DSA5103 Optimization Problem for Data Modelling

AY2024/25 Sem2 By Zhao Peiduo

Lecture 1

Nonlinear Programming A general **nonlinear programming problem (NLP)** is to minimize/maximize a function $f(x)$, subject to equality constraints $g_i(x) = 0$, $i \in [m]$, and inequality constraints $h_j(x) \le 0$, $j \in [p]$. Here, f, g_i, and h_j are functions of the variable $x = (x_1, \dots, x_n)^T \in \mathbb{R}^n$. The term definitions are as follows:

- f: **Objective function**
- $g_i(x) = 0$: **Equality constraints**
- $h_j(x) \le 0$: **Inequality constraints**

It suffices to discuss minimization problems since minimizing $f(x)$ is equivalent to maximizing $-f(x)$.

Feasible Set
$$S = \{x \in \mathbb{R}^n \mid g_1(x) = 0, \dots, g_m(x) = 0,\ h_1(x) \le 0, \dots, h_p(x) \le 0\}.$$

A point in the feasible set is a **feasible solution** or **feasible point** wbere all constraints are satisfied; otherwise, it is an **infeasible solution** or **infeasible point**. When there is no constraint, $S = \mathbb{R}^n$, we say the NLP is **unconstrained**.

Local and Global Minimizer Let S be the feasible set. Define $B_\epsilon(y) = \{x \in \mathbb{R}^n \mid \|x - y\| < \epsilon\}$ to be the open ball with center y and radius ϵ. Here, $\|x\| = \sqrt{x_1^2 + \cdots + x_n^2}$.

1. A point $x^* \in S$ is said to be a **local minimizer** of f if there exists $\epsilon > 0$ such that
$$f(x^*) \le f(x) \quad \forall x \in S \cap B_\epsilon(x^*).$$
2. A point $x^* \in S$ is said to be a **global minimizer** of f if
$$f(x^*) \le f(x) \quad \forall x \in S.$$

Interior point Let $S \subseteq \mathbb{R}^n$ be a nonempty set. An point $x \in S$ is called an **interior point** of S if
$$\exists \epsilon > 0 \quad s.t. \quad B_\epsilon(x) \subseteq S.$$

Gradient Vector Let $S \subseteq \mathbb{R}^n$ be a nonempty set. Suppose $f : S \to \mathbb{R}$, and x is an interior point of S such that f is differentiable at x. Then the **gradient vector** of f at x is
$$\nabla f(x) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(x) \\ \vdots \\ \frac{\partial f}{\partial x_n}(x) \end{bmatrix}.$$

Hessian Matrix Let $S \subseteq \mathbb{R}^n$ be a nonempty set. Suppose $f : S \to \mathbb{R}$, and x is an interior point of S such that f has second-order partial derivatives at x. Then the **Hessian** of f at x is the $n \times n$ matrix:
$$H_f(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2}(x) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(x) & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n}(x) \\ \frac{\partial^2 f}{\partial x_2 \partial x_1}(x) & \frac{\partial^2 f}{\partial x_2^2}(x) & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n}(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1}(x) & \frac{\partial^2 f}{\partial x_n \partial x_2}(x) & \cdots & \frac{\partial^2 f}{\partial x_n^2}(x) \end{bmatrix}.$$

- The ij-entry of $H_f(x)$ is $\frac{\partial^2 f}{\partial x_i \partial x_j}(x)$.
- In general, $H_f(x)$ is not symmetric. However, if f has continuous second-order derivatives, then the Hessian matrix is symmetric since ∂x_i and ∂x_j are interchangeable.

Positive (Semi)Definite Let A be a real $n \times n$ matrix.

1. A is said to be **positive semidefinite** if $x^T Ax \ge 0$, $\forall x \in \mathbb{R}^n$.
2. A is said to be **positive definite** if $x^T Ax > 0$, $\forall x \ne 0$.
3. A is said to be **negative semidefinite** if $-A$ is positive (semi)definite.
4. A is said to be **negative definite** if $-A$ is positive definite.
5. A is said to be **indefinite** if A is neither positive nor negative semidefinite.

Eigenvalue Test Theorem Let A be a real symmetric $n \times n$ matrix.

1. A is **positive semidefinite** iff every eigenvalue of A is nonnegative.
2. A is **positive definite** iff every eigenvalue of A is positive.
3. A is **negative semidefinite** iff every eigenvalue of A is nonpositive.
4. A is **negative definite** iff every eigenvalue of A is negative.
5. A is **indefinite** iff it has both a positive eigenvalue and a negative eigenvalue.

Proof for: A is positive semidefinite iff every eigenvalue of A is nonnegative

(Forward) Suppose A is positive semidefinite, show that its eigenvalues are nonnegative. By definition, a Hermitian matrix A is positive semidefinite if for all nonzero vectors $x \in \mathbb{C}^n$:
$$x^* Ax \ge 0$$

Let λ be an eigenvalue of A with corresponding eigenvector x such tha $Ax = \lambda x$. Taking the inner product of both sides with x, we obtain:
$$x^* Av = v^*(\lambda x) = \lambda(x^* x)$$

Since $v^* v$ (the squared norm of v) is always positive for nonzero v, the above equation implies $\lambda \ge 0$

(Backward) Since A is Hermitian, it has an orthonormal basis of eigenvectors $\{q_1, q_2, \dots, q_n\}$ with corresponding real eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_n$.
For any vector x, we can express it in terms of the eigenvectors as:
$$x = \sum_{i=1}^n c_i q_i$$

for some scalars c_i, and compute the quadratic form:
$$x^* Ax = \left(\sum_{i=1}^n c_i^* q_i^*\right) A \left(\sum_{j=1}^n c_j q_j\right)$$

Expanding the expression using the orthonormality of the eigenvectors:
$$x^* Ax = \sum_{i=1}^n \lambda_i |c_i|^2$$

Since we are given that all eigenvalues $\lambda_i \ge 0$, and the squared magnitudes $|c_i|^2$ are nonnegative, it follows that:
$$x^* Ax \ge 0 \quad \forall x \ne 0$$

Thus, A is positive semidefinite.

Necessary and Sufficient Conditions
Assume that $f : \mathbb{R}^n \to \mathbb{R}$ is nonlinear and differentiable. A point x^* is called a **stationary point** of f if $\nabla f(x^*) = 0$.
Necessary condition: Confine our search for global minimizers within the set of stationary points
If x^* is a local minimizer of f, then

1. x^* is a stationary point, i.e., $\nabla f(x^*) = 0$
2. The Hessian $H_f(x^*)$ is positive semidefinite

Sufficient condition: Verify that a point is indeed a local minimizer
If the following conditions hold, then x^* is a local minimizer of f.

1. x^* is a stationary point, i.e., $\nabla f(x^*) = 0$
2. The Hessian $H_f(x^*)$ is positive definite,

Convex set A set $D \in \mathbb{R}^n$ is said to be a **convex** set if for any two points x and y in D, the line segment joining x and y also lies in D. That is,
$$x, y \in D \Rightarrow \lambda x + (1 - \lambda)y \in D \quad \forall \lambda \in [0, 1].$$

Strictly convex function
Let $D \subseteq \mathbb{R}^n$ be a convex set. Consider a function $f : D \to \mathbb{R}$.

1. The function f is said to be **convex** if $f(\lambda x + (1 - \lambda)y) \le \lambda f(x) + (1 - \lambda)f(y)$, $\quad \forall x, y \in D,\ \lambda \in [0, 1]$.
2. The function f is said to be **strictly convex** if $f(\lambda x + (1 - \lambda)y) < \lambda f(x) + (1 - \lambda)f(y)$. for all distinct $x, y \in D$, $\lambda \in (0, 1)$.

For a convex f It holds that

1. any local minimizer is a global minimizer.
2. if f is strictly convex, then the global minimizer is unique.

Test for convexity of a differentiable function
Suppose that f has continuous second partial derivatives on an open convex set D in $\mathbb{R}^n$.

1. The function f is convex on D **iff** the Hessian matrix $H_f(x)$ is positive semidefinite at each $x \in D$.
2. If $H_f(x)$ is positive definite at each $x \in D$, then f is strictly convex on D.
3. If $H_f(\hat{x})$ is indefinite at some point $\hat{x} \in D$, then f is not a convex nor a concave function on D.

Eigenvalue Decomposition: The eigenvalue decomposition of $A \in \mathbb{S}^n$ is given by:
$$A = Q\Lambda Q^T = [Q_{\cdot 1} \quad \cdots \quad Q_{\cdot n}] \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} [Q_{\cdot 1} \quad \cdots \quad Q_{\cdot n}]^T$$

where Q is an orthogonal matrix whose **columns** are eigenvectors of A, Λ is a diagonal matrix with eigenvalues of A on the diagonal.
Change of bases using eigenvectors Denote the ith column of orthogonal matrix Q as q_i. Change the bases to $\{q_1, q_2\}$. With new bases,

- For any vector x, $x = Q(Q^T x)$, so its representation becomes
$$\tilde{x} = Q^T x = \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{bmatrix}$$
- Since $y = Ax = Q\Sigma Q^T x$, the representation of y is
$$\tilde{y} = \Sigma \tilde{x} = \begin{bmatrix} \lambda_1 \tilde{x}_1 \\ \lambda_2 \tilde{x}_2 \end{bmatrix}$$

Hence, the linear transformation results in a scaling of λ along the eigenvector associated with λ.

Statistical Properties Let $x_1, \ldots, x_n \in \mathbb{R}^p$ be n observations of a random variable x.

- Mean vector: $\mu = \bar{x} = \frac{1}{n}\sum_{i=1}^n x_i \in \mathbb{R}^p$
- (Sample/Empirical) Covariance matrix: $\Sigma = \frac{1}{n-1}\sum_{i=1}^n (x_i - \mu)(x_i - \mu)^T \in \mathbb{R}^{p\times p}$ (Covariance matrices are symmetric and positive semidefinite)
- Standard deviation (for $p = 1$): $\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^n (x_i - \mu)^2}$

PCA

- PCA is often used to **reduce the dimensionality** of large data sets while preserving as much information as possible.
- PCA allows us to identify the **principal directions in which the data varies**.

Let $x_1, \ldots, x_n \in \mathbb{R}^p$ be n observations of a random variable x and

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix}.$$

The mean vectors of x_i and $Q^T x_i$ (for $i = 1, \ldots, n$) are, respectively,

$$\mu = \frac{1}{n}\sum_{i=1}^n x_i, \quad \hat{\mu} = \frac{1}{n}\sum_{i=1}^n Q^T x_i = Q^T \mu.$$

Consequently, the associated covariance matrices are, respectively,

$$\Sigma = \frac{1}{n-1}\sum_{i=1}^n (x_i - \mu)(x_i - \mu)^T,$$

$$\hat{\Sigma} = \frac{1}{n-1}\sum_{i=1}^n (Q^T x_i - Q^T\mu)(Q^T x_i - Q^T \mu)^T = Q^T \Sigma Q.$$

Optimization problem of PCA

$$\max_{Q\in\mathbb{R}^{p\times k},\, Q^TQ=I} \operatorname{trace}(Q^T \Sigma Q).$$

Let the eigenvalue decomposition of Σ be

$$\Sigma = \begin{bmatrix} q_1 & \cdots & q_p \end{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_p \end{bmatrix} \begin{bmatrix} q_1 & \cdots & q_p \end{bmatrix}^T,$$

where

$$\lambda_1 \geq \cdots \geq \lambda_p \geq 0.$$

Then

$$Q = \begin{bmatrix} q_1 & \cdots & q_k \end{bmatrix}.$$

Standard PCA workflow

1. Make sure the data X are rows = observations and columns = variables.
2. Standardize the columns of X.
3. Run $[Q, X_{\text{new}}, d, \text{tsquared}, \text{explained}] = \text{pca}(X)$.
4. Using the variance% in "explained", choose k (usually 1, 2, or 3) components for visual analysis.
   - For example, if $d = (1.9087, 0.0913)$, explained= $(95.4, 4.6)$, one may choose $k = 1$ as the first principal component carries 95.4% of the information.
   - For example, if $d = (2.9108, 0.9212, 0.1474, 0.0206)$, explained= $(72.8, 23.0, 3.7, 0.5)$, one may choose $k = 2$ as the first two principal components carry 95.8% of the information.
5. Plot $X_{\text{new}}(:, 1), \ldots, X_{\text{new}}(:, k)$ on a k-dimensional plot.

