{R}^n$.
- This implies that for all $d \in \mathbb{R}^n$:

  $$0 = g'(0) = \nabla f(x^*)^T d$$
  $$0 \le g''(0) = d^T \nabla^2 f(x^*) d$$

- So, $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ must be positive semi-definite.

Second-Order Optimality Conditions (n-dim)

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

Theorem (Second Order Necessary Condition)

Let $f \in C^2$ and $x^* \in \mathbb{R}^n$ be a local minimizer. Then

$$\nabla f(x^*) = 0 \;\; \textit{and} \;\; \nabla^2 f(x^*) \textit{ is positive semi-definite.}$$

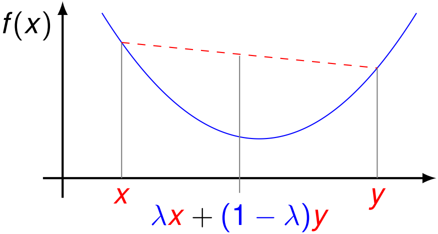
Theorem (Second Order Sufficient Condition)

Let $f \in C^2$ and $x^* \in \mathbb{R}^n$ be such that

$$\nabla f(x^*) = 0 \;\; \textit{and} \;\; \nabla^2 f(x^*) \textit{ is positive definite.}$$

Then x^* is a strict local minimizer.

Special Case: Convex Functions



Definition (Convex Function)

A function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ is convex if

$$\boxed{f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)}$$

for all points $x, y \in \mathbb{R}^n$ and all $\lambda \in (0,1)$.

Special Case: Convex Problems

- All stationary points of a convex function are global minimizers!
- f is convex if and only if $\nabla^2 f(x)$ is positive semi-definite everywhere.
- Recall: For symmetric matrix Q

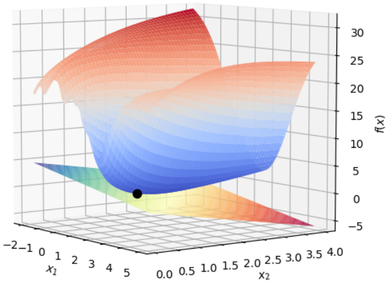
$$\boxed{\begin{array}{c} Q \text{ is positive semi-definite [definite]} \\ \Updownarrow \\ \text{All eigenvalues of } Q \text{ are } \geq 0 \ [> 0] \end{array}}$$

- For convex quadratic function $f(x) = c + g^T x + x^T Qx$:

$$\nabla f(x^*) = g + 2Qx^* = 0 \qquad \Longrightarrow \qquad \boxed{x^* = -\tfrac{1}{2} Q^{-1} g}$$

where $Q \in \mathbb{R}^{n \times n}$ is symmetric positive definite.

First Algorithm: Going Downhill



$$\boxed{f(x_k + d) \approx f(x_k) + \nabla f(x_k)^T d}$$

- To go downhill, choose direction d such that $\nabla f(x_k)^T d < 0$.
- d forms an acute angle with $-\nabla f(x_k)$.
- Steepest descent direction: $d = -\nabla f(x_k)$.

Basic Gradient Method

Given: Stopping tolerance $\epsilon > 0$.

1: Choose starting point $x_0 \in \mathbb{R}^n$ and set $k \leftarrow 0$.
2: **while** $\|\nabla f(x_k)\| > \epsilon$ **do**
3: Compute gradient step

$$\boxed{d_k = -\nabla f(x_k).}$$

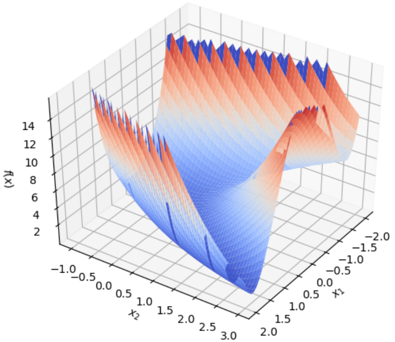
4: Take step

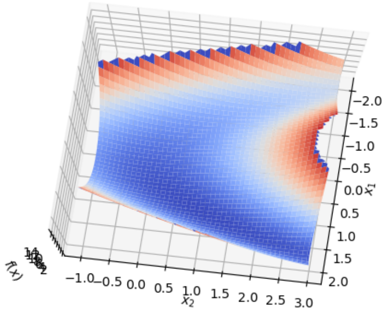
$$\boxed{x_{k+1} = x_k + d_k.}$$

5: Increase iteration counter $k \leftarrow k + 1$.
6: **end while**

Example Problem: Rosenbrock Function

$$\boxed{f(x) = 2 \cdot (x_2 - x_1^2)^2 + (x_1 - 1)^2} \qquad x^* = (1, 1)^T$$





Step Size Parameter

Problem:

- $d_k = -\nabla f(x_k)$ gives a direction.
- But its length might be inappropriate to define a step.

