**Two models of Matching Power Supply to Demand using Newton’s Method**

EMTH171

Case study 1

Jiyao Zhu email: jzh200@uclive.ac.nz

Menghao Zhan email: mzh99@uclive.ac.nz

19 September 2019

**Abstract**

This case study aims to solve two problems of matching power supply to demand with the proper use of Newton’s method. The mathematics program MATLAB is applied to sketch the plots of different variables and find the acceptable solutions.

**Exercise 1**

*Problem statement*

In this problem, a sled carrying a load is hauled up an inclined rail, pulled by the cable that is drawn by a pulley (**Figure 1**). An electric motor provides the power supply to the pulley, through a gearbox.

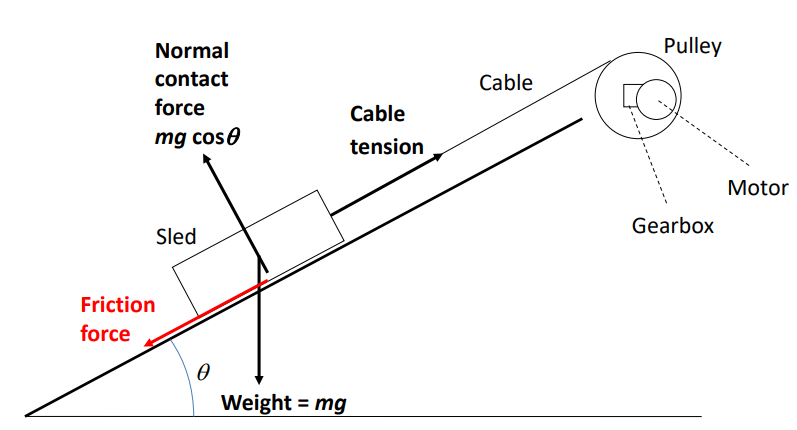
The motor power varies with motor speed as a parabola:

(1)

where , are constants with values and 314.16 rad/srespectively, and (rpm) represents the motor speed . In this case, it can be decided that the peak power output is just under 25kW at 1500 rpm.

The motor drives a gearbox with a 20:1 ratio, which means for every 20 turns of the motor shaft, the gearbox output shaft turns once. This ratio is expressed by the symbol . Given that the output shaft of the gearbox drives a winch with a drum radius of R = 0.5 m, speed of the cable V(m/s) can be determined as

(2)



**Figure1: Sled haulage system**

The total mass of the sled and its load is m = 1000 kg, and the angle of the rail to the horizontal is θ = 45°. From the figure shown above, the power demand involves that caused by both weight and friction, which are:

(3)

(4)

where is the dimensionless coefficient of friction.

Therefore, to match the power supply to demand,

(5)

an equation can be expressed as

(6)

By substituting equation (2),

(7)

To solve this problem, an appropriate speed V needs to be found to ensure power demand matches the power output of the motor, at which the sled travels at constant speed.

*Method*

Given that there is only one unknown in equation (7), the whole problem is then transferred into finding the root of the function f(V):

(8)

This is the typical problem that can be addressed by Newton’s Method. The first step is to sketch the plot of the function f(V) against V. From the plot, an initial guess of the root can be found. It is also a sound choice to take the intersection of the plot of and as initial guess.

Then by iterating from the initial guess, an improved approximate root is then found by

(9)

until the approximation is below the tolerance.

In this exercise, the derivative of the function is

(10)

*Plot of motor power and power demand*



**Figure2: Total power demand against**

*Plot of velocity versus iteration (convergence plot)*

**

**Figure3: The iterations with initial guess v=2m/s**

The iterations converge to V=2.651m/s.

*Discussion*

In figure 2, in fact two intersections are found, which means there are two solutions to this problem. One of them is finally determined as converging to 2.651m/s while another solution is V=0. When V=0, the sled along with the load inside is stationary at the bottom of the inclined rail, therefore both and equal to 0 in this case. When V=2.651m/s, the sled is pulled at a constant speed and .However, the only solution that is needed is 2.651m/s.

*Conclusion*

Apart from the case that the sled stays still at the bottom, only when V=2.651m/s can the sled travels at a constant speed

**Exercise 2**

*Problem statement*

This model focus on the power and speed of a car with petrol engine. The power output of a petrol engine is decided as follows,

(11)

with Engine output parameters of α = 420 W.s/rad and β = 0.440 W.. In this equation (rad/s) represents the engine speed, while wheel rational speed is:

(12)

Then the road speed V is given by

(13)

where and are typical gear ratios, R=0.205m is the wheel radius.

Finally the power output of engine can be expressed as

(14)

in which and K is a constant

(15)

The power demand in this model comes from four sources, which are:

• Power dissipated by aerodynamic drag:

(16)

where is the drag coefficient, is the frontal area and ρ is the density of the fluid.

• Power consumed by climbing:

*(17)*

where is the angle of the road to the horizontal and m = 1500kg is the mass.

• Power dissipated by rolling resistance:

(18)

in which is a coefficient decided by the wheel and ground properties.

• Power required to accelerate:

(19)

where is the acceleration.