Lecture 2

Gradient Descent Method Given $x_0 \in \mathbb{R}^n$, for $k = 0, 1, 2, \ldots$ do:

$$r_k = Ax_k - b,$$
$$\alpha_k = \frac{(r_k, r_k)}{(Ar_k, r_k)},$$
$$x_{k+1} = x_k - \alpha_k r_k.$$

Gradient Descent Method Example: Ax = b where A is Symmetric Positive Definite

Let

$$f(x) = \|x - x_\star\|_A^2 = (A(x - x_\star), (x - x_\star)) = (x - x_\star)^T A(x - x_\star),$$

where $x_\star$ is the solution of

$$Ax = b.$$

It is obvious that

$$f(x) = 0 \quad \text{if and only if} \quad x = x_\star.$$

Denote

$$x = x_0 + \delta_0.$$

Then,

$$\begin{aligned} f(x) &= f(x_0) + (A\delta_0, \delta_0) + 2\delta_0^T(Ax_0 - b) \\ &= f(x_0) + \delta_0^T A\delta_0 + 2\delta_0^T r_0, \end{aligned}$$

where

$$r_0 = Ax_0 - b.$$

It is clear that

$$f(x) \leq f(x_0)$$

only if

$$\delta_0^T r_0 \leq 0,$$

in particular,

$$-r_0 = b - Ax_0$$

is the negative of the gradient direction $-\nabla f$ at the point x_0.
The negative of the gradient direction is locally the direction that yields the fastest rate of decrease for f. Hence, we can choose

$$\delta_0 = -\alpha_0 r_0,$$

so that

$$\begin{aligned} f(x) &= f(x_0) + \alpha_0^2(Ar_0, r_0) - 2\alpha_0 r_0^T r_0 \\ &= f(x_0) + \alpha_0^2 r_0^T Ar_0 - 2\alpha_0 r_0^T r_0 \leq f(x_0), \end{aligned}$$

provided

$$\alpha_0 \geq 0.$$

It is obvious, we have

$$f(x) \leq f(x_0), \quad \forall 0 \leq \alpha \leq \frac{2(r_0, r_0)}{(Ar_0, r_0)}.$$

The optimal α shall satisfy

$$f(x) = \min_{\alpha_0\in\mathbb{R}} f(x_0) + \alpha_0^2(Ar_0, r_0) - 2\alpha_0 r_0^T r_0,$$

i.e.,

$$\alpha_0 = \frac{(r_0, r_0)}{(Ar_0, r_0)} \geq 0.$$

Therefore, we conclude

$$\text{If} \quad x = x_0 - \alpha_0 r_0, \quad \text{then} \quad f(x) \leq f(x_0).$$

Kantorovich Inequality Let B be any Symmetric Positive Definite real matrix and $\lambda_{\max}$ and $\lambda_{\min}$ its largest and smallest eigenvalues. Then,

$$\frac{(Bx, x)(B^{-1}x, x)}{(x, x)^2} \leq \frac{(\lambda_{\max} + \lambda_{\min})^2}{4\lambda_{\max}\lambda_{\min}}, \quad \forall x \neq 0.$$

Kantorovich Inequality Proof

Clearly, it is equivalent to show that the result is true for any unit vector x. Since B is symmetric, we have

$$B = Q^T DQ,$$

where Q is orthogonal and

$$D = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix},$$

and

$$\lambda_{\max} = \lambda_1 \geq \cdots \geq \lambda_n = \lambda_{\min} > 0.$$

We have

$$(Bx, x)(B^{-1}x, x) = (DQx, Qx)(D^{-1}Qx, Qx).$$

Setting

$$y = Qx = \begin{bmatrix} y_1 & \cdots & y_n \end{bmatrix}^T, \quad \beta_i = y_i^2.$$

Note that $\sum_{i=1}^n \beta_i = 1$, and

$$\lambda = (Dy, y) = \sum_{i=1}^n \beta_i \lambda_i$$

is a convex combination of the eigenvalues $\lambda_i, \ i = 1, \cdots, n$, and furthermore, the following relation holds,

$$(Bx, x)(B^{-1}x, x) = \lambda\psi(y),$$

with

$$\psi(y) = (D^{-1}y, y) = \sum_{i=1}^n \beta_i \frac{1}{\lambda_i}.$$

Noting that

$$\psi(y) \le \frac{1}{\lambda_1} + \frac{1}{\lambda_n} - \frac{\lambda}{\lambda_1\lambda_n}, \quad (\text{since } \sum_{i=1}^n \beta_i = 1, \text{proved later})$$

therefore,

$$(Bx, x)(B^{-1}x, x) = \lambda\psi(y) \le \lambda\left(\frac{1}{\lambda_1} + \frac{1}{\lambda_n} - \frac{\lambda}{\lambda_1\lambda_n}\right).$$

The maximum of the right-hand side is reached for

$$\lambda = \frac{\lambda_1 + \lambda_n}{2}$$

yielding

$$(Bx, x)(B^{-1}x, x) = \lambda\psi(y) \le \lambda\left(\frac{1}{\lambda_1} + \frac{1}{\lambda_n} - \frac{\lambda}{\lambda_1\lambda_n}\right)$$

$$\le \frac{\lambda_1 + \lambda_n}{4}\left(\frac{1}{\lambda_1} + \frac{1}{\lambda_n}\right)$$

Proof for $\psi(y) \le \frac{1}{\lambda_1} + \frac{1}{\lambda_n} - \frac{\lambda}{\lambda_1\lambda_n}$
Since

$$0 < \lambda_n \le \cdots \le \lambda_i \le \cdots \le \lambda_1, \quad i = 1, \ldots, n,$$

we have for any $i = 1, \ldots, n$ that

$$\lambda_1 \ge \lambda_i > 0, \quad \lambda_i - \lambda_n \ge 0, \quad i = 1, \ldots, n,$$

which gives

$$\lambda_1(\lambda_i - \lambda_n) \ge \lambda_i(\lambda_1 - \lambda_n),$$

i.e.,

$$\lambda_1\lambda_n \le \lambda_i(\lambda_1 + \lambda_n - \lambda_i),$$

and

$$\frac{1}{\lambda_i} \le \frac{\lambda_1 + \lambda_n - \lambda_i}{\lambda_1\lambda_n}.$$

Note that

$$\beta_i \ge 0, \quad \sum_{i=1}^n \beta_i = 1,$$

we get

$$\beta_i \frac{1}{\lambda_i} \le \beta_i \frac{\lambda_1 + \lambda_n - \lambda_i}{\lambda_1\lambda_n},$$

and so,

$$\sum_{i=1}^n \beta_i \frac{1}{\lambda_i} \le \sum_{i=1}^n \beta_i \frac{\lambda_1 + \lambda_n - \lambda_i}{\lambda_1\lambda_n}$$

$$= \frac{1}{\lambda_1} + \frac{1}{\lambda_n} - \frac{\sum_{i=1}^n \beta_i\lambda_i}{\lambda_1\lambda_n}.$$

This lemma helps to establish the following result regarding the convergence rate of the method.
Theorem Let A be a Symmetric Positive Definite matrix. Then, the A-norms of the error vectors

$$d_k = x_\star - x_k = -A^{-1}r_k$$

generated by the Gradient Descent Algorithm satisfy the relation

$$\|d_{k+1}\|_A \le \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}}\|d_k\|_A,$$

and so,

$$\lim_{k\to\infty} \|d_k\|_A = 0,$$

which gives

$$\lim_{k\to\infty} d_k = 0,$$

i.e., the algorithm converges for any initial guess x_0.
Proof First, we have

$$\|d_k\|_A^2 = (Ad_k, d_k) = (-r_k, d_k) = (r_k, A^{-1}r_k).$$

Then we have

$$\|d_{k+1}\|_A^2 = (Ad_{k+1}, d_{k+1}) = (-r_{k+1}, d_{k+1})$$

and by simple substitution,

$$d_{k+1} = d_k + \alpha_k r_k,$$

$$\|d_{k+1}\|_A^2 = (-r_{k+1}, d_k + \alpha_k r_k),$$

$$= (-r_{k+1}, d_k) - \alpha(r_{k+1}, r_k),$$

$$= (-r_{k+1}, d_k),$$

since

$$(r_{k+1}, r_k) = 0.$$

Thus,

$$\|d_{k+1}\|_A^2 = (-r_{k+1}, d_k),$$

$$= (-r_k + \alpha_k Ar_k, d_k),$$

$$= (-r_k, d_k) + \alpha_k(Ar_k, d_k),$$

$$= (r_k, A^{-1}r_k) - \alpha_k(Ar_k, A^{-1}r_k),$$

$$= (r_k, A^{-1}r_k) - \frac{(r_k, r_k)^2}{(Ar_k, r_k)},$$

$$= \|d_k\|_A^2\left(1 - \frac{(r_k, r_k)}{(Ar_k, r_k)} \times \frac{(r_k, r_k)}{(r_k, A^{-1}r_k)}\right).$$

The result follows by applying the Kantorovich inequality.

Unconstrained problem
To minimize a **differentiable** function f

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

Recall that a global minimizer is a local minimizer, and a local minimizer is a stationary point.

- We may try to find stationary points x, i.e., $\nabla f(x) = 0$ for solving an unconstrained problem.
- When it is difficult to solve $\nabla f(x) = 0$, we look for an approximate solution via iterative methods.

A general algorithmic framework
Choose $x^{(0)}$ and repeat

$$x^{(k+1)} = x^{(k)} + \alpha_k p^{(k)}, \quad k = 0, 1, 2, \ldots$$

until some stopping criteria is satisfied.

- $x^{(0)}$ initial guess of the solution.
- $\alpha_k > 0$ is called the step length/step size/learning rate.
- $p^{(k)}$ is a search direction.

Descent Direction
The search direction $p^{(k)}$ should be a descent direction at $x^{(k)}$.

- We say $p^{(k)}$ is a descent direction at $x^{(k)}$ if

$$\nabla f(x^{(k)})^T p^{(k)} < 0$$

- The function value f can be reduced along this descent direction with "appropriate" step length

$$\exists\delta > 0 \quad \text{such that} \quad f(x^{(k)} + \alpha_k p^{(k)}) < f(x^{(k)}) \quad \forall\alpha_k \in (0, \delta)$$

Algorithm 1 Steepest Descent Method

1: **Initialization:** Choose initial point $x^{(0)}$, tolerance $\epsilon > 0$, set $k \leftarrow 0$.
2: **while** $\|\nabla f(x^{(k)})\| > \epsilon$ **do**
3: Find the step length α_k (e.g., by a certain line search rule).
4: Update the solution:

$$x^{(k+1)} = x^{(k)} - \alpha_k \nabla f(x^{(k)})$$

5: Increment $k \leftarrow k + 1$.
6: **end while**
7: **Output:** $x^{(k)}$ (approximate solution)

One may choose to use a constant step length (say $\alpha_k = 0.1$), or find it via line search rules:

- Exact line search
- Backtracking line search

Exact line search
Exact line search tries to find α_k by solving the one-dimensional problem:

$$\min_{\alpha > 0} \quad \varphi(\alpha) := f(x^{(k)} + \alpha p^{(k)})$$

- In general, exact line search is the most difficult part of the steepest descent method.
- If f is a simple function, it may be possible to obtain an analytical solution for α_k by solving $\varphi'(\alpha) = 0$.

Contour plot A contour is a fixed height $f(x_1, x_2) = c$.

Algorithm 2 Steepest Descent Method with Exact Line Search

Initialization: Choose initial point $x^{(0)}$, tolerance $\epsilon > 0$, set $k \leftarrow 0$.
while $\|\nabla f(x^{(k)})\| > \epsilon$ **do**
  Compute search direction: $p^{(k)} = -\nabla f(x^{(k)})$.
  Find optimal step length:

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

  Update the solution:

$$x^{(k+1)} = x^{(k)} + \alpha_k p^{(k)}$$

  Increment $k \leftarrow k + 1$.
end while
Output: $x^{(k)}$ (approximate solution)

Properties of steepest descent method with exact line search
Let $\{x^{(k)}\}$ be the sequence generated by the steepest descent method with exact line search.

- **Monotonic decreasing property**:

$$f(x^{(k+1)}) < f(x^{(k)}) \quad \text{if } \nabla f(x^{(k)}) \neq 0.$$

- Suppose f is a *coercive* function with continuous first-order derivatives on $\mathbb{R}^n$. Then some subsequence of $\{x^{(k)}\}$ converges.
  The limit of any convergent subsequence of $\{x^{(k)}\}$ is a stationary point of f.

Backtracking Line Search
Backtracking line search starts with a relatively large step length and iteratively shrinks it (i.e., "backtracking") until the Armijo condition holds.

Algorithm 3 Backtracking Line Search

Choose $\bar{\alpha} > 0$, $\rho \in (0, 1)$, $c_1 \in (0, 1)$; Set $\alpha \leftarrow \bar{\alpha}$.
repeat
  Until

$$f(x^{(k)} + \alpha p^{(k)}) \leq f(x^{(k)}) + c_1 \alpha \nabla f(x^{(k)})^T p^{(k)}$$

▷ Armijo Condition
  $\alpha \leftarrow \rho\alpha$
until Armijo condition holds
return $\alpha_k = \alpha$

Notes:

- $p^{(k)}$ is a descent direction:

$$\nabla f(x^{(k)})^T p^{(k)} < 0$$

- The Armijo condition:

$$f(x^{(k)} + \alpha p^{(k)}) \leq f(x^{(k)}) + c_1 \alpha \nabla f(x^{(k)})^T p^{(k)}$$

  ensures a reasonable amount of decrease in the objective function.
- Example parameter choices:

$$\bar{\alpha} = 1, \quad \rho = 0.9, \quad c_1 = 10^{-4}$$

Algorithm 4 Steepest Descent Method with Backtracking Line Search

---

Choose $x^{(0)}$, $\epsilon > 0$, $\bar{\alpha} > 0$, $\rho \in (0, 1)$, $c_1 \in (0, 1)$; Set $k \leftarrow 0$.
while $\|\nabla f(x^{(k)})\| > \epsilon$ **do**
  Compute search direction: $p^{(k)} = -\nabla f(x^{(k)})$.
  Set $\alpha \leftarrow \bar{\alpha}$.
  repeat
    Until Armijo condition holds:

$$f(x^{(k)} + \alpha p^{(k)}) \leq f(x^{(k)}) + c_1 \alpha \nabla f(x^{(k)})^T p^{(k)}$$

    $\alpha \leftarrow \rho\alpha$.
  until Armijo condition holds
  $\alpha_k \leftarrow \alpha$.
  Update the solution:

$$x^{(k+1)} = x^{(k)} + \alpha_k p^{(k)}$$

  Increment $k \leftarrow k + 1$.
end while
return $x^{(k)}$.

