MATH 173A: Homework #6

Due on Nov 26, 2024 at 23:59pm

Professor Cloninger

Ray Tsai

A16848188

Problem 1

Perform the conjugate gradient method by hand on the problem

$$\Phi(x) = \frac{1}{2}x^T \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} x \quad - \quad \sum_{i=1}^{2} x_i,$$

where $x \in \mathbb{R}^2$. Perform the algorithm either using version 0 or 1, where the conjugate directions are initialized and chosen algorithmically.

Proof. Let $A = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, b = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and we have

$$\Phi(x) = \frac{1}{2}x^T Ax - b^T x,$$

Initialization:

$$x^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad r_0 = Ax^{(0)} - b = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \quad p_0 = -r_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Iteration 1:

$$\alpha_0 = \frac{r_0^T r_0}{p_0^T A p_0} = \frac{2}{3},$$
$$x^{(1)} = x^{(0)} + \alpha_0 p_0 = \begin{bmatrix} \frac{2}{3} \\ \frac{2}{3} \end{bmatrix},$$
$$r_1 = r_0 + \alpha_0 A p_0 = \begin{bmatrix} -1 \\ -1 \end{bmatrix} + \frac{2}{3}\begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{3} \\ -\frac{1}{3} \end{bmatrix},$$
$$\beta_1 = \frac{r_1^T r_1}{r_0^T r_0} = \frac{1}{9},$$
$$p_1 = -r_1 + \beta_1 p_0 = \begin{bmatrix} -\frac{2}{9} \\ \frac{4}{9}. \end{bmatrix}$$

Iteration 2:

$$\alpha_1 = \frac{r_1^T r_1}{p_1^T A p_1} = \frac{3}{4},$$
$$x^{(2)} = x^{(1)} + \alpha_1 p_1 = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix},$$
$$r_2 = r_1 + \alpha_1 A p_1 = \begin{bmatrix} \frac{1}{3} \\ -\frac{1}{3} \end{bmatrix} + \frac{3}{4}\begin{bmatrix} -\frac{4}{9} \\ \frac{4}{9} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$
$$\beta_2 = 0,$$
$$p_1 = 0$$

Thus, the conjugate gradient method converges to the solution $x^* = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}$ in 2 iterations. $\square$

Problem 2

Here, we will prove the inequality used in class to prove fast convergence for strongly convex functions. Let $F(x)$ be a strongly convex function with constant c. Our goal is to show

$$F(x) - F(x^*) \leq \frac{1}{2c}\|\nabla F(x)\|^2 \qquad \text{for all } x \in \mathbb{R}^d. \tag{1}$$

(a) Fix $x \in \mathbb{R}^d$ and define the quadratic function

$$q(y) = F(x) + \nabla F(x)^T(y-x) + \frac{c}{2}\|x-y\|^2.$$

Find the y^* that minimizes $q(y)$.

Proof.

$$\nabla q(y) = \nabla F(x) - c(x-y) = 0 \implies y^* = x - \frac{1}{c}\nabla F(x).$$

□

(b) Show that $q(y^*) = F(x) - \frac{1}{2c}\|\nabla F(x)\|^2$

Proof.

$$q(y^*) = F(x) - \frac{1}{c}\|\nabla F(x)\|^2 + \frac{c}{2}\left\|\frac{1}{c}\nabla F(x)\right\|^2 = F(x) - \frac{1}{2c}\|\nabla F(x)\|^2.$$

□

(c) Use the above to deduce (1).

Proof. Since $F(x)$ is strongly convex, $F(y) \geq q(y)$ for all $y \in \mathbb{R}^d$, and thus

$$F(x^*) \geq q(x^*) \geq q(y^*) \geq F(x) - \frac{1}{2c}\|\nabla F(x)\|^2 \implies F(x) - F(x^*) \leq \frac{1}{2c}\|\nabla F(x)\|^2.$$

□

(d) Explain the proof technique in your own words to demonstrate understanding of what we did.

Proof. The strong convexity property of F yields $F \geq q$. Hence by minimizing q we can obtain a lower bound on F, and rearranging the equation yields the result. □

Problem 3

Indicate whether the following functions are strongly convex. Justify your answer.

