Gear-Shift Schedule for Electric Vehicles with Multi-Speed Transmissions

TRCIM20180115-1 January, 2018

Yuhanes Dedy Setiawan Liauw and Jorge Angeles

McGill
Centre for
Intelligent Machines

Centre for Intelligent Machines, Department of Mechanical Engineering
McGill University, Montréal, Canada

Abstract

In this report we propose a gear-shift schedule for electric vehicles with multi-speed transmissions. A case study is provided whereby the gear-shift schedule for the GM EV1 with a two-speed novel modular transmission is devised. First, the efficiency map of the electric vehicle is generated. A number of ways are described with a greater detail given to a map generation using MATLAB/Simulink. Second, the gear-shift time is determined based on an established procedure. Finally, the gear-shift schedule is generated and explained in depth. Detailed procedures are explained along with the code description. Several technical problems are documented together with the solutions.

Keywords: Gear-shift schedule; Electric Vehicles; Multi-speed transmissions

1 Introduction

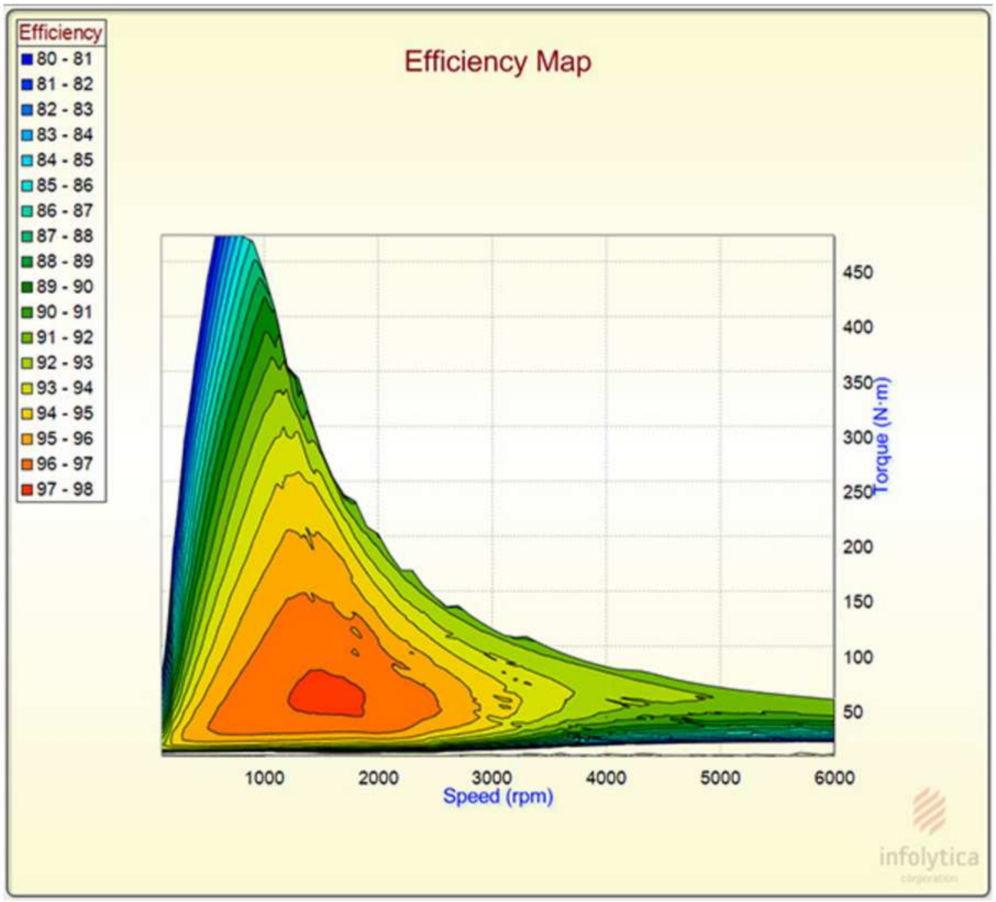
Vehicles with automatic transmissions perform gear-shifting according to a gear-shift schedule, which determines when the gear should be shifted to achieve optimal performance with minimum energy consumption. A recent development in electric vehicles includes multi-speed transmissions, where a gear-shift schedule needs to be devised for electric vehicles with automatic transmissions. However, the schedule-devising procedures for conventional vehicles are not applicable to electric vehicles because electric motors and engines have different efficiency maps. A procedure to devise gear-shift schedules for electric vehicles with multi-speed transmissions was proposed by Zhu et al. [1]. In this report, the procedure is utilized to devise a gear-shift schedule for the GM EV1 with a two-speed transmission. The practical details on performing the procedure using MATLAB/Simulink are reported along with the code description. In addition, several technical problems are documented together with the solutions.