Steepest Descent Method for Multivariate Linear Regression

Algorithm 5 Steepest Descent for Multivariate Linear Regression

---

Choose $\beta_0^{(0)}, \beta^{(0)} = (\beta_1^{(0)}, \ldots, \beta_p^{(0)})^T$ and $\epsilon > 0$; Set $k \leftarrow 0$.
while $\|\nabla L(\beta_0^{(k)}, \beta^{(k)})\| > \epsilon$ **do**
  Determine step length α_k.
  Update parameters:

$$\beta_0^{(k+1)} = \beta_0^{(k)} - \alpha_k \sum_{i=1}^{n} ((\beta^{(k)})^T x_i + \beta_0^{(k)} - y_i)$$

  for $j = 1, 2, \ldots, p$ **do**

$$\beta_j^{(k+1)} = \beta_j^{(k)} - \alpha_k \sum_{i=1}^{n} ((\beta^{(k)})^T x_i + \beta_0^{(k)} - y_i) x_{ij}$$

  end for
  Increment $k \leftarrow k + 1$.
end while
return $\beta_0^{(k)}, \beta^{(k)} = (\beta_1^{(k)}, \ldots, \beta_p^{(k)})^T$.

Normal Equation

$$\min_{\beta_0, \beta_1, \ldots, \beta_p} L(\beta_0, \beta_1, \ldots, \beta_p) = \frac{1}{2} \sum_{i=1}^{n} (\beta^T x_i + \beta_0 - y_i)^2$$

$$\hat{X}^T \hat{X} \hat{\beta} = \hat{X}^T Y$$

How to solve

$$\hat{X}^T \hat{X} \hat{\beta} = \hat{X}^T Y$$

Case 1. When $\hat{X}^T \hat{X}$ is invertible, the normal equation implies that

$$\hat{\beta} = (\hat{X}^T \hat{X})^{-1} \hat{X}^T Y$$

is the **unique** solution of linear regression.
This often happens when we face an **over-determined system** — number of training examples n is much larger than number of features p.
We have many training samples to fit but do not have enough degree of freedom.

Case 2. When $\hat{X}^T \hat{X}$ is not invertible, the normal equation will have infinite number of solutions.
$\hat{X}^T \hat{X}$ is not invertible when we face an **under-determined problem** — $n < p$.
We have too many degrees of freedom and do not have enough training samples.
We can apply any method for solving a linear system (e.g., Gaussian elimination) to obtain a solution.

Lecture 3

Classification Binary classification:

- Email: spam/not spam
- Student: fail/pass

We usually assign:

$$\text{label}\begin{cases}0, & \text{normal state/negative class, e.g., not spam}\\ 1, & \text{abnormal state/positive class, e.g., spam}\end{cases}$$

However, the label assignment can be arbitrary:

$$0 = \text{not spam}, 1 = \text{spam} \quad \text{or} \quad 0 = \text{spam}, 1 = \text{not spam}$$

Data: $x_i \in \mathbb{R}^p, y_i \in \{0, 1\}, i = 1, 2, \dots, n$.

Multi-class classification:

- Iris flower (3 species: Setosa, Versicolor, Virginica)
- Optical character recognition

Data: $x_i \in \mathbb{R}^p, y_i \in \{1, \dots, K\}, i = 1, 2, \dots, n$.

Linear Regression vs. Logistic Regression

Linear Regression

- Data $x_i, y_i \in \mathbb{R}$
- Fit: $f(x) = \beta^T x + \beta_0 = \hat{\beta}^T \hat{x}$, where $\hat{\beta} = [\beta_0; \beta]$, $\hat{x} = [1; x]$

Logistic Regression

- Data $x_i, y_i \in \{0, 1\}$
- Fit:

$$f(x) = g(\hat{\beta}^T \hat{x})$$

where

$$g(z) = \frac{1}{1 + e^{-z}} \quad (logistic function)$$

so,

$$f(x) = g(\hat{\beta}^T \hat{x}) = \frac{1}{1 + e^{-(\beta^T x + \beta_0)}}$$

Logistic Regression Decision Rule

$$f(x) = g(\hat{\beta}^T \hat{x}), \quad g(z) = \frac{1}{1 + e^{-z}}$$

$$f(x) = p(y = 1|x; \hat{\beta})$$

Predict $y = 1$ (class 1) if:

$$f(x) \geq 0.5 \quad \text{i.e.,} \quad \hat{\beta}^T \hat{x} \geq 0$$

Predict $y = 0$ (class 0) if:

$$f(x) < 0.5 \quad \text{i.e.,} \quad \hat{\beta}^T \hat{x} < 0$$

Decision Boundary

The set of all $x \in \mathbb{R}^p$ such that:

$$\beta_0 + \beta^T x = 0$$

is called the decision boundary between classes 0 and 1.

The logistic regression has a linear decision boundary; it is:

- a point when $p = 1$
- a line when $p = 2$
- a plane when $p = 3$
- in general a $(p - 1)$-dimensional subspace

Maximum Likelihood Estimation

- Data $(x_i, y_i), i = 1, 2, \dots, n, x_i \in \mathbb{R}^p, y_i \in \{0, 1\}$.
- The likelihood of a single training example (x_i, y_i) is:

$$\text{probability}(x_i \in \text{class } y_i) = \begin{cases} p(y_i = 1|x_i; \hat{\beta}) = f(x_i), & \text{if } y_i = 1\\ p(y_i = 0|x_i; \hat{\beta}) = 1 - f(x_i), & \text{if } y_i = 0\end{cases}$$

$$= f(x_i)^{y_i} [1 - f(x_i)]^{1-y_i}$$

- Assuming independence of training samples, the likelihood is:

$$\prod_{i=1}^{n} f(x_i)^{y_i} [1 - f(x_i)]^{1-y_i}$$

- Want to find $\hat{\beta}$ to maximize the log-likelihood:

$$L(\hat{\beta}) = -\log\left(\prod_{i=1}^{n} f(x_i)^{y_i} [1 - f(x_i)]^{1-y_i}\right)$$

$$= -\sum_{i=1}^{n} \left(y_i \log f(x_i) + (1 - y_i) \log(1 - f(x_i))\right)$$

For a single training example (x_i, y_i), the cost is:

$$-y_i \log f(x_i) - (1 - y_i) \log(1 - f(x_i))$$

$$= \begin{cases} -\log f(x_i), & \text{if } y_i = 1\\ -\log(1 - f(x_i)), & \text{if } y_i = 0\end{cases}$$

Simplifying the Cost Function

$$\log\left(\frac{f(x_i)}{1 - f(x_i)}\right) = \log\left(\frac{\frac{1}{1+e^{-\hat{\beta}^T \hat{x}_i}}}{1 - \frac{1}{1+e^{-\hat{\beta}^T \hat{x}_i}}}\right)$$

$$= \log(e^{\hat{\beta}^T \hat{x}_i}) = \hat{\beta}^T \hat{x}_i$$

$$\log(1 - f(x_i)) = \log\left(1 - \frac{1}{1 + e^{-\hat{\beta}^T \hat{x}_i}}\right)$$

$$= \log\left(\frac{1 + e^{\hat{\beta}^T \hat{x}_i} - 1}{1 + e^{\hat{\beta}^T \hat{x}_i}}\right)$$

$$= -\log\left(1 + e^{\hat{\beta}^T \hat{x}_i}\right)$$

Gradient of the cost function

- **Cost function**

$$L(\beta_0, \beta_1, \dots, \beta_p) = \sum_{i=1}^{n} \log(1 + e^{\beta_0 + \beta^T x_i}) - y_i(\beta_0 + \beta^T x_i)$$

$$\beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}$$

- **Calculate**

$$\frac{\partial}{\partial \beta_0} L = \sum_{i=1}^{n} \left(\frac{1}{1 + e^{-(\beta_0 + \beta^T x_i)}} - y_i\right) = \sum_{i=1}^{n} (f(x_i) - y_i)$$

$$\frac{\partial}{\partial \beta_1} L = \sum_{i=1}^{n} \left(\frac{1}{1 + e^{-(\beta_0 + \beta^T x_i)}} - y_i\right) x_{i1} = \sum_{i=1}^{n} (f(x_i) - y_i) x_{i1}$$

$$\frac{\partial}{\partial \beta_2} L = \sum_{i=1}^{n} \left(\frac{1}{1 + e^{-(\beta_0 + \beta^T x_i)}} - y_i\right) x_{i2} = \sum_{i=1}^{n} (f(x_i) - y_i) x_{i2}$$

$$\vdots$$

$$\frac{\partial}{\partial \beta_p} L = \sum_{i=1}^{n} \left(\frac{1}{1 + e^{-(\beta_0 + \beta^T x_i)}} - y_i\right) x_{ip} = \sum_{i=1}^{n} (f(x_i) - y_i) x_{ip}$$

Solution may not exist

The solution (global minimizer) of the minimization problem

$$\min_{\beta_0, \beta_1, \dots, \beta_p} \sum_{i=1}^{n} \log(1 + e^{\beta_0 + \beta^T x_i}) - y_i(\beta_0 + \beta^T x_i)$$

may not exist. (Regularization will help solve this issue)

Example. $n = 1, x_1 = -1, y_1 = 0$. Then the cost function

$$L(\beta_0, \beta_1) = \log(1 + e^{\beta_0 - \beta_1})$$

We can see that $\min L = 0$. However, this value cannot be attained.

Multi-class classification: one-vs-rest

Idea: transfer multi-class classification to multiple binary classification problems

Data: $x_i \in \mathbb{R}^p, y_i \in \{1, \dots, K\}, i = 1, 2, \dots, n$.

For each $k \in \{1, 2, \dots, K\}$

1. Construct a new label $\tilde{y}_i = 1$ if $y_i = k$ and $\tilde{y}_i = 0$ otherwise
2. Learn a binary classifier f_k with data $x_i, \tilde{y}_i$

Multi-class classifier predicts class k where k achieves the maximal value

$$\max_{k \in \{1, 2, \dots, K\}} f_k(x)$$

Overfitting

- **Underfitting:** a model is too simple and does not adequately capture the underlying structure of the data
- **Overfitting:** a model is too complicated and contains more parameters that can be justified by the data; it does not generalize well from training data to test data
- **Good fit:** a model adequately learns the training data and generalizes well to test data

Ridge regularization
In linear/logistic regression, over-fitting occurs frequently. Regularization will make the model simpler and works well for most of the regression/classification problems.

