```
% Thomas Satterly
% AAE 550
% HW 1, Part II (c)

clear;
close all;

% Setup problem
aae550.hw1.partII_setup;


% Set constraint coefficients
%cs = ones(1, numel(gs));
% for i = 1e1:1e1:1e10
rp = 25000000;
x0 = [0.44; 0.425];
[isValid, gx] = aae550.hw1.checkConstraints(gs, x0);
assert(isValid, 'Starting point not valid!');
% Scale constraint equations at starting point to the same value
%cs = [1e4 1e0 1e4 1e0 1e2 1e10 1e9 1 1 1e2];
cs = min(gx) ./ gx;
%cs = [1 1 1 1 1 1 1 1 1 1];
%cs = [1 1 1 1 1 1 1 1 1 1];
for i = 1:numel(gs)
    gs{i} = @(x) cs(i) * gs{i}(x);
end

maxErr = 1e-6;
err = inf;
fLast = inf;

minCount = 0;
iterationCount = 0;
j = 0;

epsilon = -0.2;
C = -epsilon / sqrt(rp);
while (err > maxErr)
    j = j + 1;

    % Update contraint coefficients
    dx = 1e-2;
    gradF = [f(x0 + [dx; 0]) - f(x0 - [dx; 0]); ...
        f(x0 + [0; dx]) - f(x0 - [0; dx])] ./ (2 * dx);
    for i = 1:numel(gsOrig)
        gradG = [gsOrig{i}(x0 + [dx; 0]) - gsOrig{i}(x0 - [dx;
 0]); ...
            gsOrig{i}(x0 + [0; dx]) - gsOrig{i}(x0 - [0; dx])] ./ (2 *
 dx);
        cj(i) = norm(gradF) / norm(gradG);
        gs{i} = @(x) cj(i) * gsOrig{i}(x);
    end
```

```
    % Create pseudo-objective function
    objFunc = @(x) aae550.hw1.extLinIntPenalty(f, x, rp, {g1, g2, g3,
 g4, g5, g6, ...
        g7, g8, g9, g10}, epsilon);

    options = optimoptions(@fminunc, 'Display', 'iter', 'PlotFcn',
 @optimplotfval, ...
        'HessUpdate', 'steepdesc');

    [x_opt, f_opt, exitFlag, output, grad] = fminunc(objFunc, x0,
 options);
    %[isValid, gx] = aae550.hw1.checkConstraints(gs, x_opt);

    % Record values for table
    data(j).minimization = j;
    data(j).rp = rp;
    data(j).x0 = x0;
    data(j).xOpt = x_opt;
    data(j).fOpt = f(x_opt);
    [~, data(j).gx] = aae550.hw1.checkConstraints(gsOrig, x_opt);
    data(j).iterations = output.iterations;
    data(j).exitFlag = exitFlag;

    err = abs(f_opt - fLast);
    fLast = f_opt;

    x0 = x_opt;
    rp = rp / 2;
    epsilon = -C * sqrt(rp);

    % Update counters
    minCount = minCount + 1;
    iterationCount = iterationCount + output.iterations + 1; % Oh, so
 now Matlab decides to start indecies at 0
end

% Make sure final solution is valid
[isValid, gx] = aae550.hw1.checkConstraints(gs, x_opt);
%assert(isValid, 'Solution is invalid!');

% Post data to excel table

% File name
fName = [mfilename('fullpath'), '.xlsx'];

if exist(fName, 'file') == 2
    delete(fName);
end

% Create table column titles
gCell = {};
for i = 1:numel(gs)
    gCell{i} = sprintf('g%d(x_star)', i);
```

```
end
xlswrite(fName, {'Minimization', 'r_p', 'x_0', 'x_star', 'f(x_star)',
 gCell{:}, '# of Iterations', 'Exit Flag'}, 'sheet1');

for i = 1:numel(data)
    dataCell = {};
    dataCell{1} = data(i).minimization;
    dataCell{2} = data(i).rp;
    dataCell{3} = num2str(data(i).x0');
    dataCell{4} = num2str(data(i).xOpt');
    dataCell{5} = data(i).fOpt;
    for j = 1:numel(data(i).gx)
        dataCell{end + 1} = data(i).gx(j);
    end
    dataCell{end + 1} = data(i).iterations;
    dataCell{end + 1} = data(i).exitFlag;
    xlswrite(fName, dataCell, 'sheet1', sprintf('A%d', i + 1));
end
```

Published with MATLAB® R2016a