UNIVERSITY COLLEGE LONDON

DEPARTMENT OF COMPUTER SCIENCE

COMP0083: ADVANCED TOPICS IN MACHINE LEARNING

Convex Optimization Coursework

Author:
Mr. Frederico Wieser

Student Number:
18018699

Last Updated:
January 10, 2025

Contents

1 Questions with Multiple Answers [30%]

1.1 Which One of These is a Convex Function?

1.1.1 Answer

The answer is **A**, $f(x) = \max\{ax + b, x^4 - 5, e^{-x}\}$

1.1.2 Reasoning

Given the following functions:

$$f(x) = \max\{ax + b, x^4 - 5, e^{-x}\},$$
$$g(x) = \min\{ax + b, x^4 - 5, e^{x}\},$$
$$h(x) = x^4 - 5 - e^{x},$$

we want to determine which is convex by examining their properties.

- $h(x) = x^4 - 5 - e^x$ is not convex because it is the sum of a convex term $x^4 - 5$ and a concave term $-e^x$, which does not preserve convexity.
- $g(x) = \min\{ax + b, x^4 - 5, e^x\}$ is not convex in general, as the minimum of convex functions is not necessarily convex.
- $f(x) = \max\{ax + b, x^4 - 5, e^{-x}\}$ is convex because the maximum of convex functions is convex.

Thus, by the process of elimination and convexity properties, $f(x)$ is the only convex function.

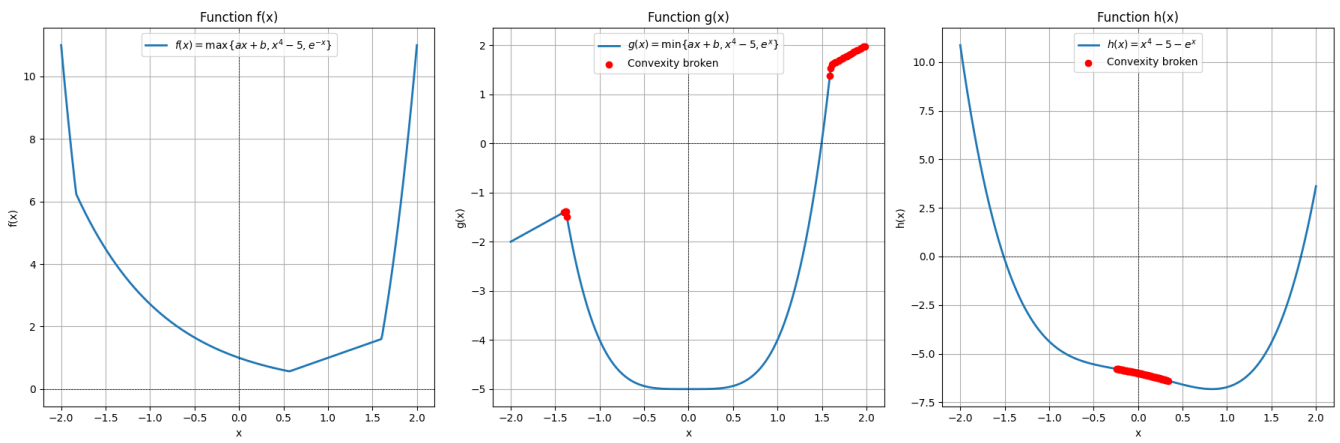

Figure 1: Python visualisation of the 3 functions h, g, and f, which shows numerically the points where they are breaking convexity according to the hessian being less than 0.

1.2 Identify the Subdifferential of the Given Function

1.2.1 Answer

The answer is **A**, as it correctly represents the graph of the subdifferential $\partial f(x)$.

1.2.2 Reasoning

Given:

$$f(x) = \begin{cases} -x, & x \in]-1, 0] \\ x^2, & x \geq 0 \end{cases}$$

We want to determine the subdifferential set $\partial f(x_0)$ at $x_0 = 0$, and so we use the definition:

$$\partial f(x_0) = \{\lambda \in \mathbb{R} \mid f(x) \geq f(x_0) + \lambda(x - x_0), \forall x \in \mathbb{R}\}.$$

Evaluating at $x_0 = 0$:

- For $x > 0$, we have $f(x) = x^2$ and $f(0) = 0$, leading to, $x^2 \geq \lambda x \quad \Rightarrow \quad \lambda \leq x$.
- For $x = 0$, the inequality trivially holds.
- For $x < 0$, we have $f(x) = -x$, and thus implying $-x \geq \lambda x$ which gives that $-1 \leq \lambda$.

Thus, combining the conditions, we conclude:

$$\lambda \in [-1, x].$$

At $x = 0$, this simplifies to:

$$\lambda \in [-1, 0].$$

Therefore, the subdifferential at $x_0 = 0$ is:

$$\partial f(0) = [-1, 0].$$

From the provided answer choices, option (A) correctly represents the graph of the subdifferential, as it clearly shows a vertical line from -1 to the origin which represents the set of values of the partial derivative we just derived.

1.3 Gradient of a Quadratic Function

1.3.1 Answer

The answer is **A**, $\nabla_x f(x) = (A + A^T)x + b$.

1.3.2 Reasoning

We are given the function:

$$f(x) = \langle Ax, x\rangle + \langle x, b\rangle + c.$$

Using the definition of the inner product: $\langle u, v\rangle = u^T v$ we can rewrite $f(x)$:

$$f(x) = x^T Ax + x^T b + c.$$

Since $x^T Ax$ is a quadratic form, we differentiate using the rule:

$$\nabla_x (x^T Ax) = (A + A^T)x.$$

The derivative of the linear term $x^T b$ is simply b, and the constant term c vanishes. Thus, the gradient is:

$$\nabla_x f(x) = (A + A^*)x + b.$$

1.4 Fenchel Conjugate of a Scaled Function

1.4.1 Answer

The answer is **A**, $f^*(u) = g^*\left(\frac{u}{3}\right)$.

