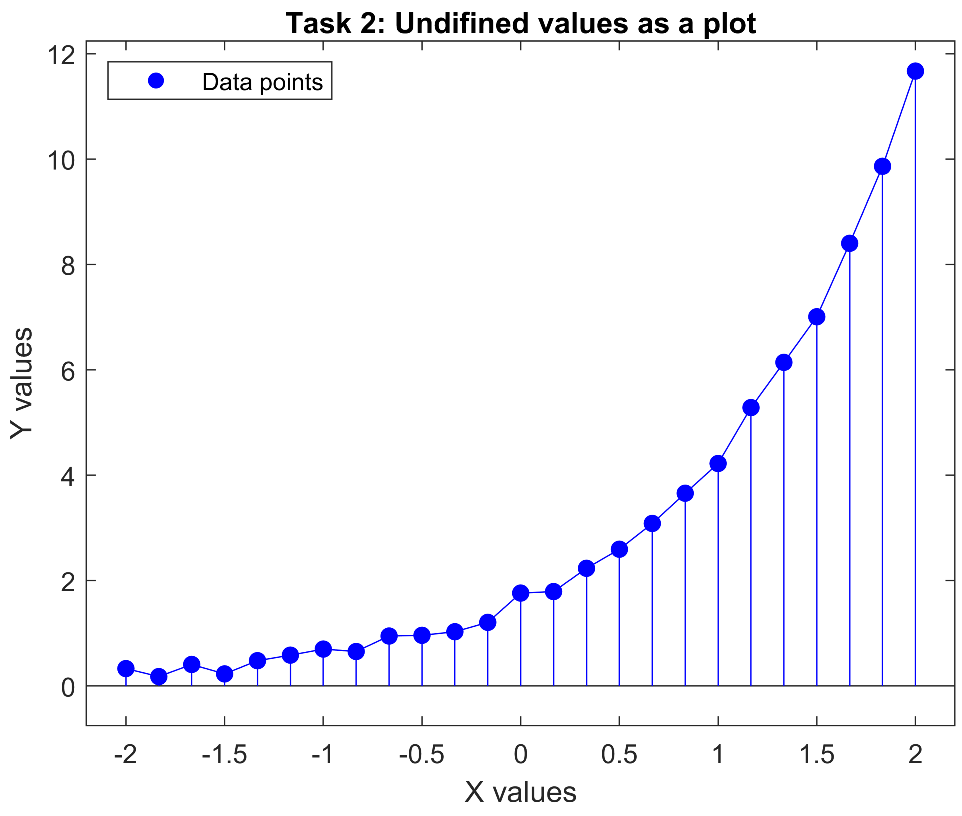
ADJUSTMENT THEORY I

Exercise 9: Adjustment Calculation - part IV

Author: Group G

Date: 19.01.2022

```
clc;
clear all;
close all;
format default;

%-------------------------------------------------------------------------
%   Task 2
%-------------------------------------------------------------------------
%-------------------------------------------------------------------------
%   Observations and initial values for the unknowns
%-------------------------------------------------------------------------
%Load data
data = load("Exercise9Task2.txt");

%Plot the data
x = data(:,1);
y = data(:,2);

plot(x,y,'-b')
hold on;
scatter(x,y,'b','filled')
Buffer = 5; % in pecrent
xlim([min(x)-(max(x)- min(x)) * Buffer / 100, ...
    max(x)+(max(x)- min(x)) * Buffer / 100]);
ylim([-min(y)-(max(y)- min(y)) * Buffer / 100, ...
    max(y)+(max(y)- min(y)) * Buffer / 100]);
stem(x,y,'b')
hold off;
xlabel('X values');
ylabel('Y values');
title('Task 2: Undifined values as a plot');
legend_cell = {'','Data points'};
legend(legend_cell,'Location','northwest')
```



Appropriate functional model:

$$Y = ae^{\mathrm{bx}_i} + c \quad \rightarrow \varphi_i$$

Discription of the data:

Our observation are the undifined Y value from the data Table.