2 The Gear-shift Schedule

A gear-shift schedule is designed for the GM EV1 with a two-speed transmission in this report. The details of the generation of the gear-shifting schedule are available in a paper by Zhu et al. [1]. To begin with, the efficiency map of the EV motor was reproduced to determine the characteristics of the motor for the Simplified Federal Urban Drive Schedule (SFUDS) driving cycle. The efficiency map is accessible either from the motor manufacturer or from commercial motor software, such as MotorSolve, from Infolytica. This package has the capability to generate the efficiency map of a motor under design, as exemplified in Fig. 1. Although MotorSolve can yield an accurate efficiency map, extensive details of the motor design that may not be available for public are mandatory to generate the map. Alternatively, the efficiency map can be produced by means of Larminie and Lowry's efficiency map model [2]. Some motor parameters are necessary, such as the motor copper, iron, windage and motor constant losses, which are available from the motor catalogue, measured, or requested from the manufacturer. Furthermore, the efficiency map can be measured using dynamometers. For the purposes of this study, the GM EV1 motor efficiency map was produced by means of Larminie and Lowry's model, as displayed in Fig. 2. The numbers on the lines account for the efficiency. The MATLAB script to design the gear-shift schedule is provided in Appendix A for reference.

After the efficiency map of the GM EV1 motor was generated, the efficiency map for the vehicle taking the two-speed transmission into account was created. This was simply done by manipulating the motor efficiency map, namely, the maximum speed and torque of the map are divided and multiplied by the transmission gear ratio, respectively. The maximum torque of the vehicle efficiency map for gear 1 is the maximum torque of the motor efficiency map times 4, which is the first gear ratio. Likewise, the vehicle efficiency map for the second gear ratio was obtained by multiplying the motor efficiency map by 2.67. In other words, the vertical axis is turned into the transmission output torque.

In order to devise a gear-shifting schedule, the horizontal axis of the vehicle efficiency maps was converted to the vehicle speed, as opposed to the motor speed. The relationship between the



MotorSolve can now generate linear and nonlinear efficiency maps

Figure 1: Efficiency map generated by MotorSolve [3]

motor speed and vehicle speed is given by [2]:

$$\omega = \gamma \frac{v}{r} \text{ rad s}^{-1} \tag{1}$$

where ω is the motor speed, γ the gear ratio, v the vehicle speed, and r the wheel radius.

After adjusting the motor efficiency map with the corresponding gear ratio, the vehicle efficiency map for that particular gear ratio was generated. The maximum torques and speeds of the GM EV1 motor with and without the transmission are listed in Table 1. The vehicle efficiency maps for gears 1 and 2 are depicted in Fig. 3. The vertical and horizontal axes were intentionally set to the maximum to clearly express the modification of the efficiency maps due to the application of our MST. As illustrated in the figure, the EVs with MSTs have multiple efficiency maps corresponding to the gear ratios of the MST that can be advantageous for particular cases. The consecutive efficiency map is indicated in Fig 4. Please note that the efficiency values in the vehicle efficiency map have to be converted back to the values in the motor efficiency map when the efficiency is calculated to design the gear-shift schedule.