1.4.2 Reasoning

Let $f(x) = g(3x)$ where $g \in \Gamma_0(\mathbb{R})$ (closed and convex). The Fenchel conjugate of $f(x)$ is defined by

$$f^*(u) = \sup_{x\in\mathbb{R}} \{ux - f(x)\}.$$

Substitute $f(x) = g(3x)$ into this expression:

$$f^*(u) = \sup_{x\in\mathbb{R}} \{ux - g(3x)\}.$$

Make the change of variable $y = 3x$. Then $x = \frac{y}{3}$ and $dy = 3\,dx$. Substituting y into the supremum gives

$$f^*(u) = \sup_{y\in\mathbb{R}} \left\{u\frac{y}{3} - g(y)\right\} = \sup_{y\in\mathbb{R}} \left\{\left(\frac{u}{3}\right)y - g(y)\right\}.$$

But by definition of the Fenchel conjugate g^* of g,

$$g^*(p) = \sup_{y\in\mathbb{R}} \{py - g(y)\}.$$

Hence, identifying $p = \frac{u}{3}$, we get

$$\sup_{y\in\mathbb{R}} \left\{\left(\frac{u}{3}\right)y - g(y)\right\} = g^*\left(\frac{u}{3}\right).$$

Therefore,

$$f^*(u) = g^*\left(\frac{u}{3}\right).$$

2 Theory on Convex Analysis and Optimization [40%]

2.1 Computing Fenchel Conjugates

2.1.1 $f(x) = \begin{cases} +\infty, & x \leq 0 \\ -\log x, & x > 0 \end{cases}$

We are given the function:

$$f : \mathbb{R} \to (-\infty, +\infty], \quad f(x) = \begin{cases} +\infty, & x \leq 0, \\ -\log x, & x > 0. \end{cases}$$

Where the Fenchel conjugate f^* is defined as:

$$f^*(u) = \sup_{x \in \mathbb{R}} \{ux - f(x)\}.$$

Since $f(x) = +\infty$ for $x \leq 0$, any $x \leq 0$ makes the term $ux - f(x) = ux - (+\infty) = -\infty$, which cannot maximize anything. Hence, the supremum is effectively only over $x > 0$, so:

$$f^*(u) = \sup_{x>0} \{ux - (-\log x)\} = \sup_{x>0} \{ux + \log x\}.$$

Consider the function we are trying to maximise over:

$$\phi(x) = ux + \log x, \quad x > 0.$$

Now we will look for critical points by differentiating:

$$\phi'(x) = u + \frac{1}{x}.$$

Setting $\phi'(x) = 0$ to find stationary points:

$$u + \frac{1}{x} = 0 \quad \Rightarrow \quad \frac{1}{x} = -u.$$

Since $x > 0$, we must have $-u > 0$, i.e., $u < 0$. In that case:

$$x = -\frac{1}{u}, \quad \text{(positive if } u < 0).$$

Next, $\phi''(x) = -1/x^2 < 0$, so ϕ is strictly concave in $x > 0$. Hence, if $u < 0$, the critical point at $x = -1/u$ is indeed the global maximizer in $(0, \infty)$. Plugging $x = -1/u$ into $\phi(x)$:

$$\phi\left(-\frac{1}{u}\right) = u\left(-\frac{1}{u}\right) + \log\left(-\frac{1}{u}\right) = -1 + \log\left(-\frac{1}{u}\right).$$

Rewriting the logarithm:

$$-1 + \log\left(-\frac{1}{u}\right) = -1 - \log(-u).$$

Thus, for $u < 0$, the supremum is finite and equals $-1 - \log(-u)$.

Now let us consider other values of u:

- **Case** $u > 0$.
  As $x \to +\infty$, we have $ux \to +\infty$ and $\log x \to +\infty$, so $\phi(x) \to +\infty$. Hence the supremum is $+\infty$.

- **Case** $u = 0$.
  Then $\phi(x) = \log x$, which again grows unbounded as $x \to +\infty$. So the supremum is also $+\infty$.

Hence, for all $u \geq 0$, $f^*(u) = +\infty$. Putting it all together, we get that the Fenchel conjugate of the given function is:

$$\boxed{f^*(u) = \begin{cases} -1 - \ln(-u), & u < 0, \\ +\infty, & u \geq 0. \end{cases}}$$

2.1.2 $f(x) = 2x^2$

The Fenchel conjugate of a function $f : \mathbb{R} \to \mathbb{R}$ is defined as:

$$f^*(u) = \sup_{x \in \mathbb{R}} \{ux - f(x)\}.$$

Substituting $f(x) = 2x^2$, we arrive at:

$$f^*(u) = \sup_{x \in \mathbb{R}} \{ux - 2x^2\}.$$

Define the function to be maximized:

$$\phi(x) = ux - 2x^2.$$

Differentiate with respect to x:

$$\phi'(x) = u - 4x.$$

Setting $\phi'(x) = 0$ to find the critical point,

$$u - 4x = 0 \quad \Rightarrow \quad x = \frac{u}{4}.$$

Substituting $x = \frac{u}{4}$ into $\phi(x)$:

$$f^*(u) = u \cdot \frac{u}{4} - 2\left(\frac{u}{4}\right)^2 = \frac{u^2}{4} - 2 \cdot \frac{u^2}{16} = \frac{u^2}{4} - \frac{u^2}{8} = \frac{u^2}{8}.$$

