```matlab
%--------------------------------------------------------------------------
%
%          ADJUSTMENT THEORY I
%    Exercise 14: Adjustment Calculation
%
%   Author          : Anastasia Pasioti
%   Version         : October 12, 2018
%   Last changes    : January 31, 2023
%
%--------------------------------------------------------------------------

clc;
clearvars;
close all;
format long g;

%--------------------------------------------------------------------------
%   Task 2
%--------------------------------------------------------------------------
%--------------------------------------------------------------------------
%   Observations and initial values for the unknowns
%--------------------------------------------------------------------------
%Load files
dir = load('directions.txt');               %[gon]
coord = load('Points.txt');             %[m]     %Error-free

for i = 1:size(coord,1)
    eval(['y' num2str(coord(i,1)) '=' num2str(coord(i,2)) ';']);
    eval(['x' num2str(coord(i,1)) '=' num2str(coord(i,3)) ';']);
end

%Vector of observations
L = dir(:,3)*pi/200; %[rad]

%Number of observations
no_n = length(L);

%Initial values for the unknowns
x3 = 250;
y3 = 500;
w3 = 1;

%Vector of initial values for the unknowns
X_0 = [x3 y3 w3]';

%Number of unknowns
no_u = length(X_0);

%Number of constraints
no_b = 1;
```

```
%Redundancy
r =  no_n - no_u + no_b;

%-----------------------------------------------------------------------------
%  Stochastic model
%-----------------------------------------------------------------------------
s_dir = 0.001 * pi / 200;

%VC Matrix of the observations
S_LL = s_dir^2*eye(no_n);

%Theoretical standard deviation
sigma_0 = 0.001;     %a priori

%Cofactor matrix of the observations
Q_LL = 1/sigma_0^2*S_LL;

%Weight matrix
P = inv(Q_LL);

%-----------------------------------------------------------------------------
%  Adjustment
%-----------------------------------------------------------------------------
%break-off conditions
epsilon = 10^-5;
delta = 10^-13;
max_x_hat = Inf;
Check2 = Inf;

%Number of iterations
iteration = 0;

while max_x_hat>epsilon || Check2>delta

     %Observations as functions of the approximations for the unknowns

     L_0(1,1) = direction(y3, x3, y1, x1, w3);
     L_0(2,1) = direction(y3, x3, y2, x2, w3);
     L_0(3,1) = direction(y3, x3, y4, x4, w3);
     L_0(4,1) = direction(y3, x3, y5, x5, w3);
     L_0(5,1) = direction(y3, x3, y6, x6, w3);

     %Vector of reduced observations
     l = L - L_0;

     %Design matrix C with the elements from the Jacobian matrix J
     C = [(x3-x7)/25 (y3-y7)/25 0];

     %Design matrix with the elements from the Jacobian matrix J
```

```
A(1,1) = dr_dx_from(y3,x3,y1,x1);
A(1,2) = dr_dy_from(y3,x3,y1,x1);
A(1,3) = -1;

A(2,1) = dr_dx_from(y3,x3,y2,x2);
A(2,2) = dr_dy_from(y3,x3,y2,x2);
A(2,3) = -1;

A(3,1) = dr_dx_from(y3,x3,y4,x4);
A(3,2) = dr_dy_from(y3,x3,y4,x4);
A(3,3) = -1;

A(4,1) = dr_dx_from(y3,x3,y5,x5);
A(4,2) = dr_dy_from(y3,x3,y5,x5);
A(4,3) = -1;

A(5,1) = dr_dx_from(y3,x3,y6,x6);
A(5,2) = dr_dy_from(y3,x3,y6,x6);
A(5,3) = -1;
%Normal matrix
N = A' * P * A;

N_ext = [N C'; C 0];

%Vector of right hand side of normal equations
n = A' * P * l;

w = distance(y3,x3,y7,x7) - 25.0;

n_ext = [n; -w];

%Inversion of normal matrix / Cofactor matrix of the unknowns
Q = inv(N_ext);
Q_xx = Q(1:no_u,1:no_u);

%Solution of the normal equations
x_solution = Q * n_ext;
x_hat = x_solution(1:no_u);

%Update
X_hat = X_0 + x_hat;
X_0 = X_hat;

x3 = X_hat(1);
y3 = X_hat(2);
w3 = X_hat(3);

k = x_solution(end);
```

```
         %Check 1
        max_x_hat = max(abs(x_hat));

        %Vector of residuals
        v = A * x_hat - l;

        %Vector of adjusted observations
        L_hat = L + v;

        %Objective function
        vTPv = v' * P * v;

        %Functional relationships without the observations

        phi_X_hat(1,1) = direction(y3, x3, y1, x1, w3);
        phi_X_hat(2,1) = direction(y3, x3, y2, x2, w3);
        phi_X_hat(3,1) = direction(y3, x3, y4, x4, w3);
        phi_X_hat(4,1) = direction(y3, x3, y5, x5, w3);
        phi_X_hat(5,1) = direction(y3, x3, y6, x6, w3);

        %Check 2
        Check2 = max(abs(L_hat - phi_X_hat));

        %Update number of iterations
         iteration = iteration+1;

end

if Check2<=delta
    disp('Everything is fine!')
else
    disp('Something is wrong.')
end
```

```
Everything is fine!
```

```
%Empirical reference standard deviation
s_0 = sqrt(vTPv/r);

%VC matrix of adjusted unknowns
S_XX_hat = s_0^2*Q_xx;

%Standard deviation of the adjusted unknowns
s_X = sqrt(diag(S_XX_hat));        %[m]

%Cofactor matrix of adjusted observations
Q_LL_hat = A*Q_xx*A';

%VC matrix of adjusted observations
```

```
S_LL_hat = s_0^2*Q_LL_hat;

%Standard deviation of the adjusted observations
s_L_hat = sqrt(diag(S_LL_hat));

%Cofactor matrix of the residuals
Q_vv = Q_LL-Q_LL_hat;

%VC matrix of residuals
S_vv = s_0^2*Q_vv;

%Standard deviation of the residuals
s_v = sqrt(diag(S_vv));

k
```

```
k =
    -0.000175322880108143
```

```
table(X_hat, s_X, 'RowNames', {'x3' 'y3' 'w3'})
```

ans = 3×2 table

	X_hat	s_X
1 x3	242.860035744546	0.00209077738156555
2 y3	493.700954509788	0.0064195533452118
3 w3	-2.07215393344185	1.74689257387742e-05

```
table(L, v, L_hat, s_v, s_L_hat)
```

ans = 5×5 table

	L	v	L_hat	s_v	s_L_hat
1	3.25012525499336	1.71629755210726e-05	3.25014241796888	1.8066377433144e-05	1.19092953165416e-05
2	0.73046270346045	-5.61422177995341e-06	0.73045708923867	1.01412274417518e-05	1.91149369464844e-05
3	1.32959898001921	1.78048551851497e-06	1.32960076050473	1.86234099530095e-05	1.10178904612253e-05
4	1.81466402732613	1.55455612817252e-05	1.81467957288741	1.86773997083038e-05	1.09261177290777e-05
5	2.44398786769324	-2.88748005413593e-05	2.4439589928927	1.67260509313012e-05	1.37282383697592e-05