Remedy:

- Introduce a step size parameter $\alpha > 0$:

$$\boxed{x_{k+1} = x_k + \alpha \cdot d_k}.$$

Gradient Method with Step Size

- Given:
  - Stopping tolerance $\epsilon > 0$
  - Step size parameter $\alpha > 0$.

1: Choose starting point $x_0 \in \mathbb{R}^n$ and set $k \leftarrow 0$.
2: **while** $\|\nabla f(x_k)\| > \epsilon$ **do**
3: Compute gradient step

$$\boxed{d_k = -\nabla f(x_k).}$$

4: Take step

$$\boxed{x_{k+1} = x_k + \alpha \cdot d_k.}$$

5: Increase iteration counter $k \leftarrow k + 1$.
6: **end while**

Convergence of Gradient Descent Method

- Choice of step size parameter α:
  - Gradient method does not converge if α is too large.
  - Can be tricky to tune.
  - Converges if $\alpha \in (0, \frac{2}{L})$, where L is Lipschitz constant of $\nabla f(x)$.
- (Slow) linear rate of convergence:

$$f(x_{k+1}) - f(x^*) \leq c \cdot (f(x_k) - f(x^*))$$

  for a constant $c \in (0, 1)$.

- Maybe we can do better if we utilize second-order Taylor expansion?

A Second-Order Method

- At an iterate x_k, consider quadratic Taylor model:

$$\boxed{q_k(x_k + d) = f(x_k) + \nabla f(x_k)^T d + \frac{1}{2} d^T \nabla^2 f(x_k) d}$$

Given: Stopping tolerance $\epsilon > 0$.

1: Choose starting point $x_0 \in \mathbb{R}^n$ and set $k \leftarrow 0$.
2: **while** $\|\nabla f(x_k)\| > \epsilon$ **do**
3: Compute the minimizer d_k of

$$\min_{d \in \mathbb{R}^n} q_k(x_k + d).$$

4: Take step

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

5: Increase iteration counter $k \leftarrow k + 1$.
6: **end while**

Second-Order Steps

$$\boxed{q(x_k + d) = f(x_k) + \nabla f(x_k)^T d + \frac{1}{2} d^T \nabla^2 f(x_k) d}$$

- What is the minimizer of $q_k(x_k + d)$?
- Use formula for quadratic functions:

$$\boxed{d_k = -[\nabla^2 f(x_k)]^{-1} \nabla f(x_k)}$$

- This assumes that $\nabla^2 f(x_k)$ is positive definite.
- Computationally, **NEVER compute the inverse!**
- Instead solve the linear system

$$\boxed{\nabla^2 f(x_k) \cdot d = -\nabla f(x_k).}$$

- Can be done for very large problems if $\nabla^2 f(x_k)$ is structured.

Alternative: Newton's Method

- Recall: First-order optimality condition: $\nabla f(x^*) = 0$.
- This is a nonlinear system of equations:

$$\boxed{F(x^*) = 0} \qquad\qquad F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$$

- Newton's method is very efficient for solving those.