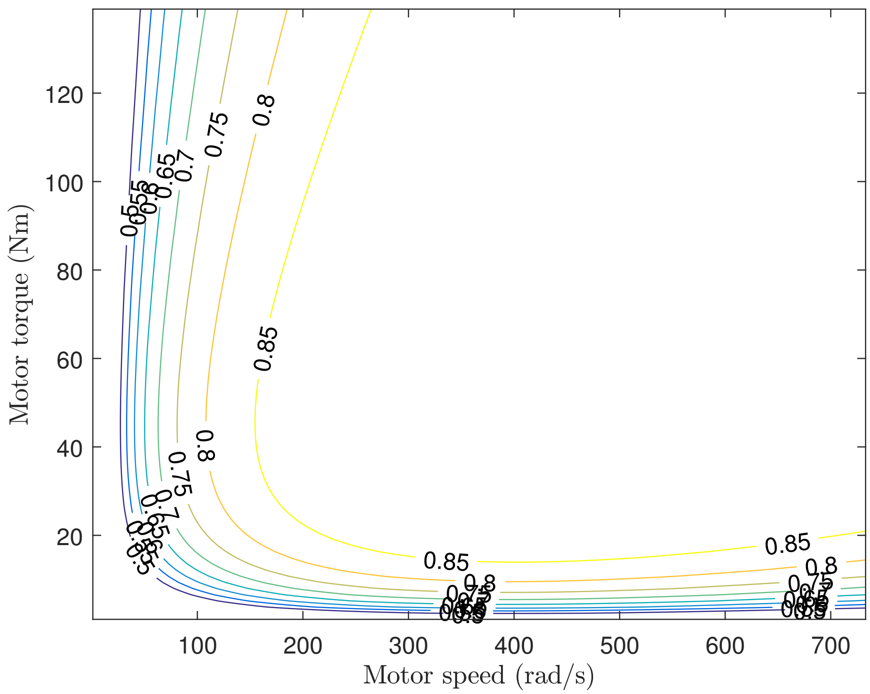


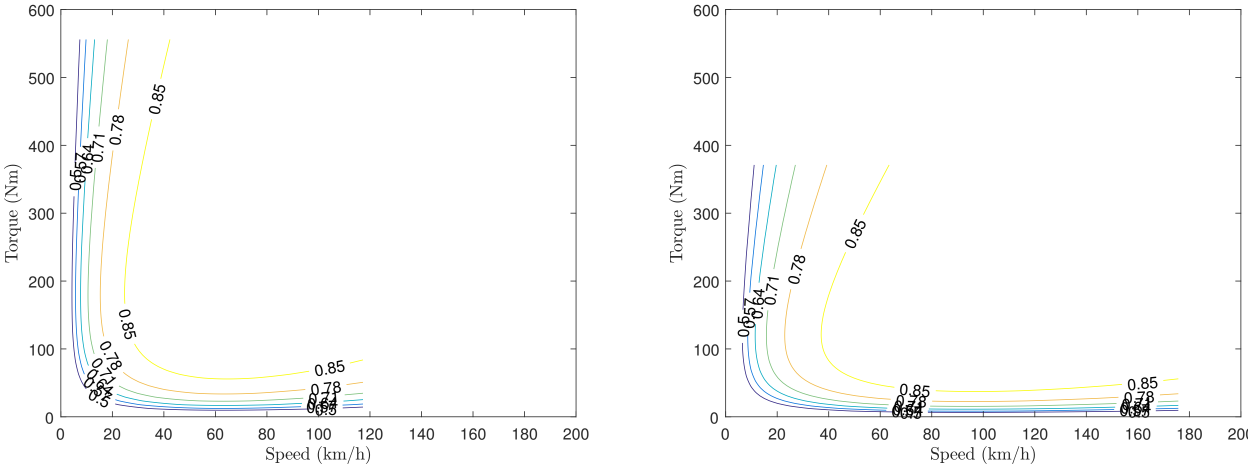
Figure 2: The GM EV1 motor efficiency map

Table 1: Adjusted GM EV1 performance

Gear ratio	**1:1**	**4:1**	**2.67:1**
T (N.m)	139	556	371.13
ω (rad/s)	733	183.25	274.53
v (km/h)	-	117.29	175.72

The next step was to draw a constant-traction-torque line on the consecutive map. Utilizing the efficiency map MATLAB script by Larminie's and Lowry's [2], this step posed a challenge because the efficiency map data of the first gear does not carry the same interval as that of the second gear. The solution was to retrieve the torque data that have the minimum difference between gear 1 and gear 2. Gear 2 has a lower torque than gear 1; therefore, the data interval in gear 2 is smaller than that in gear 1. The technique exploited the MATLAB function `meshgrid` for torque 1 and torque 2, as in line 78 of the MATLAB script in Appendix A. Subsequently, the difference was obtained in line 79 and the minimum in line 80. The `for loop` in lines 88–98 locate the minimum difference and determine the best pair.

The efficiency was calculated when efficiency maps were generated. Therefore, we only needed to locate the data row at hand in MATLAB workspace. First, a particular torque that is to be compared between gear 1 and gear 2 was selected. Then, the efficiency of the torque in question for both gear ratios was located. Second, the line intersection between the two efficiency lines was detected. The efficiency lines can be drawn but it is not necessary. We only need to detect the



(a) Gear 1

(b) Gear 2

Figure 3: Two-speed GM EV1 efficiency maps

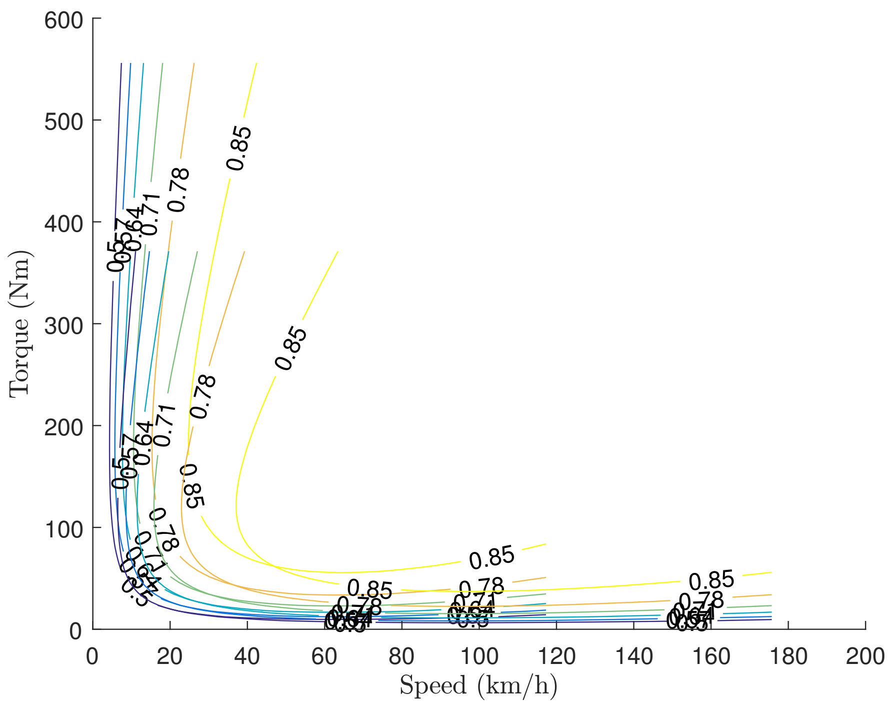


Figure 4: The consecutive GM EV1 efficiency map

intersection point that indicates the vehicle speed in which it is efficient to shift the gear. The code lines 100-116 are intended to find the efficiency and, in turn, the point of equal efficiency for the two gear ratios. The previous `meshgrid` technique was reused. The exact intersection point