Ultimately, the total power demand can be determined as

(20)

Similarly, in this model a balance between power demand and output is still needed, which should be considered in four different cases respectively:

1. 2. 3. 4.

In each case the proper speed V should be solved with the help of MATLAB script.

*Method*

Similarly, Newton’s method is applied in this model, which starts with sketching the plots of power supply and demand in each case to find an appropriate where two lines intersect. After iterating several times by equation (9) when the approximation is below tolerance, the final root can be determined. What is important in this model is that if there are more than 1 intersection between the plot of demand and output, each one should have a corresponding initial guess as well as a final solution.

*Key equations*

The initial equations need to be adjusted to the forms that Newton’s method can work. To solve this problem, the balance between power demand and output should be rearranged as

(21)

The derivative of the function is

(22)

*Plot the engine output power as a function of engine speed*

**

**Figure4: Power output of a petrol engine as a function of speed**

*Plot of case 1*

**

*Plot of case 2*

**

*Plot of case 3*

**

*Plot of case 4*

**

**Figures5, 6, 7, 8: All types of power supply and demand in four cases**

*Discussion*

An appropriate initial guess can make a difference when solving problems with Newton’s method. As is shown in the tables, on one hand, if the initial guess is too far away from the acceptable solution, it would take too many times of iterations. On the other hand, given that an initial guess can only correspond to one of the roots of the equation, if an inappropriate initial was chosen, it may finally lead to the solution that is not expected. For example in case 1 and 2, the initial guess V=10 would only converge to the solution V = 0, but that is not the constant speed people seek. Hence, such kind of initial guess is perceived as a wrong choice.

**Tables1, 2, 3, 4: Number of iterations of different initial guesses**

|  |  |
| --- | --- |
| Case1 | Number of Iterations |
| 10 | Wrong |
| 30 | 9 |
| 40 | 5 |
| 50 | 4 |
| 55 | 3 |

|  |  |
| --- | --- |
| Case2 | Number of Iterations |
| 10 | Wrong |
| 40 | 5 |
| 30 | 4 |
| 35 | 4 |

|  |  |
| --- | --- |
| Case3 | Number of Iterations |
| 35 | 11 |
| 100 | 5 |
| 65 | 3 |

|  |  |
| --- | --- |
| Case3 | Number of Iterations |
| 65 | 5 |
| 45 | 4 |

*Conclusions*

The solutions along with number of iterations in 4 cases are shown below.

**Table5: The solutions and the numbers of iterations in four cases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case | Gradient  (°) | Acceleration  (m/s2) | Initial Guess (m/s) | Solution  (m/s) | Numbers of Iterations |
| 1 | 0 | 0 | 55 | 54.8832 | 3 |
| 2 | 10.° | 0 | 35 | 32.3904 | 4 |
| 3 | -5° | 0 | 70 | 65.1241 | 3 |
| 4 | 0 | 1 | 45 | 42.0609 | 4 |

**Appendix Code**

% EMTH171

% Case\_Study1: Exercise 1: hauling a sled up incline with cable

% mzh99/jzh200 | Jiyao Zhu & Menghao Zhan

clear

clc

close all

% Known values

m = 1000; % Sled mass (Kg)

r\_g = 20; % Gear radius (m)

r\_p = 0.5; % Pulley radius (m)

ang = pi/4;% The angle of slope

f = 0.2; % Coefficent of freaction

g = 9.81; % Gravity (m/s^2)

b = 314.16;% rad/s

%--------------------------------Function----------------------------

p\_o = @(v) (b\*(r\_g\*v)/r\_p)-((r\_g.^2\*v.^2)/(r\_p.^2));

p\_d = @(v) m\*g\*sin(ang)\*v + f\*m\*g\*cos(ang)\*v;

%---------------------------------Values-----------------------------

vMin = 0;

vMax = 1000;

vstep = 0.001;

xarray = vMin : vstep : vMax;

%---------------------------------Plotting---------------------------

figure(2)

plot(xarray,p\_o(xarray), 'DisplayName', 'Motor output/W');

hold on

plot(xarray,p\_d(xarray), 'DisplayName', 'Total power demand/W');

hold off

title('Exercise 1')

xlabel('Sled speed/m/s')

ylabel('Power/W')

xlim([0,12])

legend

% EMTH171

% Case\_Study1: Exercise 1: hauling a sled up incline with cable

% mzh99/jzh200 | Jiyao Zhu & Menghao Zhan

p\_o = @(v) (b\*(r\_g\*v)/r\_p)-((r\_g.^2\*v.^2)/(r\_p.^2));

p\_d = @(v) m\*g\*sin(ang)\*v + f\*m\*g\*cos(ang)\*v;

d\_o = @(v) (b\*r\_g/r\_p)-((2\*r\_g.^2\*v)/(r\_p.^2));

d\_d = @(v) m\*g\*sin(ang) + f\*m\*g\*cos(ang);