- Ridge regularization:

$$\lambda\|\beta\|^2 = \lambda \sum_{j=1}^{p} \beta_j^2$$

λ: regularization parameter, $\|\beta\|^2$: regularizer

- It is differentiable. It forces β_j's to be small
- Extreme case: suppose λ is a huge number, it will push all β_j's to be zero and the model will be naive

Ridge regularized problems

- Logistic regression + ridge regularization (Gradient methods can be used, a solution exists)

$$\min_{\beta_0,\beta_1,\ldots,\beta_p} \sum_{i=1}^{n} \log(1 + e^{\beta_0+\beta^T x_i}) - y_i(\beta_0 + \beta^T x_i) + \lambda \sum_{j=1}^{p} \beta_j^2$$

- Linear regression + ridge regularization (Apply either normal equation or gradient methods)

$$\min_{\beta_0,\beta_1,\ldots,\beta_p} \frac{1}{2} \sum_{i=1}^{n} (\beta^T x_i + \beta_0 - y_i)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

Lasso regularization

- Lasso (Least Absolute Shrinkage and Selection Operator) regularization:

$$\lambda\|\beta\|_1 = \lambda \sum_{j=1}^{p} |\beta_j|$$

- It is non-differentiable. It forces some β_j's to be exactly zero
- It can be used for feature selection (model selection). It selects important features (removing non-informative or redundant features)
- When λ is larger, fewer features will be selected

Lasso regularized problems

- Logistic regression + lasso regularization (Gradient methods are no longer applicable)

$$\min_{\beta_0,\beta_1,\ldots,\beta_p} \sum_{i=1}^{n} \log(1 + e^{\beta_0+\beta^T x_i}) - y_i(\beta_0 + \beta^T x_i) + \lambda \sum_{j=1}^{p} |\beta_j|$$

- Linear regression + lasso regularization (Gradient methods are no longer applicable)

$$\min_{\beta_0,\beta_1,\ldots,\beta_p} \frac{1}{2} \sum_{i=1}^{n} (\beta^T x_i + \beta_0 - y_i)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

- In the following, we always assume $\beta_0 = 0$. Note that the intercept should be zero $\beta_0 = 0$ if the data is standardized. Given feature matrix $X \in \mathbb{R}^{n\times p}$ and response vector $Y \in \mathbb{R}^p$, the famous lasso problem:

$$\min_{\beta\in\mathbb{R}^p} \frac{1}{2}\|X\beta - Y\|^2 + \lambda\|\beta\|_1$$

Lecture 4

Norms
Definition: A vector norm on $\mathbb{R}^n$ is a function $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ satisfying:

1. $\|x\| \geq 0 \quad \forall x \in \mathbb{R}^n$, and $\|x\| = 0 \iff x = 0$
2. $\|\alpha x\| = |\alpha|\|x\| \quad \forall \alpha \in \mathbb{R}, x \in \mathbb{R}^n$
3. $\|x + y\| \leq \|x\| + \|y\| \quad \forall x, y \in \mathbb{R}^n$

Examples:

- $\|x\|_1 = \sum_{i=1}^n |x_i|$ (ℓ_1 norm)
- $\|x\|_2 = \left(\sum_{i=1}^n |x_i|^2\right)^{1/2}$ (ℓ_2 norm)
- $\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}, \quad 1 \leq p < \infty$ (ℓ_p norm)
- $\|x\|_\infty = \max_{1\leq i\leq n} |x_i|$ (ℓ_∞ norm)
- $\|x\|_{W,p} = \|Wx\|_p$, where W is a fixed nonsingular matrix, $1 \leq p \leq \infty$

Inner product
Definition:
The trace of a square matrix $C \in \mathbb{R}^{n\times n}$ is

$$\mathrm{Tr}(C) = \sum_{i=1}^{n} C_{ii}.$$

For matrices $A, B \in \mathbb{R}^{m\times n}$, define the standard inner product:

$$\langle A, B\rangle = \mathrm{Tr}(A^T B) = \sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}B_{ij}.$$

Properties:

- $\|A\|_F^2 := \langle A, A\rangle \geq 0$
- $\langle A, A\rangle = 0$ if and only if $A = 0$
- $\langle C, aA + bB\rangle = a\langle C, A\rangle + b\langle C, B\rangle$
- $\langle A + B, A + B\rangle = \langle A, A\rangle + 2\langle A, B\rangle + \langle B, B\rangle$

(i.e., $\|A + B\|_F^2 = \|A\|_F^2 + 2\langle A, B\rangle + \|B\|_F^2$)

Projection onto a closed convex set
Theorem (Projection theorem): Let C be a closed convex set of $\mathbb{R}^n$.

(1) For every z, there exists a unique minimizer of

$$\min_{x\in C} \frac{1}{2}\|x - z\|^2,$$

denoted as $\Pi_C(z)$, the projection of z onto C.

(2) $x^* := \Pi_C(z)$ is the projection of z onto C if and only if

$$\langle z - x^*, x - x^*\rangle \leq 0 \quad \forall x \in C.$$

Proof:
(1) By definition, there exists $x_k \in C$ such that

$$\min_{x\in C} \|z - x\| \leq \|z - x_k\| < \min_{x\in C} \|z - x\| + \frac{1}{k}.$$

It follows that $\{x_k\}$ is bounded. Since C is closed, there exists a convergent subsequence $\{x_{k_l}\}$ such that $x_{k_l} \to x^* \in C$. Therefore,

$$\|z - x^*\| = \min_{x\in C} \|z - x\|.$$

For uniqueness, suppose $x^* \neq \tilde{x}$ both satisfy $x^*, \tilde{x} \in C$ and

$$\|z - x^*\| = \|z - \tilde{x}\| = \min_{x\in C} \|z - x\|.$$

Then,

$$2\|z - x^*\|^2 = \|z - x^*\|^2 + \|z - \tilde{x}\|^2 = 2\left\|z - \frac{x^* + \tilde{x}}{2}\right\|^2 + \frac{1}{2}\|x^* - \tilde{x}\|^2.$$

Since C is convex, $\frac{x^*+\tilde{x}}{2} \in C$. Thus,

$$\left\|z - \frac{x^* + \tilde{x}}{2}\right\|^2 \geq \min_{x\in C} \|z - x\|^2 = \|z - x^*\|^2,$$

which implies

$$2\|z - x^*\|^2 \geq 2\|z - x^*\|^2 + \frac{1}{2}\|x^* - \tilde{x}\|^2.$$

Thus, $\|x^* - \tilde{x}\|^2 \leq 0$ and hence $x^* = \tilde{x}$.
(2) Now, let $x^* = \Pi_C(z)$, and for any $x \in C$, since C is convex,

$$\lambda x + (1 - \lambda)x^* \in C, \quad \forall \lambda \in (0, 1).$$

By minimality,

$$\|z - x^*\|^2 \leq \|z - (\lambda x + (1 - \lambda)x^*)\|^2.$$

Expanding the right-hand side,

$$\|z - (\lambda x + (1 - \lambda)x^*)\|^2 = \|z - x^*\|^2 - 2\lambda\langle z - x^*, x - x^*\rangle + \lambda^2\|x - x^*\|^2.$$

Thus,

$$0 \leq -2\lambda\langle z - x^*, x - x^*\rangle + \lambda^2\|x - x^*\|^2.$$

Dividing by $\lambda > 0$ and taking $\lambda \to 0^+$, we get

$$\langle z - x^*, x - x^*\rangle \leq 0.$$

Conversely, if $\langle z - x^*, x - x^*\rangle \leq 0 \quad \forall x \in C$, then

$$\|z - x\|^2 = \|z - x^*\|^2 + 2\langle z - x^*, x^* - x\rangle + \|x - x^*\|^2 \geq \|z - x^*\|^2,$$

thus x^* minimizes $\|z - x\|$ over C.

arg min
The notation

$$\arg\min_x f(x)$$

denotes the solution set of x for which $f(x)$ attains its minimum (argument of the minimum).

Extended real-valued function
Definition.
Let $\mathcal{X}$ be a Euclidean space (e.g., $\mathcal{X} = \mathbb{R}^n$ or $\mathbb{R}^{m\times n}$). Let $f : \mathcal{X} \to (-\infty, +\infty]$ be an extended real-valued function.

1. The **effective domain** of f is defined as

$$\mathrm{dom}(f) := \{x \in \mathcal{X} \mid f(x) < +\infty\}.$$

2. f is said to be **proper** if $\mathrm{dom}(f) \neq \emptyset$.
3. f is said to be **closed** if its epi-graph

$$\mathrm{epi}(f) := \{(x, \alpha) \in \mathcal{X} \times \mathbb{R} \mid f(x) \leq \alpha\}$$

is closed.
4. f is said to be **convex** if its epi-graph is convex.

Extended real-valued function (continued)

- For a real-valued function $f : \mathcal{X} \to \mathbb{R}$, the convexity of $\mathrm{epi}(f)$ coincides with the following condition:

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y), \quad \forall x, y \in \mathrm{dom}(f),\ \lambda \in [0,1].$$

(Exercise: (1) $\iff$ (2))
- A convex function $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ can be extended to a convex function on all of $\mathbb{R}^n$ by setting $f(x) = +\infty$ for $x \notin D$.

Proof of Exercise (1) $\iff$ (2)
(1) $\Rightarrow$ (2):
Assume $\mathrm{epi}(f)$ is convex. Take any $(x, f(x)) \in \mathrm{epi}(f)$ and $(y, f(y)) \in \mathrm{epi}(f)$. For any $\lambda \in [0,1]$, by convexity of $\mathrm{epi}(f)$,

$$\lambda(x, f(x)) + (1-\lambda)(y, f(y)) = (\lambda x + (1-\lambda)y, \lambda f(x) + (1-\lambda)f(y)) \in \mathrm{epi}(f).$$

Hence,

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y).$$

Thus, (2) holds.
(2) $\Rightarrow$ (1):
Assume

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y), \quad \forall x, y \in \mathrm{dom}(f),\ \lambda \in [0,1].$$

Let $(x, \alpha), (y, \beta) \in \mathrm{epi}(f)$. Then $f(x) \leq \alpha$ and $f(y) \leq \beta$.
For any $\lambda \in [0,1]$,

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y) \leq \lambda\alpha + (1-\lambda)\beta.$$

Therefore,

$$(\lambda x + (1-\lambda)y, \lambda\alpha + (1-\lambda)\beta) \in \mathrm{epi}(f),$$

which shows $\mathrm{epi}(f)$ is convex. Thus, (1) holds.

Indicator function
Let C be a nonempty set in $\mathcal{X}$.
The **indicator function** of C is

$$\delta_C(x) = \begin{cases} 0, & \text{if } x \in C, \\ +\infty, & \text{if } x \notin C. \end{cases}$$

Let $f(x) = \delta_C(x)$.
Then:

- $\mathrm{dom}(f) = C$, and f is proper.
- $\mathrm{epi}(f) = C \times [0, +\infty)$ is closed if C is closed.
- $\mathrm{epi}(f)$ is convex if C is convex.

Dual/polar cone
Definition (Cone):
A set $C \subseteq \mathcal{X}$ is called a **cone** if $\lambda x \in C$ when $x \in C$ and $\lambda \geq 0$.
Definition (Dual and polar cone):
The **dual cone** of a set $C \subseteq \mathcal{X}$ is

$$C^* := \{y \in \mathcal{X} \mid \langle x, y\rangle \geq 0 \quad \forall x \in C\}.$$

The **polar cone** of C is $C^\circ = -C^*$.
If $C^* = C$, then C is said to be **self-dual**.

- C^* is always a convex cone, even if C is neither convex nor a cone.

Normal cone
Definition (Normal cone):
Let C be a convex set in $\mathcal{X}$ and $\bar{x} \in C$. The normal cone of C at $\bar{x} \in C$ is defined by

$$N_C(\bar{x}) := \{z \in \mathcal{X} \mid \langle z, x - \bar{x}\rangle \leq 0 \quad \forall x \in C\}.$$

By convention, we let $N_C(\bar{x}) = \emptyset$ if $\bar{x} \notin C$.

Proposition:
Let $C \subseteq \mathcal{X}$ be a nonempty convex set and $\bar{x} \in C$. Then:

1. $N_C(\bar{x})$ is a closed convex cone.
2. If $\bar{x} \in \mathrm{int}(C)$ (interior point), then $N_C(\bar{x}) = \{0\}$.
3. If C is a cone, then $N_C(\bar{x}) \subseteq C^\circ$.

Proof:
(1) Proof of closeness:
Let $\{z_k\}$ be a sequence in $N_C(\bar{x})$ such that

$$z_k \to z \quad \text{(in norm)}.$$

We want to show $z \in N_C(\bar{x})$. Since $z_k \in N_C(\bar{x})$, we have

$$\langle z_k, x - \bar{x}\rangle \leq 0 \quad \forall x \in C, \quad \forall k.$$

Fix $x \in C$. By continuity of the inner product and $z_k \to z$,

$$\langle z_k, x - \bar{x}\rangle \to \langle z, x - \bar{x}\rangle.$$

Since each $\langle z_k, x - \bar{x}\rangle \leq 0$, taking limits gives

$$\langle z, x - \bar{x}\rangle \leq 0.$$

Thus $z \in N_C(\bar{x})$. Hence, $N_C(\bar{x})$ is closed.

First, we prove that $N_C(\bar{x})$ is a cone. Let $z \in N_C(\bar{x})$ and $\lambda \geq 0$. By definition,

$$\langle z, x - \bar{x}\rangle \leq 0 \quad \forall x \in C$$

implies

$$\langle \lambda z, x - \bar{x}\rangle = \lambda\langle z, x - \bar{x}\rangle \leq 0 \quad \forall x \in C.$$

Thus $\lambda z \in N_C(\bar{x})$.
Next, if $z_1, z_2 \in N_C(\bar{x})$, then

$$\langle z_1, x - \bar{x}\rangle \leq 0 \quad \text{and} \quad \langle z_2, x - \bar{x}\rangle \leq 0 \quad \forall x \in C.$$

Adding, we get

$$\langle z_1 + z_2, x - \bar{x}\rangle \leq 0 \quad \forall x \in C,$$

so $z_1 + z_2 \in N_C(\bar{x})$. Therefore, $N_C(\bar{x})$ is convex.
(2) Let $z \in N_C(\bar{x})$. Since $\bar{x} \in \mathrm{int}(C)$, there exists $\epsilon > 0$ such that $\bar{x} + tz \in C$ for all $|t| < \epsilon$. By definition of the normal cone,

$$0 \geq \langle z, (\bar{x} + tz) - \bar{x}\rangle = t\|z\|^2.$$

For small positive $t > 0$, $t\|z\|^2 \geq 0$, so $t\|z\|^2 \leq 0$. Hence, $\|z\|^2 = 0$, thus $z = 0$. Therefore, $N_C(\bar{x}) = \{0\}$.
(3) Suppose C is a cone. Then for any $x \in C$, $\bar{x} + x \in C$. For any $z \in N_C(\bar{x})$, we have

$$\langle z, x\rangle = \langle z, (\bar{x} + x) - \bar{x}\rangle \leq 0 \quad \forall x \in C.$$

Hence, $z \in C^\circ$. Thus, $N_C(\bar{x}) \subseteq C^\circ$.