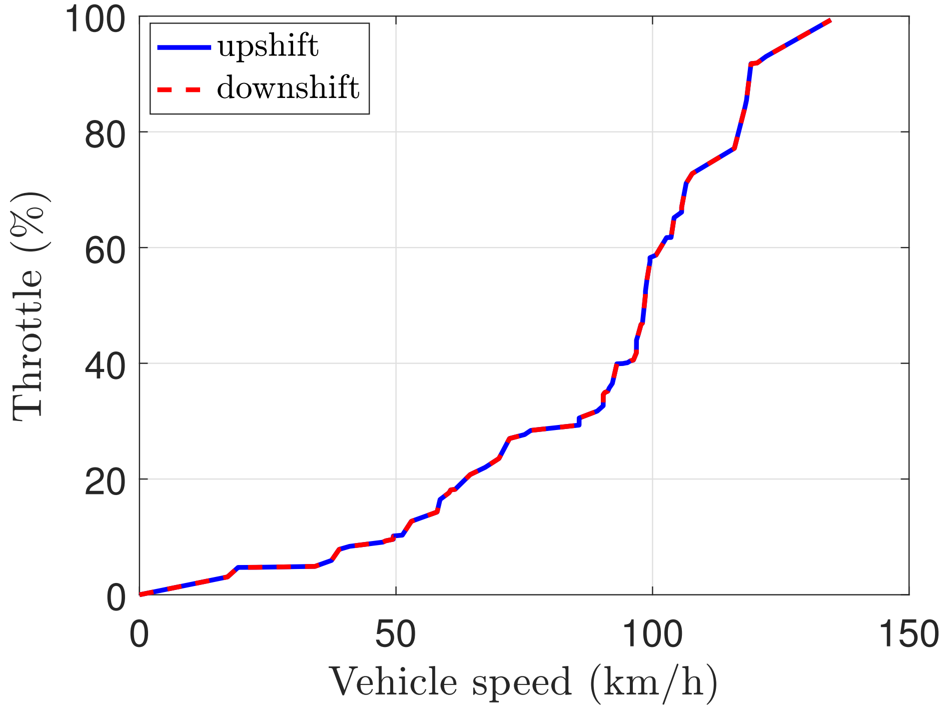


Figure 5: Gear-shift schedule for the GM EV1 with a two-speed transmission

cannot be detected because the two vector arrays speed 1 and 2 have different data intervals. Therefore, `meshgrid` was employed to find the pairs with minimum difference, as indicated in line 108. Finally, the two closest speed arrays were found, then averaged to obtain the speed of interest.

The next step was to find the throttle (accelerator pedal position) of the foregoing speed in two different gear ratios with an input-to-output power ratio as

$$\% = \frac{P_{out}}{\eta P_{in}} = \frac{T\omega}{\eta P_{in}} \tag{2}$$

where % is the throttle, P_{in} the maximum power (around 2535 W), T the maximum motor torque, ω the maximum motor speed, and η the motor efficiency.

It should be noted that the torques and speeds were retrieved from the vehicle efficiency map (as opposed to the motor efficiency map); therefore, these values must be converted to the motor efficiency map before being inputted to Eq. (2). The script is given in lines 118–145. Two gear-shifting schedules for upshift and downshift were generated, as displayed in Fig. 5. The lines were too close, but this was expected. A solution is provided where the proper downshift shifting line is computed as suggested by Zhu et al. [1], namely,

$$V_{down} = (1 - A_n)V_{up} \tag{3}$$

where V_{down} is the downshifting vehicle speed, A_n the offset coefficient (0.4–0.45), and V_{up} the

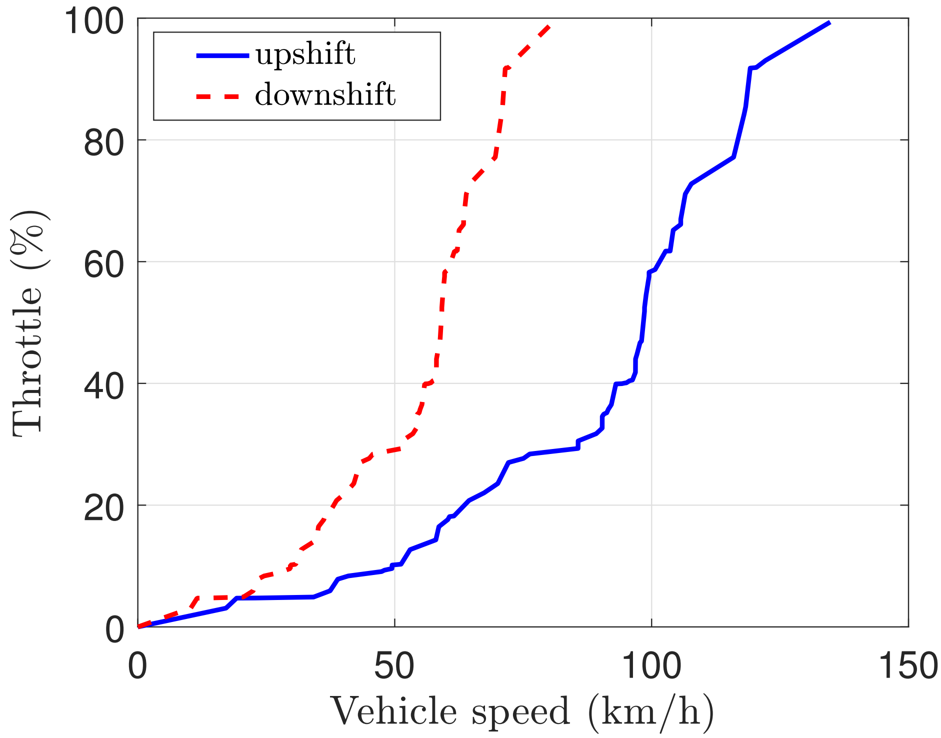


Figure 6: Modified gear-shift schedule

upshifting vehicle speed. The offset coefficient A_n of 0.4 is selected for the purposes of this study.

