课程名 | 标题名

123456789 **作者名** **教师名老师**, **学期**, 截止时间：**DDL**

一级标题

p

Problem 1

使用 `#prob` 命令来创建一个带**编号**的问题框。

或者

使用 `#pt("...")` 命令来创建一个不带编号、可自定义文本的问题框。

比如，本框使用的命令是 `#pt("或者")`。

而如果你想只想要一个问题框：

使用 `#p` 命令来创建一个空白的问题框。

Problem 2

1. because

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

if $g(\boldsymbol{x}) = \theta f(\boldsymbol{x})$,

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

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

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

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

2.

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

all of the elements that satisfy

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

and

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

are in the set

$$\partial h(\boldsymbol{x})$$

hence

$$\partial f(\boldsymbol{x}) + \partial g(\boldsymbol{x}) \subseteq \partial h(\boldsymbol{x})$$

3. 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 $\operatorname{sgn}(x) \in \partial \|x\|_1$.

Problem 3

2. Not differentiable at $\boldsymbol{x} = \mathbf{0}$, and h is convex.

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

.

let it be 0, we have

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

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% 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
```