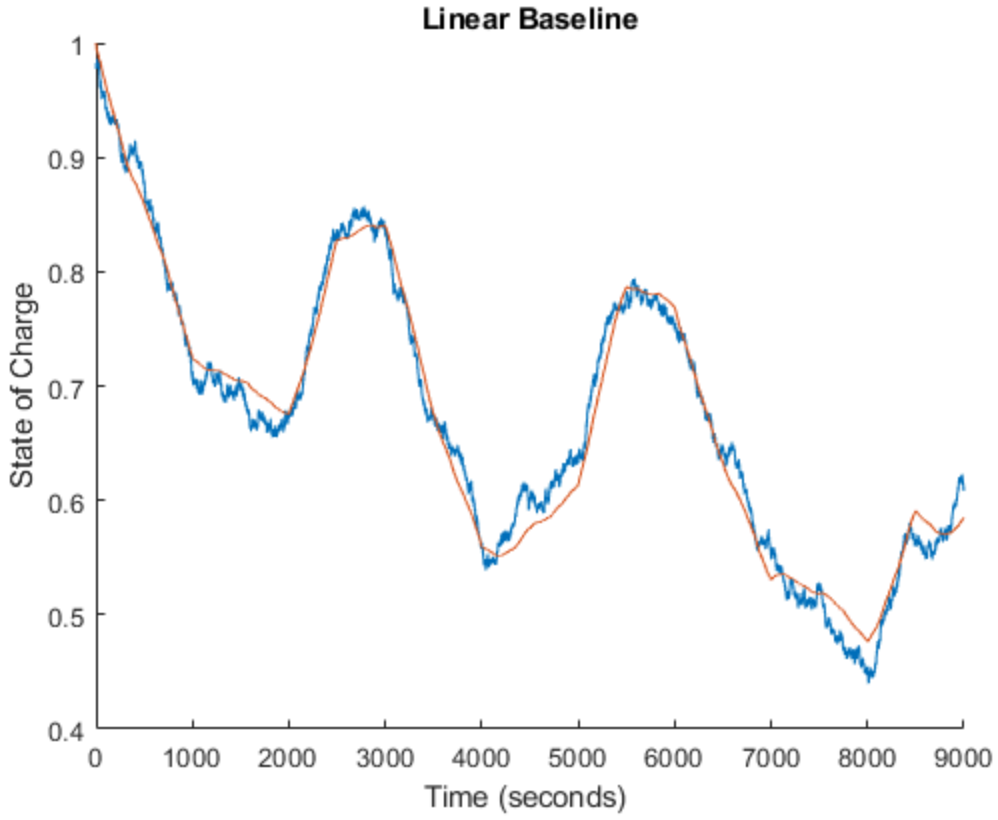
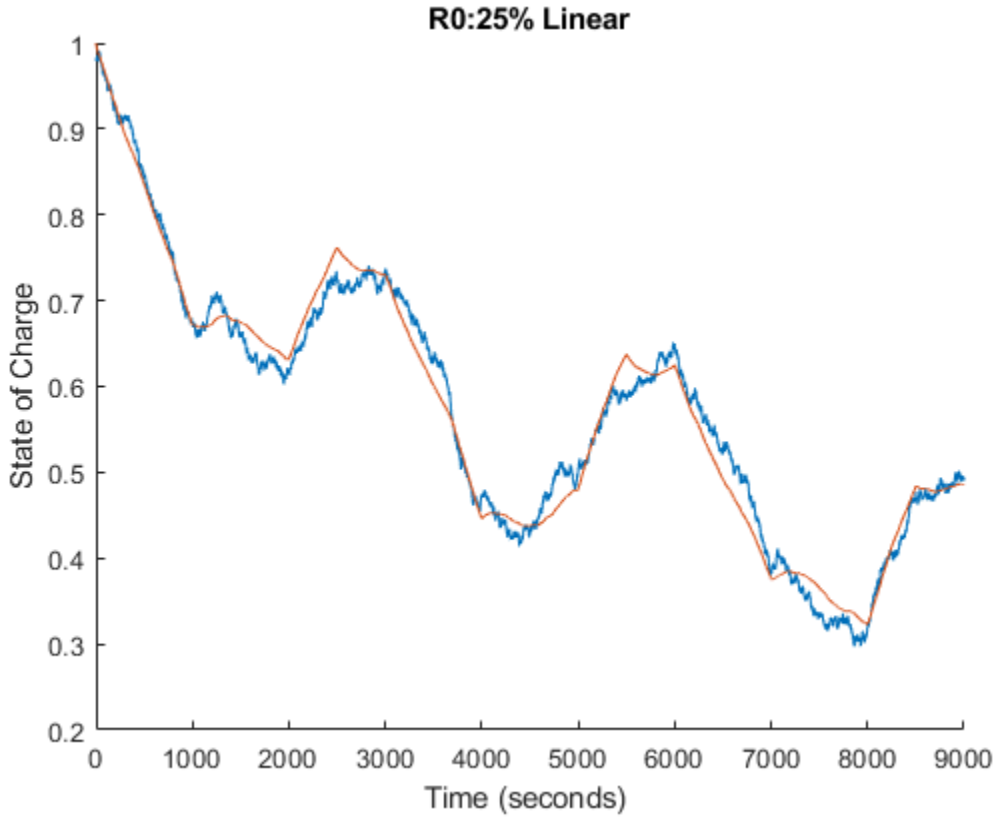
Table of Contents