The MATLAB script to modify the downshifting line is given in lines 146–154 and the modified gear-shift schedule is included in Fig. 6. The horizontal axis is the vehicle speed, whereas the vertical axis the throttle or accelerator pedal position. The two plots pertain to the upshift and downshift lines. When the vehicle speed and throttle passes these lines, the gear ratio will shift accordingly.

3 Range Simulation

Range simulation was conducted by modifying Larminie and Lowry's MATLAB code [2] that was written for a range-prediction of the standard (single-speed) GM EV1. The gear-shift time for the drive cycle SFUDS is included in Fig. 7, where the time history of the gear ratio throughout SFUDS drive cycle is plotted over the SFUDS cycle to indicate when the gear-shifting occurs. Apparently, the second gear ratio is more efficient for the SFUDS cycle.

The MATLAB script to generate the gear-ratio shifting is included in Appendix B. First, on lines 8–14, a relationship between throttle and acceleration was determined. In this case, a simple linear relationship between throttle and acceleration, where the acceleration is divided by the maximum value of acceleration in the drive cycle, was employed. Please note that this is

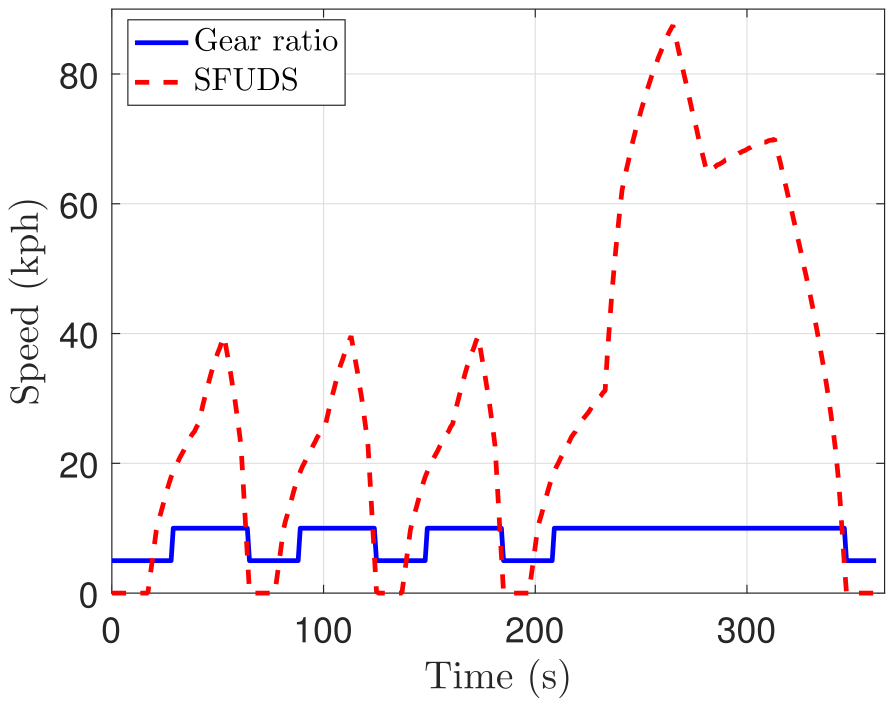


Figure 7: Gear-shift time

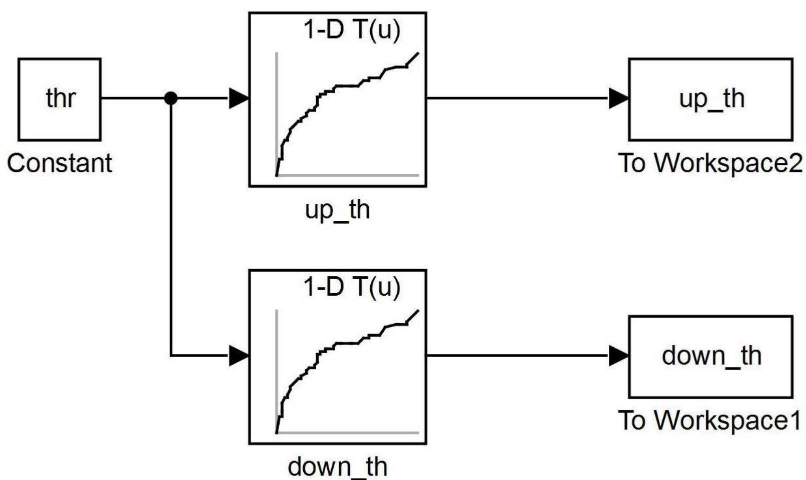


Figure 8: Shifting thresholds in Simulink

true when the acceleration is positive. In contrast, when the motor operates as a generator for regenerative braking, the driver supposedly will not press the throttle. Hence, a dummy value 0.015 is employed in that case. Three succeeding lines in the code are included to plot the throttle and the acceleration to verify their relationship.

