CSCI 5654-Fall16: Assignment # 4
Your Name: Robert Werthman

P1.

(A)
Let $\max(2x_1 + 3x_2 - 5x_3,\ x_1,\ x_2,\ 2) \leq t$. We can then form the following linear program:

$$\begin{array}{lllll} \min & t & & & \\ \textsf{s.t.} & +2x_1 & +3x_2 & -5x_3 & \leq t \\ & +2x_1 & -x_2 & +x_3 & \leq t \\ & x_1, & x_2 & & \leq t \end{array}$$

(B)
Let $t_1,\, t_2,\, t_3,\, t_4 \geq 0$.
Let $|x_1 + x_2| \leq t_1$, $|x_2 - x_3| \leq t_2$, $|x_3 - x_1| \leq t_3$, $|x_1 + x_2 + x_3| \leq t_4$.
You now have the linear problem

$$\begin{array}{llllll} \min & +t_1 & +t_2 & +t_3 & +t_4 & \\ \textsf{s.t.} & +x_1 & +x_2 & & & \leq t_1 \\ & -x_1 & -x_2 & & & \leq t_1 \\ & & +x_2 & -x_3 & & \leq t_2 \\ & & -x_2 & +x_3 & & \leq t_2 \\ & -x_1 & & +x_3 & & \leq t_3 \\ & +x_1 & & -x_3 & & \leq t_3 \\ & +x_1 & +x_2 & +x_3 & & \leq t_3 \\ & -x_1 & -x_2 & -x_3 & & \leq t_3 \\ & t_1, & t_2 & t_3 & t_4 & \geq 0 \end{array}$$

(C)
Let $\max(|x_1|,\ |x_2|,\ |x_3|,\ |x_1 + x_2|) \leq t$.
You now have the linear problem

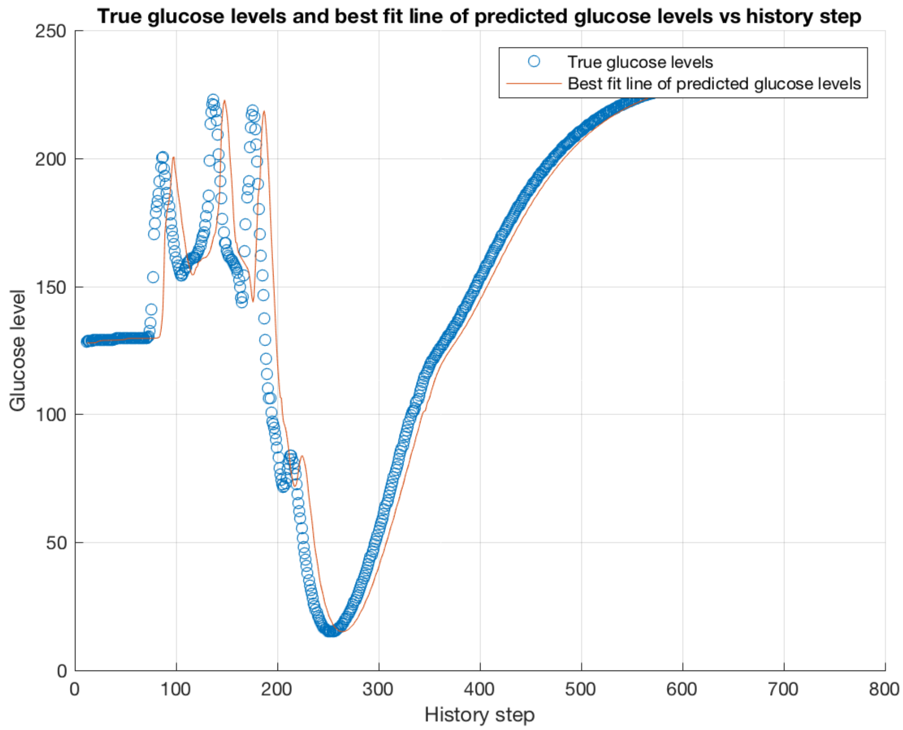
$$\begin{array}{lllll} \min & t & & & \\ \textsf{s.t.} & +x_1 & -x_2 & & \leq 5 \\ & & +x_2 & & \leq 3 \\ & +x_1 & & & \leq t \\ & -x_1 & & & \leq t \\ & & +x_2 & & \leq t \\ & & -x_2 & & \leq t \\ & & & +x_3 & \leq t \\ & & & -x_3 & \leq t \\ & +x_1 & +x_2 & & \leq t \\ & -x_1 & -x_2 & & \leq t \\ & t & & & \geq 0 \end{array}$$

P2.