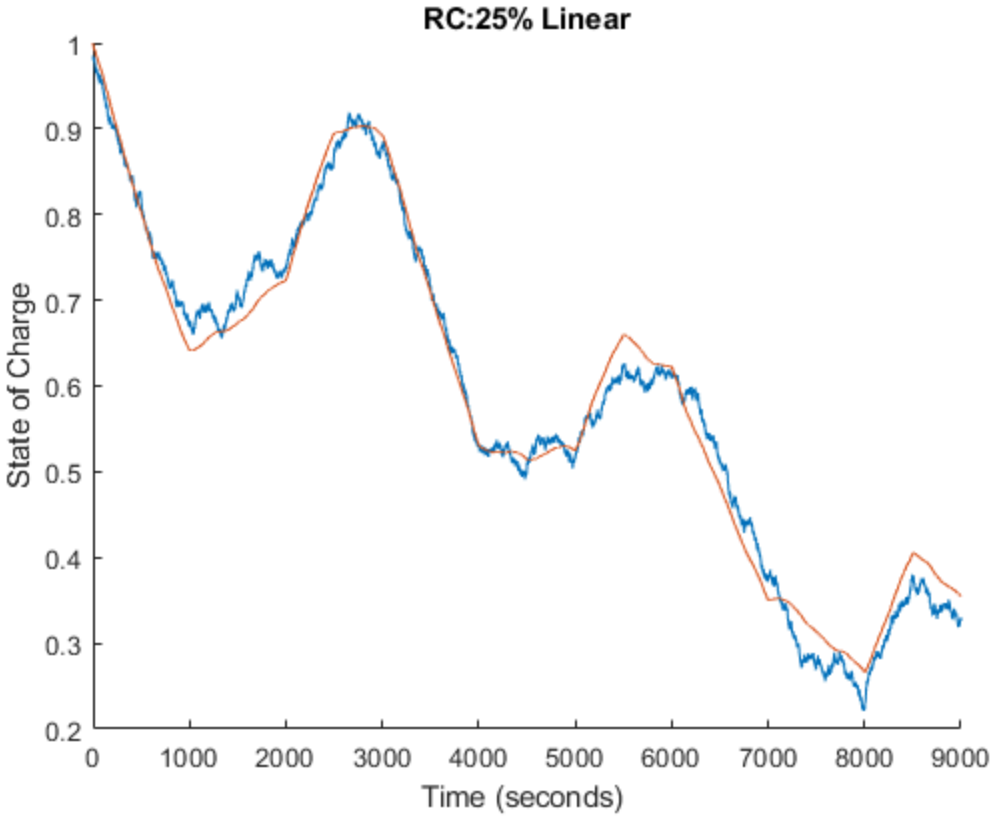
MAE 273A Project Script

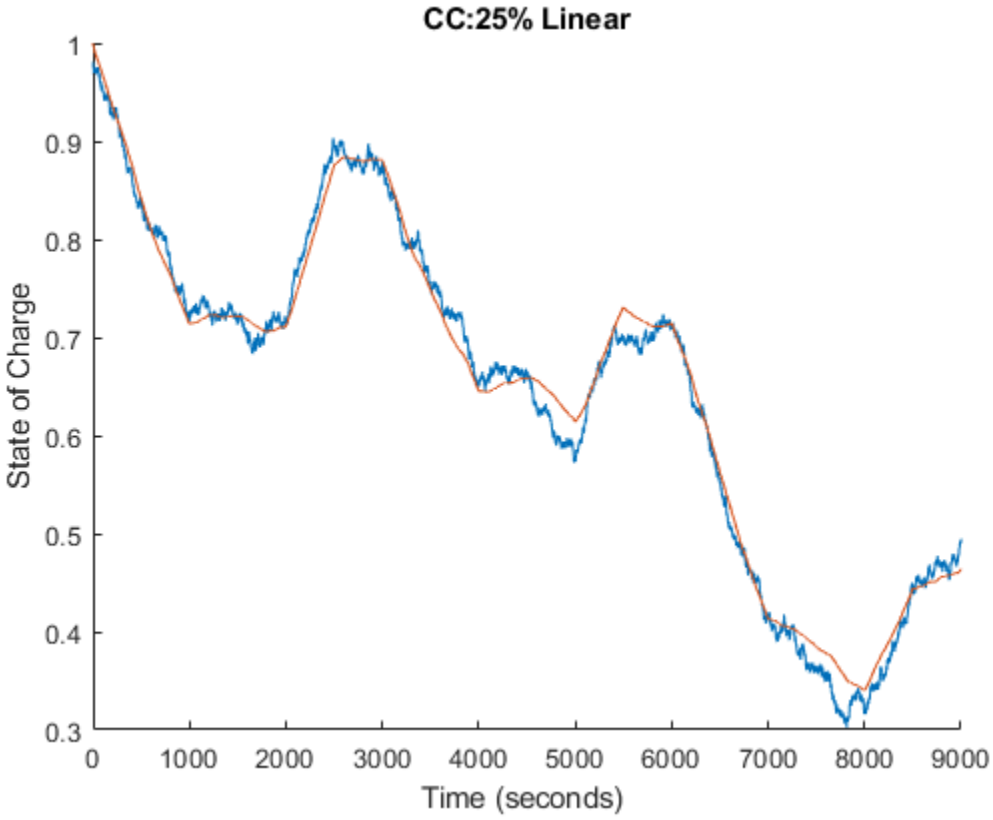
```
clear all

List =
 ["FirstOrderTruth_BASELINE_linear.mat","FirstOrderTruth_R0_25_linear.mat","FirstO
NameList = ["Linear Baseline", "R0:25% Linear", "RC:25%
 Linear","CC:25% Linear","R0:50% Linear","RC:50% Linear","CC:50%
 Linear","R0:20% RC: 20%Linear","Nonlinear Baseline ","R0:25%
 Nonlinear","RC:25% Nonlinear","CC:25% Nonlinear","R0:50%
 Nonlinear","RC:50% Nonlinear","CC:50% Nonlinear", "R0:20% RC:20%
 Nonlinear" ];


for k = 1:length(List)
    load(List(k))

s = tf('s');

%battery model parameters
Rc = 0.015;    %Ohms
Cc = 2400;     %F
Cbat = 5*3600;
alpha =0.65;
R0 = 0.01;     %Ohms
Vocv0 = 3.435; %V

%tunning parameters
K = 1;         %gain
% zeta = 0.707;  %damping ratio
zeta = 0.5;  %damping ratio

wn = 75;      %natural frequency

%continuous time ss model
A = [-1/(Rc*Cc) 0; 0 0];
B = [1/Cc; -1/Cbat];
C = [-1 alpha];
D = -R0;

A1 = A(1,1);
B1 = B(1,1);
A2 = A(2,2);
```

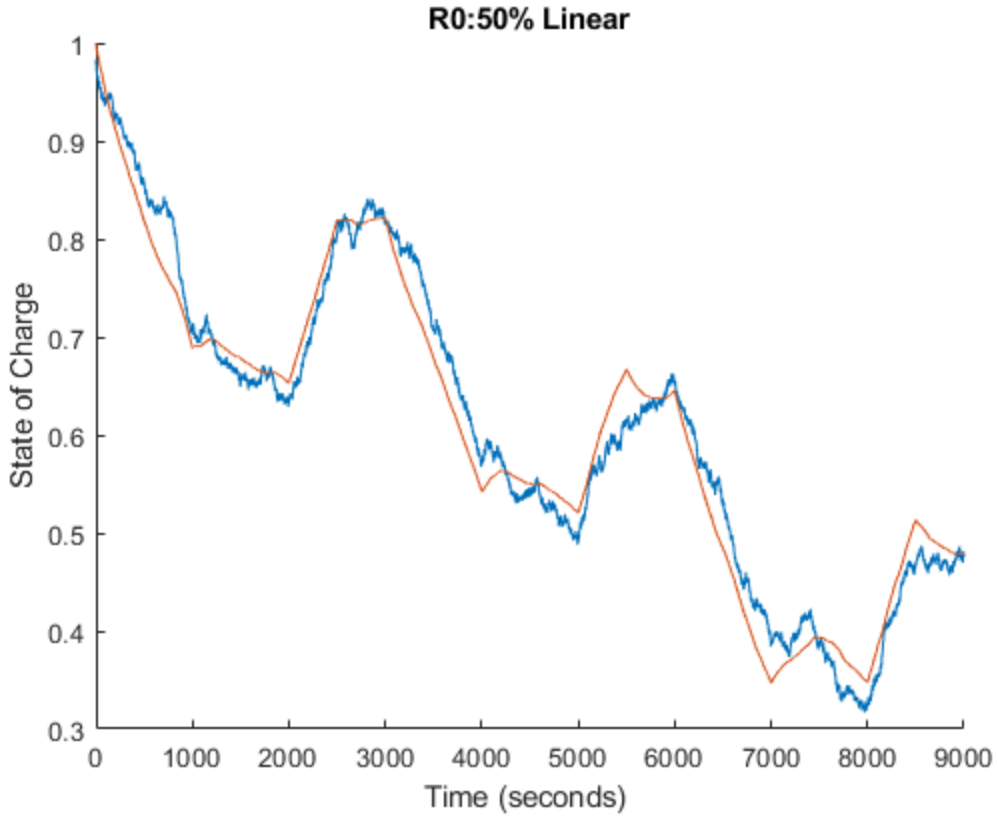
```
B2 = B(2,1);
C2 = alpha;

SI = [s 0;0 s];
Gp = C*(inv(SI-A))*B+D;       %plant
T = minreal(K*wn^2/(s^2+2*zeta*wn*s+wn^2)); %complimentary
Y = minreal(T/Gp);            %youla
S = minreal(1-T);             %sensitivity
Gc = minreal(Y/S);            %controller
L = minreal(Gc*Gp);                    %open loop TF
sysTF = minreal(Gc*Gp/(1+Gc*Gp));      %actual sys TF
[num, den] = tfdata(Gc, 'v'); %get numerator and denominator of Gc tf

out = sim('Estimator_Simulink_trial1') ;   %run simulink model

est_soc = out.SOC_est;
tout = out.tout;

YOULA_ACTUAL (:,k) = SOC_act;
YOULA_ESTIMATED(:,k) = est_soc;

figure()
hold on
plot(t,SOC_act);
plot(tout,est_soc);

title(NameList(k))
xlabel('Time (seconds) ')
ylabel('State of Charge')

end
```