(a) $f(x) = x$

Proof. Since $\nabla^2 f(x) = 0$, f is not strongly convex, as the Hessian is not positive definite. □

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

Proof. Since $\nabla^2 f(x) = 2$, f is strongly convex with constant $c = 2$. □

(c) $f(x) = \log(1 + e^x)$

Proof.

$$\begin{aligned} f'(x) &= \frac{e^x}{1+e^x} = \frac{1}{1+e^{-x}}, \\ f''(x) &= \frac{e^x}{(1+e^x)^2}. \end{aligned}$$

But then $\inf f''(x) = 0$, so f is not strongly convex. □

Question 4

Let $A \in \mathbb{R}^{n\times n}$ be a diagonal matrix with diagonal entries

$$A_{ii} = i, \quad \text{i.e. the entries run from 1 to } n,$$

and let $b \in \mathbb{R}^n$ a vector with all 1 entries. Define the function

$$f(x) = \frac{1}{2}x^T Ax - b^T x.$$

We want to compare the convergence behavior of conjugate gradient (version 0 or 1) and gradient descent. Do the following for $n = 20$ and $n = 100$ with initialization $x^{(0)} = 0$.

In [122…
```
import numpy as np
from matplotlib import pyplot as plt
```

Part A

Find the optimal solution x^* by solving $Ax = b$ using a Matlab/Python linear equation solver (or by hand and hard code the answer).

In [123…
```
n = 20

def A(n):
  return np.diag(np.arange(1, n+1))

def b(n):
  return np.ones(n)

def x_opt(n):
  return np.linalg.solve(A(n), b(n))
```

Part B

Program and run the gradient descent method for f with a fixed stepsize. Run the method for n iterations. You may experiment with the stepsize until you see something that works or use a stepsize dictated by a theorem in the class.

In [124…
```
def f(x, n):
  return 1/2 * x.T @ A(n) @ x - b(n) @ x

def df(x, n):
  return A(n) @ x - b(n)

def gd(x, n, mu = 2e-2):
  return x - mu * df(x, n)
```

In [125…
```
N = [20, 100]

gd_x_values = [[], []]
gd_f_values = [[], []]

for n in N:
  x = np.zeros(n)
  for i in range(n):
    gd_x_values[N.index(n)].append(np.linalg.norm(x - x_opt(n)))
    gd_f_values[N.index(n)].append(f(x, n) - f(x_opt(n), n))
    x = gd(x, n)
```

Part C

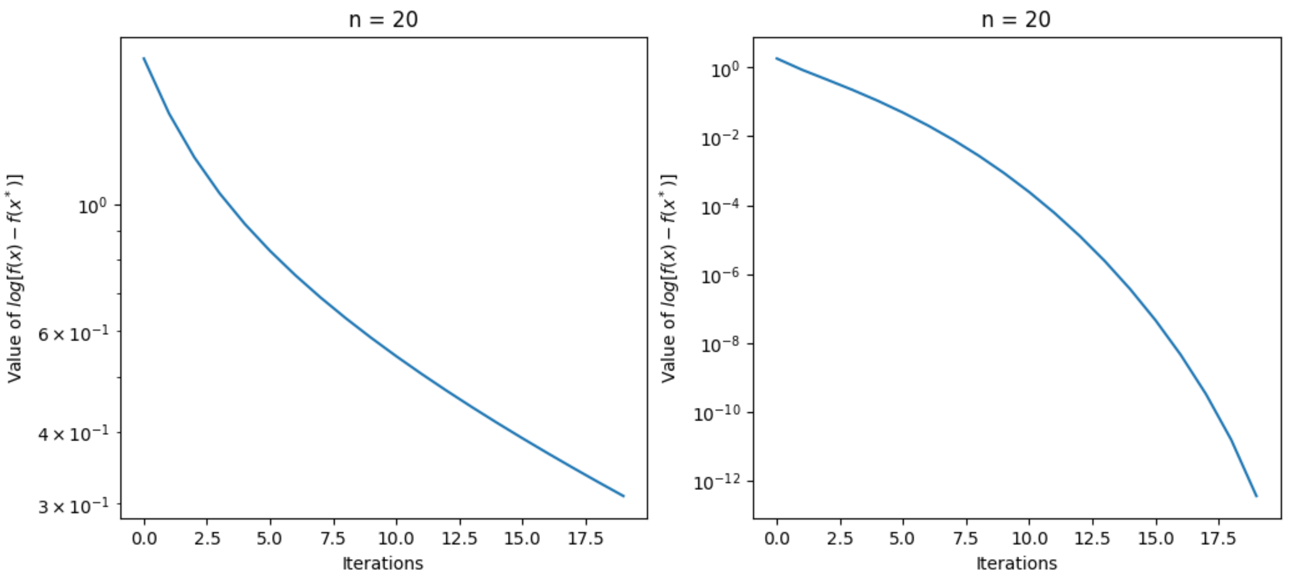
Program and run the conjugate gradient (version 0 or 1) for f. Run the method for n iterations.