Thus, the Fenchel conjugate of $f(x) = 2x^2$ is:

$$\boxed{f^*(u) = \frac{u^2}{8}.}$$

2.1.3 $\iota_{[-1,1]}$ (zero on $[-1,1]$, $+\infty$ otherwise)

We have the function:

$$f(x) = \iota_{[-1,1]}(x) = \begin{cases} 0, & x \in [-1,1], \\ +\infty, & \text{otherwise.} \end{cases}$$

Its Fenchel conjugate f^* is given by

$$f^*(u) = \sup_{x \in \mathbb{R}} \{ux - f(x)\}.$$

Since $f(x) = +\infty$ outside $[-1,1]$, the supremum reduces to x in the interval $[-1,1]$. Within $[-1,1]$, $f(x) = 0$. Hence,

$$f^*(u) = \sup_{x \in [-1,1]} \{ux - 0\} = \sup_{x \in [-1,1]} (ux).$$

Now let's inspect the different values u can take on a case by case basis:

1. **Case** $u > 0$:
   The maximum of ux over $x \in [-1,1]$ occurs at $x = 1$, giving $u \cdot 1 = u$.
2. **Case** $u < 0$:
   The maximum of ux over $x \in [-1,1]$ occurs at $x = -1$, giving $u \cdot (-1) = -u$.
3. **Case** $u = 0$:
   Then $ux = 0$ for any x, and the supremum is 0.

In short, the supremum is $\max\{u, -u\} = |u|$. Therefore, the Fenchel conjugate is

$$\boxed{f^*(u) = |u|.}$$

2.2 Jensen's Inequality and Applications

2.2.1 Proving Jensen's Inequality by Induction

We proceed by induction on n, the number of points. For $n = 1$, we trivially have:

$$f(x_1) \leq f(x_1),$$

which is true, under the equality case. For $n = 2$, the inequality follows directly from the definition of convexity, which has to be true since f is a convex function:

$$f(\lambda_1 x_1 + \lambda_2 x_2) \leq \lambda_1 f(x_1) + \lambda_2 f(x_2),$$

where $\lambda_1 + \lambda_2 = 1$. This is precisely the convexity condition. Assuming that Jensen's inequality holds for some $n = k$:

$$f\left(\sum_{i=1}^{k} \lambda_i x_i\right) \leq \sum_{i=1}^{k} \lambda_i f(x_i),$$

for any convex function f, any $x_1, x_2, \ldots, x_k$, and any weights $\lambda_1, \lambda_2, \ldots, \lambda_k$ satisfying $\sum_{i=1}^{k} \lambda_i = 1$. Now, we must show that Jensen's inequality holds for $n = k+1$, in order to finish this proof by induction. That is, given $x_1, x_2, \ldots, x_k, x_{k+1}$ and weights $\lambda_1, \lambda_2, \ldots, \lambda_k, \lambda_{k+1}$ satisfying $\sum_{i=1}^{k+1} \lambda_i = 1$, we must prove:

$$f\left(\sum_{i=1}^{k+1} \lambda_i x_i\right) \leq \sum_{i=1}^{k+1} \lambda_i f(x_i).$$

In order to make the proof more readable, let:

$$t = \sum_{i=1}^{k} \lambda_i.$$

Since the total sum $\sum_{i=1}^{k+1} \lambda_i = 1$, we have $t = 1 - \lambda_{k+1}$. We now rewrite the convex combination explicitly as:

$$\sum_{i=1}^{k+1} \lambda_i x_i = t\left(\sum_{i=1}^{k} \frac{\lambda_i}{t} x_i\right) + \lambda_{k+1} x_{k+1}.$$

By convexity, we apply it to the two points $\sum_{i=1}^{k} \frac{\lambda_i}{t} x_i$ and x_{k+1}:

$$f\left(\sum_{i=1}^{k+1} \lambda_i x_i\right) = f\left(t \sum_{i=1}^{k} \frac{\lambda_i}{t} x_i + \lambda_{k+1} x_{k+1}\right).$$

Since f is convex, we get:

$$f\left(\sum_{i=1}^{k+1} \lambda_i x_i\right) \leq t f\left(\sum_{i=1}^{k} \frac{\lambda_i}{t} x_i\right) + \lambda_{k+1} f(x_{k+1}).$$

By the induction hypothesis, we apply Jensen's inequality to f at k points:

$$f\left(\sum_{i=1}^{k} \frac{\lambda_i}{t} x_i\right) \leq \sum_{i=1}^{k} \frac{\lambda_i}{t} f(x_i).$$

Multiplying both sides by t, we obtain:

$$t f\left(\sum_{i=1}^{k} \frac{\lambda_i}{t} x_i\right) \leq t \sum_{i=1}^{k} \frac{\lambda_i}{t} f(x_i) = \sum_{i=1}^{k} \lambda_i f(x_i).$$

Thus, we conclude:

$$\boxed{f\left(\sum_{i=1}^{k+1} \lambda_i x_i\right) \leq \sum_{i=1}^{k} \lambda_i f(x_i) + \lambda_{k+1} f(x_{k+1}) = \sum_{i=1}^{k+1} \lambda_i f(x_i).}$$

This is exactly Jensen's inequality for $k+1$ points, completing the induction.

