

# **CS101: Assignment #4**

Due on April 3 at 23:59

*Prof. Turing, Alan* , Spring 2023, April 3 at 23:59

**Alice Bob**

## Problem 1

a) when  $\min(\|x\|_2)$ ,  $x = x^*$  is the solution to the problem, which is  $x^* = \begin{pmatrix} \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \end{pmatrix}$

b) We have a matrix  $A = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 0 \end{pmatrix}$ , the projection operator is

$$P = A(A^T A)^{-1} A^T = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

hence,

$$x^* = Pv = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 1 \end{pmatrix}.$$

c) We have a matrix  $A = \begin{pmatrix} 1 & -1 \\ -1 & 1 \\ 2 & 2 \end{pmatrix}$ , the projection operator is

$$P = A(A^T A)^{-1} A^T = \begin{pmatrix} \frac{1}{2} & -\frac{1}{2} & 0 \\ -\frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

hence,

$$x^* = Pv = \begin{pmatrix} \frac{1}{2} \\ -\frac{1}{2} \\ 0 \end{pmatrix}.$$

## Problem 2

a) we know that:

$$\text{prox}_\varphi(z) = \arg \min_{x \in \mathbb{R}} \left\{ \frac{1}{2} \|x - z\|^2 + \phi(x - c) \right\}.$$

let  $x' = x - c$

$$\text{prox}_\varphi(z) = \arg \min_{x \in \mathbb{R}} \left\{ \frac{1}{2} \|x' - (z - c)\|^2 + \phi(x' + c - c) \right\} + c = \text{prox}_\phi(z - c) + c.$$

b) if we want to  $f(x) = \frac{1}{2} \|x - z\|^2 + \phi(x)$  to be minimized, we need to find the  $x$  that makes the derivative of the function equal to zero.

we know

$$\partial f(x) = \begin{cases} x - z + \lambda & \text{when } x > 0 \\ [x - z - \lambda, x - z + \lambda] & \text{when } x = 0 \\ x - z - \lambda & \text{when } x < 0 \end{cases}$$

. Hence, let

$$\partial f(x) = 0$$

, we have

$$\text{prox}_\phi(z) = x^* = \begin{cases} z - \lambda & \text{when } z > \lambda \\ [z - \lambda, z + \lambda] & \text{when } z \in [-\lambda, \lambda] \\ z + \lambda & \text{when } z < -\lambda \end{cases}$$

c) if  $\varphi(x) = \lambda|x - c|$ , where  $c \in \mathbb{R}$  and  $\lambda > 0$ . Use the result from part a.

$$\text{prox}_\varphi(z) = \text{prox}_\phi(z - c) + c = \begin{cases} z - \lambda & \text{when } z > \lambda + c \\ [z - \lambda, z + \lambda] & \text{when } z \in [-\lambda + c, \lambda + c] \\ z + \lambda & \text{when } z < -\lambda + c \end{cases}$$

### Problem 3

a) If we take the derivative of  $\frac{1}{2}\|\mathbf{x} - \mathbf{x}^{t-1}\|^2 + \gamma g(\mathbf{x})$ , we have

$$\mathbf{x}^t = \text{prox}_{\gamma g}(\mathbf{x}^{t-1}) = \mathbf{x}^{t-1} - \gamma \nabla g(\mathbf{x}^t)$$

b) By the convexity of  $g$ , we know that  $g(\mathbf{x}^t) + \nabla g(\mathbf{x}^t)^T (\mathbf{x}^{t-1} - \mathbf{x}^t) \leq g(\mathbf{x}^{t-1})$ . Hence, we have

$$g(\mathbf{x}^t) \leq g(\mathbf{x}^{t-1}) - \nabla g(\mathbf{x}^t)^T (\mathbf{x}^{t-1} - \mathbf{x}^t) = g(\mathbf{x}^{t-1}) - \gamma \nabla \|g(\mathbf{x}^t)\|_2^2$$

c) because  $\mathbf{x}^t = \mathbf{x}^{t-1} - \gamma \nabla g(\mathbf{x}^t)$  which is a gradient descent method, so

$$-\infty < g(\mathbf{x}^t) \leq g(\mathbf{x}^{t-1})$$

and we have

$$g(\mathbf{x}^t) \leq g(\mathbf{x}^{t-1}) - \gamma \nabla \|g(\mathbf{x}^t)\|_2^2$$