Proposition:
Let $C \subseteq \mathcal{X}$ be a nonempty closed convex set. Then for any $y \in C$,

$$u \in N_C(y) \quad \iff \quad y = \Pi_C(y + u),$$

where $\Pi_C(\cdot)$ is the projection onto C.

Proof:
"$\Rightarrow$" Suppose $u \in N_C(y)$. Then

$$\langle u, x - y\rangle \leq 0 \quad \forall x \in C.$$

Thus,

$$\langle (y + u) - y, x - y\rangle \leq 0 \quad \forall x \in C,$$

which implies that $y = \Pi_C(y + u)$.
"$\Leftarrow$" Suppose $y = \Pi_C(y + u)$. Then for all $x \in C$,

$$\langle u, x - y\rangle = \langle (y + u) - y, x - y\rangle \leq 0,$$

which implies $u \in N_C(y)$.

Subdifferential
Definition:
Let $f : \mathcal{X} \to (-\infty, +\infty]$ be a convex function.
We call v a **subgradient** of f at $x \in \operatorname{dom}(f)$ if

$$f(z) \geq f(x) + \langle v, z - x \rangle \quad \forall z \in \mathcal{X}.$$

The set of all subgradients at x is called the **subdifferential** of f at x, denoted by

$$\partial f(x) = \{v \mid f(z) \geq f(x) + \langle v, z - x \rangle \quad \forall z \in \mathcal{X}\}.$$

By convention, $\partial f(x) = \emptyset$ for any $x \notin \operatorname{dom}(f)$.

Subdifferential and optimization
Subgradient is an extension of the gradient:

- If f is differentiable at x, then
$$\partial f(x) = \{\nabla f(x)\}.$$

Proof:
Suppose $v \in \partial f(x)$. Then
$$f(x + h) \geq f(x) + \langle v, h \rangle \quad \forall h.$$

Take $h = t(v - \nabla f(x))$ with $t > 0$, and use Taylor's expansion:

$$f(x + h) = f(x) + \langle \nabla f(x), h \rangle + o(\|h\|).$$

So
$$\langle \nabla f(x), h \rangle + o(\|h\|) \geq \langle v, h \rangle.$$

Dividing by t and letting $t \to 0^+$, we conclude $v = \nabla f(x)$.

Theorem:
Let $f : \mathcal{X} \to (-\infty, +\infty]$ be a proper convex function.
Then $\bar{x} \in \mathcal{X}$ is a global minimizer of $\min_{x \in \mathcal{X}} f(x)$ if and only if

$$0 \in \partial f(\bar{x}).$$

Proof:
By the subgradient inequality,
$$f(z) \geq f(\bar{x}) + \langle v, z - \bar{x} \rangle \quad \forall z \in \mathcal{X}.$$

Take $v = 0 \in \partial f(\bar{x})$, then
$$f(z) \geq f(\bar{x}),$$

so $\bar{x}$ is a global minimizer.

Lipschitz continuous
Definition (Lipschitz continuous):
A function $F : \mathbb{R}^n \to \mathbb{R}^m$ is said to be **locally Lipschitz continuous** if for any open set $\mathcal{O} \subseteq \mathbb{R}^n$, there exists a constant L such that
$$\|F(x) - F(y)\| \leq L\|x - y\| \quad \forall x, y \in \mathcal{O}.$$

If $\mathcal{O} = \mathbb{R}^n$, then F is called **globally Lipschitz continuous**.
Examples:

- $f(x) = |x|, \quad x \in \mathbb{R}$ is globally Lipschitz continuous with Lipschitz constant $L = 1$.
- $f(x) = x^2, \quad x \in \mathbb{R}$ is locally Lipschitz continuous but not globally Lipschitz continuous.

Fenchel conjugate
Definition:
Let $f : \mathcal{X} \to [-\infty, +\infty]$.
The (Fenchel) conjugate of f is defined as

$$f^*(y) = \sup\{\langle y, x \rangle - f(x) \mid x \in \mathcal{X}\}, \quad y \in \mathcal{X}.$$

Remark:

- f^* is always closed and convex, even if f is neither convex nor closed.
- If $f : \mathcal{X} \to (-\infty, +\infty]$ is closed and proper convex, then

$$(f^*)^* = f.$$

Proposition:
Let $f : \mathcal{X} \to (-\infty, +\infty]$ be a closed proper convex function. The following are equivalent:

1. $f(x) + f^*(y) = \langle x, y \rangle$
2. $y \in \partial f(x)$
3. $x \in \partial f^*(y)$

The equivalence $y \in \partial f(x) \iff x \in \partial f^*(y)$ means that ∂f^* is the **inverse** of ∂f (in the sense of multi-valued mappings).

Moreau envelope and proximal mapping
Let $f : \mathcal{X} \to (-\infty, +\infty]$ be a closed proper convex function. We define:

- **Moreau envelope** (Moreau-Yosida regularization) of f at x:

$$M_f(x) = \min_y \left\{ f(y) + \frac{1}{2}\|y - x\|^2 \right\}.$$

- **Proximal mapping** of f at x:
$$P_f(x) = \arg\min_y \left\{ f(y) + \frac{1}{2}\|y - x\|^2 \right\}.$$

Properties:

- $M_f(x)$ is differentiable, and its gradient is

$$\nabla M_f(x) = x - P_f(x).$$

- $P_f(x)$ exists and is unique.
- $M_f(x) \leq f(x)$.
- $\arg\min_x f(x) = \arg\min_x M_f(x)$.

(The Moreau envelope is a way to **smooth a possibly non-differentiable convex function**.)

Example 1: Projection onto a closed convex set
Let $C \subseteq \mathcal{X}$ be a nonempty closed convex set, and $f(x) = \delta_C(x)$ (indicator function). Its proximal mapping:

$$P_f(x) = \arg\min_{y \in \mathcal{X}} \left\{ \delta_C(y) + \frac{1}{2}\|y - x\|^2 \right\} = \arg\min_{y \in C} \frac{1}{2}\|y - x\|^2 = \Pi_C(x),$$

where $\Pi_C(x)$ denotes projection onto C. Its Moreau envelope:

$$M_f(x) = \frac{1}{2}\|x - \Pi_C(x)\|^2.$$

Example 2: Huber function and soft thresholding
Let $f(x) = \lambda|x|, \quad x \in \mathbb{R}$. Then its Moreau envelope (Huber function) is

$$M_f(x) = \begin{cases} \frac{1}{2}x^2, & |x| \leq \lambda, \\ \lambda|x| - \frac{\lambda^2}{2}, & |x| > \lambda. \end{cases}$$

Its proximal mapping (soft thresholding):

$$P_f(x) = \operatorname{sign}(x)\max\{|x| - \lambda, 0\}.$$

Soft thresholding operator
The soft thresholding operator $S_\lambda : \mathbb{R}^n \to \mathbb{R}^n$ is defined as:

$$S_\lambda(x) = \begin{bmatrix} \operatorname{sign}(x_1)\max\{|x_1| - \lambda, 0\} \\ \operatorname{sign}(x_2)\max\{|x_2| - \lambda, 0\} \\ \vdots \\ \operatorname{sign}(x_n)\max\{|x_n| - \lambda, 0\} \end{bmatrix},$$

for any $x = [x_1, \ldots, x_n] \in \mathbb{R}^n$ and $\lambda > 0$.

Moreau decomposition
Theorem (Moreau decomposition):
Let $f : \mathcal{X} \to (-\infty, +\infty]$ be a closed proper convex function and f^* its Fenchel conjugate. Then, for any $x \in \mathcal{X}$,

$$x = P_f(x) + P_{f^*}(x),$$

and
$$\frac{1}{2}\|x\|^2 = M_f(x) + M_{f^*}(x).$$

Example: Let $C \subseteq \mathcal{X}$ be a nonempty closed convex cone. Take $f(x) = \delta_C(x)$, so $f^*(x) = \delta_{C^\circ}(x)$. Therefore,

$$x = \Pi_C(x) + \Pi_{C^\circ}(x).$$

Remarks:

- $M_f(\cdot)$ is always differentiable, even if f is non-differentiable.
- $P_f(\cdot)$ is important for optimization algorithms (e.g., accelerated proximal gradient methods).
- For many regularizers, $P_f(\cdot)$ and $M_f(\cdot)$ have explicit expressions.

A proximal point view of gradient methods
To minimize a differentiable function $\min_\beta f(\beta)$, the gradient update is

$$\beta^{(k+1)} = \beta^{(k)} - \alpha_k \nabla f(\beta^{(k)}).$$

The gradient step can be equivalently written as ((linear approximation + proximal term):

$$\beta^{(k+1)} = \arg\min_\beta \left\{ f(\beta^{(k)}) + \langle \nabla f(\beta^{(k)}), \beta - \beta^{(k)} \rangle + \frac{1}{2\alpha_k} \|\beta - \beta^{(k)}\|^2 \right\}.$$

Optimizing composite functions
Minimize

$$\min_{\beta \in \mathbb{R}^p} f(\beta) + g(\beta),$$

where:

- $f : \mathbb{R}^p \to \mathbb{R}$ is convex, differentiable, ∇f is L-Lipschitz continuous,
- $g : \mathbb{R}^p \to (-\infty, +\infty]$ is closed, proper convex, possibly non-differentiable.

Example (Lasso):

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{2}\|X\beta - Y\|^2 + \lambda \|\beta\|_1,$$

where the first term is $f(\beta)$ and the second is $g(\beta)$. Since g is non-differentiable, gradient methods alone cannot be applied.

Proximal gradient step
Gradient step (if $g(\beta)$ disappears):

$$\beta^{(k+1)} = \beta^{(k)} - \alpha_k \nabla f(\beta^{(k)}),$$

or equivalently,

$$\beta^{(k+1)} = \arg\min_\beta \left\{ f(\beta^{(k)}) + \langle \nabla f(\beta^{(k)}), \beta - \beta^{(k)} \rangle + \frac{1}{2\alpha_k} \|\beta - \beta^{(k)}\|^2 \right\}.$$

Proximal gradient step for $f + g$:

$$\beta^{(k+1)} = \arg\min_\beta \left\{ f(\beta^{(k)}) + \langle \nabla f(\beta^{(k)}), \beta - \beta^{(k)} \rangle + g(\beta) + \frac{1}{2\alpha_k} \|\beta - \beta^{(k)}\|^2 \right\}.$$

After ignoring constant terms and completing the square:

$$\beta^{(k+1)} = \arg\min_\beta \left\{ \frac{1}{2\alpha_k} \left\| \beta - \left(\beta^{(k)} - \alpha_k \nabla f(\beta^{(k)}) \right) \right\|^2 + g(\beta) \right\}.$$

That is,

$$\beta^{(k+1)} = P_{\alpha_k g}\left(\beta^{(k)} - \alpha_k \nabla f(\beta^{(k)}) \right),$$

where $P_{\alpha_k g}$ denotes the proximal operator associated to $\alpha_k g$.

Derivation (completing the square):

$$\langle \nabla f(\beta^{(k)}), \beta \rangle + \frac{1}{2\alpha_k} \|\beta - \beta^{(k)}\|^2$$

$$= \langle \nabla f(\beta^{(k)}) - \frac{1}{\alpha_k} \beta^{(k)}, \beta \rangle + \frac{1}{2\alpha_k} \|\beta\|^2 + \text{constant}.$$

Convergence Rate of PG:
In convex problems, PG method satisfies

$$f(\beta^{(k)}) + g(\beta^{(k)}) - \min_{\beta \in \mathbb{R}^p} (f(\beta) + g(\beta)) \le O\left(\frac{1}{k}\right).$$

If stopping condition

$$f(\beta^{(k)}) + g(\beta^{(k)}) - \text{optimal value} \le 10^{-4},$$

then around $O(10^4)$ iterations needed.

Algorithm 6 Proximal Gradient (PG) Method

1: **Initialization:** Choose initial point $\beta^{(0)}$, step size $\alpha > 0$, set $k \leftarrow 0$.
2: **while** not converged **do**
3: Update the iterate:

$$\beta^{(k+1)} = P_{\alpha g}\left(\beta^{(k)} - \alpha \nabla f(\beta^{(k)}) \right)$$

4: Increment $k \leftarrow k + 1$.
5: **end while**
6: **Output:** $\beta^{(k)}$ (approximate solution)

Algorithm 7 Accelerated Proximal Gradient (APG) Method

1: **Initialization:** Choose initial point $\beta^{(0)}$, set $t_0 = t_1 = 1$, step size $\alpha > 0$, set $k \leftarrow 0$.
2: **while** not converged **do**
3: Compute extrapolated point:

$$\bar{\beta}^{(k)} = \beta^{(k)} + \frac{t_k - 1}{t_{k+1}} \left(\beta^{(k)} - \beta^{(k-1)} \right)$$

4: Proximal step:

$$\beta^{(k+1)} = P_{\alpha g}\left(\bar{\beta}^{(k)} - \alpha \nabla f(\bar{\beta}^{(k)}) \right)$$

5: Update:

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2}$$

6: Increment $k \leftarrow k + 1$.
7: **end while**
8: **Output:** $\beta^{(k)}$ (approximate solution)

Convergence Rate of APG:
In convex problems, APG method satisfies

$$f(\beta^{(k)}) + g(\beta^{(k)}) - \min_{\beta \in \mathbb{R}^p} (f(\beta) + g(\beta)) \le O\left(\frac{1}{k^2}\right).$$

If stopping condition

$$f(\beta^{(k)}) + g(\beta^{(k)}) - \text{optimal value} \le 10^{-4},$$

then around $O(10^2)$ iterations needed.