Lines 19–28 specify the velocity thresholds for the upshift and downshift in Simulink. The

Simulink file utilized to perform this task is displayed in Fig. 8. Essentially, the acceleration was converted to throttle and multiplied by 100 in line 24. This is intended to assign the upshift and downshift thresholds for different throttle values. Two look-up tables (up_th and down_th) in the Simulink file comprise data from the gear-shift schedule, one for the upshift, the other for the downshift. On the Table and Breakpoints section of the look-up tables, the speed data of the gear-shift schedule can be inputed in the Table Section, and the throttle data in the Breakpoints-1 Section. The whole data were simply copied and pasted into the box within the square brackets. The procedure was repeated for the downshift line of the gear-shift schedule.

Because this simulation procedure takes time and it is inconvenient to wait every time the simulation is executed, the data of throttle are saved, as expressed in line 30. The plot commands on lines 31 and 32 are included only for verification. After the speed thresholds were prescribed, the gear-shift time was detected according to the gear-shift schedule for the SFUDS. Lines 35–45 impose the condition that the gear will shift (simulation starts from gear 1) if the current velocity exceeds the threshold both for upshift and downshift. The gear-shift time with SFUDS on the background is plotted on lines 47–58, as depicted in Fig. 7. The gear ratio is defined in the final line 61.

The final step is to modify the `one cycle` code in Larminie and Lowry's range model. Apparently, this code was written for the range-prediction of EVs with a single speed. Therefore, the gear changes according to the gear-shift time illustrated in Fig. 7 must be incorporated in the range simulation of GM EV1 with a two-speed transmission. The modified code of one_cycle is provided in Appendix C, with only a few lines of the code modified. Therefore, only lines added or modified will be explained. First, line 11 incorporates the gear-shift time in the simulation cycle. Then, line 21 provides the data of the gear ratio of the MST. The gear ratio is determined by these inputted data because the simulation happened every second, which is the sample time of the SFUDS. Lastly, the final lines 87–89 are modified to check the tractive power P_{te} and velocities for each sample.

4 Conclusions

We reported here on the details of the devising of gear-shift schedules for electric vehicles with multi-speed transmissions. A case study is included whereby the schedule for the GM EV1 with a two-speed transmission was devised. The design was conducted in MATLAB/Simulink, the procedure documented along with the code description.

References

[1] B. Zhu, N. Zhang, P. Walker, X. Zhou, W. Zhan, Y. Wei, and N. Ke, "Gear shift schedule design for multi-speed pure electric vehicles," *Proc. IMechE, part D: Journal of Automobile Engineering*, vol. 229, no. 1, pp. 70 – 82, 2015.

[2] J. Larminie and J. Lowry, *Electric Vehicle Technology Explained, Second Edition.* John Wiley and Sons, West Sussex, England, 2012.

[3] Infolytica Press Release, "Infolytica corporation releases MotorSolve v5 for immediate distribution[1]." Accessed: 2017-12-20.

[1] http://www.infolytica.com/news/infolytica-corporation-releases-motorsolve-v5/

