Principles of Machine Learning

Lecture 3: Calculus Concepts and Optimization Basics

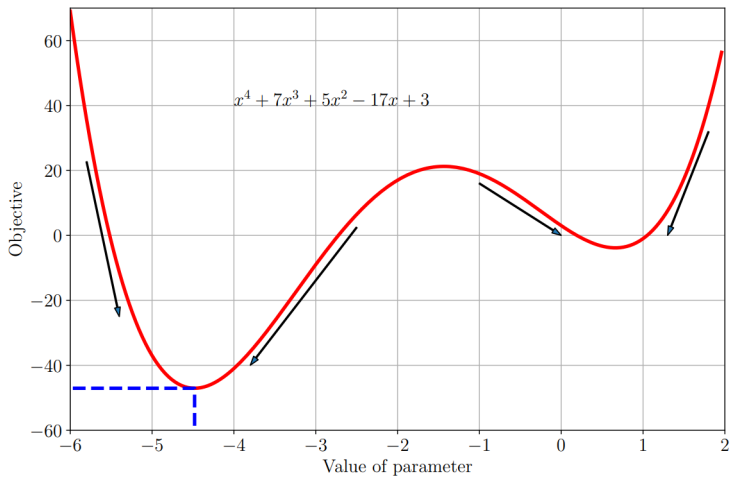
Sharif University of Technology
Dept. of Aerospace Engineering

March 9, 2025

Table of Contents

Continuous Optimization

- Training a machine learning model boils down to finding a good set of parameters.
- Given an objective function, finding the best value is done using optimization algorithms.
- Finding the best value means moving downhill, opposite to the gradient, to reach the deepest point of the objective function.
- In general, the objective function can have multiple local minima but only one global minimum. For convex functions, any local minimum is also the global minimum.



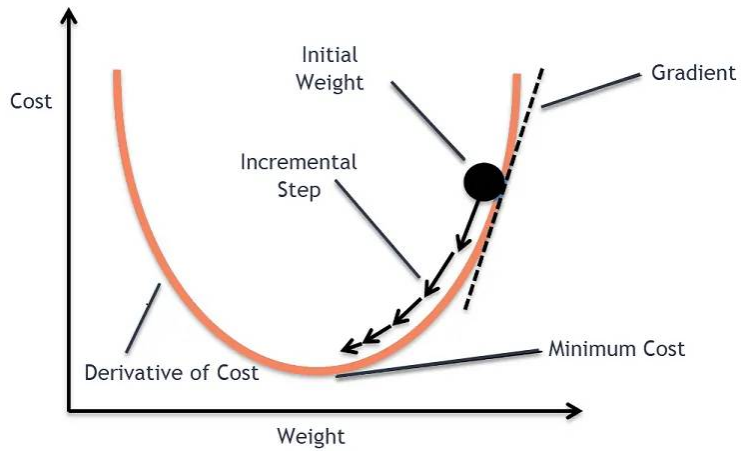
Optimization Using Gradient Descent

We consider the problem of solving for the minimum of a real-valued function

$$\min_{\mathbf{x}} f(\mathbf{x})$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is an objective function. We assume that our function f is differentiable.

- Gradient descent is a first-order optimization algorithm.
- Gradient descent: move towards the error (local) minimum.
- Compute gradient, which implies getting direction to the cost minimum.
- Take steps proportional to the negative of the gradient of the function at the current point.



Optimization Using Gradient Descent

- Gradient descent exploits the fact that $f(\mathbf{x}_0)$ decreases fastest if one moves from $\mathbf{x}_0$ in the direction of the negative gradient $-((\nabla f)(\mathbf{x}_0))^T$ of f at $\mathbf{x}_0$.
- The gradient descent iterative rule is

$$\mathbf{x}_{i+1} = \mathbf{x}_i - \gamma_i ((\nabla f)(\mathbf{x}_i))^T$$

- We start with an initial guess $\mathbf{x}_0$.
- For a suitable step-size (learning rate) γ_i, this algorithm $(f(\mathbf{x}_0) \geq f(\mathbf{x}_1) \geq \dots)$ converges to a local minimum.
- Gradient descent can be relatively slow close to the minimum: Its asymptotic rate of convergence is inferior to many other methods.

Optimization Using Gradient Descent

Examples (Gradient Descent)

Consider a quadratic function of $\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T$

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T \begin{bmatrix} 2 & 1 \\ 1 & 20 \end{bmatrix} \mathbf{x} - \begin{bmatrix} 5 & 3 \end{bmatrix} \mathbf{x}$$