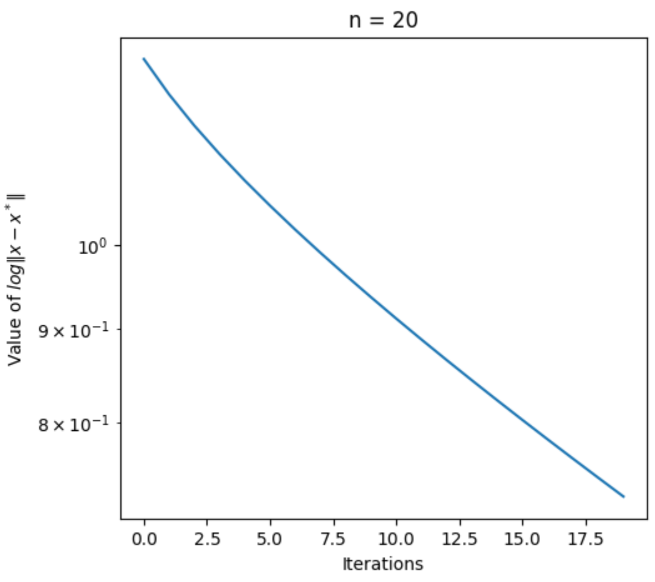
In [126…
```
N = [20, 100]

cgd_x_values = [[], []]
cgd_f_values = [[], []]

for n in N:
  x = np.zeros(n)
  r = df(x, n)
  p = -r

  for i in range(n):
    cgd_x_values[N.index(n)].append(max(np.linalg.norm(x - x_opt(n)), 1e-16)
    cgd_f_values[N.index(n)].append(max(f(x, n) - f(x_opt(n), n), 1e-16))

    alpha = r.T @ r / (p.T @ A(n) @ p)
    x += alpha * p
    r_new = r + alpha * A(n) @ p
    beta = r_new.T @ r_new / (r.T @ r)
    p = -r_new + beta * p
    r = r_new
```

Plot the $f(x^{(t)}) - f(x^*)$ for both methods in the same figure. In a different figure, plot $\|x^{(t)} - x^*\|$ for both methods. If you encounter a number smaller than 10^{-16}, set it to be 10^{-16}. In both plots, make the logarithmic scale for the vertical axis. Comment on the plots.

