Numerical optimization
Common denominator: optimization problems
Generic **unconstrained minimization** problem

\[
\min_{x \in \mathbb{X}} f(x)
\]

where

- Vector space $\mathbb{X}$ is the **search space**
- $f : \mathbb{X} \rightarrow \mathbb{R}$ is a **cost** (or **objective**) **function**
- A solution $x^* = \arg\min_{x \in \mathbb{X}} f(x)$ is the **minimizer** of $f(x)$
- The value $f(x^*)$ is the **minimum**
Local vs. global minimum

Find minimum by analyzing the local behavior of the cost function

\[ f(x^*) \leq f(x \in \mathcal{N}(x^*)) \]

\[ f(x^*) \leq f(x \in \mathbb{X}) \]
Local vs. global in real life

Broad Peak (K3), 12th highest mountain on Earth
Convex functions

A function $f : A \subseteq \mathbb{X} \rightarrow \mathbb{R}$ defined on a convex set $A$ is called convex if

$$f(\lambda x + (1 - \lambda)x') \leq \lambda f(x) + (1 - \lambda)f(x')$$

for any $x, x' \in \mathbb{X}$ and $\lambda \in [0, 1]$

For convex function local minimum = global minimum.
One-dimensional optimality conditions

Point $x^*$ is the local minimizer of a $C^2$-function $f : \mathbb{R} \to \mathbb{R}$ if

- $f'(x^*) = 0$
- $f''(x^*) > 0$

Approximate a function around $x^*$ as a parabola using Taylor expansion

$$f(x^* + dx) \approx f(x^*) + f'(x^*)dx + \frac{1}{2}f''(x^*)dx^2$$

$f'(x^*) = 0$ guarantees the minimum at $x^*$

$f''(x^*) > 0$ guarantees the parabola is convex
Gradient

In multidimensional case, linearization of the function according to Taylor

\[ f(x + dx) \approx f(x) + \langle g(x), dx \rangle_x \]

gives a multidimensional analogy of the derivative.

The function \( g : \mathbb{R}^n \rightarrow \mathbb{R}^n \), denoted as \( \nabla f(x) \), is called the gradient of \( f(x) \).

In one-dimensional case, it reduces to standard definition of derivative

\[ \nabla f(x) = \frac{f(x + dx) - f(x)}{dx} \]

\[ = \frac{df(x)}{dx} \quad (dx \rightarrow 0) \]
Gradient

In Euclidean space \( (\mathbb{X} = \mathbb{R}^N) \), \( \nabla f(x) \) can be represented in standard basis \( \{e_1, \ldots, e_N\} \) in the following way:

\[
f(x + \epsilon e_i) \approx f(x) + \langle \nabla f(x), \epsilon e_i \rangle_x
\]

\[
e_i = (0, \ldots, 1, \ldots, 0)
\]

\( i \)-th place

\[
= f(x) + \epsilon (\nabla f(x))^i
\]

\[
\Rightarrow (\nabla f(x))^i = \frac{f(x + \epsilon e_i) - f(x)}{\epsilon}
\]

\[
= \frac{\partial f(x)}{\partial x^i} \quad (\epsilon \to 0)
\]

which gives

\[
\nabla f(x) = \left( \frac{\partial f(x)}{\partial x^1}, \ldots, \frac{\partial f(x)}{\partial x^N} \right)
\]
Example 1: gradient of a matrix function

Given $\mathbb{X} = \mathbb{R}^{N \times M}$ (space of $N \times M$ real matrices) with standard inner product $\langle A, B \rangle_{\mathbb{R}^{N \times M}} = \text{trace}(A^\top B)$