The gradient is

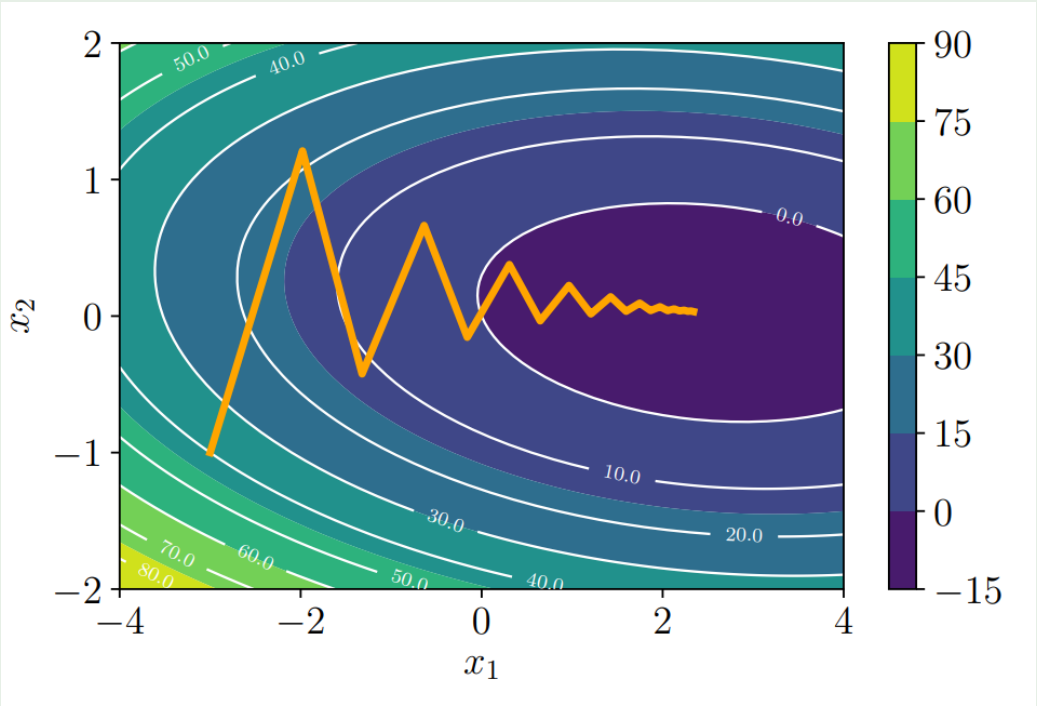
$$\nabla f(\mathbf{x}) = \mathbf{x}^T \begin{bmatrix} 2 & 1 \\ 1 & 20 \end{bmatrix} - \begin{bmatrix} 5 & 3 \end{bmatrix} = \begin{bmatrix} 2x_1 + x_2 - 5 & x_1 + 20x_2 - 3 \end{bmatrix}$$

Starting at the initial location $\mathbf{x}_0 = \begin{bmatrix} -3 & -1 \end{bmatrix}^T$ we iteratively apply the gradient descent rule to obtain a sequence of estimates that converge to the minimum value.

Optimization Using Gradient Descent

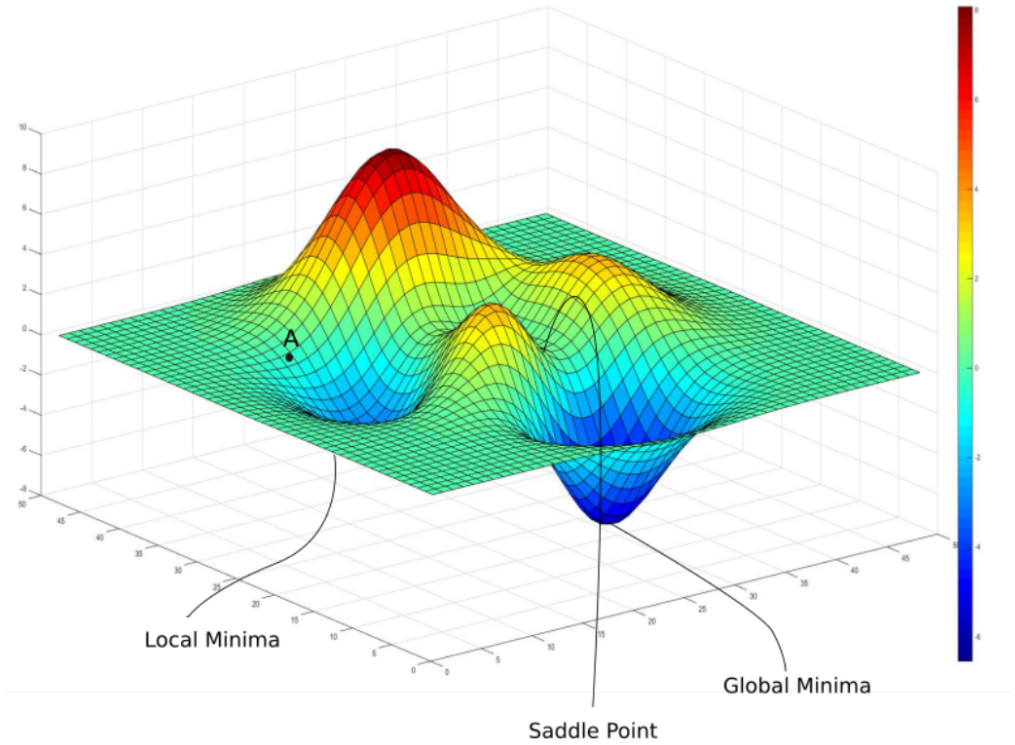
Examples (Contd.)