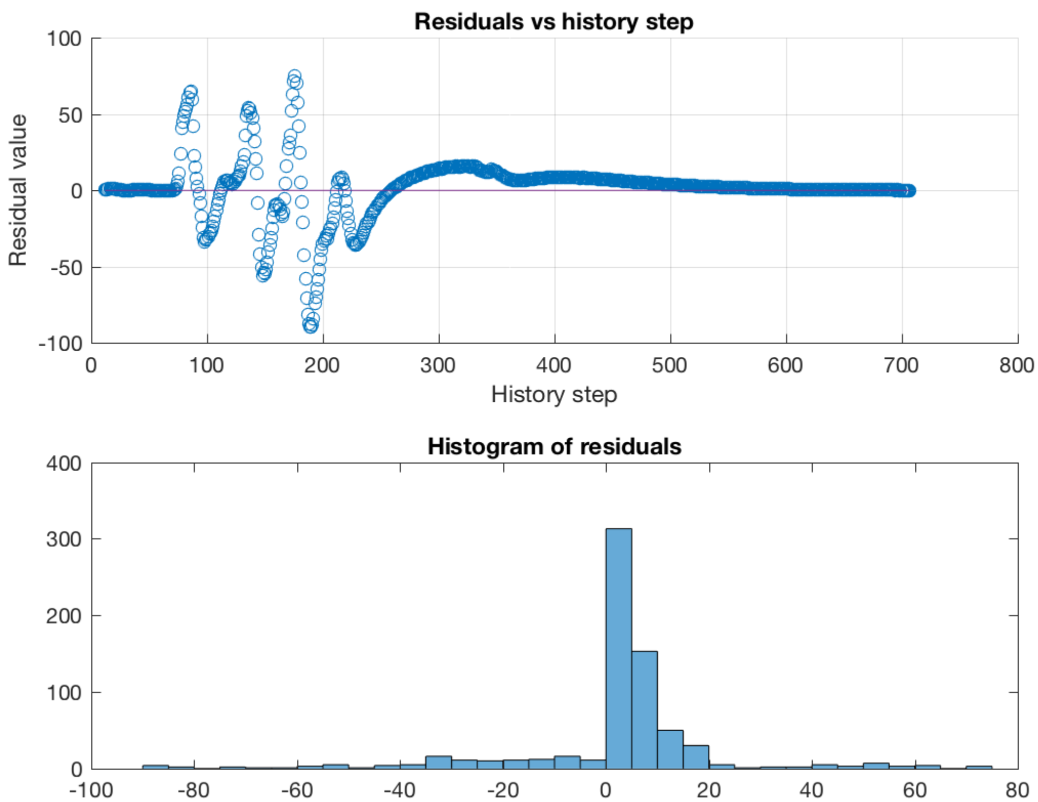
```matlab
% Code for problem 2 of homework 4.
% Read in the csv file skipping the first row
M = csvread('insulinGlucose1.csv', 1);

% Insulin input values
u = M(:,1);
% Glucose levels
G = M(:,2);

A = [];
b = [];

[m,~] = size(G(12:end));
for i = 1:m
    b = [b, G(i)];
    b = [b, -G(i)];
end

b = b';

% Initialize the objective function which is the minimum of the sum
% of the absolute values of the residuals.
% Residuals have the form y - Ax.
f = ones(1,17);

% Create the constraint matrix A which is the residuals
% of glucose levels 12 -> 707.
% If u = |x| then u = x and u = -x
[m,~] = size(G);
for t = 11:(m-1)
    a = [G(t) G(t-1) G(t-2) G(t-3) G(t-4) G(t-5) G(t-6) G(t-7) G(t-8) G(t-9) G(t-10)];
    a = horzcat(a, [u(t) u(t-1) u(t-2) u(t-3) u(t-4) u(t-5)]);
    % Append the constraint ax.
    A = [A;a];

    % Append the constraint -ax.
    a = times(a, -1);
    A = [A;a];
end

% Solve for the coefficents.
% x will have an array of the coefficient values when it is solved.
[x,fval] = linprog(f,A,b)

% Recreate the matrix A without the additional absolute value constraints.
A = [];
for t = 11:(m-1)
    a = [G(t) G(t-1) G(t-2) G(t-3) G(t-4) G(t-5) G(t-6) G(t-7) G(t-8) G(t-9) G(t-10)];
    a = horzcat(a, [u(t) u(t-1) u(t-2) u(t-3) u(t-4) u(t-5)]);
    A = [A;a];
end

% Create an array of the history steps.
t = [];
for i = 12:m
    t = [t, i];
end

% Plot the true glucose levels at each history step t_i.
figure(1);
scatter(t,G(12:end));
hold on;

% Plot the best fit line of the predicted glucose levels at each
% history step t_i.
% x stores the estimated coefficients of the best fit line.
% |x| = sqrt(x^2)
A = (A*x).^2;
A = sqrt(A);
plot(t, A);
grid on;
legend('True glucose levels', 'Best fit line of predicted glucose levels');
title('True glucose levels and best fit line of predicted glucose levels vs history step');
```

```matlab
xlabel('History step');
ylabel('Glucose level');
saveas(1, 'predictions.png');

% Plot the residuals.
figure(2);
residuals = [];
trueGlucoseValues = G(12:end);
predictedGlucoseValues = A;
[m,~] = size(predictedGlucoseValues);
for i = 1:m
    residuals = [residuals, trueGlucoseValues(i) - predictedGlucoseValues(i)];
end

subplot(2,1,1);
scatter(t, residuals);
hold on;
plot(t, zeros(m));
grid on;
title('Residuals vs history step');
xlabel('History step');
ylabel('Residual value');

subplot(2,1,2);
histogram(residuals);
title('Histogram of residuals');
saveas(2, 'residuals.png');
```