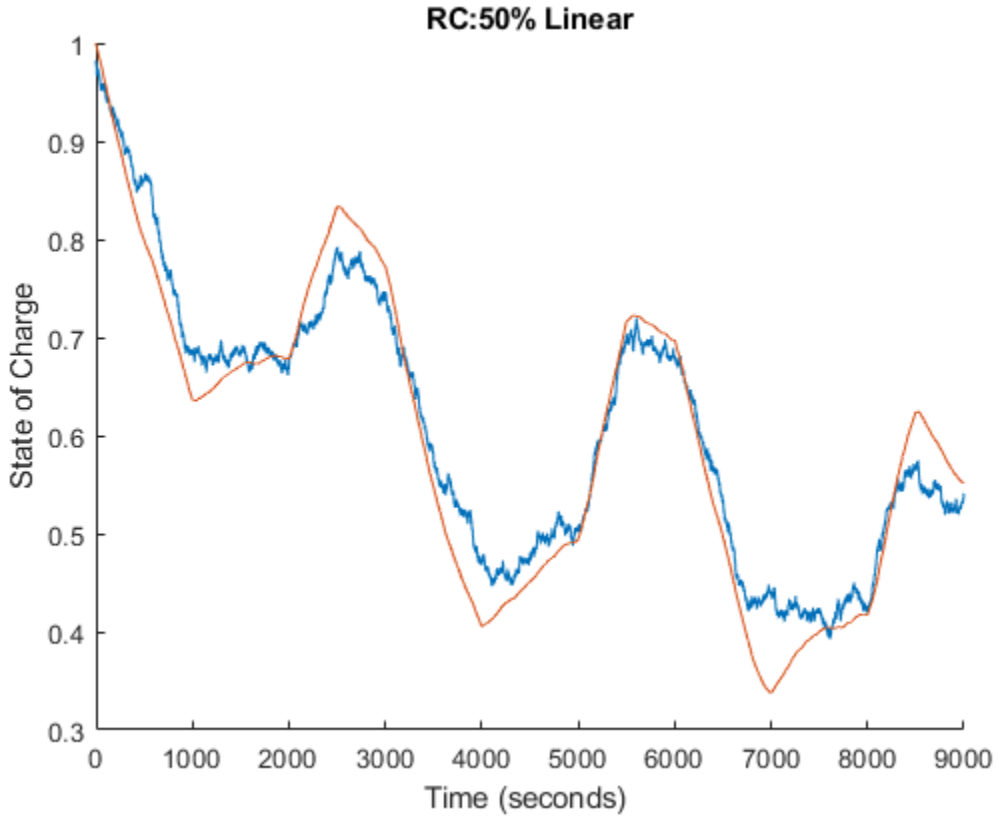

Linear Baseline
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

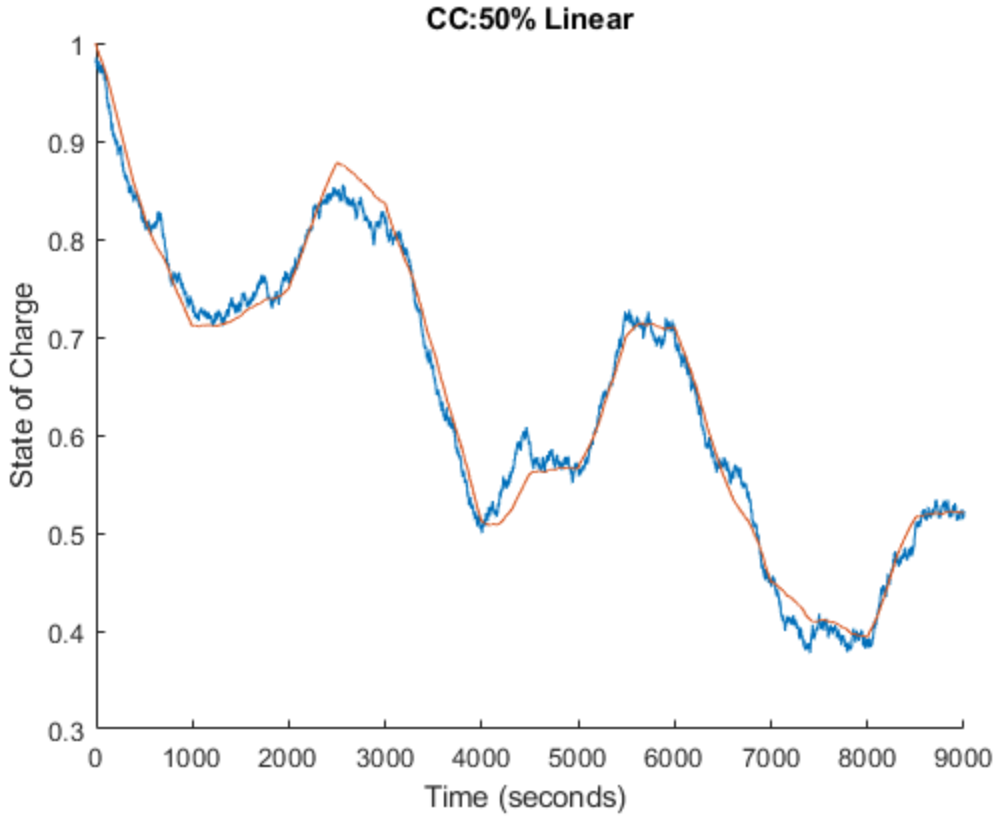

R0:25% Linear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

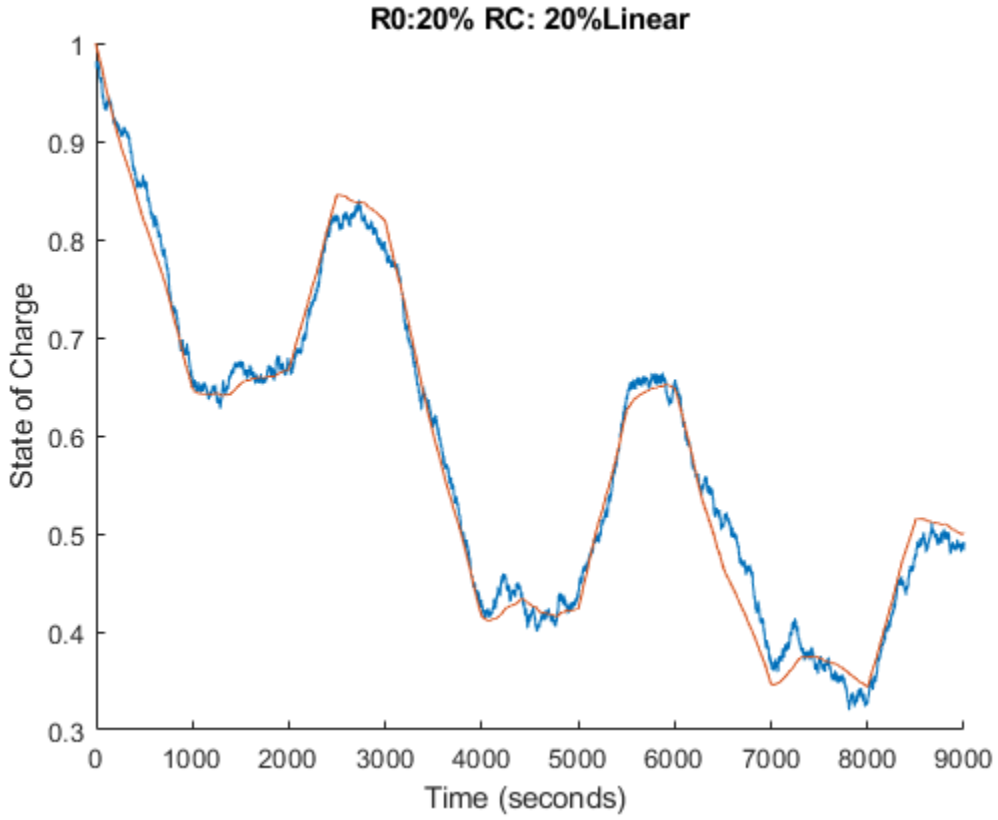

RC:25% Linear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

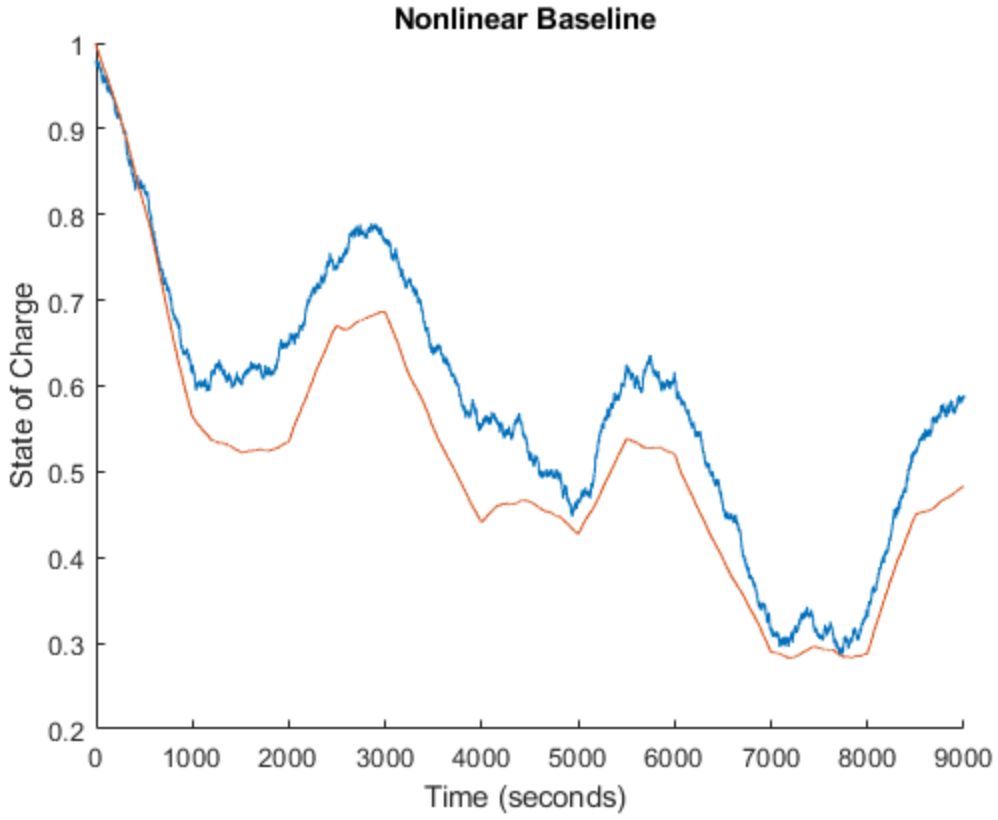

CC:25% Linear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