The gradient points in a direction that is orthogonal to the contour lines of the function we wish to optimize.



Challenges with Gradient Descent

- **Local minimum**: A local minimum is a minimum within some neighborhood that need not be (but may be) a global minimum.
- **Saddle points**: For non-convex functions, having the gradient to be 0 is not good enough. Example: $f(\mathbf{x}) = x_1^2 - x_2^2$ at $\mathbf{x} = (0, 0)$ has zero gradient but it is clearly not a local minimum as $\mathbf{x} = (0, \epsilon)$ has smaller function value.



Challenges with Gradient Descent

- If the step-size is too small, gradient descent can be slow.
- If the step-size is chosen too large, gradient descent can overshoot, fail to converge, or even diverge.

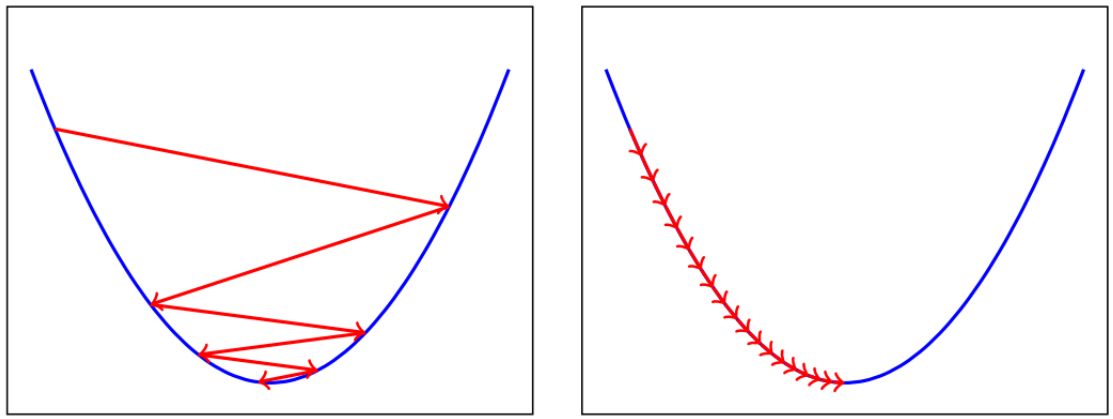


Figure: Effect of step size. Left: Big step size, Right: Small step size

Therefore, an adaptive mechanism should be employed to adjust the step size at each iteration based on changes in the function value.

Gradient Descent With Momentum

- Gradient descent with momentum incorporates a memory term to retain past updates.
- This memory dampens oscillations and smoothes out the gradient updates.
- The idea is to have a gradient update with memory to implement a moving average.
- The momentum term is useful since it averages out different noisy estimates of the gradient.

The momentum-based method update rule is:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \gamma_k \left(\left(\nabla f \right) \left(\mathbf{x}_k \right) \right)^T + \alpha \Delta \mathbf{x}_k$$

$$\Delta \mathbf{x}_k = \mathbf{x}_k - \mathbf{x}_{k-1} = \alpha \Delta \mathbf{x}_{k-1} - \gamma_{k-1} \left(\left(\nabla f \right) \left(\mathbf{x}_{k-1} \right)^T \right)$$

where $\alpha \in [0, 1]$.

Gradient Descent with Momentum

- Update rule with velocity vector (Alternative formula):
-

$$\mathbf{v}_{k+1} = \beta \mathbf{v}_k + (1 - \beta) \nabla f(\mathbf{x}_k)$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \mathbf{v}_{k+1}$$

- Parameters:
  - β: Momentum coefficient (0.5–0.9)
  - α: Learning rate
- Benefits: Accelerates convergence in valleys and dampens oscillations.

Nesterov's Accelerated Gradient (NAG)

- Key idea: Compute gradient at a *look-ahead* position:

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

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{v}_{k+1}$$

- Difference from momentum: Gradient evaluated at $\mathbf{x}_k - \beta \mathbf{v}_k$, not $\mathbf{x}_k$.
- Advantages: Faster convergence for convex functions, avoids overshooting.

Adagrad

- Adapts learning rate per parameter using historical gradients:

$$x_{k+1,i} = x_{k,i} - \frac{\alpha}{\sqrt{G_{k,i} + \epsilon}} \nabla f(x_{k,i})$$

- $G_{k,i}$: Sum of squared gradients for parameter i.
- Example: Minimize $f(\theta) = \theta^2$:

$$\theta_{k+1} = \theta_k - \frac{\alpha}{\sqrt{\sum g_i^2 + \epsilon}} g_k$$

RMSprop

- Addresses Adagrad's diminishing learning rate:

$$G_k = \beta G_{k-1} + (1-\beta)(\nabla f(x_k))^2$$

$$x_{k+1} = x_k - \frac{\alpha}{\sqrt{G_k + \epsilon}} \nabla f(x_k)$$

- Uses exponentially decaying average of squared gradients.

