```matlab
%-------------------------------------------------------------------------
%
%          ADJUSTMENT THEORY I
%  Exercise 10: Adjustment Calculation - part V
%
%   Author         : Anastasia Pasioti
%   Version        : October 11, 2018
%   Last changes   : January 18, 2023
%
%-------------------------------------------------------------------------
% Group 6
%
% Fahd Sbahi / 0501211
% Nils Lang / 0494447
% Felix Nöding / 0501313
% Egor Grekhov / 0501111
% Corban Rosenauer / 0494441
% Zilong Liu / 0501297
%-------------------------------------------------------------------------

clc;
clear vars;
close all;
format shortG;

%-------------------------------------------------------------------------
%   Task 2.1 - HA as a benchmark
%-------------------------------------------------------------------------
%-------------------------------------------------------------------------
%   Observations and redundancy
%-------------------------------------------------------------------------
%Height differences/observations
L = [10.509 5.360 -8.523 -7.348 -3.167 15.881]';   %[m]
s_L = [0.006 0.004 0.005 0.003 0.004 0.012]';      %[m]

%Benchmarks - error free
HA = 501.111;          %[m]

%Vector of observations with benchmarks
benchmark = [HA 0 0 -HA 0 HA]';
L_dash = L + benchmark;

%Number of observations
no_n = length(L);

%Number of unknowns
no_u = 3;

%Redundancy
r = no_n - no_u;
```

```
%-------------------------------------------------------------------------
%  Stochastic model
%-------------------------------------------------------------------------
%VC Matrix of the observations
S_LL = diag(s_L.^2);

%Theoretical standard deviation
sigma_0 = 1;     %a priori

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

%Weight matrix
P = inv(Q_LL);

%-------------------------------------------------------------------------
%  Adjustment
%-------------------------------------------------------------------------
%Design matrix
A = [1 0 0;
    -1 1 0;
    0 -1 1;
    0 0 -1;
    -1 0 1;
    0 1 0];

%Normal matrix
N = A' * P * A;

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

%Inversion of normal matrix / Cofactor matrix of the unknowns
Q_XX_hat = inv(N);

%Solution of normal equation
X_hat = Q_XX_hat * n;

%Estimated unknown parameters

HB = X_hat(1)     %for point B
```

```
HB =
      511.62
```

```
HC = X_hat(2)     %for point C
```

```
HC =
      516.98
```

```
HD = X_hat(3)     %for point D
```

```
HD =
      508.46
```

```

%Vector of residuals
v = A*X_hat - L_dash;

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

%Vector of adjusted observations
L_hat = L + v;

%Final check    (to check for computational errors)
Phi_X_hat = A*X_hat - benchmark;
if L_hat-Phi_X_hat<1e-12
 disp('Everything is fine!')
else
 disp('Something is wrong.')
end
```

```
Everything is fine!
```

```

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

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

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

%Cofactor matrix of adjusted observations
Q_LL_hat = A * Q_XX_hat * 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));

%--------------------------------------------------------------------------
%   Task 2.2 - HB as a benchmark
%--------------------------------------------------------------------------
%--------------------------------------------------------------------------
%   Observations and redundancy
%--------------------------------------------------------------------------
%Height differences/observations
L = [10.509 5.360 -8.523 -7.348 -3.167 15.881]';   %[m]
s_L = [0.006 0.004 0.005 0.003 0.004 0.012]';      %[m]

%Benchmarks - error free
HB = 111.105;             %[m]

%Vector of observations with benchmarks
benchmark = [-HB HB 0 0 HB 0]';
L_dash = L + benchmark;

%Number of observations
no_n = length(L);

%Number of unknowns
no_u = 3;

%Redundancy
r = no_n - no_u;

%--------------------------------------------------------------------------
%  Stochastic model
%--------------------------------------------------------------------------
%VC Matrix of the observations
S_LL = diag(s_L.^2);

%Theoretical standard deviation
sigma_0 = 1;     %a priori

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

%Weight matrix
P = inv(Q_LL);

%--------------------------------------------------------------------------
%  Adjustment
%--------------------------------------------------------------------------
%Design matrix
A = [-1 0 0;
    0 1 0;
    0 -1 1;
```

```
    1 0 -1;
    0 0 1;
    -1 1 0];

%Normal matrix
N = A' * P * A;

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

%Inversion of normal matrix / Cofactor matrix of the unknowns
Q_XX_hat = inv(N);

%Solution of normal equation
X_hat_2 = Q_XX_hat * n;

%Estimated unknown parameters

HA = X_hat_2(1)     %for point A
```

```
HA =
      100.59
```

```
HC = X_hat_2(2)     %for point C
```

```
HC =
      116.46
```

```
HD = X_hat_2(3)     %for point D
```

```
HD =
      107.94
```

```

%Vector of residuals
v_2 = A*X_hat_2 - L_dash;

%Objective function
vTPv = v_2' * P * v_2;

%Vector of adjusted observations
L_hat_2 = L + v_2;

%Final check    (to check for computational errors)
Phi_X_hat = A*X_hat_2 - benchmark;
if L_hat_2-Phi_X_hat<1e-12
 disp('Everything is fine!')
else
 disp('Something is wrong.')
end
```

```
Everything is fine!
```