Compute the gradient of the function $f(X) = \text{trace}(AX)$ where $A$ is an $M \times N$ matrix

$$f(X + dX) = \text{trace}(A(X + dX))$$

For square matrices

$$\text{trace}(B^\top) = \text{trace}(B)$$

$$= \text{trace}(AX) + \text{trace}(AdX)$$

$$= f(X) + \text{trace}(dX^\top A^\top)$$

$$= f(X) + \langle A^\top, dX \rangle_{\mathbb{R}^{N \times M}}$$

$$\nabla f(X) = A^\top$$
Example 2: gradient of a matrix function

Compute the gradient of the function $f(X) = \text{trace}(X^\top BX)$ where $B$ is an $N \times N$ matrix

\[
f(X + dX) = \text{trace}((X + dX)^\top B(X + dX)) = \text{trace}(X^\top BX) + \text{trace}(dX^\top BX) + \text{trace}(X^\top BdX) + \text{trace}(dX^\top BdX)
\]

\[\approx f(X) + \text{trace}(dX^\top BX) + \text{trace}(X^\top BdX)
\]

\[
= f(X) + \text{trace}(dX^\top (B + B^\top)X)
\]

\[
= f(X) + \langle (B + B^\top)X, dX \rangle_{\mathbb{R}^{N \times M}}
\]

\[
\nabla f(X) = (B + B^\top)X
\]
Hessian

Linearization of the gradient

$$\nabla f(x + dx) \approx \nabla f(x) + H(x)dx$$

gives a multidimensional analogy of the second-order derivative.

The function $H : \mathbb{R} \rightarrow \mathbb{R} \times \mathbb{R}$, denoted as $\nabla^2 f(x)$, is called the **Hessian** of $f(x)$.

In the standard basis, Hessian is a symmetric matrix of mixed second-order derivatives

$$\nabla^2 f(x) = \left( \frac{\partial^2 f(x)}{\partial x^i \partial x^j} \right)$$
Optimality conditions, \textit{bis}

Point $x^*$ is the local minimizer of a $C^2$-function $f : \mathbb{X} \rightarrow \mathbb{R}$ if

- $\nabla f(x^*) = 0$
- $x^T \nabla^2 f(x^*) x > 0$ for all $x \neq 0$, i.e., the Hessian is a positive definite matrix (denoted $\nabla^2 f(x^*) \succ 0$)

Approximate a function around $x^*$ as a parabola using Taylor expansion:

$$f(x^* + dx) \approx f(x^*) + \nabla f(x^*)^T dx + \frac{1}{2} dx^T \nabla^2 f(x^*) dx$$

$\nabla f(x^*) = 0$ guarantees the minimum at $x^*$

$\nabla^2 f(x^*) \succ 0$ guarantees the parabola is convex
Optimization algorithms

Descent direction

Step size
Generic optimization algorithm

- Start with some \( x^{(0)}, k = 0 \)
- Determine descent direction \( d^{(k)} \)
- Choose step size \( \alpha^{(k)} \) such that

\[
 f(x^{(k)} + \alpha^{(k)} d^{(k)}) < f(x^{(k)})
\]
- Update iterate

\[
 x^{(k+1)} = x^{(k)} + \alpha^{(k)} d^{(k)}
\]
- Increment iteration counter \( k \leftarrow k + 1 \)
- Solution \( x^* \approx x^{(k)} \)

Descent direction \hspace{1cm} Step size \hspace{1cm} Stopping criterion
Stopping criteria

- Near local minimum, $\nabla f(x) \approx 0$ (or equivalently $\|\nabla f(x)\| \approx 0$)
  Stop when **gradient norm** becomes small

  $$\|\nabla f(x^{(k)})\| \leq \epsilon$$

- Stop when **step size** becomes small

  $$\|x^{(k+1)} - x^{(k)}\| \leq \epsilon$$

- Stop when **relative objective change** becomes small

  $$\frac{f(x^{(k)}) - f(x^{(k+1)})}{f(x^{(k)})} \leq \epsilon$$
Line search

Optimal step size can be found by solving a one-dimensional optimization problem

$$\alpha^{(k)} = \arg\min_{\alpha \geq 0} f(x^{(k)} + \alpha d^{(k)})$$

One-dimensional optimization algorithms for finding the optimal step size are generically called **exact line search**
Armijo [ar-mi-xo] rule

The function **sufficiently decreases** if \( f(x + \alpha d) - f(x) < \sigma \alpha \nabla f(x)^\top d \)

**Armijo rule** (Larry Armijo, 1966): start with \( \alpha = \alpha_0 \) and decrease it by multiplying by some \( \beta \in (0, 1) \) until the function sufficiently decreases.
Descent direction

How to descend in the fastest way?