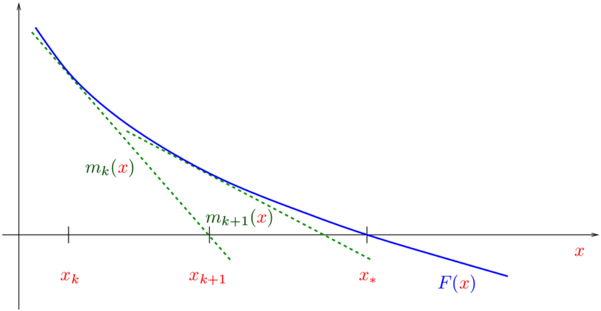
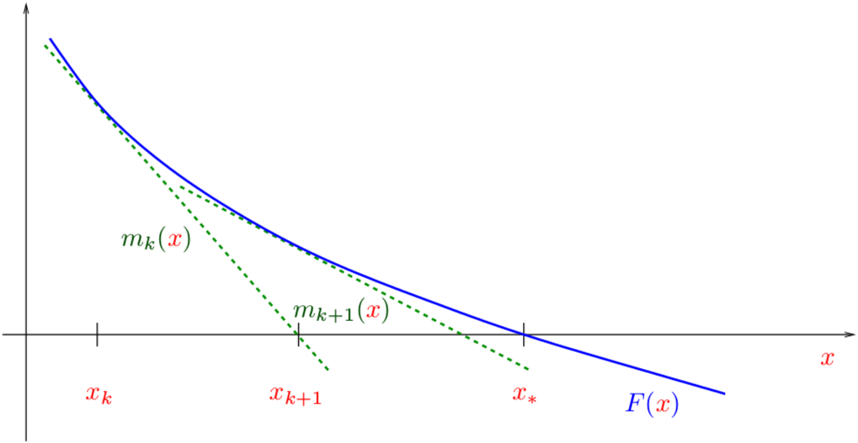


Illustration of Newton's Method



Newton's Method For System of Equations

$$\boxed{F(x^*) = 0}$$

- First-order Taylor model:

$$F(x_k + d) \approx F(x_k) + \nabla F(x_k)^T d$$

  where $\nabla F(x_k)^T$ is Jacobian matrix of F.
- Compute step as root of linear model:

$$F(x_k) + \nabla F(x_k)^T d_k = 0$$

- So

$$\boxed{d_k = -[\nabla F(x_k)^T]^{-1} F(x_k)}$$

Newton's Method For Stationary Point

- First-order optimality condition:

$$\boxed{F(x^*) = \nabla f(x^*) = 0}$$

- Newton step for $F(x^*) = 0$:

$$d_k = -[\nabla F(x_k)^T]^{-1} F(x_k)$$

- Newton step for $\nabla f(x^*) = 0$:

$$\boxed{d_k = -[\nabla^2 f(x_k)]^{-1} \nabla f(x_k)}$$

- This is the second-order step from earlier!

Two Perspectives

- Root-finding problem:
  - We can use well-established Newton's method and theory.
  - Fast local quadratic convergence rate:

$$\|x_{k+1} - x^*\| \leq M \cdot \|x_k - x^*\|^2$$

    for some constant $M > 0$, starting x_0 close to x^*.
  - "Double the number of accurate digits in every iteration"

- Model minimization:

$$\boxed{\min\ q(x_k + d) = f(x_k) + \nabla f(x_k)^T d + \frac{1}{2} d^T \nabla^2 f(x_k) d}$$

  - We keep in mind that we are not only looking for stationary points.
  - We know we need to be careful if model does not have minimizer.
    - Check if $\nabla^2 f(x_k)$ is positive definite.
    - Change steps to avoid moving towards a non-minimizer.

Generalized Model

- Quadratic model:

$$\boxed{q_k(x_k + d) = f(x_k) + \nabla f(x_k)^T d + \tfrac{1}{2} d^T B_k d}$$

where B_k is some symmetric positive definite matrix.

- Minimizer:

$$d_k = -[B_k]^{-1} \cdot \nabla f(x_k).$$

- Variants:

$$\begin{aligned} \text{Newton's method:} \quad & B_k = \nabla^2 f(x_k) \\ \text{Gradient method:} \quad & B_k = \tfrac{1}{\alpha} I \\ \text{Other methods:} \quad & B_k \text{ positive definite} \end{aligned}$$

- Is there a fast method that only uses gradient information?