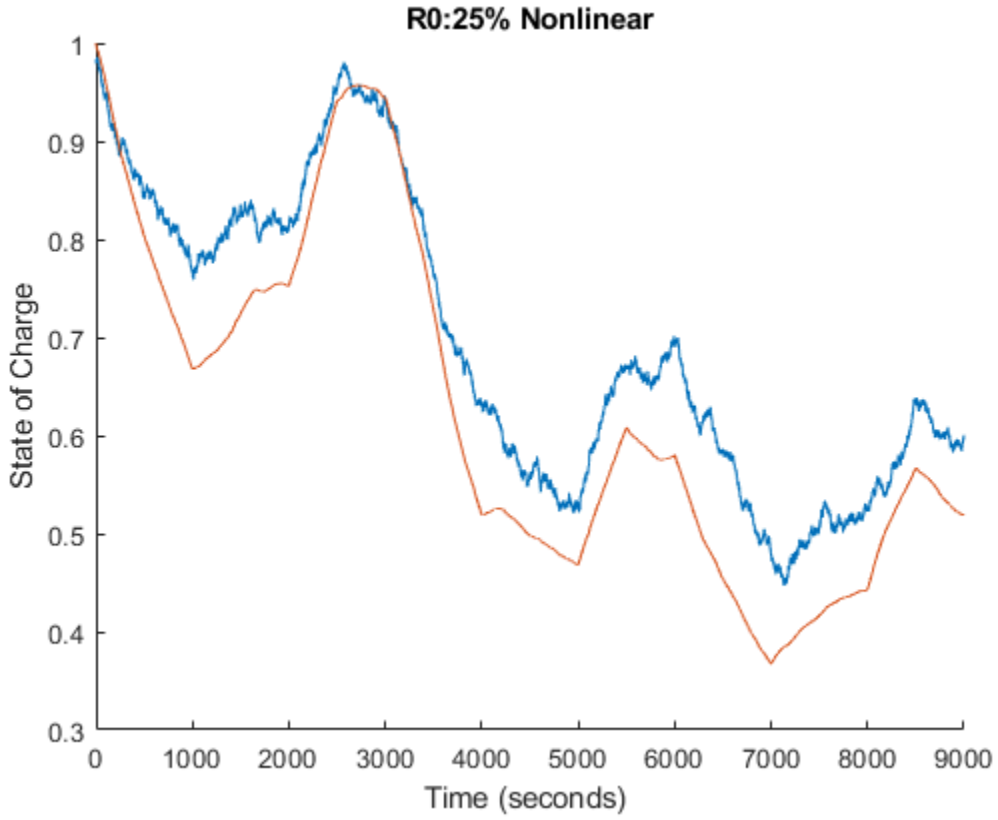

R0:50% Linear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

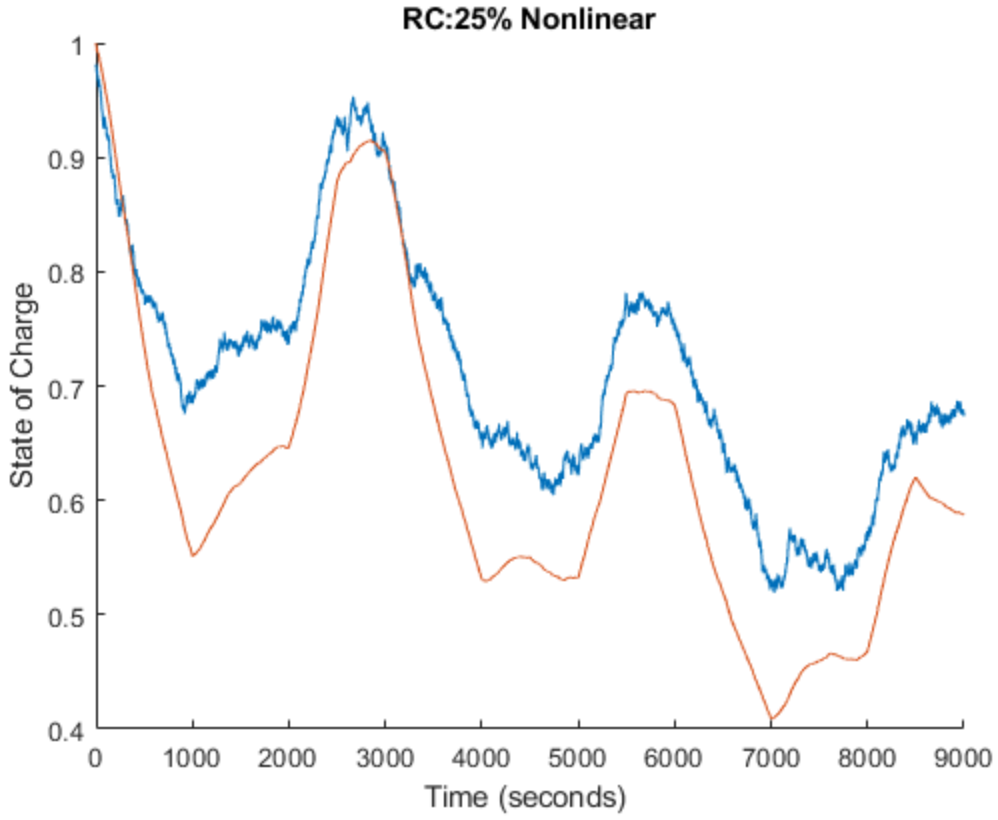

RC:50% Linear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

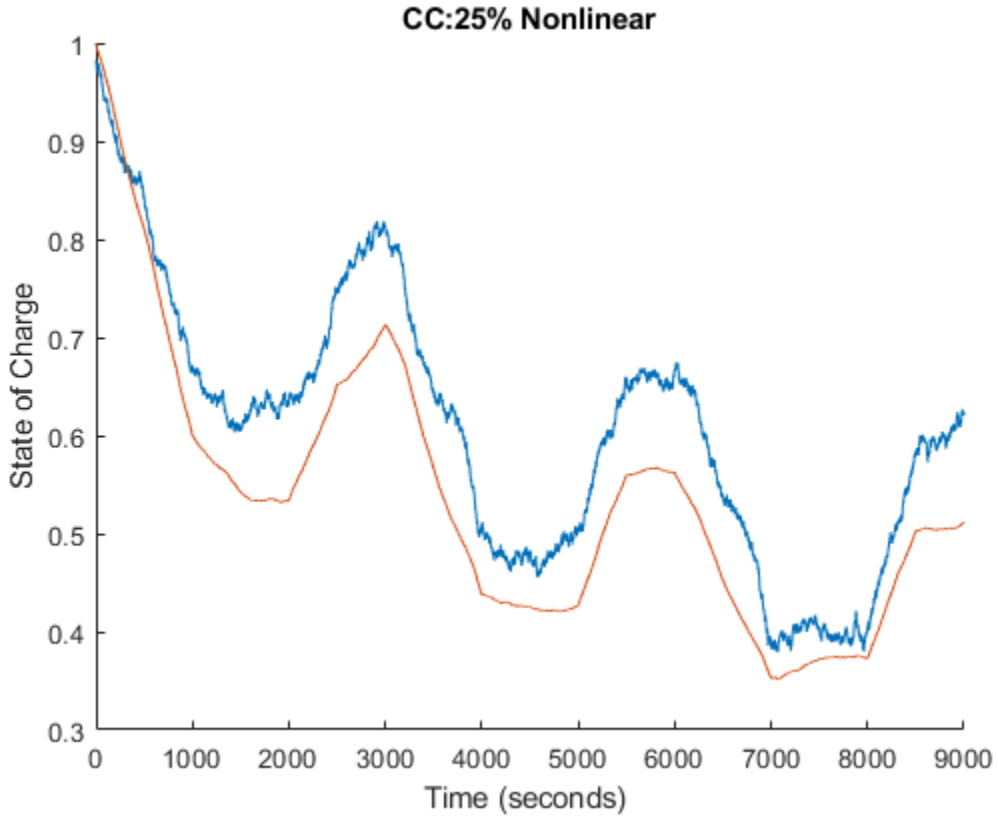

CC:50% Linear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

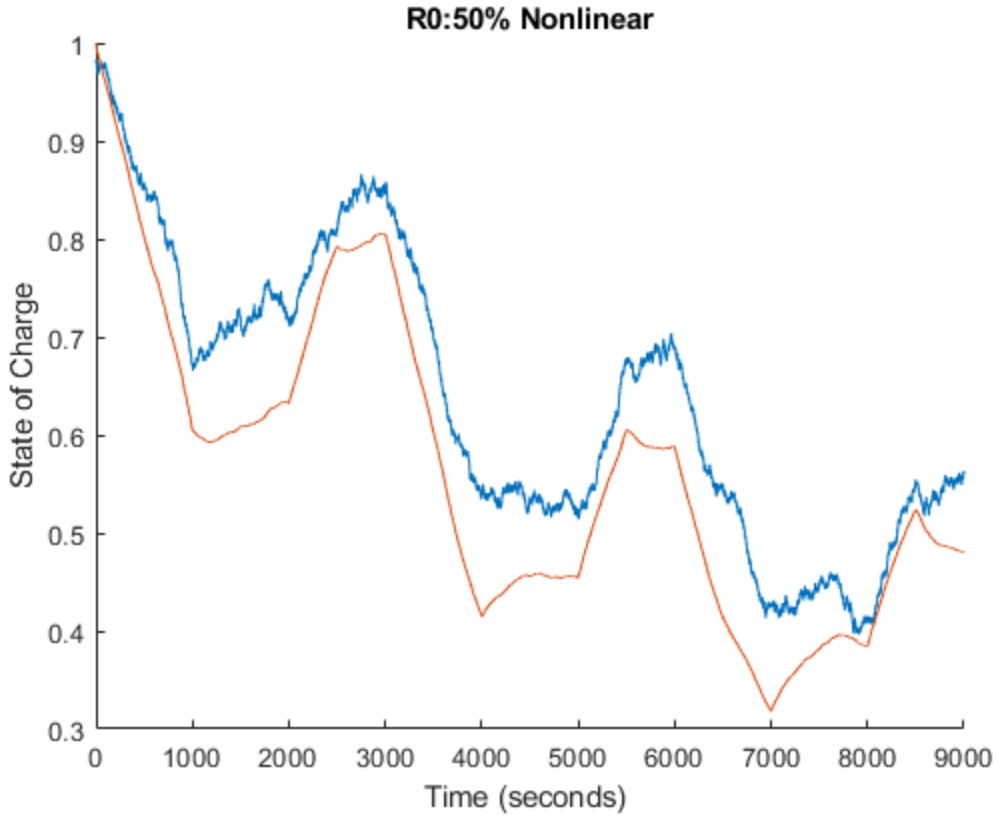

R0:20% RC: 20%Linear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