Go in the direction in which the height lines are the densest

Devil’s Tower

Topographic map
Steepest descent

\[ f(x + d) \approx f(x) + \nabla f(x)^\top d \]

**Directional derivative**: how much \( f(x) \) changes in the direction \( d \) (negative for a descent direction)

Find a unit-length direction **minimizing directional derivative**

\[ d = \arg\min_{d: ||d||=1} \nabla f(x)^\top d \]
Steepest descent

\[ \|d\|_2 = 1 \]
\[ -\nabla f(x) \]
\[ \vec{d} \]
\[ x \]

\[ d = -\frac{\nabla f(x)}{\|\nabla f(x)\|_2} \]

Normalized steepest descent

\[ \|d\|_1 = 1 \]
\[ -\nabla f(x) \]
\[ \vec{d} \]
\[ x \]

\[ d = -\text{sign} \left( \frac{\partial f(x)}{\partial x^i} \right) e_i \]

Coordinate descent (coordinate axis in which descent is maximal)
Steepest descent algorithm

- Start with some $x^{(0)}$, $k = 0$
- Compute **steepest descent direction**
  $$d^{(k)} = -\nabla f(x^{(k)})$$
- Choose **step size** using line search
  $$\alpha^{(k)} \approx \arg\min_{\alpha \geq 0} f(x^{(k)} + \alpha d^{(k)})$$
- Update iterate
  $$x^{(k+1)} = x^{(k)} + \alpha^{(k)} d^{(k)}$$
- Increment iteration counter $k \leftarrow k + 1$

Until convergence
Condition number

Condition number is the ratio of maximal and minimal eigenvalues of the Hessian $\nabla^2 f(x)$, 

$$\kappa = \frac{\lambda_{\max}}{\lambda_{\min}}$$

Problem with large condition number is called ill-conditioned

Steepest descent convergence rate is slow for ill-conditioned problems
Q-norm

\[ \| x \|_Q = \| Q^{1/2} x \|_2 \]

Function \( f(x) \)

Gradient \( \nabla f(x) \)

Descent direction \( -Q^{-1} \nabla f(x) \)

\[ y = Q^{1/2} x \]

\((Q > 0)\)

Change of coordinates

\[ x = Q^{-1/2} y \]

L_2 norm

\[ \| y \|_2 = \| Q^{1/2} x \|_2 \]

\[ \bar{f}(y) = f(Q^{-1/2} y) = f(x) \]

\[ \nabla \bar{f}(y) = Q^{-1/2} \nabla f(x) \]

\[ -Q^{-1/2} \nabla f(x) \]
Preconditioning

Using Q-norm for steepest descent can be regarded as a change of coordinates, called **preconditioning**

**Preconditioner** $Q$ should be chosen to improve the condition number of the Hessian in the proximity of the solution

In $y = Q^{1/2}x$ system of coordinates, the Hessian at the solution is

$$\nabla^2\bar{f}(y^*) = Q^{-1/2}\nabla^2 f(x^*)Q^{-1/2}$$

$$\approx I \quad \text{(a dream)}$$
Newton method as optimal preconditioner

Best theoretically possible preconditioner $Q = \nabla^2 f(x^*)$, giving descent direction

$$d = - (\nabla^2 f(x^*))^{-1} \nabla f(x)$$

Ideal condition number

$$\nabla^2 f(y^*) = Q^{-1/2} \nabla^2 f(x^*) Q^{-1/2} = I$$

Problem: the solution $x^*$ is unknown in advance

Newton direction: use Hessian as a preconditioner at each iteration

$$d = - (\nabla^2 f(x))^{-1} \nabla f(x)$$
Another derivation of the Newton method