Coefficient	Value
a_0	8.704148513061227e-14
a_1	-3.375077994860476e-14
a_2	-1.474376176702208e-13
a_3	2.131628207280301e-13
a_4	-1.243449787580175e-13
a_5	-1.056932319443149e-13
a_6	1.376676550535194e-13
a_7	-3.979039320256561e-13
a_8	3.179678742526448e-13
a_9	6.430411758628907e-13
a_{10}	0.999999999999309
b_0	1.989519660128281e-13
b_1	-2.060573933704291e-13
b_2	1.847411112976261e-13
b_3	-8.526512829121202e-14
b_4	-1.207922650792170e-13
b_5	7.815970093361102e-14



(a) Linear Regression Predictions



(b) Residuals

P3.

(A)

1.

x_1	$+2$	$-x_4$	$+x_5$	$-x_6$			$+w_6$
x_2	-8		$+2x_5$	$+x_6$	$+w_4$	$+w_5$	$+w_6$
x_3	$+4$	$+x_4$	$-2x_5$			$-w_5$	$-w_6$
w_1	-7	$+x_4$	$+x_5$	$+3x_6$	$+w_4$	$+w_5$	
w_2	-3	$+2x_4$		$+x_6$			$-w_6$
w_3	2	$+x_4$	$-2x_5$	$+x_6$		$-w_5$	$-w_6$
ζ	16		$-6x_5$		$-3w_4$	$-2w_5$	$-5w_6$

2.

x_1	-1	$-x_3$	$+x_4$	$+2x_6$	$-w_1$		$+w_4$
x_2	-4	$-x_3$	$+x_4$	$+x_6$			$+w_4$
x_5	0	$-x_3$		$+2x_6$	$-w_1$	$+w_2$	$+w_4$
w_3	-2	$+x_3$		$+x_6$			
w_5	7	$+x_3$	$-x_4$	$-5x_6$	$+2w_1$	$-w_2$	$-2w_4$
w_6	-3		$2x_4$	$+x_6$		$-w_2$	
ζ	14	$+4x_3$	$-6x_4$	$-6x_6$	$+2w_1$		$-5w_4$

3.