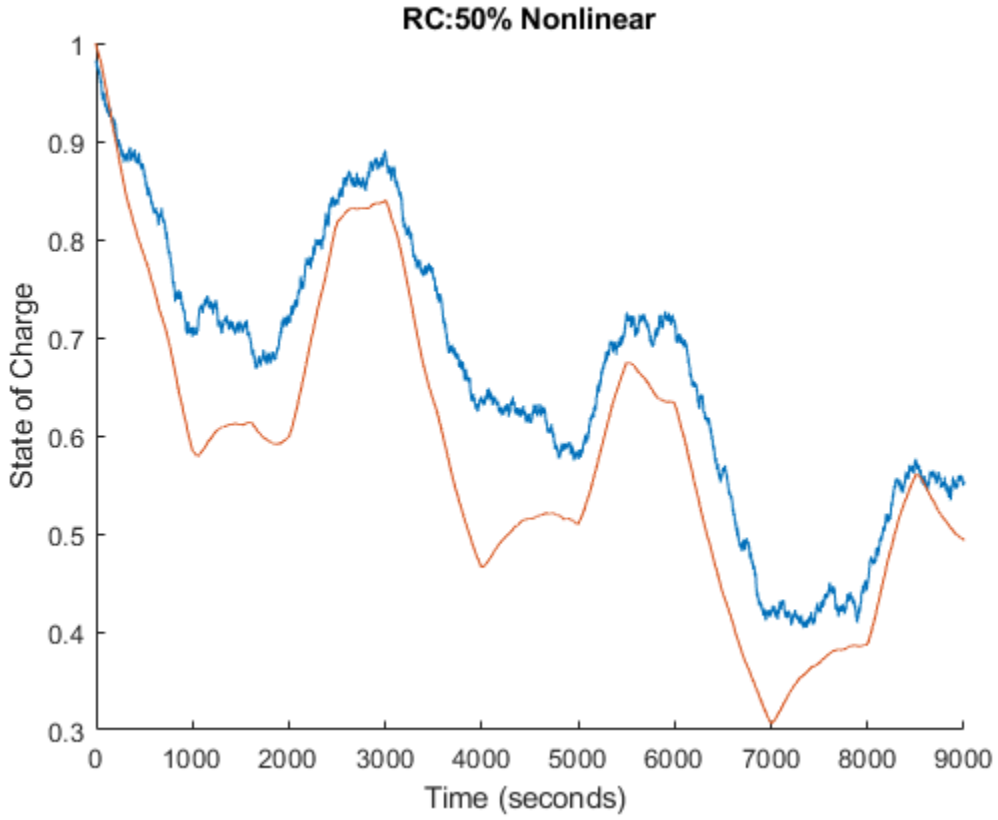

Nonlinear Baseline
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

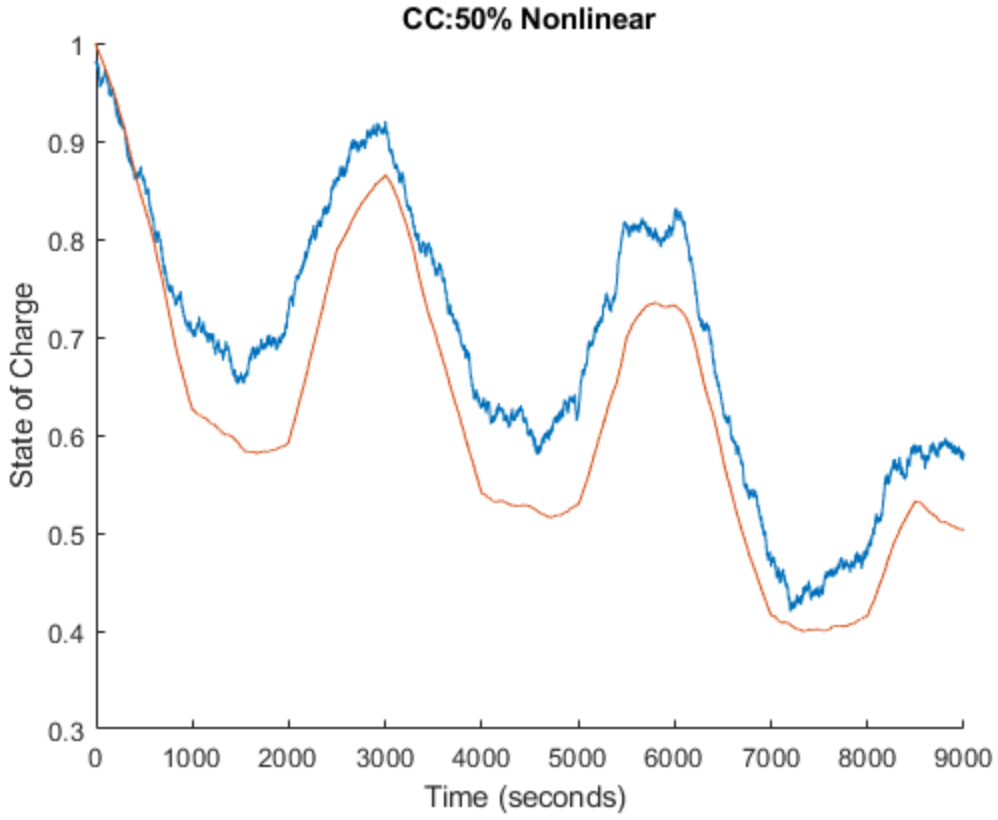

R0:25% Nonlinear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

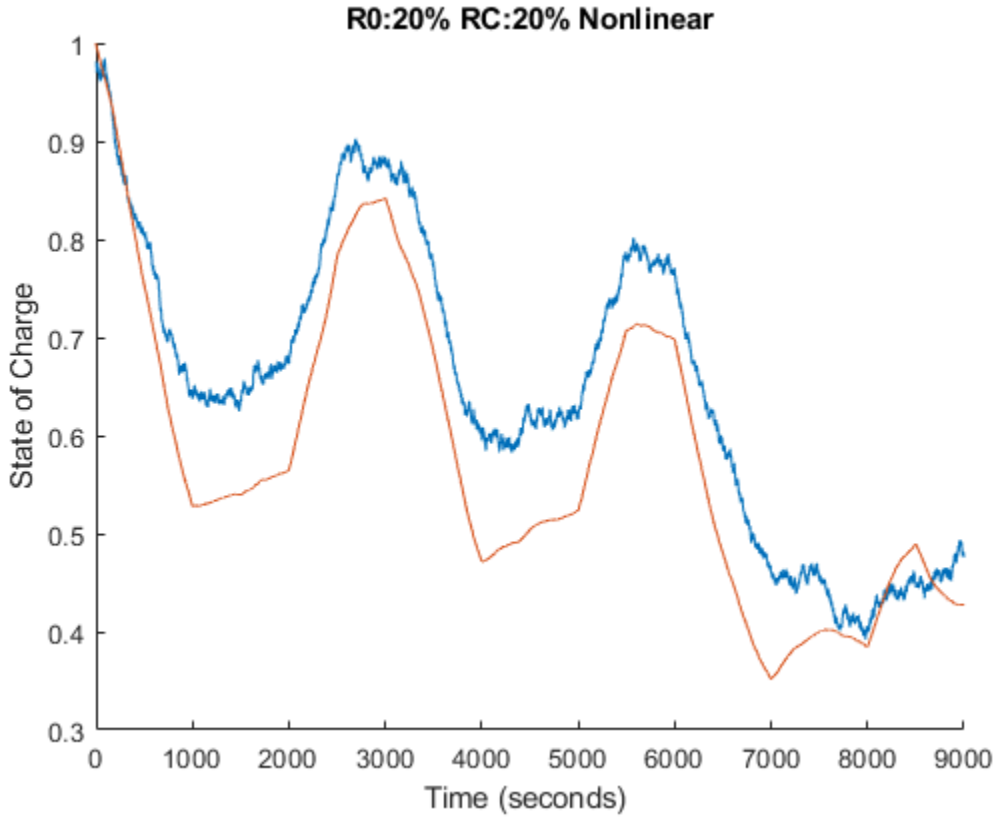

RC:25% Nonlinear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0
1000
2000
3000
4000
5000
6000
7000
8000
9000