Approximate the function as a quadratic function using second-order Taylor expansion

\[ f(x + d) \approx f(x) + \nabla f^T(x)d + \frac{1}{2}d^T \nabla^2 f(x)d \]

\[ = q(d) \quad \text{(quadratic function in } d) \]

\[ \nabla q(d) = \nabla f + \nabla^2 f(x)d = 0 \]

\[ \Rightarrow d = -(\nabla^2 f(x))^{-1}\nabla f(x) \]

Close to solution the function looks like a quadratic function; the Newton method converges fast.
Newton method

- Start with some $x^{(0)}, k = 0$
- Compute **Newton direction**
  \[ d^{(k)} = -\left(\nabla^2 f(x^{(k)})\right)^{-1}\nabla f(x^{(k)}) \]
- Choose **step size** using line search
  \[ \alpha^{(k)} \approx \arg\min_{\alpha \geq 0} f(x^{(k)} + \alpha d^{(k)}) \]
- Update iterate
  \[ x^{(k+1)} = x^{(k)} + \alpha^{(k)} d^{(k)} \]
- Increment iteration counter $k \leftarrow k + 1$

Until convergence
**Frozen Hessian**

**Observation:** close to the optimum, the Hessian does not change significantly

Reduce the number of Hessian inversions by keeping the Hessian from previous iterations and update it once in a few iterations

Such a method is called **Newton with frozen Hessian**
**Cholesky factorization**

Decompose the Hessian

\[ \nabla^2 f = LL^T \]

where \( L \) is a lower triangular matrix

Solve the Newton system

\[ LL^T d = -\nabla f \]

in two steps

- **Forward substitution** \( Ly = -\nabla f \)
- **Backward substitution** \( L^T d = y \)

Complexity: \( \frac{1}{6}N^3 + O(N^2) \), better than straightforward matrix inversion

Andre Louis Cholesky (1875-1918)
Truncated Newton

Solve the Newton system \textit{approximately}

\[ \nabla^2 f d \approx -\nabla f \]

A few iterations of \textit{conjugate gradients} or other algorithm for the solution of linear systems can be used

Such a method is called \textit{truncated} or \textit{inexact Newton}
Non-convex optimization

- Using convex optimization methods with non-convex functions does not guarantee global convergence!
- There is no theoretical guaranteed global optimization, just heuristics
Iterative majorization

Construct a majorizing function $h(x, q)$ satisfying

- $h(q, q) = f(q)$
- **Majorizing inequality:** $f(x) \leq h(x, q)$ for all $x \in \mathbb{X}$
- $h(x, q)$ is convex or easier to optimize w.r.t. $x$
Iterative majorization

- Start with some $q^{(0)}$
- Find $x^{(k+1)}$ such that
  \[ x^{(k+1)} = \arg\min_{x \in X} h(x, q^{(k)}) \]
- Update iterate
  \[ q^{(k+1)} = x^{(k)} \]
- Increment iteration counter $k \leftarrow k + 1$
- Solution $x^* \approx q^{(k)}$

Until convergence
Constrained optimization
Constrained optimization problems

Generic **constrained minimization** problem

\[
\min_{x \in \mathbb{X}} f(x) \quad \text{s.t.} \quad \begin{cases} 
  g_k(x) \leq 0 & k = 1, \ldots, K \\
  h_l(x) = 0 & l = 1, \ldots, L
\end{cases}
\]

where

- \( g_k : \mathbb{X} \to \mathbb{R} \) are **inequality constraints**
- \( h_l : \mathbb{X} \to \mathbb{R} \) are **equality constraints**
- A subset of the search space \( \mathbb{X} \) in which the constraints hold is called the **feasible set**
- A point \( x \in \mathbb{X} \) belonging to the feasible set is called a **feasible solution**

A minimizer of the problem \( \min_{x \in \mathbb{X}} f(x) \) may be infeasible!
An example

Inequality constraint: $g(x) \leq 0$

Equality constraint: $h(x) = 0$

Feasible set

Inequality constraint $g_k$ is active at point $x$ if $g_k(x) = 0$, inactive otherwise.