Adam

- Combines momentum and adaptive learning rates:

$$m_k = \beta_1 m_{k-1} + (1-\beta_1)\nabla f(x_k)$$
$$v_k = \beta_2 v_{k-1} + (1-\beta_2)(\nabla f(x_k))^2$$
$$\hat{m}_k = \frac{m_k}{1-\beta_1^k}, \quad \hat{v}_k = \frac{v_k}{1-\beta_2^k}$$
$$x_{k+1} = x_k - \frac{\alpha \hat{m}_k}{\sqrt{\hat{v}_k} + \epsilon}$$

- Includes bias correction for moments.

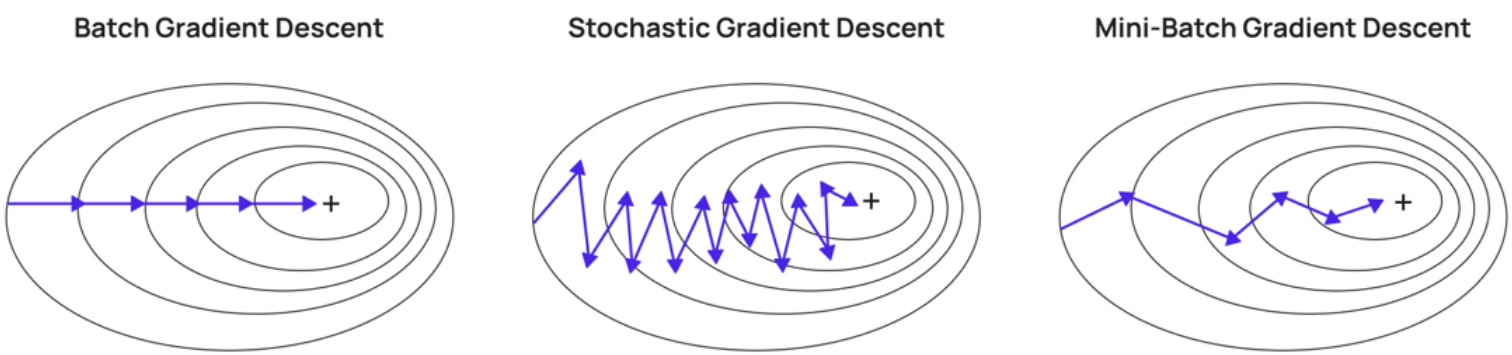
Gradient Descent Variations

- **Batch Gradient Descent**: Computes the gradient using the entire dataset before updating parameters and takes one step per epoch.
- **Stochastic Gradient Descent**: Updates parameters after computing the gradient for each individual data point and takes many steps per epoch. *Stochastic* here refers to the fact that we acknowledge that we do not know the gradient precisely, but instead only know a noisy approximation to it.
- **Mini-Batch Gradient Descent**: Uses a small random subset (mini-batch) of the dataset to compute the gradient and update parameters. Strikes a balance between Batch and Stochastic GD.

Gradient Descent Variations

Table: Comparison of Gradient Descent Algorithms

Algorithm	**Update Frequency**	**Computation Cost**	**Convergence Speed**	**Stability**	**Suitable for Large Datasets**
Batch GD	Once per epoch	High	Slow	High	No (Expensive)
SGD	Every sample	Low	Fast	Low (Noisy)	Yes
Mini-Batch GD	Every mini-batch	Moderate	Moderate	Moderate	Yes



Equality Constraints: Lagrange Multipliers

Consider a function $f(\mathbf{x})$ that needs to be minimized or maximized subject to equality constraints $h_i(\mathbf{x}) = 0$. The method of Lagrange multipliers involves constructing a new function called the Lagrangian:

- Lagrangian function:

$$\mathcal{L}(\mathbf{x}, \lambda) = f(\mathbf{x}) + \sum \lambda_i h_i(\mathbf{x})$$

- Example: Minimize $f(x, y) = x^2 + y^2$ s.t. $x + y = 1$:
  - Solution: $x = y = \frac{1}{2}$, $f_{\min} = \frac{1}{2}$

$L(x, y, \lambda) = x^2 + y^2 + \lambda(x + y - 1)$

$\frac{\partial L}{\partial \lambda} = 0 = x + y - 1$

$\frac{\partial L}{\partial x} = 2x + \lambda = 0$

$\frac{\partial L}{\partial y} = 2y + \lambda = 0$

$\hookrightarrow 2x = 1$

$\rightarrow x = 1/2$

Inequality Constraints & KKT Conditions

For optimization problems with inequality constraints $g_j(\mathbf{x}) \leq 0$, the Karush-Kuhn-Tucker (KKT) conditions extend the method of Lagrange multipliers. The Lagrangian for inequality constraints is:

$$\mathcal{L}(\mathbf{x}, \lambda, \mu) = f(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i h_i(\mathbf{x}) + \sum_{j=1}^{p} \mu_j g_j(\mathbf{x})$$