CC:25% Nonlinear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000


R0:50% Nonlinear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000


RC:50% Nonlinear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000


CC:50% Nonlinear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

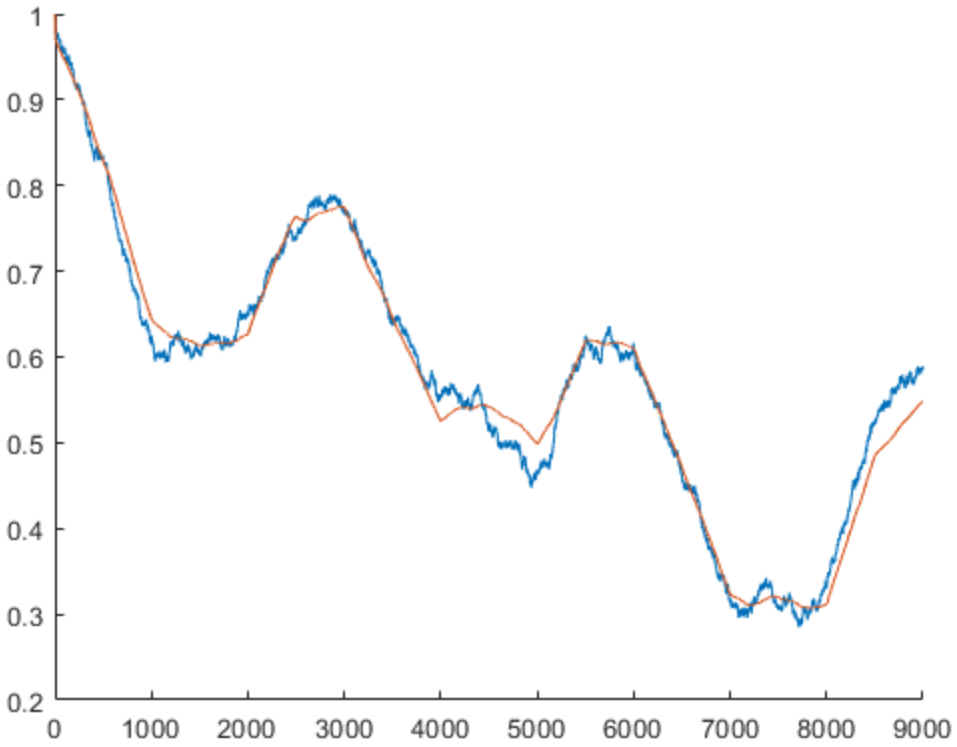

R0:20% RC:20% Nonlinear
State of Charge
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