x_1	-1		$+x_4$	$+x_5$		$-w_2$	
x_2	-6	$+x_3$	$+x_4$	$+x_5$	$+w_1$	$-w_2$	$-w_3$
x_6	2	$-x_3$					$+w_3$
w_4	-4	$+3x_3$		$+x_5$	$+w_1$	$-w_2$	$-2w_3$
w_5	5		$-x_4$	$-2x_5$		$+w_2$	$-w_3$
w_6	-1	$-x_3$	$+2x_4$			$-w_2$	$+w_3$
ζ	22	$-5x_3$	$-6x_4$	$-5x_5$	$-3w_1$	$+5w_2$	$+4w_3$

(B)

1.

x_3	2			$-x_6$	$+w_3$		
x_4	6		$+x_2$	$-2x_6$	$+w_3$	$-w_4$	
x_5	4	$-x_1$			$-w_3$		$-w_5$
w_1	3	$-x_1$	$+x_2$	$+x_6$			
w_2	9	$-2x_1$	$+x_2$	$-2x_6$		$-w_4$	$-w_5$
w_6	0	$+2x_1$	$+x_2$	$-x_6$	$+2w_3$	$-w_4$	$+w_5$
ζ	-8	$-2x_1$	$-4x_2$	$+4x_6$	$-2w_3$	$+w_4$	

2. x_6 will be the entering variable because it has the largest coefficient.
3. The equation for the entering variable column a_i is $a_i = -B^{-1}Ne_i$.
4. The leaving variable with be w_6 because it constrains the entering variable the most.
5. The basic variables will be $x_3, x_4, x_5, w_1, w_2, x_6$.

(C)

x_5	1	$+x_1$			$-x_4$	$+w_2$	
x_6	4		$+x_2$	$+x_3$	$-x_4$		$-w_4$
w_1	7	$-x_1$	$+2x_2$	$+x_3$	$-x_4$		$-w_4$
w_3	2		$+x_2$	$+2x_3$	$-x_4$		$-w_4$
w_5	1	$-2x_1$	$-x_2$	$-2x_3$	$+2x_4$	$-w_2$	$+w_4$
w_6	1		$+x_2$	$+x_3$	$+x_4$	$-w_2$	$-w_4$
ζ	4	$-2x_1$	$-2x_2$		$-2x_4$		$-w_4$

(D)
This produces a new objective function: $(-2+u)x_1+(-3+u)x_2-x_3-x_4+(1+u)x_6$. To make sure the dictionary is final and feasible we have to ensure that the objective row coefficients are ≤ 0 and the constant column values are ≥ 0. Trying different values of μ I found a bound of $-1 \leq \mu \leq 0$ as values of the μ that kept the dictionary final and feasible.