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

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

%Standard deviation of the adjusted unknowns
s_X_2 = sqrt(diag(S_XX_hat));

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

%VC matrix of adjusted observations
S_LL_hat = s_0^2 * Q_LL_hat;

%Standard deviation of the adjusted observations
s_L_hat_2 = 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_2 = sqrt(diag(S_vv));
%Comparison

table(X_hat, X_hat_2, s_X, s_X_2, 'RowNames',{'HB/HA' 'HC' 'HD'})
```

ans = 3×4 table

	X_hat	X_hat_2	s_X	s_X_2
1 HB/HA	511.62	100.59	0.0022953	0.0022953
2 HC	516.98	116.46	0.0026363	0.0021329
3 HD	508.46	107.94	0.0017607	0.001962

Conclusion: the distrubution of the standard deviations changed with the different benchmark. Therefore, SDs depend on the chosen benchmark value.

First values in each SD column reprsent SD of HB and HA respectively, and these values are equal, because they are derived from the same observation (HA to HB).

HC w.r.t. HA has the largest SD because it has the largest corresponding observation value in this configuration.

```
table(L, v, v_2, L_hat, L_hat_2)
```

ans = 6×5 table

	L	v	v_2	L_hat	L_hat_2
1	10.509	0.0037117	0.0037117	10.513	10.513
2	5.36	-0.00024395	-0.00024395	5.3598	5.3598
3	-8.523	-0.0018625	-0.0018625	-8.5249	-8.5249
4	-7.348	0.00039467	0.00039467	-7.3476	-7.3476
5	-3.167	0.0018936	0.0018936	-3.1651	-3.1651
6	15.881	-0.0085322	-0.0085322	15.872	15.872

Conclusion: the adjustment results are the same regardless of the chosen benchmark. That means that residuals and adjusted observations values depend only on the configuration of the network.

```
table(s_v, s_v_2, s_L_hat, s_L_hat_2)
```

ans = 6×4 table

	s_v	s_v_2	s_L_hat	s_L_hat_2
1	0.0031618	0.0031618	0.0022953	0.0022953
2	0.0014951	0.0014951	0.0021329	0.0021329
3	0.0023233	0.0023233	0.0022811	0.0022811
4	0.00084638	0.00084638	0.0017607	0.0017607
5	0.0017132	0.0017132	0.001962	0.001962
6	0.0073561	0.0073561	0.0026363	0.0026363

Conclusion: SDs of the residuals and adjusted observations are the same regardless of the chosen benchmark, which means that thier values depend only on the configuration of the network.