A MATLAB Script to Devise the Gear-shifting Schedule

The MATLAB script to design gear-shift schedule is as follows:

```
close all;clear all;clc;

%% efficiency map for gear 1
r = 0.2159; % in m, wheel radius for 235/45R17 taken from Zhu's
G1 = 4; % first gear ratio
max_torque = 8.03; % Nm
max_speed = 315; % rad/s
max_output_torque1 = G1*max_torque;
a1 = G1/r/3.6; % \omega rad/s = a1 * vehicle speed km/h, 3.6 converts m/s to km/h
max_vehicle_speed1 = max_speed/a1;
x=linspace(1,max_vehicle_speed1); % N.B. km/h, not rpm
y=linspace(1,max_output_torque1);
% Allocate motor loss constants.
kc=0.2; % For copper losses
ki=0.08; % For iron losses
kw=1e-7; % For windage losses
ConL=60; % For constant motor losses
[X,Y]=meshgrid(x,y);
Output_power=((a1*X).*(Y/G1)); % Torque x speed = power
B=((Y/G1).^2)*kc; % Copper losses
% Appendices: MATLAB? Examples 291
C=(a1*X)*ki; % Iron losses
D=((a1*X).^3)*kw; % Windage losses
Input_power = Output_power + B + C + D + ConL;
Z = Output_power./Input_power;
% We now set the efficiencies for which a contour
% will be plotted.
% V=[0.7,0.8,0.85,0.9,0.91,0.92,0.925,0.93];
V=[0.5:0.07:0.87];
box off
grid off
[C,h] = contour(X,Y,Z,V);
xlabel('Speed/km.h^-^1'), ylabel('Torque/N.m');
clabel(C,h)
hold on
V=[0,550];
[C] = contour(X,Y,Output_power,V, 'red', 'linewidth', 2);
clabel(C)

%% efficiency map for gear 2
G2 = 2.67; % second gear ratio
a2 = G2/r/3.6; % constant for motor speed rad/s to vehicle speed km/h, 3.6 converts m/s to km/h
max_output_torque2 = G2*max_torque;
max_vehicle_speed2 = max_speed/a2;
x2=linspace(1,max_vehicle_speed2); % N.B. km/h, not rpm
y2=linspace(1,max_output_torque2);
```

```
% Allocate motor loss constants.
kc=0.2; % For copper losses
ki=0.08; % For iron losses
kw=1e-7; % For windage losses
ConL=60; % For constant motor losses
[X2,Y2]=meshgrid(x2,y2);

Output_power=((a2*X2).*(Y2/G2)); % Torque x speed = power
B2=((Y2/G2).^2)*kc; % Copper losses
% Appendices: MATLAB? Examples 291
C2=(a2*X2)*ki; % Iron losses
D2=((a2*X2).^3)*kw; % Windage losses
Input_power = Output_power + B2 + C2 + D2 + ConL;
Z2 = Output_power./Input_power;
% We now set the efficiencies for which a contour
% will be plotted.
% V=[0.7,0.8,0.85,0.9,0.91,0.92,0.925,0.93];
V2=[0.5:0.07:0.87];
box off
grid off
[C,h] = contour(X2,Y2,Z2,V2);
xlabel('Speed/km.h^-^1'), ylabel('Torque/N.m');
clabel(C,h)
hold on
V=[0,550];
[C] = contour(X2,Y2,Output_power,V, 'red', 'linewidth', 2);
clabel(C)

%% step 2 a constant-traction-torque line
format long
[X3,Y3] = meshgrid(Y(:,1),Y2(:,1));
A=abs(X3-Y3);
c=min(A);
l=zeros(length(c),5);

% find the minimum torque difference because
% I dont have the exact torque for two
% different gear ratio due to two different
% linspace

for i=1:length(c)
% position = find(A==c(i));
[row,col,v]=find(A==c(i));
% [m,n] = size(A);
% [Q,R]=quorem(sym(position),sym(m));
l(i,1)=Y(col,1);
l(i,2)=Y2(row,1);
l(i,3)=c(i);
l(i,4)=col; % the column of Y that has minimum difference
l(i,5)=row; % the column of Y2 that has minimum difference
end
```

```
%% step 3 calculate the motor efficiency but
% only until 66 which is around 21 Nm
n=65; % number of points to compare
speed=zeros(n,1);
Eff=zeros(n,4);
for j=2:n+1;
[X4,Y4] = meshgrid(Z(:,l(j,4)),Z2(:,l(j,5)));
A2=abs(X4-Y4);
c2=min(min(A2));
[row,col,v]=find(A2==c2);
speed(j,1) = (X(1,col)+X2(1,row))/2;
Eff(j,1) = Z(col,l(j,4)); % efficiency in gear 1
Eff(j,2) = Z2(row,l(j,5)); % efficiency in gear 2
Eff(j,3) = c2; % efficiency difference
Eff(j,4) = Z(col,l(j,4))-Z2(row,l(j,5)); % double check
end
sorted_speed = sort(speed);

%% step 4-5 throttle values

max_power = 2535.79;
throttle_gear=zeros(n,2);
for k=2:n+1;
eff = Eff(k,1);
% first gear 4:1
T = l(k,1)/G1;
omega=speed(k,1)*a1;
throttle_gear(k,1) = 100*T*omega/max_power/eff;
% throttle_gear(k,1) = 100*sqrt(T*omega/max_power/eff);
% throttle_gear(k,1) = 100*nthroot(T*omega/max_power/eff,4);


% second gear 4:1
T2 = l(k,1)/G2;
omega2=speed(k,1)*a2;
throttle_gear(k,2) = 100*T2*omega2/max_power/eff;
% throttle_gear(k,2) = 100*sqrt(T2*omega2/max_power/eff);
% throttle_gear(k,2) = 100*nthroot(T2*omega2/max_power/eff,4);
end
sorted_throttle = sort(throttle_gear);
figure(2);
plot(sorted_speed(:,1),sorted_throttle(:,1));
hold on;
grid on;
plot(sorted_speed(:,1),sorted_throttle(:,2));
xlabel('Speed/km.h^-^1'), ylabel('Throttle/%');
%% adjusting the downshift curves
figure(3)
An = 0.4;
new_downshift = (1-An)*sorted_speed(:,1);
plot(sorted_speed(:,1),sorted_throttle(:,1));
```

```
hold on;
grid on;
plot(new_downshift(:,1),sorted_throttle(:,2))
xlabel('Speed/km.h^-^1'), ylabel('Throttle/%');
```