We also have an error-free observation of the density of the undifined X value.

The unknow parameter we want the calculate are a, b and c.

Linear or non-linear?

To determine if the problem is a linear or non-linear adjustment problem we have to look at the unkowns we want to calculate. Only they show if we need to solve the problem linear or non-linear. The other variables dosen't matter for this classification, because they are not the unkowns we want to determine.

In this case we see that our unknow b dosen't scale linear in the equation $\ldots e^{\mathrm{bx}_i}\ldots$, but exponential. That indicates that we need a non-linear solution to solve the problem. Even though the other to unknown parameters a and c scale linear, the adjustment problem is still non-linear because at least one paramter does not scale linear.

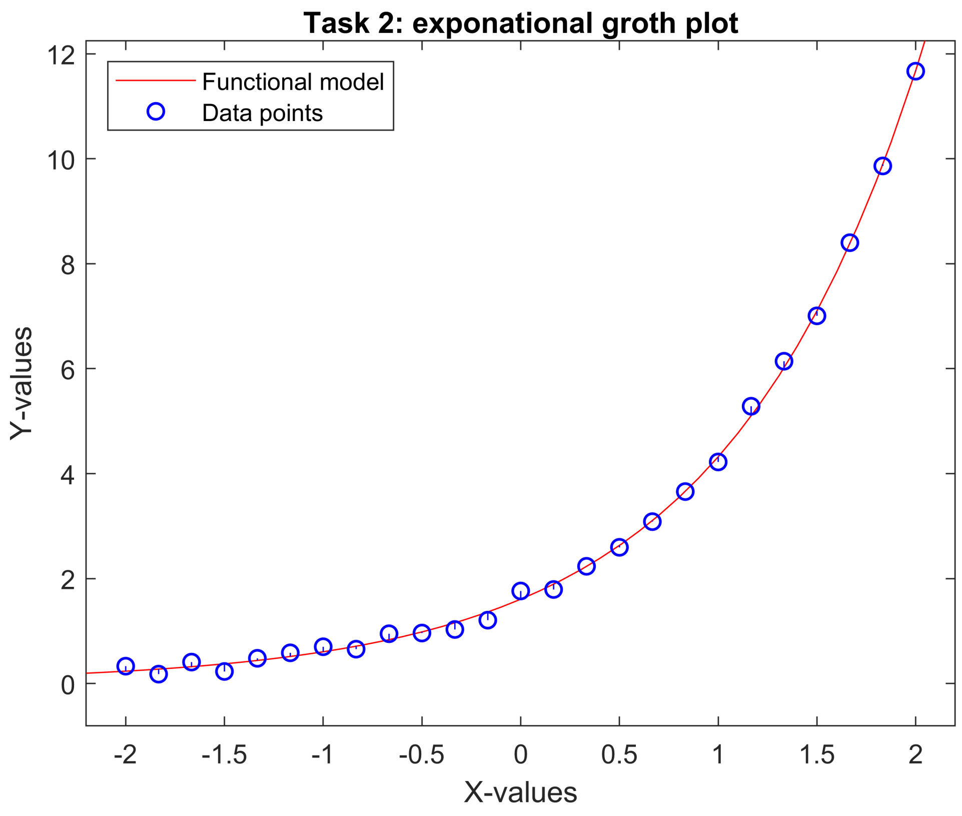
```
%Error-free values
%x = x;

%Vector of observations
L = y;
```

```
%Number of observations
no_n = length(y);

%Initial values for the unknowns
a = 1.7637;
b = log(2);
c = 0;

%Vector of initial values for the unknowns
X_0 = [a,b,c]';

%Number of unknowns
no_u = length(X_0);

%Redundancy
r = no_n - no_u;

%--------------------------------------------------------------------------
%  Stochastic model
%--------------------------------------------------------------------------
%VC Matrix of the observations
s_y = 0.15;  %[m]
S_LL = eye(no_n) * s_y^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
%--------------------------------------------------------------------------
%break-off conditions
epsilon = 10^-8;
delta = 10^-12;
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 = a*exp(b*x) + c;

     %Vector of reduced observations
     l = L - L_0;
```