Accelerated Proximal Gradient (APG) Methods

- Backtracking line search can also be used for finding step length α_k.
- For simplicity, we often take a constant step length. It should satisfy

$$\alpha \in \left(0, \frac{1}{L}\right),$$

  where L is the Lipschitz constant of $\nabla f(\cdot)$ (typically unknown).
- APG methods enjoy the same computational cost per iteration as PG methods.
- Iteration complexity:

$$\text{APG:} \quad O\left(\frac{1}{k^2}\right), \quad \text{PG:} \quad O\left(\frac{1}{k}\right).$$

APG for Lasso
Solve:

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{2}\|X\beta - Y\|^2 + \lambda\|\beta\|_1 \quad \text{(lasso problem)},$$

where

$$f(\beta) = \frac{1}{2}\|X\beta - Y\|^2, \quad g(\beta) = \lambda \|\beta\|_1.$$

Then

$$\nabla f(\beta) = X^T (X\beta - Y) \quad \text{with Lipschitz constant} \quad L = \lambda_{\max}(X^T X).$$

Choose step length $\alpha = 1/L$.
APG iterations:

$$\bar{\beta}^{(k)} = \beta^{(k)} + \frac{t_k - 1}{t_{k+1}}(\beta^{(k)} - \beta^{(k-1)}),$$

$$\beta^{(k+1)} = S_{\lambda/L}\left(\bar{\beta}^{(k)} - \frac{1}{L}X^T(X\bar{\beta}^{(k)} - Y)\right),$$

where $S_{\lambda/L}$ is the soft-thresholding operator.
APG is also applicable to "logistic regression + lasso regularization".

Restart Strategy

- To speed up APG, restart the algorithm after a fixed number of iterations.
- Use the latest iterate as the starting point for a new APG round.
- A reasonable choice is to restart every 100 or 200 iterations.

Lecture 5

Idea of Support Vector Machine (SVM)

- Data: $x_i \in \mathbb{R}^p, \quad y_i \in \{-1, 1\}$ (instead of $\{0, 1\}$ in logistic regression), $i = 1, \ldots, n$.
- The two classes are assumed to be linearly separable.
- Aim: Learn a linear classifier: $f(x) = \mathrm{sign}(\beta^T x + \beta_0)$.
- Question: What is the "best"separating hyperplane?
- SVM answer: The hyperplane with **maximum margin**.
- Margin = distance to the closest data points.

For the separating hyperplane with maximum margin:

Distance to positive class = Distance to negative class.

Normal Cone of a Hyperplane
Hyperplane:

$$H = H_{\beta,\beta_0} = \{x \in \mathbb{R}^p \mid \beta^T x + \beta_0 = 0\},$$

which is:

- A linear decision boundary,
- A $(p-1)$-dimensional subspace, closed, convex.

For any $\bar{x} \in H$, the normal cone:

$$N_H(\bar{x}) = \{\lambda\beta \mid \lambda \in \mathbb{R}\}.$$

We can show that:

$$\beta \in N_H(\bar{x}), \quad \text{i.e.,} \quad \langle \beta, z - \bar{x}\rangle \le 0 \quad \forall z \in H.$$

(Since for any $z, \bar{x} \in H$, $\beta^T z + \beta_0 = \beta^T \bar{x} + \beta_0 = 0$).

Distance of a Point to a Hyperplane
To compute distance of x to hyperplane H:

$$\text{Distance} = \frac{|\beta^T x + \beta_0|}{\|\beta\|}.$$

Key steps:

1. Projection $\bar{x} = \Pi_H(x)$ implies $x - \bar{x} \in N_H(\bar{x}) \Rightarrow x - \bar{x} = \lambda\beta$.
2. Use that $\bar{x} \in H$: $\beta^T \bar{x} + \beta_0 = 0$.
3. Solve for λ and compute distance.

Important: The distance is **invariant to scaling** of β, β_0.

Maximize Margin
Define the margin:

$$\gamma(\beta, \beta_0) = \min_{i=1,\ldots,n} \frac{|\beta^T x_i + \beta_0|}{\|\beta\|}.$$

Constraints:

$$\beta^T x_i + \beta_0 \ge 0 \quad \text{when} \quad y_i = 1,$$

$$\beta^T x_i + \beta_0 \le 0 \quad \text{when} \quad y_i = -1,$$

or equivalently:

$$y_i(\beta^T x_i + \beta_0) \ge 0, \quad \forall i \in [n].$$

Therefore, the optimization becomes:

$$\max_{\beta,\beta_0} \left\{ \min_{i=1,\ldots,n} \frac{|\beta^T x_i + \beta_0|}{\|\beta\|} \right\} \quad \text{subject to} \quad y_i(\beta^T x_i + \beta_0) \ge 0 \quad \forall i.$$

Simplify the Optimization Problem
Using scale invariance:

$$\max_{\beta,\beta_0} \frac{1}{\|\beta\|} \quad \text{s.t.} \quad y_i(\beta^T x_i + \beta_0) \ge 0, \quad \min_i |\beta^T x_i + \beta_0| = 1.$$

Thus, equivalent to:

$$\min_{\beta,\beta_0} \|\beta\|^2 \quad \text{subject to} \quad y_i(\beta^T x_i + \beta_0) \ge 1, \quad \forall i.$$

Notes:

- "$\Rightarrow$"relies on $y_i \in \{-1, 1\}$.
- "$\Leftarrow$"we minimize $\|\beta\|$.

SVM
SVM is a quadratic programming (QP) problem — it can be solved by generic QP solvers:

$$\min_{\beta,\beta_0} \quad \frac{1}{2}\|\beta\|^2 \quad \text{s.t.} \quad y_i(\beta^T x_i + \beta_0) \ge 1 \quad \forall i \in [n].$$

- Later, we will discuss the **Lagrangian duality** and derive the dual problem.
- The dual problem will help us use **kernels** (introduced later).
- The dual problem also provides a more efficient algorithm, especially when $n \ll p$.

Support Vectors
Support vectors are data points x_i satisfying tight constraints:

$$y_i(\beta^T x_i + \beta_0) = 1.$$

- Support vectors must exist.
- Number of support vectors $\ll$ sample size n.
- Removing a support vector may change the hyperplane.

Lagrangian
For a general nonlinear programming problem (NLP) (called primal problem (P)):

$$(P) \quad \min_{x \in \mathbb{R}^p} f(x) \quad \text{s.t.} \quad g_i(x) = 0,\ i \in [m], \quad h_j(x) \le 0,\ j \in [l].$$

Define the Lagrangian:

$$L(x, v, u) = f(x) + \sum_{i=1}^{m} v_i g_i(x) + \sum_{j=1}^{l} u_j h_j(x),$$

where $v = [v_1, \ldots, v_m] \in \mathbb{R}^m$, $u = [u_1, \ldots, u_l] \in \mathbb{R}^l_+$.
Define the **Lagrange dual function**:

$$\theta(v, u) = \min_x L(x, v, u).$$

Properties of the Lagrangian Dual Function

- To compute $\theta(v, u)$, solve $\min_x L(x, v, u)$.
- If f, g_i, and h_j are convex and differentiable, may use $\partial L/\partial x = 0$.
- $\theta(v, u)$ is always concave even if (P) is non-convex.
- For any feasible x, $\theta(v, u) \le f(x)$ (weak duality).

Lagrangian Dual Problem
The dual function $\theta(v, u)$ provides a lower bound:

$$\theta(v, u) \le f(x) \quad \text{for all primal feasible } x.$$

Therefore, we search for the largest lower bound:

$$(D) \quad \max_{v,u} \theta(v, u) \quad \text{s.t.} \quad v \in \mathbb{R}^m,\ u \in \mathbb{R}^l_+.$$

v_i, u_j are called **dual variables** or **Lagrange multipliers**.

Primal and Dual Relationship
For primal problem (P) and dual problem (D):

$$\text{(P)} \quad \min_{x \in \mathbb{R}^p} f(x) \quad \text{s.t.} \quad g_i(x) = 0, \quad h_j(x) \le 0,$$

$$\text{(D)} \quad \max_{v,u} \theta(v, u) = \min_x \left(f(x) + \sum_{i=1}^{m} v_i g_i(x) + \sum_{j=1}^{l} u_j h_j(x) \right).$$

Key duality concepts:

- **Weak duality:** Optimal value for (D) $\le$ Optimal value for (P).
- **Strong duality:** Under certain conditions, (D) optimal value = (P) optimal value.

KKT Assumptions:

- $f, h_j : \mathbb{R}^p \to \mathbb{R}$ are differentiable and convex
- $g_i : \mathbb{R}^p \to \mathbb{R}$ are affine: $g_i(x) = a_i^T x + b_i$
- **Slater's condition** holds: there exists $\hat{x}$ such that

$$g_i(\hat{x}) = 0, \quad \forall i, \quad h_j(\hat{x}) < 0, \quad \forall j$$

Under these assumptions, strong duality holds and there exist solutions x^* and (u^*, v^*) satisfying the KKT conditions:

$$\nabla f(x^*) + \sum_{i=1}^{m} v_i^* \nabla g_i(x^*) + \sum_{j=1}^{l} u_j^* \nabla h_j(x^*) = 0$$

$$g_i(x^*) = 0, \quad h_j(x^*) \le 0, \quad u_j^* \ge 0, \quad u_j^* h_j(x^*) = 0, \quad \forall i, j$$

KKT points and Complementary slackness:

- (x^*, u^*, v^*) (or simply x^*) is called a **KKT point** if it satisfies the KKT conditions.
- (x^*, u^*, v^*) is a KKT solution if and only if x^* is optimal for (P) and (u^*, v^*) is optimal for (D).

$$u_j^* h_j(x^*) = 0 \quad \forall j$$

Complementary slackness means:

- $h_j(x^*) < 0 \Rightarrow u_j^* = 0$
- $u_j^* > 0 \Rightarrow h_j(x^*) = 0$

Dual of SVM:
Consider the primal SVM problem:

$$\min_{\beta, \beta_0} \frac{1}{2}\|\beta\|^2 \quad \text{s.t.} \quad 1 - y_i(\beta^T x_i + \beta_0) \le 0, \quad \forall i$$

Step 1: Write the Lagrangian:

$$L(\beta, \beta_0, \alpha) = \frac{1}{2}\|\beta\|^2 + \sum_{i=1}^{n} \alpha_i (1 - y_i(\beta^T x_i + \beta_0))$$

where $\alpha_i \ge 0$.
Step 2: The dual function is

$$\theta(\alpha) = \min_{\beta, \beta_0} L(\beta, \beta_0, \alpha)$$

Expanding:

$$L = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{n} \alpha_i y_i x_i^T \beta - \left(\sum_{i=1}^{n} \alpha_i y_i\right) \beta_0 + \sum_{i=1}^{n} \alpha_i$$

Setting the gradient to zero:

$$\frac{\partial}{\partial \beta} L = \beta - \sum_{i=1}^{n} \alpha_i y_i x_i = 0$$

$$\frac{\partial}{\partial \beta_0} L = - \sum_{i=1}^{n} \alpha_i y_i = 0$$

Thus:

$$\beta = \sum_{i=1}^{n} \alpha_i y_i x_i$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

Substituting into L, we get:

$$\theta(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle$$

The dual problem becomes:

$$\max_{\alpha} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \quad \text{s.t.} \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \quad \alpha_i \ge 0$$

KKT conditions for SVM:

$$\sum_{i=1}^{n} \alpha_i^* y_i x_i = \beta^*, \quad \sum_{i=1}^{n} \alpha_i^* y_i = 0$$

$$y_i((\beta^*)^T x_i + \beta_0^*) \ge 1, \quad \alpha_i^* \ge 0$$

$$\alpha_i^*(1 - y_i((\beta^*)^T x_i + \beta_0^*)) = 0$$

Given α^*:

$$\beta^* = \sum_{i=1}^{n} \alpha_i^* y_i x_i$$

Choose any support vector x_k with $\alpha_k^* > 0$, then

$$\beta_0^* = y_k - \sum_{i=1}^{n} \alpha_i^* y_i \langle x_i, x_k \rangle$$

Support vectors and sparsity:

- $\alpha_i^* > 0 \quad \Rightarrow \quad y_i((\beta^*)^T x_i + \beta_0^*) = 1$
- Most α_i^* are 0; thus the solution is sparse.