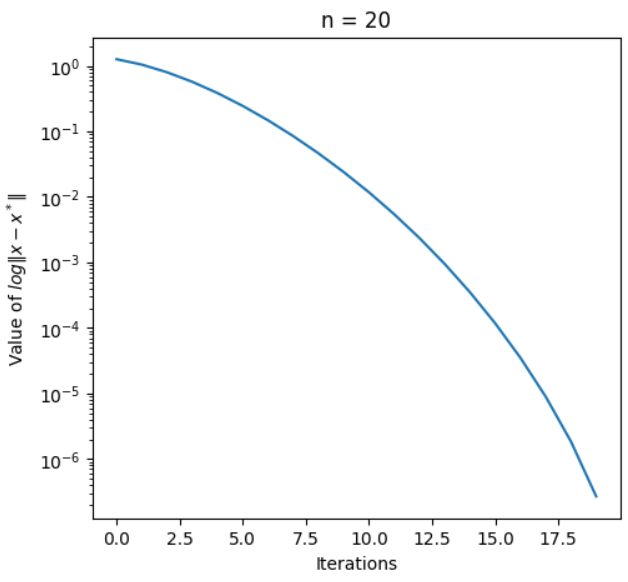
In [127…

```
plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
plt.plot(range(20), gd_f_values[0])
plt.yscale('log')
plt.xlabel(f"Iterations")
plt.ylabel(r"Value of $log [f(x) - f(x^*)]$")
plt.title(f"n = 20")

plt.subplot(1, 2, 2)
plt.plot(range(20), cgd_f_values[0])
plt.yscale('log')
plt.xlabel(f"Iterations")
plt.ylabel(r"Value of $log [f(x) - f(x^*)]$")
plt.title(f"n = 20")

plt.show()

plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
plt.plot(range(20), gd_x_values[0])
plt.yscale('log')
plt.xlabel(f"Iterations")
plt.ylabel(r"Value of $log \|x - x^*\|$")
plt.title(f"n = 20")

plt.subplot(1, 2, 2)
plt.plot(range(20), cgd_x_values[0])
plt.yscale('log')
plt.xlabel(f"Iterations")
plt.ylabel(r"Value of $log \|x - x^*\|$")
plt.title(f"n = 20")

plt.show()

plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
plt.plot(range(100), gd_f_values[1])
plt.yscale('log')
plt.xlabel(f"Iterations")
plt.ylabel(r"Value of $log [f(x) - f(x^*)]$")
plt.title(f"n = 100")

plt.subplot(1, 2, 2)
plt.plot(range(100), cgd_f_values[1])
plt.yscale('log')
plt.xlabel(f"Iterations")
plt.ylabel(r"Value of $log [f(x) - f(x^*)]$")
plt.title(f"n = 100")

plt.show()

plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
plt.plot(range(100), gd_x_values[1])
plt.yscale('log')
plt.xlabel(f"Iterations")
plt.ylabel(r"Value of $log \|x - x^*\|$")
plt.title(f"n = 100")

plt.subplot(1, 2, 2)
plt.plot(range(100), cgd_x_values[1])
plt.yscale('log')
plt.xlabel(f"Iterations")
plt.ylabel(r"Value of $log \|x - x^*\|$")
plt.title(f"n = 100")

plt.show()
```

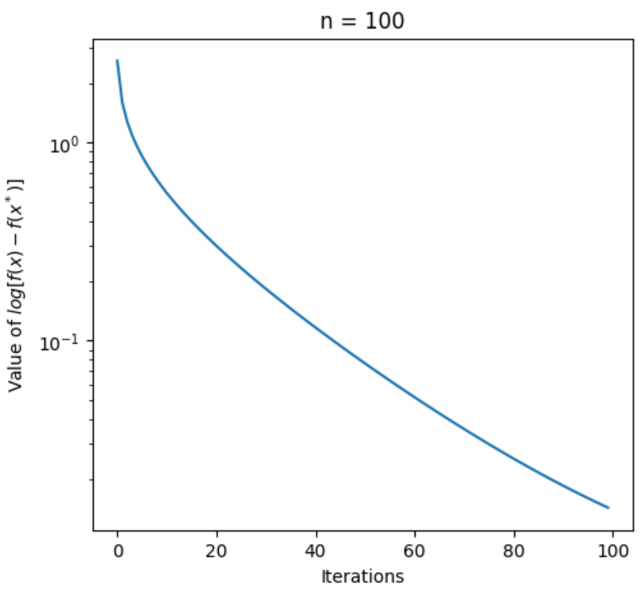




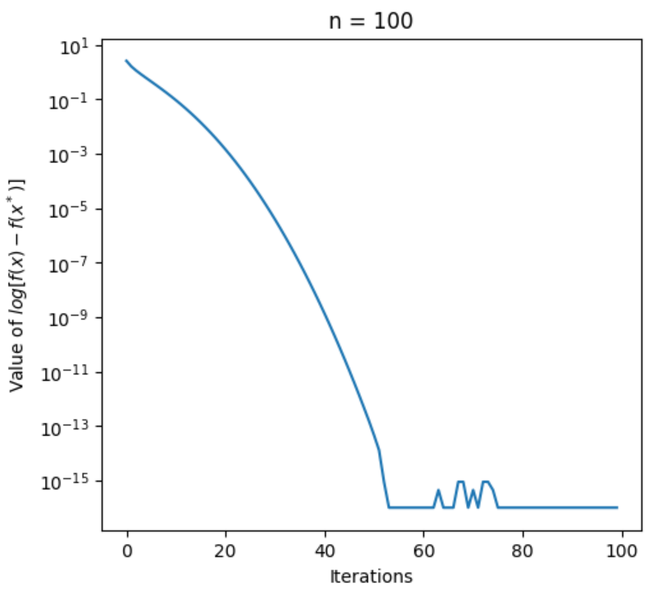
n = 20
10^0
9×10^{-1}
8×10^{-1}
Value of $log\|x - x^*\|$
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5
Iterations