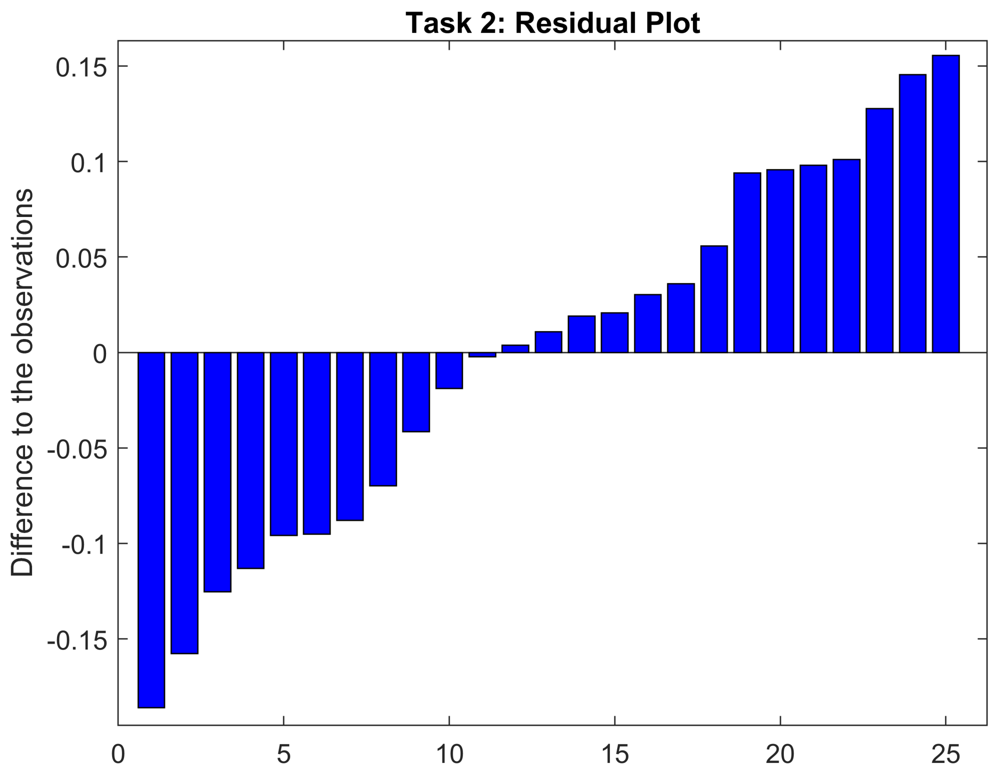
```
    %Design matrix with the elements from the Jacobian matrix J
    A(:,1) = exp(b*x);
    A(:,2) = a*x .* exp(b*x);
    A(:,3) = 1;

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

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

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

    %Solution of the normal equations
    x_hat = Q_xx * n;

    %Update
    X_0 = X_0 + x_hat;

    a = X_0(1);
    b = X_0(2);
    c = X_0(3);

    %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
    Psy_X_hat = a*exp(b*x) + c;

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

%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));

%--------------------------------------------------------------------------
%  Plot
%--------------------------------------------------------------------------

%Residual Plot

Fu = @(x) a*exp(b*x) + c;
fplot(Fu,'red')
xlabel('X-values');
ylabel('Y-values');
title('Task 2: exponational groth plot');
Buffer = 5; % in pecrent
xlim([min(x)-(max(x)- min(x)) * Buffer / 100, ...
    max(x)+(max(x)- min(x)) * Buffer / 100]);
ylim([-min(L_0)-(max(L_0)- min(L_0)) * Buffer / 100, ...
    max(L_0)+(max(L_0)- min(L_0)) * Buffer / 100]);
hold on;
plot(x,Psy_X_hat-v,'bo','linewidth',1);
for i = 1:no_n

    plot([x(i),x(i)],[Psy_X_hat(i),Psy_X_hat(i)-v(i)],'blue')

end
legend_cell = {'Functional model','Data points'};
legend(legend_cell,'Location','northwest')
hold off;
```



```
bar(sort(v),'blue')

Buffer = 5; % in pecrent
xlim([0 + 0 * Buffer / 100, ...
    length(v) + length(v) * Buffer / 100]);
ylim([min(sort(v)) + min(sort(v)) * Buffer / 100, ...
    max(sort(v)) + max(sort(v)) * Buffer / 100]);
title('Task 2: Residual Plot');
ylabel('Difference to the observations');
hold off;
```