- KKT Conditions:
  - **Stationarity:** $\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \lambda, \mu) = 0$
  - **Primal Feasibility:** $h_i(\mathbf{x}) = 0$ and $g_j(\mathbf{x}) \leq 0$
  - **Dual Feasibility:** $\mu_j \geq 0$
  - **Complementary Slackness:** $\mu_j g_j(\mathbf{x}) = 0$
- Example: Minimize $f(x, y) = x^2 + y^2$ s.t. $x + y = 1$, $x \leq y$:
  - Solution: Same as equality case, $\mu \geq 0$

Constrained Optimization and Lagrange Multipliers

We consider the constrained optimization problem

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & g_i(\mathbf{x}) \leq 0 \quad \text{for all} \quad i = 1, \ldots, m. \end{aligned}$$

We associate the *Lagrangian* to the above problem by introducing the *Lagrange multipliers* $\lambda_i \geq 0$ as

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i g_i(\mathbf{x}) = f(\mathbf{x}) + \boldsymbol{\lambda}^T \mathbf{g}(\mathbf{x})$$

Definition (Lagrangian dual problem)

The associated Lagrangian dual problem to the above primal problem is

$$\begin{aligned} \max_{\boldsymbol{\lambda} \in \mathbb{R}^m} \quad & \mathcal{D}(\boldsymbol{\lambda}) \\ \text{subject to} \quad & \boldsymbol{\lambda} \succeq \mathbf{0} \end{aligned}$$

where $\boldsymbol{\lambda}$ are the dual variables and $\mathcal{D}(\boldsymbol{\lambda}) = \min_{\mathbf{x} \in \mathbb{R}^d} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})$.

Constrained Optimization and Lagrange Multipliers

Definition (minimax inequality)

minimax inequality says that for any function with two arguments $\phi(\mathbf{x}, \mathbf{y})$, the maxmin is less than the minmax:

$$\max_{\mathbf{y}} \min_{\mathbf{x}} \phi(\mathbf{x}, \mathbf{y}) \leq \min_{\mathbf{x}} \max_{\mathbf{y}} \phi(\mathbf{x}, \mathbf{y})$$

Definition (weak duality)

weak duality says that Swapping the order of the minimum and maximum results in a smaller value. Thus, primal values ($\mathbf{x}$) are always greater than or equal to dual values.

$$\min_{\mathbf{x} \in \mathbb{R}^d} \max_{\boldsymbol{\lambda} \succeq \mathbf{0}} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) \geq \max_{\boldsymbol{\lambda} \succeq \mathbf{0}} \min_{\mathbf{x} \in \mathbb{R}^d} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}).$$

Remark: We can model equality constraints by replacing them with two inequality constraints. That is for each equality constraint $h_j(\mathbf{x}) = 0$ we equivalently replace it by two constraints $h_j(\mathbf{x}) \leq 0$ and $h_j(\mathbf{x}) \geq 0$.

Convex Optimization

- When $f(\cdot)$ is a convex function, and when the constraints involving are convex sets, this is called a *convex optimization problem*.
- In the convex optimization setting, the optimal solution of the dual problem is the same as the optimal solution of the primal problem.
- Convex sets are sets such that a straight line connecting any two elements of the set lie inside the set.

Definition (Convex set)

A set $\mathcal{C}$ is a *convex set* if for any $x, y \in \mathcal{C}$ and for any scalar θ with $0 \leq \theta \leq 1$, we have

$$\theta x + (1 - \theta) y \in \mathcal{C}.$$

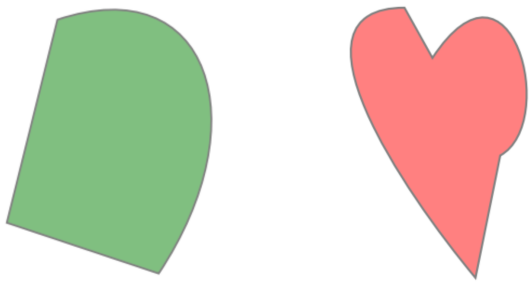
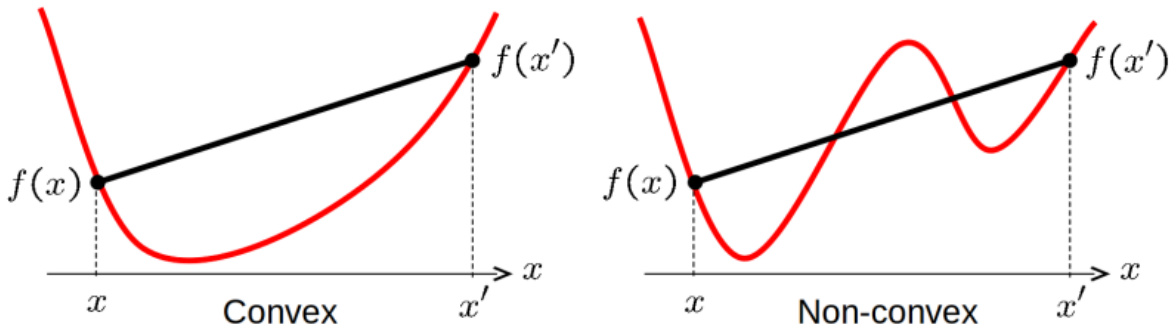


Figure: Left: Example of a convex set, Right: Example of a nonconvex set.