%------------------------------Interation----------------------------

v = 2; %Iniitial guess

N = 5; %Guess times

store = [];

store(1) = v ;

for ii = 1 : N

v = v -((p\_d(v)-p\_o(v))/((d\_d(v)-d\_o(v))));

store(ii+1) = v;

end

%---------------------------------Plotting---------------------------

figure(2)

yarray = store;

xarray = [0:N];

plot(xarray,yarray,'-or');

title('Convergence plot: V versus number of iterations')

xlabel('Number of iterations')

ylabel('V(m/s)')

%EMTH171

%Case\_Study1: Exercise 1: hauling a sled up incline with cable

clear

clc

close all

% Motor out power(W)/ Sharf speed (RPM)

% Known values

a = 420;

b = 0.44;

r\_f = 3.50;

r\_g = 0.8;

r\_w = 0.205;

%--------------------------------Function----------------------------

p\_o = @(w) (a\*(w/(30/pi)))-(b\*(w/(30/pi)).^2);

%---------------------------------Values-----------------------------

vMin = 0;

vMax = 7000;

vstep = 1;

xarray = vMin : vstep : vMax;

%---------------------------------Plotting---------------------------

figure(1)

yarray = p\_o(xarray);

plot(xarray,yarray/1000);

title('Mathrmatical model of engine power supply')

xlabel('Engine speed/rpm')

ylabel('Power output/kW')

xlim([0,8000])

% EMTH171

% Case\_Study1: Exercise 2:

% Analyzing a car of petrol power supply system with several factors

% mzh99/jzh200 | Jiyao Zhu & Menghao Zhan

clear

clc

close all

% Motor out power(W)/ Sharf speed (RPM)

% Known values

m = 1500; % Mass of car (Kg)

c\_d = 0.3; % Drag coefficient

area = 2; % Frontal area (m^2)

c\_rr = 0.010;% Rolling resistance

a = 420; % Engine parameters a (W.s/rad)

b = 0.44; % Engine parameters b (W.s 2 /rad 2)

r\_f = 3.50; % Final drive ratio (m)

r\_g = 0.8; % Top gear with a gearbox ratio (m)

r\_w = 0.205; % Wheel radius (m)

g = 9.81; % Gravity coefficient (m/s^2)

air = 1.2; % Air density (Kg/m^3)

% ===================Changeable Values in 4 Cases====================

angle = 0; % Gradient (degree)

acc = 1; % Acceleration (m/s^2)

K = r\_w/(r\_f\*r\_g); % K

%--------------------------------Function----------------------------

% Power demand from drag resistance

p\_d = @(v) c\_d.\*area.\*(1/2).\*v.^3\*air;

% Power demand from rolling resistance

p\_rr = @(v) c\_rr.\*m\*g.\*cos(angle).\*v;

% Power demand from gradient

p\_r = @(v) m.\*g.\*sin(angle).\*v;

% Power demand from acceleration

p\_a = @(v) m.\*acc.\*v;

% Engine power

p\_e = @(v) ((a\*v)/ K) - (b\*v.^2)/K.^2;

% Total power demand

p\_t = @(v) p\_d(v) + p\_r(v) + p\_rr(v) + p\_a(v)

%---------------------------------Values-----------------------------

vMin = 0;

vMax = 100;

vstep = 0.01;

xarray = vMin : vstep : vMax;

%---------------------------------Plotting---------------------------

figure(1)

plot(xarray, p\_d(xarray), 'DisplayName', 'Drag')

title('Road car power')

ylim([-8e4 150000]);

xlabel('Petrol powered car speed m/s')

ylabel('Power W')

hold on

plot(xarray, p\_t(xarray), 'DisplayName', 'Total power demand')

plot(xarray, p\_e(xarray), 'DisplayName', 'Engine power')

plot(xarray, p\_rr(xarray), 'DisplayName', 'Rolling resistance')

hold off

legend

% ==================Changeable Values in 4 Cases=====================

angle = 18\*pi ; % Gradient (degree)

acc = 0; % Acceleration (m/s^2)

K = r\_w/(r\_f\*r\_g); % K

%--------------------------------Function----------------------------

f = @(v) c\_d\*area\*(1/2)\*air\*v.^3 + m\*g\*sin(angle).\*v + ...

c\_rr\*m\*g\*cos(angle).\*v + m\*acc.\*v - (a\*v./K) + ((b\*v.^2)/K^2);

d = @(v) 3\*c\_d\*area\*(1/2)\*air\*v.^2 + m\*g\*sin(angle) + ...

c\_rr\*m\*g\*cos(angle) + m\*acc - a/K + ((2\*b\*v)./K^2);

%---------------------------------Values-----------------------------

N = 1000; % 100 Iterations

v = 35 ; % Initial guess

tol = 1e-4; % Tolerance value

%-----------------------------Processing-----------------------------

for ii = 1 : N

v(ii+1) = v(ii) - f(v(ii))/d(v(ii));

if abs(v(ii+1)-v(ii)) < tol

break

end

end

%---------------------------------Plotting---------------------------

xArray = 0 : 1 : ii; % X-axis

yArray = v % Y-axis

figure(1)

plot(xArray, yArray,'-p')

ylabel('Velocity m/s')

xlabel('Number of Iterations')