Decision boundary:
The decision boundary is:

$$0 = (\beta^*)^T x + \beta_0^* = \sum_{i=1}^{n} \alpha_i^* y_i \langle x_i, x \rangle + \beta_0^* = \sum_{\alpha_i^* > 0} \alpha_i^* y_i \langle x_i, x \rangle + \beta_0^*$$

Thus, the decision boundary depends only on support vectors.

Primal vs Dual
Primal

$$\min_{\beta, \beta_0} \quad \frac{1}{2}\|\beta\|^2 \quad \text{s.t.} \quad y_i(\beta^T x_i + \beta_0) \ge 1, \quad i \in [n]$$

Classifier

$$f(x) = \text{sign}(\beta^T x + \beta_0)$$

Dual

$$\max_{\alpha} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \quad \text{s.t.} \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \quad \alpha_i \ge 0, \quad i \in [n]$$

Classifier

$$f(x) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \langle x_i, x \rangle + \beta_0\right)$$

Many α_i are zero (sparse solutions).

- Optimize $p + 1$ variables for primal, n variables for dual
- When $n \ll p$, it might be more efficient to solve the dual
- Dual problem only involves $\langle x_i, x_j \rangle$ — allowing the use of kernels

Feature mapping

- Recall feature expansion, for example,

$$x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix} \quad \xrightarrow{\text{feature expansion}} \quad \begin{bmatrix} x_{i1} \\ x_{i2} \\ x_{i1}^2 \\ x_{i2}^2 \\ x_{i1} x_{i2} \end{bmatrix}$$

- Let ϕ denote the feature mapping, which maps from original features to new features

$$\phi\left(\begin{bmatrix} z_1 \\ z_2 \end{bmatrix}\right) = \begin{bmatrix} z_1 \\ z_2 \\ z_1^2 \\ z_2^2 \\ z_1 z_2 \end{bmatrix}$$

- Instead of using the original feature vectors x_i, we may apply SVM using new features $\phi(x_i)$
- New feature space can be very high dimensional

Kernel:
Primal

$$\min_{\beta, \beta_0} \quad \frac{1}{2}\|\beta\|^2 \quad \text{s.t.} \quad y_i(\beta^T x_i + \beta_0) \ge 1, \quad i \in [n]$$

Dual

$$\max_{\alpha} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \quad \text{s.t.} \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \quad \alpha_i \ge 0, \quad i \in [n]$$

Using dual:

- For feature expansion, simply replace $\langle x_i, x_j \rangle$ with $\langle \phi(x_i), \phi(x_j) \rangle$
- Given a feature mapping ϕ, we define the corresponding **kernel**

$$K(a, b) = \langle \phi(a), \phi(b) \rangle, \quad a, b \in \mathbb{R}^p$$

- Usually computing $K(a, b)$ may be very cheap, even though computing $\phi(a), \phi(b)$ may be expensive
- The dual of SVM only requires the computation of kernels $K(x_i, x_j)$. Explicitly calculating $\phi(x_i)$ is not necessary

Common kernels

- Polynomials of degree d
$$K(a,b) = (a^T b)^d$$
- Polynomials up to degree d
$$K(a,b) = (a^T b + 1)^d$$
- Gaussian kernel — polynomials of all orders

$$K(a,b) = \exp\left(-\frac{\|a-b\|^2}{2\sigma^2}\right), \quad \sigma > 0$$

Kernel

- SVM can be applied in high dimensional feature spaces, without explicitly applying the feature mapping
- The two classes might be separable in high dimensional space, but not separable in the original feature space
- Kernels can be used efficiently in the dual problem of SVM because the dual only involves inner products

SVM with soft constraints

When the two classes are not separable, no feasible separating hyperplane exists. We **allow the constraints to be violated slightly** ($C > 0$ is given)

$$\min_{\beta,\beta_0,\epsilon} \quad \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^n \epsilon_i \quad \text{s.t.} \quad y_i(\beta^T x_i + \beta_0) \geq 1 - \epsilon_i, \quad \epsilon_i \geq 0, \quad i \in [n]$$

$$\epsilon_i = \begin{cases} 1 - y_i(\beta^T x_i + \beta_0), & \text{if } y_i(\beta^T x_i + \beta_0) < 1 \\ 0, & \text{if } y_i(\beta^T x_i + \beta_0) \geq 1 \end{cases} = \max\{1 - y_i(\beta^T x_i + \beta_0), 0\}$$

SVM with soft constraints solves

$$\min_{\beta,\beta_0} \quad \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^n \max\{1 - y_i(\beta^T x_i + \beta_0), 0\}$$

(ridge regularization + hinge-loss function)

SVM vs. logistic regression

SVM with soft constraints

$$\min_{\beta,\beta_0} \quad C\sum_{i=1}^n \max\{1 - y_i(\beta^T x_i + \beta_0), 0\} + \frac{1}{2}\|\beta\|^2$$

Hinge-loss

$$\text{hinge-loss} = \max\{1 - z, 0\}, \quad z = y_i(\beta^T x_i + \beta_0) \quad \text{hope } z \geq 1$$

Logistic regression with ridge regularization

$$\min_{\beta,\beta_0} \quad \sum_{i=1}^n \log\left(1 + e^{-z}\right) + \lambda\|\beta\|^2$$

Logistic-loss

$$\text{logistic-loss} = \log(1 + e^{-z}), \quad z = y_i(\beta^T x_i + \beta_0) \quad \text{hope } z \gg 0$$

SVM with soft constraints: dual

SVM with soft constraints:

$$\min_{\beta,\beta_0,\epsilon} \quad \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^n \epsilon_i \quad \text{s.t.} \quad 1 - \epsilon_i - y_i(\beta^T x_i + \beta_0) \leq 0, \quad -\epsilon_i \leq 0, \quad i \in [n]$$

Find the dual problem.
1. For $\alpha \in \mathbb{R}^n_+, r \in \mathbb{R}^n_+$, the Lagrangian $L(\beta, \beta_0, \epsilon, \alpha, r)$ is

$$L = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^n \epsilon_i + \sum_{i=1}^n \alpha_i(1 - \epsilon_i - y_i(\beta^T x_i + \beta_0)) - \sum_{i=1}^n r_i\epsilon_i$$

2. The dual function is

$$\theta(\alpha, r) = \min_{\beta,\beta_0,\epsilon} L(\beta, \beta_0, \epsilon, \alpha, r)$$

By setting

$$\frac{\partial}{\partial\beta}L = \beta - \sum_{i=1}^n \alpha_i y_i x_i = 0, \quad \frac{\partial}{\partial\beta_0}L = -\sum_{i=1}^n \alpha_i y_i = 0 \quad \frac{\partial}{\partial\epsilon_i}L = C - \alpha_i - r_i = 0$$

We obtain

$$\theta(\alpha, r) = \begin{cases} \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle, & \text{if } \sum_{i=1}^n \alpha_i y_i = 0 \text{ and } \alpha_i + r_i = C \\ -\infty, & \text{otherwise} \end{cases}$$

3. The dual problem

$$\max_\alpha \quad \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle \quad \text{s.t.} \quad \sum_{i=1}^n \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C, \quad i \in [n]$$

Lecture 6

Coordinate-wise Minimizer
Definition (Coordinate-wise minimizer)
For any $f : \mathbb{R}^n \to (-\infty, +\infty]$, we say $\bar{x}$ is a coordinate-wise minimizer of f if $\bar{x} \in \text{dom} f$ and

$$f(\bar{x} + de_i) \geq f(\bar{x}) \quad \forall i \in [n],\ d \in \mathbb{R},$$

where $e_i \in \mathbb{R}^n$ is the i-th standard basis vector.

Examples:
- When $n = 2$:
$$f(\bar{x}_1 + d, \bar{x}_2) \geq f(\bar{x}_1, \bar{x}_2), \quad f(\bar{x}_1, \bar{x}_2 + d) \geq f(\bar{x}_1, \bar{x}_2) \quad \forall d \in \mathbb{R}$$
- When $n = 3$:
$$f(\bar{x}_1 + d, \bar{x}_2, \bar{x}_3) \geq f(\bar{x}_1, \bar{x}_2, \bar{x}_3), \quad \text{etc.}$$

(1) is equivalent to
$$\bar{x}_i \in \arg\min_{x_i} f(\bar{x}_1, \ldots, \bar{x}_{i-1}, x_i, \bar{x}_{i+1}, \ldots, \bar{x}_n) \quad \forall i \in [n].$$

Question: Is a coordinate-wise minimizer a global minimizer?

Coordinate-wise Minimizer: Differentiable
Claim: A coordinate-wise minimizer $\bar{x}$ of a convex function f is a global minimizer if f is differentiable at $\bar{x}$.
Proof: Since f is differentiable at $\bar{x}$,

$$\bar{x}_i \in \arg\min_{x_i} f(\bar{x}_1, \ldots, \bar{x}_{i-1}, x_i, \bar{x}_{i+1}, \ldots, \bar{x}_n)$$

implies
$$\nabla_i f(\bar{x}) = \frac{\partial}{\partial x_i} f(\bar{x}) = 0.$$

Thus, $\nabla f(\bar{x}) = (\nabla_1 f(\bar{x}), \ldots, \nabla_n f(\bar{x})) = 0$, so $\bar{x}$ is a global minimizer.

Question: Same question for non-differentiable f?

Coordinate-wise Minimizer: Non-Differentiable
Claim: A coordinate-wise minimizer $\bar{x}$ of a convex function f is **not necessarily** a global minimizer when f is not differentiable at $\bar{x}$.
Example:
$$f(x_1, x_2) = \begin{cases} (x_1 + 10)^2 + (x_2 - 10)^2, & \text{if } x_1 \geq x_2, \\ (x_1 - 10)^2 + (x_2 + 10)^2, & \text{if } x_1 < x_2 \end{cases}$$

$$= x_1^2 + x_2^2 + 20|x_1 - x_2| + 200$$

The global minimizer is at $(0, 0)$.

Coordinate-wise Minimizer: Separable Non-Differentiable
Claim: If the non-differentiable part is separable:

$$\min_{x \in \mathbb{R}^n} F(x) := f(x) + \sum_{i=1}^n r_i(x_i),$$

where f is convex differentiable and each r_i is closed proper convex, then a coordinate-wise minimizer of F is a global minimizer.
Proof:
- Define $r(x) := \sum_{i=1}^n r_i(x_i)$.
- Then
$$\bar{x}_i \in \arg\min_{x_i} f(\bar{x}_1, \ldots, x_i, \ldots, \bar{x}_n) + r_i(x_i) \quad \forall i$$

  implies
$$0 \in \nabla_i f(\bar{x}) + \partial r_i(\bar{x}_i) \quad \forall i \quad \Longleftrightarrow \quad 0 \in \nabla f(\bar{x}) + \partial r(\bar{x}).$$

Coordinate Descent Method
Target problem:

$$\min_{x\in\mathbb{R}^n} F(x) := f(x) + \sum_{i=1}^{n} r_i(x_i),$$

where f is convex differentiable, and each r_i is closed proper convex.
Why this problem?
- A coordinate descent method will search for a coordinate-wise minimizer.
- In general, a coordinate-wise minimizer is not necessarily a global minimizer.
- For this target problem, a coordinate-wise minimizer is indeed a global minimizer.

Algorithm (Coordinate Descent Method)
Choose $x^{(0)} \in \mathrm{dom}(F)$, set $k \leftarrow 0$.
Repeat until convergence:

$$x_1^{(k+1)} \leftarrow \arg\min_{x_1} f(x_1, x_2^{(k)}, \ldots, x_n^{(k)}) + r_1(x_1)$$

$$x_2^{(k+1)} \leftarrow \arg\min_{x_2} f(x_1^{(k+1)}, x_2, x_3^{(k)}, \ldots, x_n^{(k)}) + r_2(x_2)$$

$$\vdots$$

$$x_n^{(k+1)} \leftarrow \arg\min_{x_n} f(x_1^{(k+1)}, \ldots, x_{n-1}^{(k+1)}, x_n) + r_n(x_n)$$

Update $k \leftarrow k+1$.
End (repeat)

Remarks
- $x^{(k)}$ has a subsequence converging to a global minimizer x^*.
- Coordinates can be updated in any order (commonly cyclic).
- After updating $x_i^{(k+1)}$, use the new value immediately (no parallelization).
- Block coordinate descent generalizes this to groups of variables.
- No global convergence guarantee for non-convex F.

Lecture 7

Dimension Reduction
- Dimension reduction aims to represent each data point as a linear combination of a small number of basis vectors.
- Given n data points $a_1, a_2, \ldots, a_n \in \mathbb{R}^m$, dimension reduction looks for basis vectors

$$u_1, u_2, \ldots, u_r \in \mathbb{R}^m$$

such that each data point is well-approximated by a linear combination of the basis vectors:

$$a_j \approx \sum_{i=1}^{r} u_i v_{ji} \quad \text{or} \quad a_j \approx u_1 v_{j1} + u_2 v_{j2} + \cdots + u_r v_{jr}, \quad j \in [n]$$

where $v_{ji} \in \mathbb{R}$.

