Contents


```
% Logistic Regression
%
% Gradient method with a fixed step
%
% U. S. Kamilov, CIG, WUSTL, 2021.

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

prepare workspace

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

clear; close all; home;

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

load the variables of the optimization problem

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

load('dataset.mat');

[p, n] = size(A);

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

set up the function and its gradient

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
lambda = 0.02;
mu = 0.001;
evaluateFunc = @(x) (1/n)*sum(log(1+exp(-b.*(A'*x))));
evaluateGrad = @(x) (1/n)*A*(-b.*exp(-b.*(A'*x))./(1+exp(-b.*(A'*x))));

evaluateFunc_g = @(x) lambda*norm(x,1);
evaluateGrad_g = @(x) lambda*sign(x);
evaluateFunc_gsmooth = @(x) (x.^2/(2*mu))*(norm(x,1) < mu) ...
                          + (abs(x) - mu/2)*(norm(x,1) > mu);
evaluateGrad_gsmooth = @(x) (x/mu)*(norm(x,1) < mu) + (sign(x))*(norm(x,1) > mu);

evaluateFunc_h = @(x) evaluateFunc(x) + evaluateFunc_g(x);
evaluateGrad_h = @(x) evaluateGrad(x) + evaluateGrad_g(x);
evaluateGrad_hsmooth = @(x) evaluateGrad(x) + evaluateGrad_gsmooth(x);
```

```
prox_g = @(y) (abs(y) - stepSize*lambda).*sign(y);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

parameters of the gradient method

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

xInit = zeros(p, 1); % zero initialization
stepSize = 0.1; % step-size of the gradient method
tol = 1e-6; % stopping tolerance
maxIter = 100; % maximum number of iterations
thetaPast = 1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

optimize

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

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

% iterate
for iter = 1:maxIter

    % gradient at w
    grad = evaluateGrad(x);

    % update
    %xNext = x - stepSize*grad;

    % SGD
    %xNext = x - stepSize*evaluateGrad_h(x);

    %AGM
    theta = (1+sqrt(1+4*thetaPast^2))/2;
    beta_t = (thetaPast - 1)/theta;
    s = x + beta_t*(x - xPast);
    xNext = prox_g(s - stepSize*evaluateGrad_hsmooth(s));


    % evaluate the objective
    funcNext = evaluateFunc(xNext);

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

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

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % 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('cost: %.4e', objVals(iter)));
    xlim([1 maxIter]);
    set(gca, 'FontSize', 16);
    drawnow;

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

    % check stopping criterion
    if(norm(grad) < tol)
        break;
    end

    % update w
    thetaPast = theta;
    xPast = x;
    x = xNext;
end

% save for plotting
cost_gm = objVals;
save('plotfile.mat', 'cost_gm');
```

```
Unrecognized function or variable 'stepSize'.

Error in gm>@(y)(abs(y)-stepSize*lambda).*sign(y) (line 39)
prox_g = @(y) (abs(y) - stepSize*lambda).*sign(y);

Error in gm (line 75)
    xNext = prox_g(s - stepSize*evaluateGrad_hsmooth(s));
```

Published with MATLAB® R2020b