Nonlinear Estimator Plant with Nonlinear BASELINE data

```
List =
 ["FirstOrderTruth_BASELINE_linear.mat","FirstOrderTruth_R0_25_linear.mat","FirstO


load('OCV_table.mat')
load('OCV_slope_table.mat')
load(List(9));

s = tf('s');

%battery model parameters
Rc = 0.015;    %Ohms
Cc = 2400;     %F
Cbat = 5*3600;
alpha =0.65;
R0 = 0.01;     %Ohms
Vocv0 = 3.435; %V

%tunning parameters
K = 1;         %gain
% zeta = 0.707;  %damping ratio
zeta = 0.5;  %damping ratio

wn = 75;      %natural frequency

%continuous time ss model
A = [-1/(Rc*Cc) 0; 0 0];
B = [1/Cc; -1/Cbat];
C = [-1 alpha];
D = -R0;

A1 = A(1,1);
B1 = B(1,1);
A2 = A(2,2);
B2 = B(2,1);
C2 = alpha;

SI = [s 0;0 s];
Gp = C*(inv(SI-A))*B+D;       %plant
T = minreal(K*wn^2/(s^2+2*zeta*wn*s+wn^2)); %complimentary
Y = minreal(T/Gp);            %youla
S = minreal(1-T);             %sensitivity
Gc = minreal(Y/S);            %controller
L = minreal(Gc*Gp);                    %open loop TF
sysTF = minreal(Gc*Gp/(1+Gc*Gp));      %actual sys TF
[num, den] = tfdata(Gc, 'v'); %get numerator and denominator of Gc tf

out = sim('Estimator_Simulink_trial2') ;   %run simulink model
```

```
est_soc = out.SOC_est;
tout = out.tout;

figure()
hold on
plot(t,SOC_act);
plot(tout,est_soc);
% %% Hinf Optimization Based Controller
%
%
% List =
 ["FirstOrderTruth_BASELINE_linear.mat","FirstOrderTruth_R0_25_linear.mat","FirstO
% NameList = ["Linear Baseline", "R0:25% Linear", "RC:25%
 Linear","CC:25% Linear","R0:50% Linear","RC:50% Linear","CC:50%
 Linear","R0:20% RC: 20%Linear","Nonlinear Baseline ","R0:25%
 Nonlinear","RC:25% Nonlinear","CC:25% Nonlinear","R0:50%
 Nonlinear","RC:50% Nonlinear","CC:50% Nonlinear", "R0:20% RC:20%
 Nonlinear" ];
%
%
% for k = 1:length(List)
%     load(List(k))
% s=tf('s');
%
% Rc = 0.015;    %Ohms
% Cc = 2400;     %F
% Cbat = 5*3600;
% alpha =0.65;
% R0 = 0.01;     %Ohms
% Vocv0 = 3.435; %V
%
% %tunning parameters
% K = 1;         %gain
% % zeta = 0.707;  %damping ratio
% zeta = 0.87;  %damping ratio
%
% wn = 60;      %natural frequency
%
% %continuous time ss model
% A = [-1/(Rc*Cc) 0; 0 0];
% B = [1/Cc; -1/Cbat];
% C = [-1 alpha];
% D = -R0;
%
% A1 = A(1,1);
% B1 = B(1,1);
% A2 = A(2,2);
% B2 = B(2,1);
% C2 = alpha;
%
%
% Gp = (-64800*s^2 - 2093.4*s - 0.65)/(18000*s*(36*s+1));  %plant
% sysg=ss(Gp);
% [Ag,Bg,Cg,Dg]=ssdata(sysg);
```

```
%
% Ag=Ag-0.08*eye(1); % slightly shift A to avoid poles on jw axis
%
% [num,den]=ss2tf(Ag,Bg,Cg,Dg);
% Gpn=tf(num,den); % perturbed plant
%
% % %Hinf shaping filter
% W1=(s+43)/(2*s+0.01); %(BW >= 65rad/s(CL at -6dB) so step input resp
 shows tracking at high freq) *input data at 10Hz
% W2=0.05;
% % W3=0.5;
% W3 = makeweight(1/2,43,50);  %(low freq gain, cross over freq, high
 freq gain)
%                               %^^to make it look like a
 differentiator!
%
% %Hinf Controller Computation
% ssga_=augtf(Gpn,W1,W2,W3);
% [sys3,sscl,GAM]=hinfsyn(ssga_);  %sys3=controller, sscl=CL tf (w/z),
%
% % Hinf Controller
% Gc=minreal(tf(sys3));
%
% [num, den] = tfdata(Gc, 'v'); %get numerator and denominator of Gc
 tf
%
% out = sim('Estimator_Simulink_trial1') ;   %run simulink model
%
% est_soc = out.SOC_est;
% tout = out.tout;
%
% figure()
% hold on
% plot(t,SOC_act);
% plot(tout,est_soc);
%
% end
```