A point $x$ is regular if the gradients of equality constraints $\nabla h_l(x)$ and of active inequality constraints $\nabla g_k(x)$ are linearly independent.
Lagrange multipliers

Main idea to solve constrained problems: arrange the objective and constraints into a single function

\[ \mathcal{L}(x, \lambda, \mu) = f(x) + \sum_{k=1}^{K} \lambda_k g_k(x) + \sum_{l=1}^{L} \mu_l h_l(x) \]

\[ = f(x) + g(x)^\top \lambda + h(x)^\top \mu \]

and minimize it as an unconstrained problem

\[ \mathcal{L}(x, \lambda, \mu) \] is called **Lagrangian**

\( \lambda \) and \( \mu \) are called **Lagrange multipliers**
KKT conditions

If $x^*$ is a regular point and a local minimum, there exist Lagrange multipliers $\lambda^*$ and $\mu^*$ such that

- $g_k(x^*) \leq 0$ for all $k = 1, \ldots, K$ and $h_l(x^*) = 0$ for all $l = 1, \ldots, L$
- $g(x^*)^T \lambda^* = 0$ such that $\lambda_i > 0$ for active constraints and zero for inactive constraints
- $\nabla L(x^*, \lambda^*, \mu^*) = \nabla f(x) + g(x^*)^T \lambda^* + h(x^*)^T \mu^* = 0$

Known as Karush-Kuhn-Tucker conditions

Necessary but not sufficient!
KKT conditions

Sufficient conditions:

If the objective $f(x)$ is **convex**, the inequality constraints $g_k(x)$ are **convex** and the equality constraints $h_l(x)$ are **affine**, the KKT conditions are sufficient.

In this case, $x^*$ is the solution of the constrained problem (global constrained minimizer).
Geometric interpretation

Consider a simpler problem: $\min_{x \in X} f(x) \quad \text{s.t.} \quad h(x) = 0$

The gradient of objective and constraint must line up at the solution

Equality constraint $h(x) = 0$
Penalty methods

Define a penalty aggregate

\[ F_p(x) = f(x) + \sum_{k=1}^{K} \varphi_p(g_k(x)) + \sum_{l=1}^{L} \psi_p(h_l(x)) \]

where \( \varphi : \mathbb{R} \rightarrow \mathbb{R} \) and \( \psi : \mathbb{R} \rightarrow \mathbb{R} \) are parametric penalty functions

For larger values of the parameter \( p \), the penalty on the constraint violation is stronger
Penalty methods

\[ \varphi(t) = e^t - 1 \]

Inequality penalty

\[ \varphi_p(t) = p^{-1} \varphi(pt) \]

\[ \varphi'_p(t) = \varphi'(pt) \]

\[ \lim_{p \to \infty} \varphi_p(t) = \begin{cases} 
0 & t \leq 0 \\
\infty & t > 0 
\end{cases} \]

Equality penalty

\[ \psi_p(t) = p^{-1} \psi(pt) \]

\[ \psi'_p(t) = \psi'(pt) \]

\[ \lim_{p \to \infty} \psi_p(t) = \begin{cases} 
0 & t = 0 \\
\infty & \text{else} 
\end{cases} \]
Penalty methods

- Start with some $x_p^{(0)}$ and initial value of $p$
- Find
  \[ x_p^* = \arg\min_{x \in X} F_p(x) \]
  
  by solving an unconstrained optimization problem initialized with $x_p^{(0)}$
- Set $p' = \beta p$
- Set $x_p^{(0)} = x_p^*$
- Update $p \leftarrow p'$
- Solution $x^* = x_p^*$

Until convergence