2.2.2 Proving Convexity of the Negative Log Function

We wish to prove that the function $f(x) = -\log x$ is convex on its natural domain $(0, \infty)$, which is sufficient as the logarithm of negative numbers is undefined. To prove that $-\log x$ is convex on $(0, \infty)$, it suffices to show that its derivative is non-decreasing. We first compute the derivative of $f(x)$:

$$f'(x) = -\frac{1}{x}.$$

Next, we check whether $f'(x)$ is increasing on $(0, \infty)$. If $0 < a < b$, then

$$f'(a) = -\frac{1}{a}, \quad f'(b) = -\frac{1}{b}.$$

Since $a < b$ implies $\frac{1}{a} > \frac{1}{b}$, we obtain

$$-\frac{1}{a} < -\frac{1}{b},$$

which shows that $f'(x)$ is strictly increasing. Alternatively, we confirm this using the second derivative:

$$f''(x) = \frac{d}{dx}\left(-\frac{1}{x}\right) = \frac{1}{x^2} > 0, \quad \forall x > 0.$$

Since $f''(x) > 0$ on $(0, \infty)$, we conclude that $f'(x)$ is strictly increasing, and thus $f(x)$ is strictly convex.

2.2.3 Proving the Arithmetic-Geometric Mean Inequality

Applying Jensen's inequality, which states that for a convex function f:

$$f\left(\sum_{i=1}^{n} \lambda_i x_i\right) \leq \sum_{i=1}^{n} \lambda_i f(x_i),$$

where $\lambda_i \geq 0$ and $\sum_{i=1}^{n} \lambda_i = 1$, we set $f(x) = -\log x$ and obtain:

$$-\log\left(\sum_{i=1}^{n} \lambda_i x_i\right) \leq \sum_{i=1}^{n} \lambda_i (-\log x_i).$$

Rearranging gives:

$$\log\left(\sum_{i=1}^{n} \lambda_i x_i\right) \geq \sum_{i=1}^{n} \lambda_i \log x_i.$$

Choosing equal weights $\lambda_i = \frac{1}{n}$ for all $i = 1, \ldots, n$, since they sum to 1:

$$\log\left(\frac{1}{n}\sum_{i=1}^{n} x_i\right) \geq \frac{1}{n}\sum_{i=1}^{n} \log x_i.$$

Taking the exponential on both sides:

$$\frac{1}{n}\sum_{i=1}^{n} x_i \geq \exp\left(\frac{1}{n}\sum_{i=1}^{n} \log x_i\right).$$

Since the right-hand side is the geometric mean:

$$\frac{1}{n}\sum_{i=1}^{n} x_i \geq \sqrt[n]{x_1 x_2 \cdots x_n}.$$

Thus, we have proven the Arithmetic-Geometric Mean inequality:

$$\boxed{\sqrt[n]{x_1 \cdots x_n} \leq \frac{1}{n}\sum_{i=1}^{n} x_i,}$$

for all $x_1, \ldots, x_n \in \mathbb{R}^+$.

2.3 Convex Function Maximization Over a Polytope

Let C be a polytope, defined as the convex hull of its vertices $C = \text{co}(a_1, a_2, \ldots, a_m)$. By definition, any point $x \in C$ can be expressed as a convex combination of the vertices:

$$x = \sum_{i=1}^{m} \lambda_i a_i, \quad \text{where } \lambda_i \geq 0, \quad \sum_{i=1}^{m} \lambda_i = 1.$$

Since f is a convex function, applying Jensen's Inequality gives:

$$f(x) = f\left(\sum_{i=1}^{m} \lambda_i a_i\right) \leq \sum_{i=1}^{m} \lambda_i f(a_i).$$

This inequality shows that the function value at x is bounded above by a convex combination of the function values at the vertices. Since each $f(a_i)$ is a value of the function at a vertex, we can bound their weighted sum by the maximum function value over the vertices:

$$\sum_{i=1}^{m} \lambda_i f(a_i) \leq \sum_{i=1}^{m} \lambda_i \max_i f(a_i).$$

Because $\max_i f(a_i)$ is constant with respect to i, we factor it out:

$$\sum_{i=1}^{m} \lambda_i \max_i f(a_i) = \max_i f(a_i) \cdot \sum_{i=1}^{m} \lambda_i.$$

Using the property that $\sum_{i=1}^{m} \lambda_i = 1$, we obtain:

$$\sum_{i=1}^{m} \lambda_i \max_i f(a_i) = \max_i f(a_i).$$

Thus, we conclude:

$$f(x) \leq \max_i f(a_i).$$

This inequality shows that the function $f(x)$, when maximized over C, cannot exceed the maximum function value attained at one of the vertices. That is,

$$\max_{x \in C} f(x) = \max_{1 \leq i \leq m} f(a_i).$$

Thus, the maximum of a convex function on a polytope is always attained at a vertex.

2.4 Proving Joint Convexity of $f(x,y) = \|2x - y\|_2^2$

We need to show that the function $f(x,y) = \|2x - y\|_2^2$ is convex as a function of both x and y. We begin by defining auxiliary functions, which will help us in our proof of joint convexity. First, we define the affine mapping:

$$\phi : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n, \quad \phi(x,y) = 2x - y.$$

Since $\phi(x,y)$ is an affine transformation, it is convex. Next, we define:

$$\varphi : \mathbb{R}^n \to \mathbb{R}, \quad \varphi(z) = \|z\|_2^2.$$

Since the squared norm function is convex, and the composition of a convex function with an affine function is convex, it follows that:

$$f(x,y) = \varphi \circ \phi(x,y) = \|2x - y\|_2^2$$

is convex. To further verify this is true, let us prove the general form of this. We want to prove that for $a_1, a_2 \in \Omega$, and for convex functions h and g, We have:

$$\begin{aligned}(g \circ h)(\lambda a_1 + (1-\lambda)a_2) &= g(h(\lambda a_1 + (1-\lambda)a_2)) \\ &\leq g(\lambda h(a_1) + (1-\lambda)h(a_2)) \\ &\leq \lambda g(h(a_1)) + (1-\lambda)g(h(a_2)) \\ &= \lambda(g \circ h)(a_1) + (1-\lambda)(g \circ h)(a_2).\end{aligned}$$