Definition (Convex function)

Let function $f : \mathbb{R}^D \to \mathbb{R}$ be a function whose domain is a convex set. The function f is a *convex function* if for all $\mathbf{x}, \mathbf{y}$ in the domain of f, and for any scalar θ with $0 \leq \theta \leq 1$, we have

$$f(\theta\mathbf{x} + (1-\theta)\mathbf{y}) \leq \theta f(\mathbf{x}) + (1-\theta)f(\mathbf{y}).$$



Remark: A concave function is the negative of a convex function.

Convex Optimization

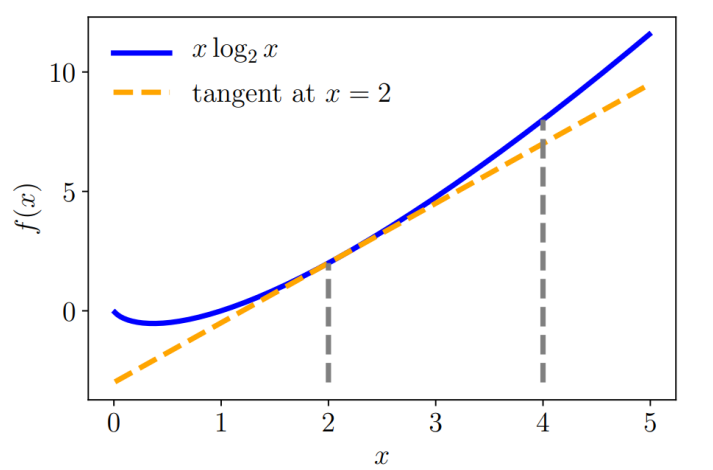
- If a function $f : \mathbb{R}^n \to \mathbb{R}$ is differentiable, we can specify convexity in terms of its gradient $\nabla_{\mathbf{x}} f(\mathbf{x})$.
- A function $f(\mathbf{x})$ is convex if and only if for any two points $\mathbf{x}, \mathbf{y}$ it holds that

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla_{\mathbf{x}} f(\mathbf{x})^T (\mathbf{y} - \mathbf{x}).$$

- If a function $f(\mathbf{x})$ is twice differentiable, that is, the Hessian exists for all values in the domain of $\mathbf{x}$, then the function $f(\mathbf{x})$ is convex if and only if $\nabla^2_{\mathbf{x}} f(\mathbf{x})$ is positive semi-definite.
- A nonnegative weighted sum of convex functions is convex.

Examples

The negative entropy $f(x) = x \log_2 x$ is convex for $x > 0$. It can be seen that the function is convex.



We check the calculations for two points $x = 2$ and $x = 4$. Note that to prove convexity of $f(x)$ we would need to check for all points $x \in \mathbb{R}$.
Consider a point midway between the two points (that is $\theta = 0.5$). Then the left-hand side is $f(0.5 \times 2 + 0.5 \times 4) = \log_2 3 \approx 4.75$.
The right-hand side is $0.5 \times (2 \log_2 2) + 0.5 \times (4 \log_2 4) = 5$. And therefore the definition is satisfied.

Examples (Contd.)

Using the gradient method, we have:

$$\nabla_x(x\log_2 x) = \log_2 x + x\left(\frac{1}{x\log_e 2}\right) = \log_2 x + \left(\frac{1}{\log_e 2}\right)$$

Using the same tow test points $x = 2$ and $x = 4$, the left-hand side of the relation is given by $f(4) = 8$. The right-hand side is

$$\begin{aligned} f(\mathbf{x}) + \nabla_{\mathbf{x}}^T(\mathbf{y} - \mathbf{x}) &= f(2) + \nabla f(2) \times (4 - 2) \\ &= 2 + \left(1 + \frac{1}{\log_e 2}\right) \times 2 \approx 6.9. \end{aligned}$$

Convex Optimization

- If f is a convex function, and $\alpha > 0$ is a nonnegative scalar, then the function αf is convex.
- If f_1 and f_2 are convex functions, then we have by the definition

$$f_1(\theta \mathbf{x} + (1-\theta)\mathbf{y}) \leq \theta f_1(\mathbf{x}) + (1-\theta) f_1(\mathbf{y})$$
$$f_2(\theta \mathbf{x} + (1-\theta)\mathbf{y}) \leq \theta f_2(\mathbf{x}) + (1-\theta) f_2(\mathbf{y})$$