hence

$$0 \leq \gamma \nabla \|g(\mathbf{x}^t)\|_2^2 \leq 0$$

if

$$t \rightarrow +\infty$$

### Problem 4

a) because

$$\partial f(x) = \{v \in \mathbb{R}^n : f(y) \geq f(x) + v^T(y - x), \forall y \in \mathbb{R}^n\}$$

if  $g(x) = \theta f(x)$ ,

$$\partial g(x) = \{v \in \mathbb{R}^n : g(y) \geq g(x) + v^T(y - x), \forall y \in \mathbb{R}^n\}$$

$$\partial g(x) = \{v \in \mathbb{R}^n : \theta f(y) \geq \theta f(x) + v^T(y - x), \forall y \in \mathbb{R}^n\}$$

$$\partial g(x) = \left\{ v \in \mathbb{R}^n : f(y) \geq f(x) + \frac{v^T}{\theta}(y - x), \forall y \in \mathbb{R}^n \right\}$$

$$\partial g(x) = \theta \{v \in \mathbb{R}^n : f(y) \geq f(x) + v^T(y - x), \forall y \in \mathbb{R}^n\} = \theta \partial f(x)$$

b)

$$\partial h(x) = \{v \in \mathbb{R}^n : f(y) + g(y) \geq f(x) + g(x) + v^T(y - x), \forall y \in \mathbb{R}^n\}$$

all of the elements that satisfy

$$f(y) \geq f(x) + v^T(y - x), \forall y \in \mathbb{R}^n$$

and

$$g(y) \geq g(x) + v^T(y - x), \forall y \in \mathbb{R}^n$$

are in the set

$$\partial h(x)$$

hence

$$\partial f(x) + \partial g(x) \subseteq \partial h(x)$$

c) we know that

$$\partial \|x\|_1 = \begin{cases} 1 & \text{when } x > 0 \\ [-1, 1] & \text{when } x = 0 \\ -1 & \text{when } x < 0 \end{cases}$$

.

hence  $\text{sgn}(x) \in \partial \|x\|_1$ .

## Problem 5

b) Not differentiable at  $x = 0$ , and  $h$  is convex.

c)

$$\nabla \left[ \frac{1}{2} \|x - y\|_2^2 + \gamma \lambda \|x\|_1 \right] = \begin{cases} x - y + \gamma \lambda & \text{when } x > 0 \\ [x - y - \gamma \lambda, x - y + \gamma \lambda] & \text{when } x = 0 \\ x - y - \gamma \lambda & \text{when } x < 0 \end{cases}$$

let it be 0, we have

$$\text{prox}_{\gamma g(y)} = x^* = \begin{cases} y - \gamma \lambda & \text{when } y > \lambda \\ [y - \gamma \lambda, y + \gamma \lambda] & \text{when } y \in [-\gamma \lambda, \gamma \lambda] \\ y + \gamma \lambda & \text{when } y < -\gamma \lambda \end{cases}$$

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% load the variables of the optimization problem
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

load('dataset.mat');

[p, n] = size(A);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% set up the function and its gradient
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

evaluate_f = @(x) (1/n)*sum(log(1+exp(-b.*(A'*x))));
evaluate_gradf = @(x) (1/n)*A*(-b.*exp(-b.*(A'*x))./(1+exp(-b.*(A'*x))));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% parameters of the gradient method
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

xInit = zeros(p, 1); % zero initialization
stepSize = 1; % step-size of the gradient method
maxIter = 1000; % maximum number of iterations

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% optimize
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% initialize
x = xInit;

% keep track of cost function values
objVals = zeros(maxIter, 1);

% iterate
for iter = 1:maxIter

    % update
    xNext = x - stepSize*evaluate_gradf(x);

```

```

% evaluate the objective
funcNext = evaluate_f(xNext);

% store the objective and the classification error
objVals(iter) = funcNext;

fprintf(['[%d/%d] [step: %.1e] [objective: %.1e]\n',...
        iter, maxIter, stepSize, objVals(iter)];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% begin visualize data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% plot the evolution
figure(1);
set(gcf, 'Color', 'w');
semilogy(1:iter, objVals(1:iter), 'b-',...
         iter, objVals(iter), 'b*', 'LineWidth', 2);
grid on;
axis tight;
xlabel('iteration');
ylabel('objective');
title(sprintf('GM (f = %.2e)', objVals(iter)));
xlim([1 maxIter]);
set(gca, 'FontSize', 16);
drawnow;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% end visualize data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% update w
x = xNext;
end

```