2.5 Optimization Problem Analysis

2.5.1 Compute the Dual Problem

We consider the convex optimization problem:

$$\min_{x\in C} \frac{1}{2}\|x\|^2, \quad C = \{x : \|Ax - b\|_\infty \le \varepsilon\},$$

where $\varepsilon > 0$, $A \in \mathbb{R}^{n\times d}$, and $b \in \mathbb{R}^n$. This is a strongly convex quadratic minimization problem subject to an affine constraint in the ℓ_∞-norm. To derive the dual problem, we apply Fenchel-Rockafellar duality theory. To apply Fenchel-Rockafellar duality, we rewrite the problem in the standard form:

$$\min_x f(x) + g(Ax),$$

where:

$$f(x) = \frac{1}{2}\|x\|^2, \quad g(Ax) = \iota_{\{u:\|u-b\|_\infty\le\varepsilon\}}(Ax)$$

Fenchel-Rockafellar duality states that if $f(x)$ and $g(Ax)$ are convex, then the dual problem is given by:

$$\max_{y\in\mathbb{R}^n}\Big\{ - f^*(-A^\top y) - g^*(y)\Big\}.$$

where $f^*(y)$ and $g^*(y)$ are the Fenchel conjugates of f and g, respectively, defined by:

$$f^*(y) = \sup_{x\in\mathbb{R}^d}\left[\langle x, y\rangle - \frac{1}{2}\|x\|^2\right].$$

Expanding the expression by completing the square:

$$\begin{aligned} f^*(y) &= \sup_{x\in\mathbb{R}^d}\left[\langle x, y\rangle - \frac{1}{2}\|x\|^2\right] \\ &= \sup_{x\in\mathbb{R}^d}\left[-\frac{1}{2}\|x - y\|^2 + \frac{1}{2}\|y\|^2\right] \\ &= \frac{1}{2}\|y\|^2. \end{aligned}$$

The Fenchel conjugate $g^*(y)$ is computed as follows, using the substitution $u - b = v$:

$$g^*(y) = \sup_{\|u-b\|_\infty\le\varepsilon} \langle u, y\rangle = \sup_{\|v\|_\infty\le\varepsilon} \langle b + v, y\rangle = b^\top y + \sup_{\|v\|_\infty\le\varepsilon} \langle v, y\rangle = b^\top y + \varepsilon\|y\|_1.$$

Substituting the computed conjugates into the dual formulation:

$$\max_y\Big\{ - f^*(-A^\top y) - g^*(y)\Big\} = \max_y\Big\{ - \frac{1}{2}\|A^\top y\|^2 - \Big(b^\top y + \varepsilon\|y\|_1\Big)\Big\}.$$

Rearranging gives us that the dual problem is:

$$\boxed{\max_y\Big\{ - \frac{1}{2}\|A^\top y\|^2 - b^\top y - \varepsilon\|y\|_1\Big\}.}$$

2.5.2 Check Strong Duality

Strong duality states that the optimal values of the primal and dual problems coincide. To guarantee this, we must verify a qualification condition, which in Fenchel-Rockafellar duality is:

$$0 \in \text{int}\Big(\text{dom } g - A(\text{dom } f)\Big).$$

For our problem, since $f(x) = \frac{1}{2}\|x\|^2$ is finite everywhere, dom $f = \mathbb{R}^d$, so $A(\text{dom } f) = \text{range}(A)$. Meanwhile, dom $g = \{u : \|u - b\|_\infty \leq \varepsilon\}$, which represents a closed box centered at b with radius ε. We require:

$$0 \in \text{int}\Big(\{u - b : \|u - b\|_\infty \leq \varepsilon\} - \text{range}(A)\Big).$$

This condition holds if there exists x_0 such that $\|Ax_0 - b\|_\infty < \varepsilon$, ensuring the feasibility region has a non-empty interior. This condition holds because the set $\{z : \|z\|_\infty \leq \varepsilon\}$ is an open box centred at 0 whenever $\varepsilon > 0$, ensuring that its Minkowski sum with $-\text{range}(A)$ has a non-empty interior. Consequently, there must exist a x_0 such that $\|Ax_0 - b\|_\infty < \varepsilon$, meaning the qualification condition is satisfied. Since this holds for any $\varepsilon > 0$, strong duality follows.

2.5.3 Karush-Kuhn-Tucker (KKT) Conditions

The KKT conditions provide necessary and sufficient conditions for optimality when strong duality holds. The primal feasibility condition requires that x satisfies the constraint $\|Ax - b\|_\infty \leq \epsilon$. The dual feasibility condition follows from the subdifferential of $g^*(u)$, giving $\|u\|_1 \leq \epsilon$. The stationarity condition is derived from Theorem 8.1.1ofo the notes, specifically the relation:

$$-A^*u \in \partial f(x), \quad u \in \partial g(Ax).$$

Since $f(x) = \frac{1}{2}\|x\|^2$, the subdifferential is $\partial f(x) = \{x\}$, which simplifies the first stationarity condition to be:

$$x + A^*u = 0.$$

For $g(z) = \iota_{\{\|z-b\|_\infty \leq \epsilon\}}(z)$, the subdifferential $\partial g(z)$ is the normal cone $N_{\{\|z-b\|_\infty \leq \epsilon\}}(z)$, which takes the form

$$\partial g(z) = \begin{cases} \{u \in \mathbb{R}^n : u_i = 0 \text{ if } z_i \in (-\epsilon + b_i, \epsilon + b_i), \\ u_i \geq 0 \text{ if } z_i = \epsilon + b_i, \\ u_i \leq 0 \text{ if } z_i = -\epsilon + b_i\}, & \text{if } z \in \{\|z - b\|_\infty \leq \epsilon\}, \\ \emptyset, & \text{otherwise.} \end{cases}$$

Complementary slackness follows from the condition that u belongs to the subdifferential $\partial g(Ax)$, leading to the requirement

$$u_i(Ax - b)_i = 0, \quad \forall i = 1, \ldots, n.$$

Thus, the KKT conditions for this problem are summarized as the feasibility conditions $\|Ax - b\|_\infty \leq \epsilon$ and $\|u\|_1 \leq \epsilon$, the stationarity condition $x + A^*u = 0$, and the complementary slackness relation $u_i(Ax - b)_i = 0$. These conditions provide necessary and sufficient criteria for optimality given strong duality.