Summing up both sides gives us

$$f_1(\theta \mathbf{x} + (1-\theta)\mathbf{y}) + f_2(\theta \mathbf{x} + (1-\theta)\mathbf{y})$$
$$\leq \theta f_1(\mathbf{x}) + (1-\theta) f_1(\mathbf{y}) + \theta f_2(\mathbf{x}) + (1-\theta) f_2(\mathbf{y})$$

where the right-hand side can be rearranged to

$$\theta(f_1(\mathbf{x}) + f_2(\mathbf{x})) + (1-\theta)(f_1(\mathbf{y}) + f_2(\mathbf{y})),$$

completing the proof that the sum of convex functions is convex. Combining the preceding two facts, we see that $\alpha f_1(\mathbf{x}) + \beta f_2(\mathbf{x})$ is convex for $\alpha, \beta > 0$. This closure property can be extended for nonnegative weighted sums of more than two convex functions.

Convex Optimization

Definition

A constrained optimization problem is called a *convex optimization problem* if

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & g_i(\mathbf{x}) \leq 0 \quad \text{for all} \quad i = 1, \ldots, m \\ & h_j(\mathbf{x}) = 0 \quad \text{for all} \quad j = 1, \ldots, n, \end{aligned}$$

where all functions $f(\mathbf{x})$ and $g_i(\mathbf{x})$ are convex functions, and all $h_j(\mathbf{x}) = 0$ are convex sets.

Convex Optimization

- Consider the special case when all the preceding functions are linear, i.e. with $\mathbf{A} \in \mathbb{R}^{m \times d}$ and $\mathbf{b} \in \mathbb{R}^m$, we have

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^d} \quad & \mathbf{c}^T \mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \preceq \mathbf{b}, \end{aligned}$$

- This is known as a *linear program*.
- It has d variables and m linear constraints.
- The Lagrangian is given by

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = \mathbf{c}^T \mathbf{x} + \boldsymbol{\lambda}^T (\mathbf{A}\mathbf{x} - \mathbf{b}) = \left(\mathbf{c} + \mathbf{A}^T \boldsymbol{\lambda}\right)^T \mathbf{x} - \boldsymbol{\lambda}^T \mathbf{b}.$$

Taking the derivative of $\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})$ wrt $\mathbf{x}$ and setting it to zero gives us

$$\mathbf{c} + \mathbf{A}^T \boldsymbol{\lambda} = \mathbf{0}.$$

Therefore, the dual Lagrangian is $\mathcal{D}(\boldsymbol{\lambda}) = -\boldsymbol{\lambda}^T \mathbf{b}$ which we would like to maximize.

Convex Optimization

- In this case, the dual optimization problem is

$$\begin{aligned} \max_{\boldsymbol{\lambda} \in \mathbb{R}^m} \quad & -\mathbf{b}^T \boldsymbol{\lambda} \\ \text{subject to} \quad & \mathbf{c} + \mathbf{A}^T \boldsymbol{\lambda} = \mathbf{0} \\ & \boldsymbol{\lambda} \succeq \mathbf{0}. \end{aligned}$$

- This is also a linear program with m variables.
- We have the choice of solving the primal or the dual program depending on whether m or d is larger.

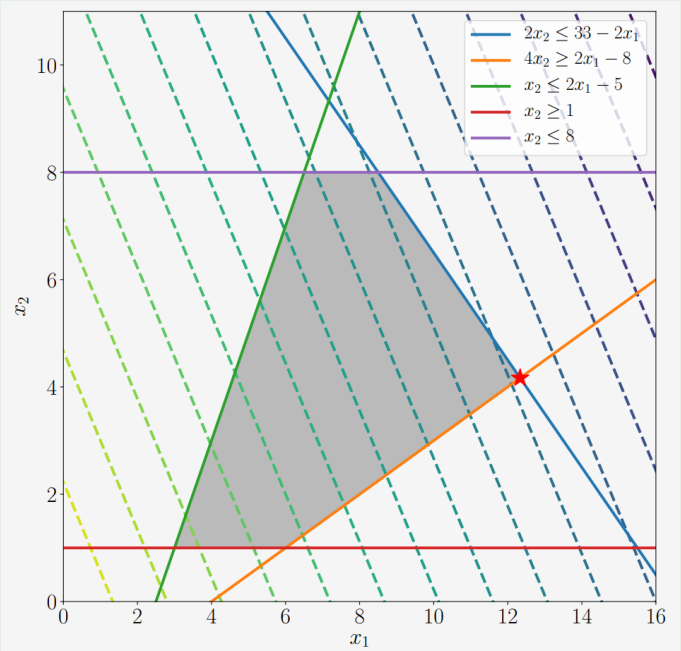
Examples (Linear Program)

Consider the linear program

$$\min_{\mathbf{x}\in\mathbb{R}^2} \quad -\begin{bmatrix}5 & 3\end{bmatrix}\mathbf{x}$$

$$\text{subject to} \quad \begin{bmatrix}2 & 2 & -2 & 0 & 0\\ 2 & -4 & 1 & -1 & 1\end{bmatrix}^T \mathbf{x} \preceq \begin{bmatrix}33 & 8 & 5 & -1 & 8\end{bmatrix}^T$$

The optimal value must lie in the shaded (feasible) region, and is indicated by the star.