Secant Method in 1-Dim

$$\boxed{f'(x^*) = 0} \qquad\qquad f : \mathbb{R} \longrightarrow \mathbb{R}$$

- Newton step

$$d_k = -f''(x_k)^{-1} f'(x_k)$$

- Suppose $f''(x_k)$ cannot be evaluated. Can we estimate it?
- Derivative

$$f''(x) = \lim_{y \to x} \frac{f'(x) - f'(y)}{x - y}$$

- Let's suppose we have $x_k, x_{k-1}, \ldots$ and $f'(x_k), f'(x_{k-1}), \ldots$.
- In step computation, replace

$$f''(x_k) \quad \text{with} \quad \frac{f'(x_k) - f'(x_{k-1})}{x_k - x_{k-1}}.$$

Secant Method for *n*-Dim

- Secant step for $f'(x^*) = 0$

$$d_k = -B_k^{-1} f'(x_k) \qquad \text{where} \qquad f''(x_k) \approx B_k = \frac{f'(x_k) - f'(x_{k-1})}{x_k - x_{k-1}}$$

- Note: B_k satisfies the secant condition:

$$B_k(x_k - x_{k-1}) = f'(x_k) - f'(x_{k-1})$$

- What can we do in *n* dimensions?
- Choose a matrix B_k that satisfies the secant condition and compute step

$$d_k = -B_k^{-1} \nabla f(x_k)$$

Secant Condition in Second-Order Method

- Quadratic model in algorithm:

$$q_k(x_k + d) = f(x_k) + \nabla f(x_k)^T d + \frac{1}{2} d^T B_k d$$

- We would like to mimic Newton's method: $B_k \approx \nabla^2 f(x_k)$
- The Hessian approximation should satisfy the secant condition:

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

- There are $\frac{n(n+1)}{2}$ independent entries in the symmetric matrix B_k.
- The secant condition has only n equations.
- For $n > 1$, B_k is not uniquely defined.

Quasi-Newton Methods

$$\boxed{q_k(x_k + d) = f(x_k) + \nabla f(x_k)^T d + \tfrac{1}{2} d^T B_k d}$$

- Idea: Generate a sequence $B_0, B_1, \ldots$ of Hessian approximations satisfying secant condition.

Given: Stopping tolerance $\epsilon > 0$.

Choose x_0 and B_0, and set $k \leftarrow 0$.
while $\|\nabla f(x_k)\| > \epsilon$ **do**
    Compute the minimizer d_k of $q_k(x_k + d)$.
    Take step $x_{k+1} = x_k + d_k$.
    Compute B_{k+1} from some update formula.
    Increase iteration counter $k \leftarrow k + 1$.
end while

Quasi-Newton Update Formula

- Want B_{k+1} to satisfy secant condition:

$$B_{k+1} \cdot s_k = y_k$$

  where $s_k = x_{k+1} - x_k$ and $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$.
- Suppose we believe that B_k is a good approximation of Hessian.
- Idea: Choose symmetric matrix B that is closest to B_k and has desired properties

$$\boxed{\begin{array}{rl} \min\limits_{B \in \mathbb{R}^{n \times n}} & \|B - B_k\| \\ \text{s.t.} & B \cdot s_k = y_k, \quad B = B^T \end{array}}$$

- A variation of this leads to the BFGS formula.

BFGS Formula

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

- Named after Broyden, Fletcher, Goldfarb, and Shanno.

Properties:

- B_{k+1} satisfies secant condition.
- If B_k is symmetric, then B_{k+1} is symmetric.
- If B_k is pos. def. and $s_k^T y_k > 0$, then B_{k+1} is pos. def.
- In practice, use version that approximates $H_k \approx [\nabla^2 f(x_k)]^{-1}$.
  - Then no need to solve linear system, just compute $d_k = -H_k \nabla f(x_k)$.

BFGS Formula

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

- Most-used quasi-Newton update.
- Requires same amount of derivative evaluations as gradient method.
- Converges typically much faster than gradient method.
  - Can prove local superlinear convergence under (strong) assumptions.

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

- B_k is a dense matrix, not suitable for large n.
- There is a “limited-memory” version (L-BFGS) for large n.

Our Algorithm So Far

Given: Stopping tolerance $\epsilon > 0$.

1: Choose x_0 and set $k \leftarrow 0$.
2: **while** $\|\nabla f(x_k)\| > \epsilon$ **do**
3: Compute or update B_k.
4: Compute step $d_k = -B_k^{-1}\nabla f(x_k)$.
5: Take step $x_{k+1} = x_k + d_k$.
6: Increase iteration counter $k \leftarrow k + 1$.
7: **end while**

Concerns:

- Sometimes, this basic algorithm fails to converge.
- The iterates might cycle or diverge.