2.5.4 Derive a Convergence Rate for Primal Iterates

FISTA accelerates first-order methods for problems of the form $\min_x f(x) + g(x)$. Applied to the dual problem:

$$\min_y \frac{1}{2}\|A^\top y\|^2 + b^\top y + \varepsilon\|y\|_1,$$

it achieves the convergence rate:

$$d(y_k) - d(y^*) = \mathcal{O}\left(\frac{1}{k^2}\right).$$

Since $f(x) = \frac{1}{2}\|x\|^2$ is strongly convex, we can bound the squared primal error:

$$\frac{1}{2}\|x_k - x^*\|^2 \leq d(y_k) - d(y^*),$$

which implies:

$$\|x_k - x^*\|^2 = \mathcal{O}\left(\frac{1}{k^2}\right),$$

and thus:

$$\boxed{\|x_k - x^*\| = \mathcal{O}\left(\frac{1}{k}\right).}$$

This confirms the accelerated convergence rate of the primal iterates under FISTA.

3 Solving the Lasso Problem [30%]

Consider the problem:

$$\min_{x \in \mathbb{R}^d} \frac{1}{2n} \|Ax - y\|_2^2 + \lambda \|x\|_1$$

with data generation given in Python. In order to complete this task I have used the following functions to assist:

```python
def soft_threshold(x, threshold):
    """Soft thresholding operator. proximal operator of the L1 norm.
    prox_{ lambda * | |_1 }(x) = sign(x) max(|x|_1 - lambda, 0)
    """
    return np.sign(x) * np.maximum(np.abs(x) - threshold, 0)
```

```python
def lasso_objective(A, x, y, lam):
    residual = A @ x - y
    return 0.5 * np.mean(residual**2) + lam * la.norm(x, 1)
```

3.1 Implementing the Proximal Stochastic Gradient Algorithm (PSGA)

Implement PSGA:

$$x_{k+1} = \text{prox}_{\gamma_k \lambda \|\cdot\|_1} \left(x_k - \gamma_k (\langle a_{i_k}, x \rangle - y_{i_k}) a_{i_k} \right)$$

where $\gamma_k = \frac{n}{\|A\|_2^2 \sqrt{k+1}}$.

3.1.1 Code

```python
def psga_lasso(A, y, lam, max_iters, seed=42):
    """Proximal Stochastic Gradient Algorithm for Lasso

    Args:
        A (np.ndarray): Design matrix of shape (n, d)
        y (np.ndarray): Response vector of shape (n,)
        lam (float): Regularization parameter
        max_iters (int): Maximum number of iterations
        seed (int): Random seed

    Returns:
        np.ndarray: Estimated coefficients of shape (d,)
        list: Objective values at each iteration
    """
    np.random.seed(seed)
    n, d = A.shape

    # Compute the Lipschitz constant
```

```
    step_size = 1 / (la.norm(A, 2)**2 * np.sqrt(np.arange(1, max_iters + 1)))

    # Initialize the coefficients
    x = np.zeros(d)
    obj_vals = np.zeros(max_iters)

    for k in range(max_iters):
        i = np.random.randint(0, n)
        a_i = A[i]
        grad = (a_i @ x - y[i]) * a_i
        x = soft_threshold(x - step_size[k] * grad, step_size[k] * lam)

        obj_vals[k] = lasso_objective(A, x, y, lam)

    return obj_vals
```

3.2 Implementing the Randomized Coordinate Proximal Gradient Algorithm (RCPGA)

Implement RCPGA:

$$x_{k+1}^{(j)} = \begin{cases} \text{prox}_{\gamma_j \lambda |\cdot|}\left(x_k^{(j)} - \frac{\gamma_j}{n}\langle a_j, Ax_k - y\rangle\right), & j = j_k \\ x_k^{(j)}, & \text{otherwise} \end{cases}$$

where $\gamma_j = \frac{n}{\|a_j\|_2^2}$.

3.2.1 Code

```
def rcpga_lasso(A, y, lam, max_iters, seed=42):
    """Randomized Coordinate Proximal Gradient Algorithm for Lasso

    Args:
        A (ndarray): Design matrix of shape (n, d)
        y (ndarray): Response vector of shape (n,)
        lam (float): Regularization parameter
        max_iters (int): Maximum number of iterations
        seed (int): Random seed

    Returns:
        ndarray: Estimated coefficients of shape (d,)
        list: Objective values at each iteration
    """
    np.random.seed(seed)
    n, d = A.shape
    step_sizes = n / np.sum(A**2, axis=0)
```

```python

    x = np.zeros(d)
    r = -y.copy()
    obj_vals = np.zeros(max_iters)

    for k in range(max_iters):
        j = np.random.randint(0, d)
        a_j = A[:, j]
        grad = (a_j @ r) / n
        old_x = x[j]
        x[j] = soft_threshold(old_x - step_sizes[j] * grad, step_sizes[j] * lam)
        r += (x[j] - old_x) * a_j

        obj_vals[k] = lasso_objective(A, x, y, lam)

    return obj_vals
```

3.3 Comparing Algorithm Performance

We set $\lambda = 0.1$ to balance sparsity and data fidelity (Figure 5). A smaller λ approximates ordinary least squares, while a larger one enforces stronger regularization, leading to excessive coefficient shrinkage.