Convex Optimization

- Consider the case of a convex quadratic objective function, where the constraints are affine, i.e.,

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^d} \quad & \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \preceq \mathbf{b}, \end{aligned}$$

  where $\mathbf{A} \in \mathbb{R}^{m\times d}$, $\mathbf{b} \in \mathbb{R}^m$, and $\mathbf{c} \in \mathbb{R}^d$.
- The square matrix $\mathbf{Q} \in \mathbb{R}^{d\times d}$ is positive definite and therefore objective function is convex.
- This is known as a *quadratic program*.
- It has d variables and m linear constraints.

Convex Optimization

- The Lagrangian is given by

$$\begin{aligned}\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) &= \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{c}^T\mathbf{x} + \boldsymbol{\lambda}^T(\mathbf{A}\mathbf{x} - \mathbf{b}) \\ &= \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} + \left(\mathbf{c} + \mathbf{A}^T\boldsymbol{\lambda}\right)^T\mathbf{x} - \boldsymbol{\lambda}^T\mathbf{b}\end{aligned}$$

- Taking the derivative of $\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})$ wrt $\mathbf{x}$ and setting it to zero gives us

$$\mathbf{Q}\mathbf{x} + (\mathbf{c} + \mathbf{A}^T\boldsymbol{\lambda}) = \mathbf{0}.$$

- Assuming that $\mathbf{Q}$ is invertible, we get

$$\mathbf{x} = -\mathbf{Q}^{-1}(\mathbf{c} + \mathbf{A}^T\boldsymbol{\lambda}).$$

Convex Optimization

- A convex set can be equivalently described by its supporting hyperplane.
- A hyperplane is called a *supporting hyperplane* of a convex set if it intersects the convex set, and the convex set is contained on just one side of it.
- Convex functions can be equivalently described by a function of their gradient.
- The *Legendre-Fenchel transform* is a transformation (in the sense of a Fourier transform) from a convex differentiable function $f(\mathbf{x})$ to a function that depends on the tangents $s(\mathbf{x}) = \nabla_{\mathbf{x}} f(\mathbf{x})$.
- The Legendre-Fenchel transform is also known as the *convex conjugate* and is closely related to duality.

Convex Optimization

Definition (Convex Conjugate)

The *convex conjugate* of a function $f : \mathbb{R}^D \to \mathbb{R}$ is a function f^* defined by

$$f^*(\mathbf{s}) = \sup_{\mathbf{x} \in \mathbb{R}^D} \left(\langle \mathbf{s}, \mathbf{x} \rangle - f(\mathbf{x}) \right) = \sup_{\mathbf{x} \in \mathbb{R}^D} \left(\mathbf{s}^T \mathbf{x} - f(\mathbf{x}) \right)$$

The preceding convex conjugate definition does not need the function f to be convex nor differentiable.

Examples

To illustrate the application of convex conjugates, consider the quadratic function

$$f(\mathbf{y}) = \frac{\lambda}{2} \mathbf{y}^T \mathbf{K}^{-1} \mathbf{y}$$

based on a positive definite matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$. The primal variable is $\mathbf{y} \in \mathbb{R}^n$ and the dual variable is $\boldsymbol{\alpha} \in \mathbb{R}^n$.

Examples (Contd.)

Based on the definition of convex conjugate, we obtain:

$$f^*(\boldsymbol{\alpha}) = \sup_{\mathbf{y}\in\mathbb{R}^n} \langle \mathbf{y}, \boldsymbol{\alpha} \rangle - \frac{\lambda}{2}\mathbf{y}^T\mathbf{K}^{-1}\mathbf{y}.$$

We can find the maximum by taking the derivative wrt $\mathbf{y}$ and setting it zero:

$$\frac{\partial \left[\langle \mathbf{y}, \boldsymbol{\alpha} \rangle - \frac{\lambda}{2}\mathbf{y}^T\mathbf{K}^{-1}\mathbf{y}\right]}{\partial \mathbf{y}} = \left(\boldsymbol{\alpha} - \lambda\mathbf{K}^{-1}\mathbf{y}\right)^T$$

Thus, when the gradient is zero we have $\mathbf{y} = \frac{1}{\lambda}\mathbf{K}\boldsymbol{\alpha}$ and

$$f^*(\boldsymbol{\alpha}) = \frac{1}{\lambda}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \frac{\lambda}{2}\left(\frac{1}{\lambda}\mathbf{K}\boldsymbol{\alpha}\right)^T\mathbf{K}^{-1}\left(\frac{1}{\lambda}\mathbf{K}\boldsymbol{\alpha}\right) = \frac{1}{2\lambda}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha}.$$