Dimension reduction is equivalently written in matrix form:

$$[a_1, a_2, \ldots, a_n] \approx [u_1, u_2, \ldots, u_r] \begin{bmatrix} v_{11} & \cdots & v_{n1} \\ \vdots & \ddots & \vdots \\ v_{1r} & \cdots & v_{nr} \end{bmatrix}$$

where

$$A \in \mathbb{R}^{m\times n}, \quad U \in \mathbb{R}^{m\times r}, \quad V^T \in \mathbb{R}^{r\times n}.$$

- Each column of A is a **data point**.
- Each column of U is a **basis vector**.
- Each column of V^T contains the **coordinates** in the basis U.
- Typically, $r \ll \min(m, n)$.

Variants of Dimension Reduction
Dimension reduction mainly differs in:
1. **Error measure** can vary:
   - NMF uses $\|A - UV^T\|^2 = \|A - UV^T\|_F^2$.
   - Alternative: ℓ_1 norm $\|A - UV^T\|_1$.
2. **Different constraints** on U and V:
   - NMF constraint: $U \geq 0, V \geq 0$ (nonnegative matrix factorization).
   - Orthogonal constraint: $V^T V = I_r$.
   - Symmetric constraint: $V = U$, then $A \approx UU^T$, with $m = n$.

NMF (Non-negative Matrix Factorization)
Given nonnegative matrix $A \in \mathbb{R}_+^{m\times n}$ and rank r, NMF finds nonnegative matrices $U \in \mathbb{R}_+^{m\times r}$, $V \in \mathbb{R}_+^{n\times r}$ solving:

$$\min_{U,V} \quad \frac{1}{2}\|A - UV^T\|_F^2 \quad \text{s.t.} \quad U \geq 0,\ V \geq 0$$

Notes:
- The objective is non-convex w.r.t (U, V) but bi-convex.
- NMF is a popular dimension reduction technique.

Variants of NMF
- Standard NMF can vary by error measure, constraints, or regularization.
- Different variants are suited for different applications.

Model	Objective Function	Constraints
NMF	$\frac{1}{2}\|A - UV^T\|_F^2$	$U \geq 0, V \geq 0$
Symmetric NMF	$\frac{1}{2}\|A - UV^T\|_F^2$	$U \geq 0, V \geq 0, \mathbf{V{=}U}$

Algorithms for NMF
Focus: BCD-type (block coordinate descent) methods.
- BCD: U and V are divided into blocks (columns, entries, etc.), updated successively.
- HALS (Hierarchical Alternating Least Squares) updates one column of U or V at a time.

Algorithm 8 HALS-1 Algorithm

1: **for** $j = 1$ to J **do**
2: $\quad v_j \leftarrow \arg\min_{v\geq 0} f(v_1, \ldots, v_{j-1}, v, v_{j+1}, \ldots, v_J, u_1, \ldots, u_J)$
3: $\quad u_j \leftarrow \arg\min_{u\geq 0} f(v_1, \ldots, v_J, u_1, \ldots, u_{j-1}, u, u_{j+1}, \ldots, u_J)$
4: **end for**

Algorithm 9 HALS-2 Algorithm

1: **for** $j = 1$ to J **do**
2: $\quad v_j \leftarrow \arg\min_{v\geq 0} f(v_1, \ldots, v_{j-1}, v, v_{j+1}, \ldots, v_J, u_1, \ldots, u_J)$
3: **end for**
4: **for** $j = 1$ to J **do**
5: $\quad u_j \leftarrow \arg\min_{u\geq 0} f(v_1, \ldots, v_J, u_1, \ldots, u_{j-1}, u, u_{j+1}, \ldots, u_J)$
6: **end for**

Update Orders:

$$\text{HALS-1: } v_1 \to u_1 \to v_2 \to u_2 \to \cdots \to v_J \to u_J$$

$$\text{HALS-2: } v_1 \to \cdots \to v_J \to u_1 \to \cdots \to u_J$$

Lemma 1 (Solutions to subproblems of HALS methods)
Given a matrix $B \in \mathbb{R}^{M\times N}$ and a nonzero $v \in \mathbb{R}^N$, we have

$$\frac{[Bv]_+}{v^T v} = \arg\min_{u\geq 0} \|B - uv^T\|^2$$

Similarly, given a $B \in \mathbb{R}^{M\times N}$ and a nonzero $u \in \mathbb{R}^M$,

$$\frac{[B^T u]_+}{u^T u} = \arg\min_{v\geq 0} \|B - uv^T\|^2$$

The unique optimal solution is guaranteed due to $v \neq 0$ or $u \neq 0$.

Algorithm 10 HALS-1-Modified

1: Initialize nonnegative matrices U and V.
2: **repeat**
3: $\quad$ **for** $j = 1$ to J **do**
4: $\quad\quad v_j \leftarrow [(A - UV^T + u_j v_j^T)^T u_j]_+/(u_j^T u_j)$
5: $\quad\quad u_j \leftarrow [(A - UV^T + u_j v_j^T) v_j]_+/(v_j^T v_j)$
6: $\quad$ **end for**
7: **until** convergence criterion is reached

Algorithm 11 HALS-2-Modified

Initialize nonnegative matrices U and V.
repeat
  for $j = 1$ to J **do**
    $v_j \leftarrow [(A - UV^T + u_j v_j^T)^T u_j]_+ / (u_j^T u_j)$
  end for
  for $j = 1$ to J **do**
    $u_j \leftarrow [(A - UV^T + u_j v_j^T) v_j]_+ / (v_j^T v_j)$
  end for
until convergence criterion is reached

Theorem 2
If the columns of U^k and V^k remain nonzero throughout all the iterations of HALS-1 (or HALS-2), and the minimum of all subproblems is attained at each iteration, then every limit point of the sequence $\{(U^k, V^k)\}$ generated by HALS-1 (or HALS-2) is a stationary point of NMF.

ARkNLS: A Rank-k NLS based NMF Framework
Let

$$U = [U_1 \; \cdots \; U_q], \quad V = [V_1 \; \cdots \; V_q], \quad U_i \in \mathbb{R}^{m\times k}, \quad V_i \in \mathbb{R}^{n\times k}.$$

For simplicity, assume $r/k = q$ is an integer. Then

$$f(U, V) = \|A - UV^T\|_F^2 = \left\| U_1 V_1^T + \cdots + U_q V_q^T - A \right\|_F^2.$$

The optimization can be broken into solving:

$$V_i = \arg\min_{V\geq 0} \left\| U_i V^T - \left(A - \sum_{\ell\neq i} U_\ell V_\ell^T \right) \right\|_F^2, \quad i = 1, \ldots, q,$$

$$U_i = \arg\min_{U\geq 0} \left\| V_i^T U^T - \left(A - \sum_{\ell\neq i} U_\ell V_\ell^T \right)^T \right\|_F^2, \quad i = 1, \ldots, q.$$

Algorithm 12 ARkNLS: Alternating Rank-k Nonnegative Least Squares Framework for NMF

Require: Assume $A \in \mathbb{R}^{m\times n}$, $r \leq \min(m, n)$ given.
Initialize $U \in \mathbb{R}^{m\times r}$, $V \in \mathbb{R}^{n\times r}$ with $\{U, V\} \geq 0$.
Partition as in (2), where each block U_i, V_i has k columns.
Normalize the columns of U.
repeat
  for $i = 1, \ldots, q$ **do**
    Update V_i by solving rank-k NLS subproblem (3).
  end for
  for $i = 1, \ldots, q$ **do**
    Update U_i by solving rank-k NLS subproblem (4).
  end for
until a stopping criterion is satisfied

Convergence Property of ARkNLS
Theorem 3
If U_i^k and V_i^k, for $i = 1, \ldots, q$, are of full column rank throughout all iterations, and the unique minimizers in (3) and (4) are attained at each updating step, then every limit point of the sequence $\{(U, V)^{(k)}\}$ generated by ARkNLS algorithm is a stationary point of NMF.

ARkNLS is a general framework where $k > 0$ can be any integer. When $k = 1$, it reduces to HALS.

Subproblems (3) and (4)
The subproblems are NLS with multiple right-hand sides:

$$\min_{Y\geq 0} \|GY - B\|_F^2$$

where

- $G = U_i$, $B = A - \sum_{\ell\neq i} U_\ell V_\ell^T$ for (3),
- $G = V_i$, $B = (A - \sum_{\ell\neq i} U_\ell V_\ell^T)^T$ for (4).

Hence, solving (5) is the key problem in ARkNLS.

Recursive Formula for Rank-k NLS Problem
Problem (5) can be decoupled into independent NLS problems:

$$\min_{Y(:,j)\geq 0} \|GY(:, j) - B(:, j)\|_F^2$$

To solve (5) in closed-form, we first solve the rank-k NLS problem:

$$\min_{y\geq 0} \|Gy - b\|, \quad b \in \mathbb{R}^m, \quad G \in \mathbb{R}^{m\times k}, \quad \operatorname{rank}(G) = k.$$

Theorem 4
Assume $G \in \mathbb{R}^{m\times k}$, $g_{k+1} \in \mathbb{R}^m$, and $b \in \mathbb{R}^m$ are given, and $[G \; g_{k+1}]$ has full column rank. Denote the unique solution of rank-k NLS (7) as $s(G, b) \in \mathbb{R}^k$.
Then the unique solution to the rank-$(k + 1)$ NLS problem

$$\begin{bmatrix} y^\star \\ y_{k+1}^\star \end{bmatrix} = \arg\min_{y\geq 0, y_{k+1}\geq 0} \left\| \begin{bmatrix} G & g_{k+1} \end{bmatrix} \begin{bmatrix} y \\ y_{k+1} \end{bmatrix} - b \right\|$$

is given by

$$\begin{cases} y_{k+1}^\star = \dfrac{1}{\|g_{k+1}\|^2} [g_{k+1}^T (b - G \cdot s(G, b))]_+ \\ y^\star = s(G, b - g_{k+1} y_{k+1}^\star) \end{cases}$$

Lemma 5
Given a continuous convex function $f(z)$, and two nonempty closed convex sets $\mathcal{T}$ and $\mathcal{C}$ with $\mathcal{T} \cap \mathcal{C} \neq \emptyset$,

$$\tilde{z} = \arg\min_{z\in\mathcal{T}} f(z)$$

Assume $\tilde{z}$ is finite.
Then, solving

$$\min_{z\in\mathcal{T}\cap\mathcal{C}} f(z)$$

satisfies:

- If $\tilde{z} \in \mathcal{C}$, then $z^\star = \tilde{z}$.
- If $\tilde{z} \notin \mathcal{C}$, there exists $z^\star \in \mathcal{T} \cap C_{\text{edge}}$ with $f(z^\star) = f(\tilde{z})$.

Proof of Lemma 5
Assume $\hat{z} \in \mathcal{T} \cap \mathcal{C}$ is finite and

$$\hat{z} = \arg\min_{z\in\mathcal{T}\cap\mathcal{C}} f(z)$$

Then

$$z^\star = (1 - t)\tilde{z} + t\hat{z} \in \mathcal{T} \cap C_{\text{edge}}$$

for some $t \in (0, 1)$.
Since f is convex,

$$f(z^\star) = f(\tilde{z}) = f(\hat{z})$$

thus $z^\star$ solves the problem.

Lemma 6
Assume $G \in \mathbb{R}^{m\times k}$, $\operatorname{rank}(G) = k$, and $b \in \mathbb{R}^m$.
Then the solution to the rank-k NLS problem (7) is unique.
Theorem 4 can be used to recursively derive solutions for any $k \geq 1$.

Corollary 7
Assume

$$G = [g_1 \quad g_2 \quad g_3] \in \mathbb{R}^{m\times 3}, \quad \operatorname{rank}(G) = 3.$$

The unique solution to the rank-3 NLS problem

$$\begin{bmatrix} y_1^\star \\ y_2^\star \\ y_3^\star \end{bmatrix} = \arg\min_{y\geq 0} \|Gy - b\|$$

is given by

$$\begin{cases} y_3^\star = \dfrac{1}{\|g_3\|^2} [g_3^T (b - Gp)]_+ \\ y_2^\star = \dfrac{1}{\|g_2\|^2} [g_2^T (b - g_3 y_3^\star - Gw)]_+ \\ y_1^\star = \dfrac{1}{\|g_1\|^2} [g_1^T (b - g_2 y_2^\star - g_3 y_3^\star)]_+ \end{cases}$$

where

$$w = -g_2^T\, g_1 \left(\frac{b^T g_1 \|g_2\|^2 - b^T g_2\, g_1^T g_2}{\|g_1\|^2 \|g_2\|^2 - (g_1^T g_2)^2} \right)$$

$$p = \left(\frac{b^T g_2 \|g_1\|^2 - b^T g_1\, g_1^T g_2}{\|g_1\|^2 \|g_2\|^2 - (g_1^T g_2)^2} \right)$$

and

$$\tilde{p} = \left(\frac{b^T g_1 \|g_3\|^2 - b^T g_3\, g_1^T g_3}{\|g_1\|^2 \|g_3\|^2 - (g_1^T g_3)^2} \right) p$$