The Residuals look fine. Looks like there is no blunder in the data. The resudials are also no well normaly, but instead more linearly distributed. But we whould need more residuals to define the distribution more accurate.

```
%Tables
table(x,L,L_hat,s_L_hat,v,s_v,'VariableNames',...
{'X-Value','L (Y)'...
'L_hat',...
's_L_hat',...
'v',...
's_v'})
```

```
ans = 25×6 table
```

	X-Value	L (Y)	L_hat	s_L_hat	v	s_v
1	-2	0.3307	0.2356	0.0374	-0.0951	0.0943
2	-1.8333	0.1790	0.2746	0.0361	0.0956	0.0949
3	-1.6667	0.4087	0.3207	0.0346	-0.0880	0.0954
4	-1.5000	0.2297	0.3751	0.0330	0.1454	0.0960
5	-1.3333	0.4808	0.4394	0.0313	-0.0414	0.0965
6	-1.1667	0.5851	0.5152	0.0296	-0.0699	0.0971
7	-1	0.7006	0.6048	0.0280	-0.0958	0.0976
8	-0.8333	0.6549	0.7106	0.0264	0.0557	0.0980
9	-0.6667	0.9487	0.8356	0.0251	-0.1131	0.0983

	X-Value	L (Y)	L_hat	s_L_hat	v	s_v
10	-0.5000	0.9623	0.9831	0.0241	0.0208	0.0986
11	-0.3333	1.0297	1.1573	0.0236	0.1276	0.0987
12	-0.1667	1.2075	1.3630	0.0238	0.1555	0.0986
13	0	1.7637	1.6060	0.0247	-0.1577	0.0984
14	0.1667	1.7918	1.8928	0.0262	0.1010	0.0980
15	0.3333	2.2338	2.2316	0.0282	-0.0022	0.0975
16	0.5000	2.5956	2.6316	0.0304	0.0360	0.0968
17	0.6667	3.0849	3.1039	0.0325	0.0190	0.0961
18	0.8333	3.6579	3.6617	0.0342	0.0038	0.0955
19	1	4.2224	4.3203	0.0353	0.0979	0.0952
20	1.1667	5.2843	5.0983	0.0355	-0.1860	0.0951
21	1.3333	6.1417	6.0163	0.0352	-0.1254	0.0952
22	1.5000	7.0071	7.1011	0.0355	0.0940	0.0951
23	1.6667	8.4009	8.3820	0.0396	-0.0189	0.0934
24	1.8333	9.8634	9.8937	0.0520	0.0303	0.0872
25	2	11.6690	11.6799	0.0756	0.0109	0.0676

```
table([1.7637,log(2),0]',X_0,s_X,'VariableNames',...
{'X',...
'X_hat',...
's_X_hat'},...
'RowNames',{'a','b','c'})
```

```
ans = 3×3 table
```

	X	X_hat	s_X_hat
1 a	1.7637	1.5861	0.0520
2 b	0.6931	0.9974	0.0164
3 c	0	0.0198	0.0488