Figure 2 compares PSGA and RCPGA convergence, showing RCPGA's faster progress due to its coordinate-wise updates, while PSGA updates all coordinates based on a single sample, increasing variance. The log-scale plot (Figure 3) further highlights this difference.

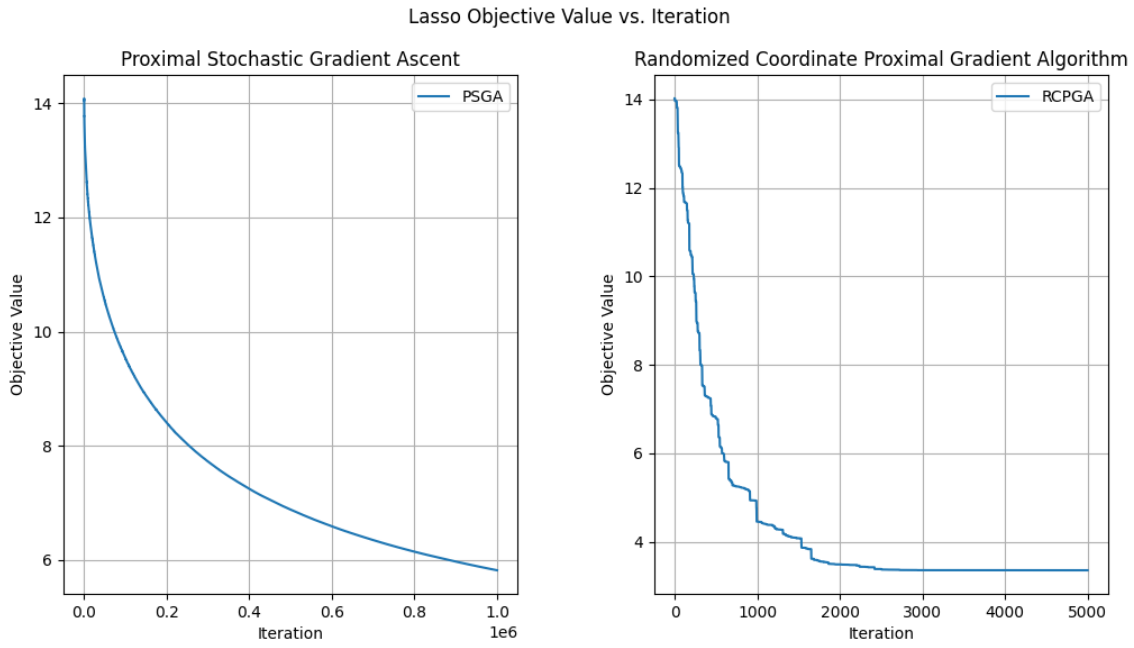

Figure 2: Lasso objective value vs. iteration for PSGA and RCPGA, demonstrating RCPGA's faster convergence.

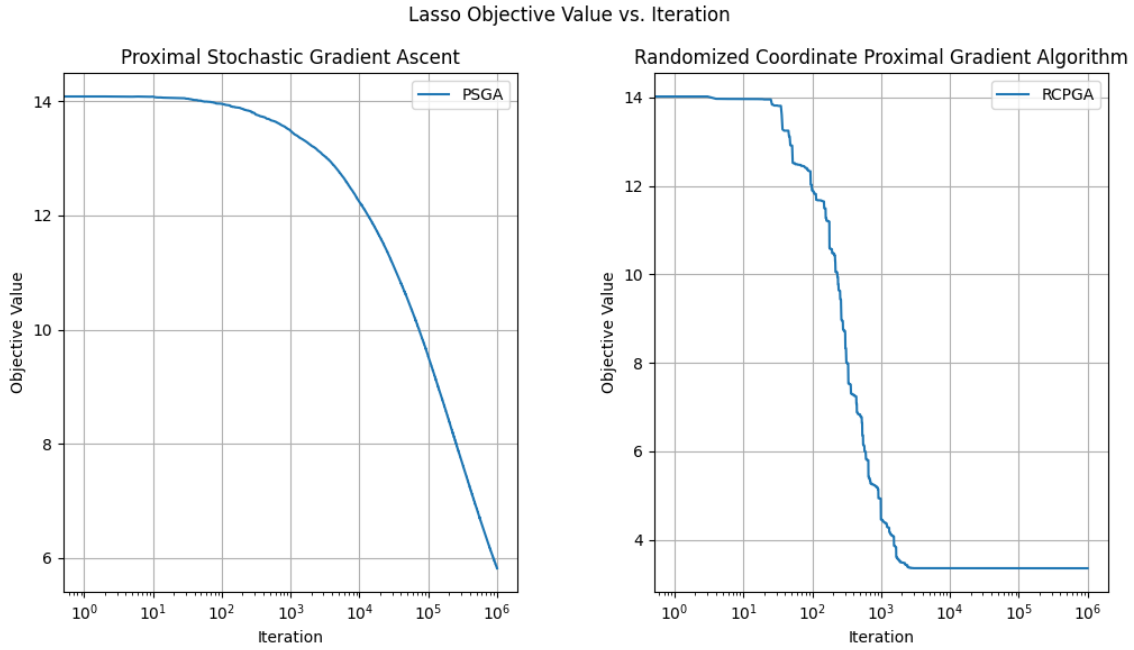

Figure 3: Lasso objective value vs. iteration on a logarithmic scale, emphasizing PSGA's slower decay.