(E)
We need to ensure that the constant column of the dictionary is ≥ 0 to ensure the dictionary is feasible. Trying different values of λ I found a boud of $-1 \leq \lambda \leq +1$.

```
% Code for problem 3 of homework 4.
A = [+1 -1 0 0 0 -1 +1 0 0 0 0 0;...
     +1 0 0 -1 -1 0 0 +1 0 0 0 0;...
     0 0 -1 0 0 -1 0 0 +1 0 0 0;...
     0 -1 -1 +1 0 +1 0 0 0 +1 0 0;...
     +1 0 +1 0 +1 +1 0 0 0 0 +1 0;...
     -1 0 0 -1 +1 -1 0 0 0 0 0 +1];

b = [3 -1 -2 +4 +6 -2]';

c = [-2 -3 -1 -1 0 +1 0 0 0 0 0 0];

% P3 a1
xB = [1 2 3 7 8 9];
xN = [4 5 6 10 11 12];
B = A(:, xB);
N = A(:, xN);
Binv = inv(B);
% Dictionary constant value column.
Binv*b;
% Dictionary coefficient value rows.
-Binv*N;

cB = c(:, xB);
cN = c(:, xN);
% Objective function constant value.
cB*(Binv*b);
% Object function coefficient values.
cN - cB*(Binv*N);

% P3 a2
xB = [5 6 7 8 9 10];
xN = [1 2 3 4 11 12];


B = A(:, xB);
N = A(:, xN);
Binv = inv(B);
% Dictionary constant value column.
Binv*b;
% Dictionary coefficient value rows.
-Binv*N;

cB = c(:, xB);
cN = c(:, xN);
% Objective function constant value.
cB*(Binv*b);
% Object function coefficient values.
cN - cB*(Binv*N);

% P3 a3
xB = [1 2 6 10 11 12];
xN = [3 4 5 7 8 9];

B = A(:, xB);
N = A(:, xN);
Binv = inv(B);
% Dictionary constant value column.
Binv*b;
% Dictionary coefficient value rows.
-Binv*N;

cB = c(:, xB);
cN = c(:, xN);
% Objective function constant value.
cB*(Binv*b);
% Object function coefficient values.
cN - cB*(Binv*N);

% p3 b
xB = [3 4 5 7 8 12];
xN = [1 2 6 9 10 11];

B = A(:, xB);
N = A(:, xN);
```

```
Binv = inv(B);
% Dictionary constant value column.
Binv*b;
% Dictionary coefficient value rows.
-Binv*N;

cB = c(:, xB);
cN = c(:, xN);
% Objective function constant value.
cB*(Binv*b);
% Object function coefficient values.
cN - cB*(Binv*N);

% Choose the third variable as the entering variable.
% Get the column of the entering variable.
a_i = -Binv*N(:, 3);

% Determine the leaving variable row index by finding the value that
% constraints the entering variable the most.
-(Binv*b)./a_i;

% Get the leaving variable row.
a_j = -(Binv*N);
a_j = a_j(6,:);

% P3 c
f = [-2 -3 -1 -1 0 1];
Y = [1 -1 0 0 0 -1;1 0 0 -1 -1 0;0 0 -1 0 0 -1;0 -1 -1 1 0 1;1 0 1 0 1 1;-1 0 0 -1 1 -1];
x = [3 -1 -2 4 6 -2];
options = optimoptions('linprog','Algorithm','dual-simplex');
[x,fval] = linprog(-f, Y, x, [], [], zeros(size(f)), [], options)

xB = [5 6 7 9 11 12];
xN = [1 2 3 4 8 10];

B = A(:, xB);
N = A(:, xN);
Binv = inv(B);
% Dictionary constant value column.
Binv*b
% Dictionary coefficient value rows.
-Binv*N

cB = c(:, xB);
cN = c(:, xN);
% Objective function constant value.
cB*(Binv*b)
% Object function coefficient values.
cN - cB*(Binv*N)

% P3 d
% z = (-2+u)x_1 + (-3+u)x_2 -x_3 -x_4 + (1+u)x_6
u = 0;
c = [(-2+u) (-3+u) -1 -1 0 (+1+u) 0 0 0 0 0 0];

xB = [5 6 7 9 11 12];
xN = [1 2 3 4 8 10];

B = A(:, xB);
N = A(:, xN);
Binv = inv(B);
% Dictionary constant value column.
Binv*b
% Dictionary coefficient value rows.
-Binv*N

cB = c(:, xB);
cN = c(:, xN);
% Objective function constant value.
cB*(Binv*b)

% Check if all values in constant column are >= 0
% This means the dictionary is feasible.
if all(Binv*b>=0)
    disp('Dictionary is feasible.');
```

```
else
    disp('Dictionary is not feasible.');
end

% Object function coefficient values.
cN - cB*(Binv*N)

% Check if all obejctive function coefficients are <= 0.  This means the dictionary is final.
if all(cN - cB*(Binv*N)<=0)
    disp('Dictionary is final.');
else
    disp('Dictionary is not final.');
end

% P3 e
y = 1;
b = b + (y.*[1 1 0 0 0 -1])';
c = [-2 -3 -1 -1 0 +1 0 0 0 0 0 0];

xB = [5 6 7 9 11 12];
xN = [1 2 3 4 8 10];

B = A(:, xB);
N = A(:, xN);
Binv = inv(B);
% Dictionary constant value column.
Binv*b;
% Dictionary coefficient value rows.
-Binv*N;

cB = c(:, xB);
cN = c(:, xN);
% Objective function constant value.
cB*(Binv*b);

% Check if all values in constant column are >= 0
% This means the dictionary is feasible.
Binv*b
if all(Binv*b>=0)
    disp('Dictionary is feasible.');
else
    disp('Dictionary is not feasible.');
end

% Object function coefficient values.
cN - cB*(Binv*N);
```