B MATLAB Script to Compute the Gear-shifting Time

The MATLAB script to create the gear-shift time is given below.

```
%% shiftTime5report.m

close all;clear all;clc;
load shiftData; % accel in m/s^2 and XDATA
sfuds; % kph
N=length(V); % Find out how many readings

for c=1:N
    if accel(c) > 0
        throttle(c) = accel(c)/1.8465;
    else
        throttle(c) = 0.015;
    end
end
% plot(XDATA,accel)
% hold on
% plot(XDATA,throttle,'r')

%% velocity threshold
% Up_th = zeros(1,N);
% Down_th = zeros(1,N);
% for c=1:N
%     clear up_th
%     thr=100*throttle(c);
%     sim('thresholdGMEV1');
%     Up_th(c) = up_th(1,1);
%     Down_th(c) = down_th(1,1);
% end

load speedThresholdGMEV1; % Up_th and Down_th in kph
% plot(XDATA,Up_th);hold on;
% plot(XDATA,Down_th);

%% gear shift time
gear = zeros(1,N);
for c=2:N
    gear(1) = 1;
    if V(c)>Up_th(c) && gear(c-1) == 1
        gear(c) = gear(c-1)+1;
    elseif V(c)<Down_th(c) && gear(c-1) == 2
        gear(c) = gear(c-1)-1;
    else
        gear(c)=gear(c-1);
    end
end
% yyaxis left;
plot(XDATA,5*gear, '-k', 'LineWidth', 2);
```

```
hold on;
% yyaxis right;
plot(XDATA,V, '--r', 'LineWidth', 2)
grid on;
ylim([0 90]);xlim([0 365]);
set(legend('Gear ratio','SFUDS'), 'interpreter','latex')
ylabel('Speed (kph)','interpreter','latex','FontSize',13)
xlabel('Time~(s)','interpreter','latex','FontSize',13)
% annotation('textbox','String','1st gear ratio','interpreter','latex','FontSize
    ',12)
% annotation('textbox','String','2nd gear ratio','interpreter','latex','FontSize
    ',12)
set(gca,'FontSize',15);


Gratio=[34 15];
```

C MATLAB Script of Modified Range-prediction Simulation

The MATLAB script of the modified range-prediction simulation is provided below.

```
% one_cycle_2speedAPC1report

% ******************************
% ONE CYCLE
% This script file performs one cycle, of any
% drive cycle of N points with any vehicle and
% for lead acid or NiCad batteries.
% All the appropriate variables must be set
% by the calling program.
% *******************************
load gearAPC1
for C=2:N
    accel=V(C) - V(C-1);
    Fad = 0.5 * 1.25 * area * Cd * V(C)^2; % Equ. 7.2
    Fhc = 0; % Eq. 7.3, assume flat
    Fla = 1.05 * mass * accel;

    % The mass is increased modestly to compensate for
    % the fact that we have excluded the moment of inertia
    Pte = (Frr + Fad + Fhc + Fla)*V(C); %Equ 7.9 & 7.23
    omega = Gratio(gear(C)) * V(C);
    if omega == 0 % Stationary
        Pte=0;
        Pmot_in=0; % No power into motor
        Torque=0;
        eff_mot=0.5; % Dummy value, to make sure not zero.
    elseif omega > 0 % Moving
        if Pte < 0
            Pte = Regen_ratio * Pte; % Reduce the power if
        end; % braking, as not all will be by the motor.

        % We now calculate the output power of the motor,
        % Which is different from that at the wheels, because
        % of transmission losses.
        if Pte>=0
            Pmot_out=Pte/G_eff; % Motor power> shaft power
        elseif Pte<0
            Pmot_out=Pte * G_eff; % Motor power diminished
        end; % if engine braking.
        Torque=Pmot_out/omega; % Basic equation, P=T * omega
        if Torque>0 % Now use equation 7.24
            eff_mot=(Torque*omega)/((Torque*omega)+((Torque^2)*kc)+(omega*ki)+((omega^3)*kw)+ConL);
        elseif Torque<0
```

```
            eff_mot=(-Torque*omega)/((-Torque*omega) + ((Torque^2)*kc)+(omega*ki)
                +((omega^3)*kw)+ConL);
        end;
        if Pmot_out >= 0
            Pmot_in = Pmot_out/eff_mot; % Equ 7.25
        elseif Pmot_out < 0
            Pmot_in = Pmot_out * eff_mot;
        end;
    end;

    Pbat = Pmot_in + Pac; % Equation 7.27

    if bat_type=='NC'
        E=open_circuit_voltage_NC(DoD(C-1),NoCells);
    elseif bat_type=='LA'
        E=open_circuit_voltage_LA(DoD(C-1),NoCells);
    else
        error('Invalid battery type');
    end;

    if Pbat > 0 % Use Equ. 2.20
        I = (E - ((E*E) - (4*Rin*Pbat))^0.5)/(2*Rin);
        CR(C) = CR(C-1) +((I^k)/3600); %Equation 2.18
    elseif Pbat==0
        I=0;
    elseif Pbat <0
        % Regenerative braking. Use Equ. 2.22, and
        % double the internal resistance.
        Pbat = - 1 * Pbat;
        I = (-E + (E*E + (4*2*Rin*Pbat))^0.5)/(2*2*Rin);
        CR(C) = CR(C-1) - (I/3600); %Equation 2.23
    end;

    DoD(C) = CR(C)/PeuCap; %Equation 2.19

    if DoD(C)>1
        DoD(C) =1;
    end

    % Since we are taking one second time intervals,
    % the distance traveled in metres is the same
    % as the velocity. Divide by 1000 for km.
    D(C) = D(C-1) + (V(C)/1000);
    XDATA(C)=C; % See Section 7.4.4 for the use
    YDATA(1,C)=Pte;
    YDATA(2,C) = V(C)./.3556; % omega output, .3556m is the wheel radius
    YDATA(3,C) = YDATA(1,C)./YDATA(2,C);
end;
% Now return to calling program.
```