Figure 4 illustrates the convergence behaviour of both algorithms, along with PSGA's ergodic mean sequence. The ergodic mean $\bar{x}^k$ (Figure 6) smooths fluctuations but results in slightly higher objective values than x^k. This aligns with theoretical results indicating that ergodic sequences improve stability but converge more slowly.

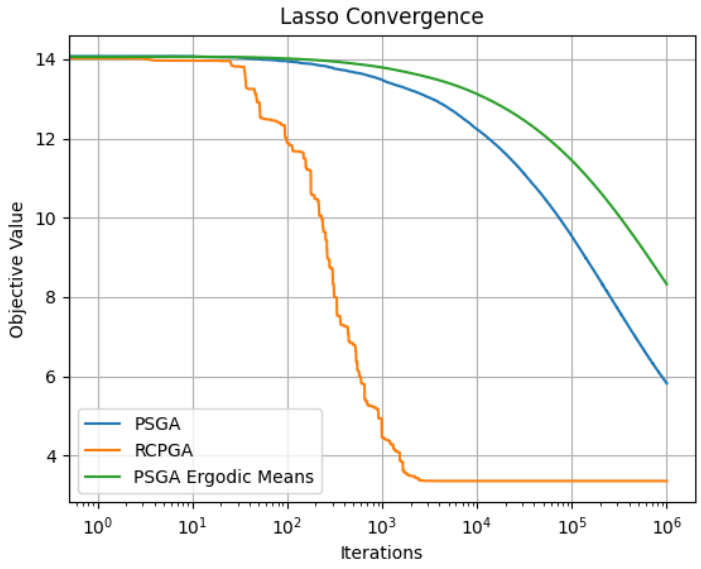

Figure 4: Lasso convergence comparison of PSGA, RCPGA, and PSGA Ergodic Means, highlighting convergence trends.

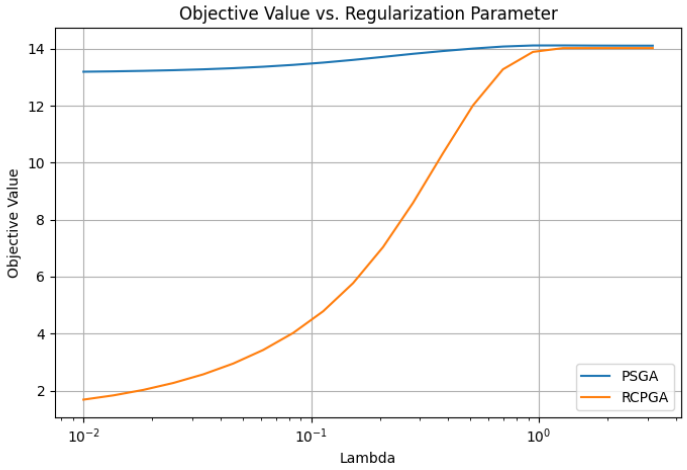

Figure 5: Objective value vs. regularization parameter λ for PSGA and RCPGA, showing the effect of regularization strength.

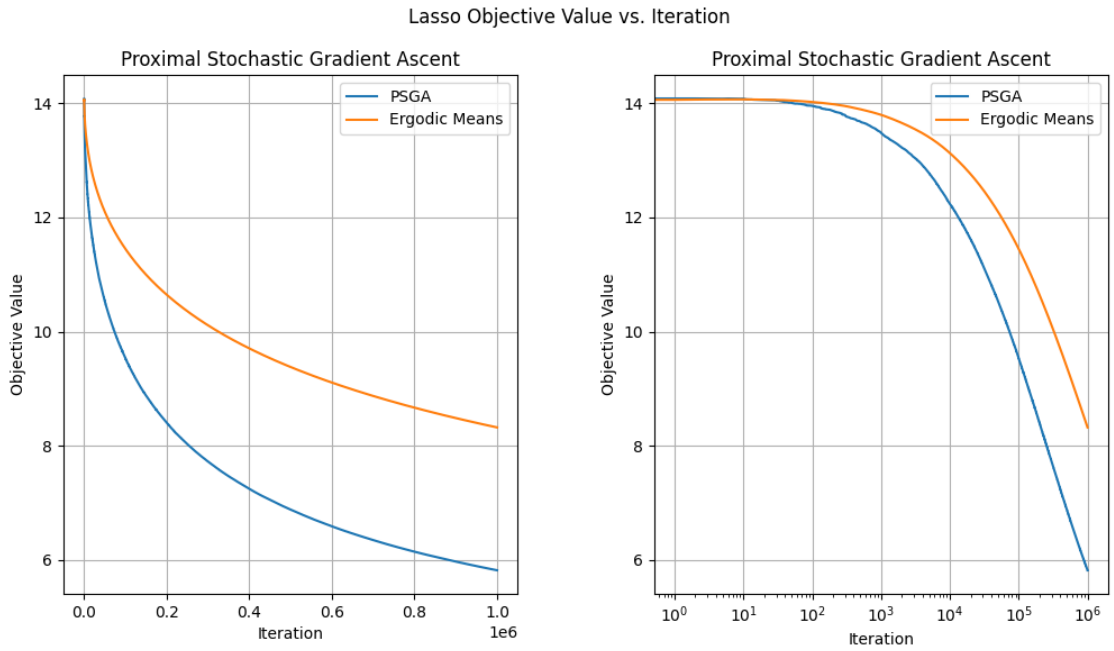

Figure 6: Ergodic means sequence convergence for PSGA, illustrating its smoothing effect on fluctuations.