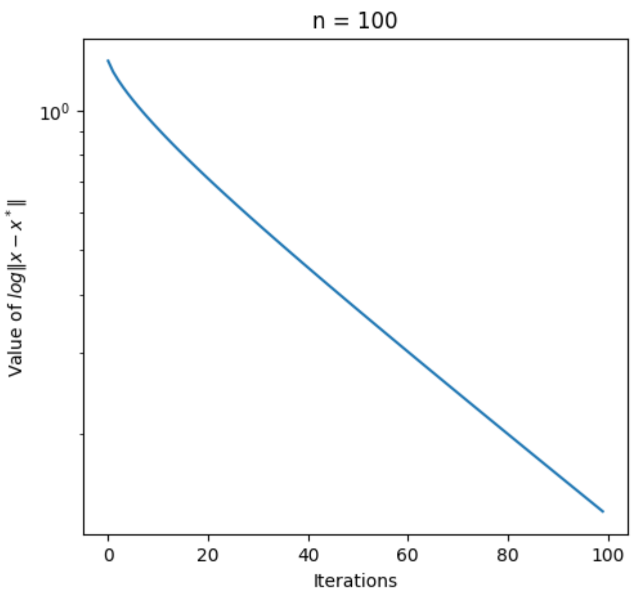
n = 20
10^0 10^{-1} 10^{-2} 10^{-3} 10^{-4} 10^{-5} 10^{-6}
Value of $log\|x - x^*\|$
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5
Iterations



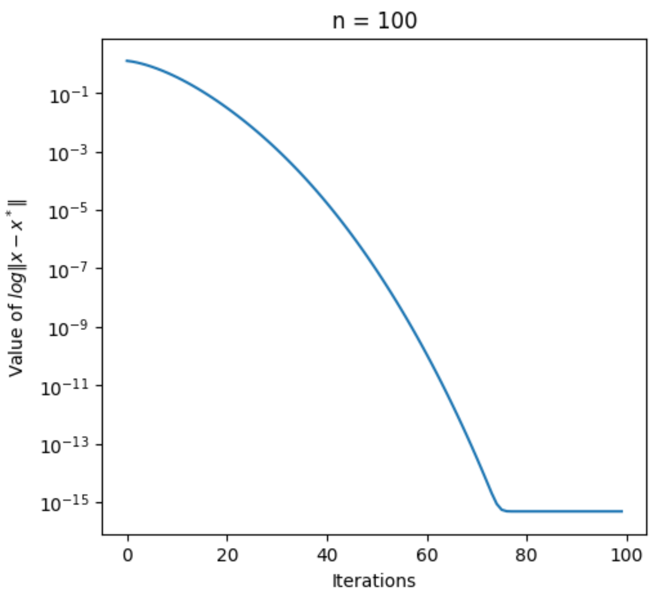
n = 100
10^0
10^{-1}
Value of $log[f(x) - f(x^*)]$
0 20 40 60 80 100
Iterations



n = 100
10^1 10^{-1} 10^{-3} 10^{-5} 10^{-7} 10^{-9} 10^{-11} 10^{-13} 10^{-15}
Value of $log[f(x) - f(x^*)]$
0 20 40 60 80 100
Iterations



n = 100
10^0
Value of $log\|x - x^*\|$
0 20 40 60 80 100
Iterations



n = 100
10^{-1} 10^{-3} 10^{-5} 10^{-7} 10^{-9} 10^{-11} 10^{-13} 10^{-15}
Value of $log\|x - x^*\|$
0 20 40 60 80 100
Iterations

The conjugate gradient descent method converges significantly faster than the standard gradient descent. The conjugate gradient descent method indeed converges within n iterations, agreeing with the theorem we learned.