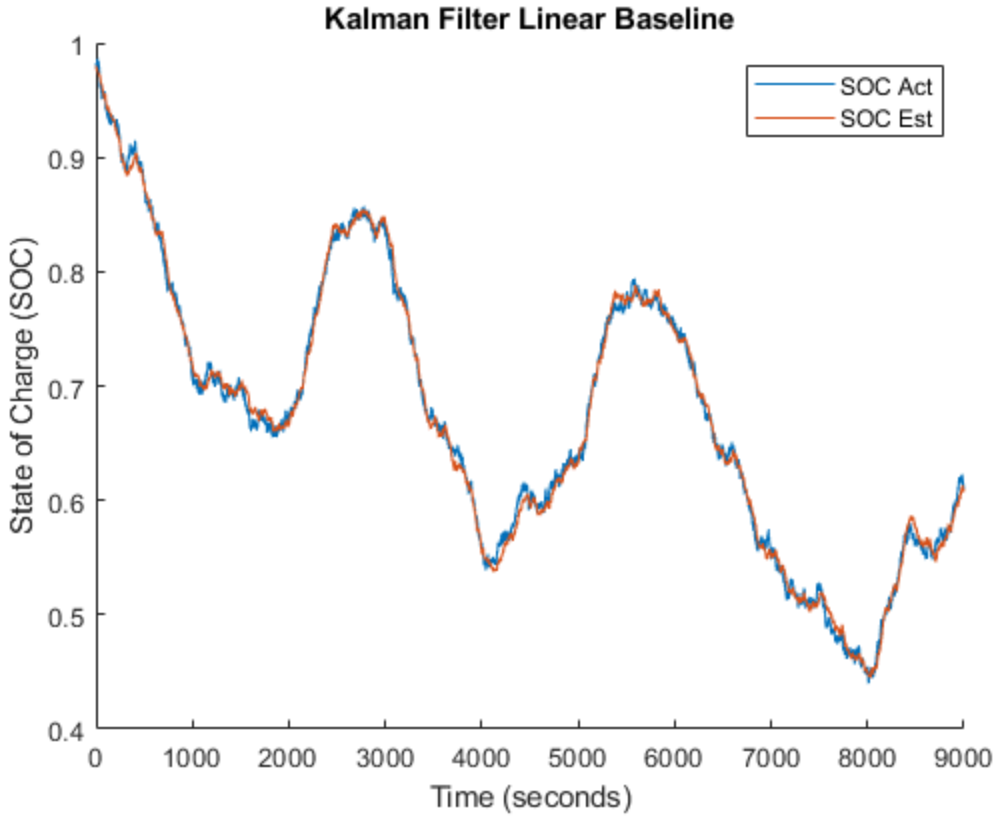


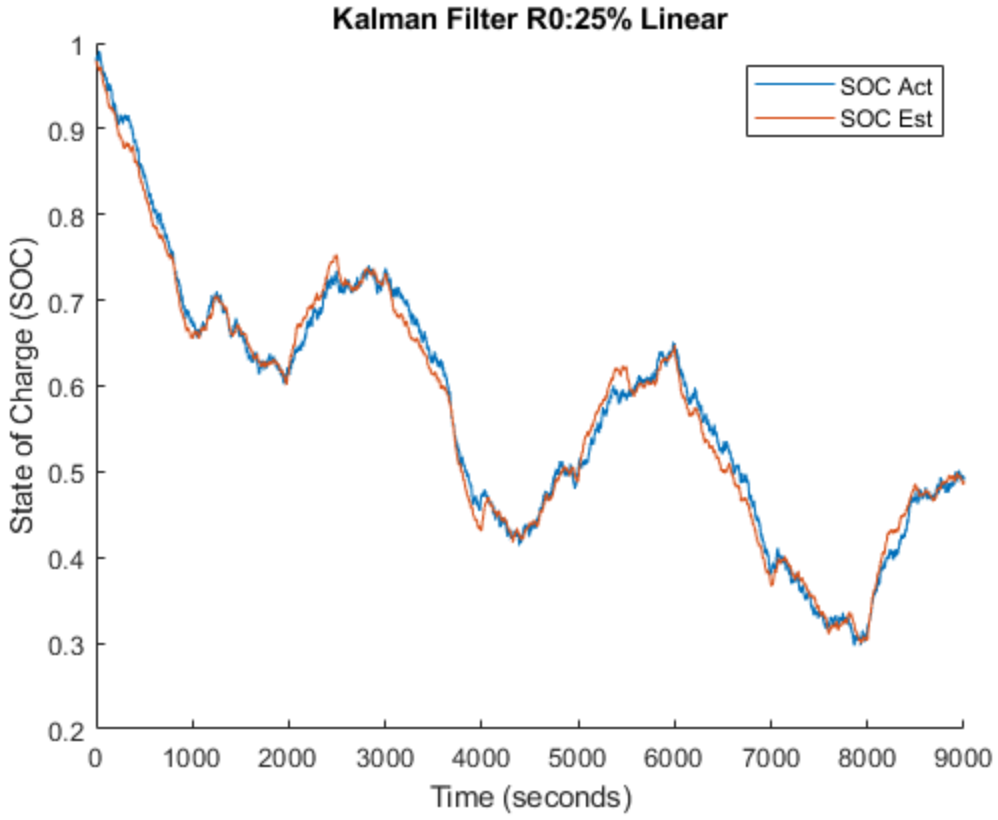
Kalman Filter

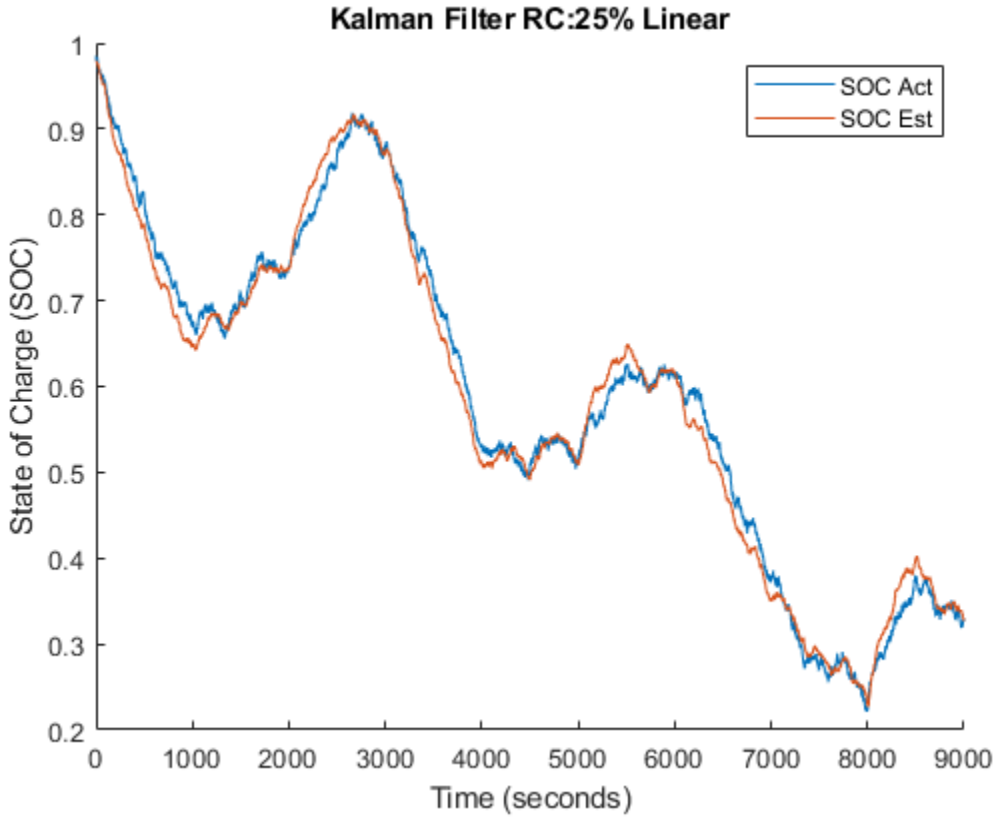
```
List =
 ["FirstOrderTruth_BASELINE_linear.mat","FirstOrderTruth_R0_25_linear.mat","FirstO
NameList = ["Linear Baseline", "R0:25% Linear", "RC:25%
 Linear","CC:25% Linear","R0:50% Linear","RC:50% Linear","CC:50%
 Linear","R0:20% RC: 20%Linear","Nonlinear Baseline ","R0:25%
 Nonlinear","RC:25% Nonlinear","CC:25% Nonlinear","R0:50%
 Nonlinear","RC:50% Nonlinear","CC:50% Nonlinear", "R0:20% RC:20%
 Nonlinear" ];


for i = 1:length(List)

    load(List(i));

R0 = 0.01;
Rc = 0.015;
Cc = 2400;
Cbat = (5*3600);
alpha = .65;
Voc_0 = 3.435;
SOC0 = 1;

dt = .1;
```

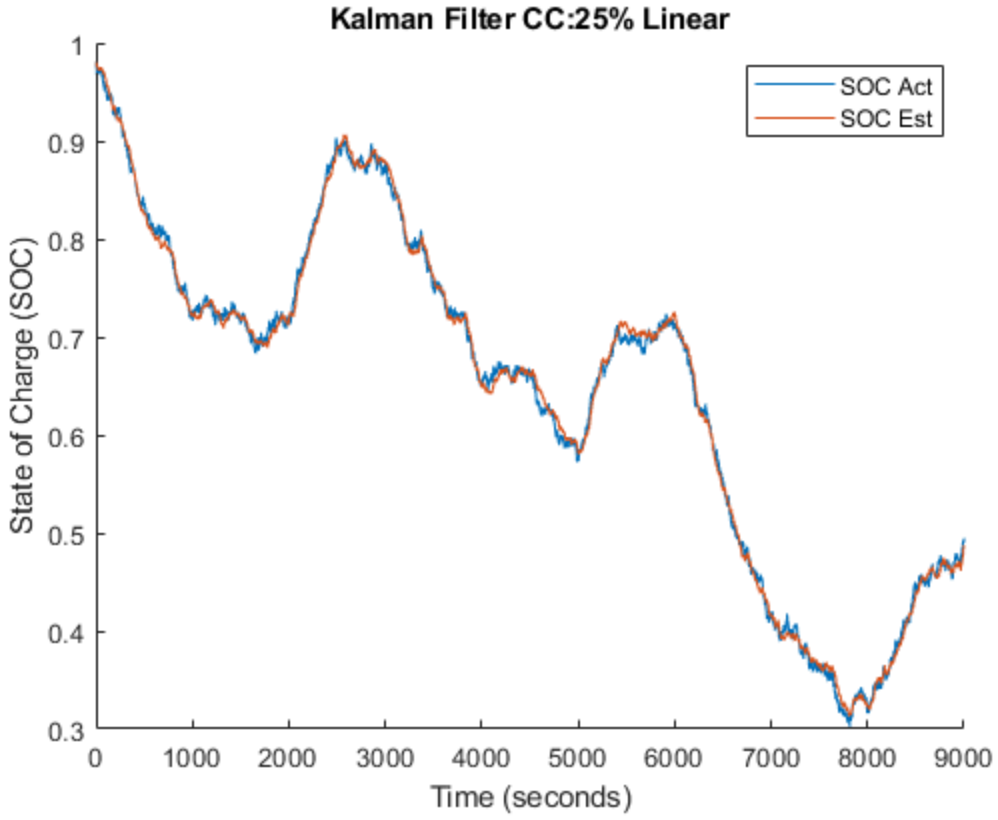
```
%System Dynamics
%Linear Model
A = [1 0 ; 0 (1-(dt/(Rc*Cc)))];
B = [-(dt/Cbat) ; (dt/Cc)];
C = [alpha -1];
D = [-R0];

%Kalman Model
Ak = [1];
Bk = [-(dt/Cbat)];
Ck=[alpha];
Dk=[R0];

% Kalman Filter

wk_mean = 0;
Q = 2.5*10^-7;
vk_mean = 0;
R = 1*10^-4;
P0 = 0;

Q = 2.5*10^-9;


% Set Initial Conditions
P(1) = P0;
x1(1) = .98; % SOC
x2(1) = 0;  % Vc
x1_hat(1) = .98;

for k = 2:1:length(t)
    %State Equations:
    x1(k) = x1(k-1)-(dt/Cbat)*I(k-1); %+normrnd(0,Q);  % SOC
    x2(k) = (1-(dt/(Rc*Cc)))*x2(k-1)+(dt/Cc)*I(k-1);  % Vc

    % Open Loop:
    x1_ol(k) = Ak*x1(k-1)+Bk*I(k-1);

    % Model Prediction:
    x1_hat_prev(k) = Ak*x1_hat(k-1)+ Bk*I(k-1);
    P_prev = Ak*P(k-1)*Ak' + Q;

    %Estimated Output:
    V_hat = alpha*x1_hat_prev(k)-x2(k)- R0*I(k)+ Voc_0;

    % Measurement Update:
    x1_hat(k) = x1_hat_prev(k) +
 P_prev*Ck'*inv(Ck*P_prev*Ck'+R)*(V(k)-V_hat);
    P(k) = P_prev -P_prev*Ck'*inv(Ck*P_prev*Ck'+R)*Ck*P_prev;


end
```

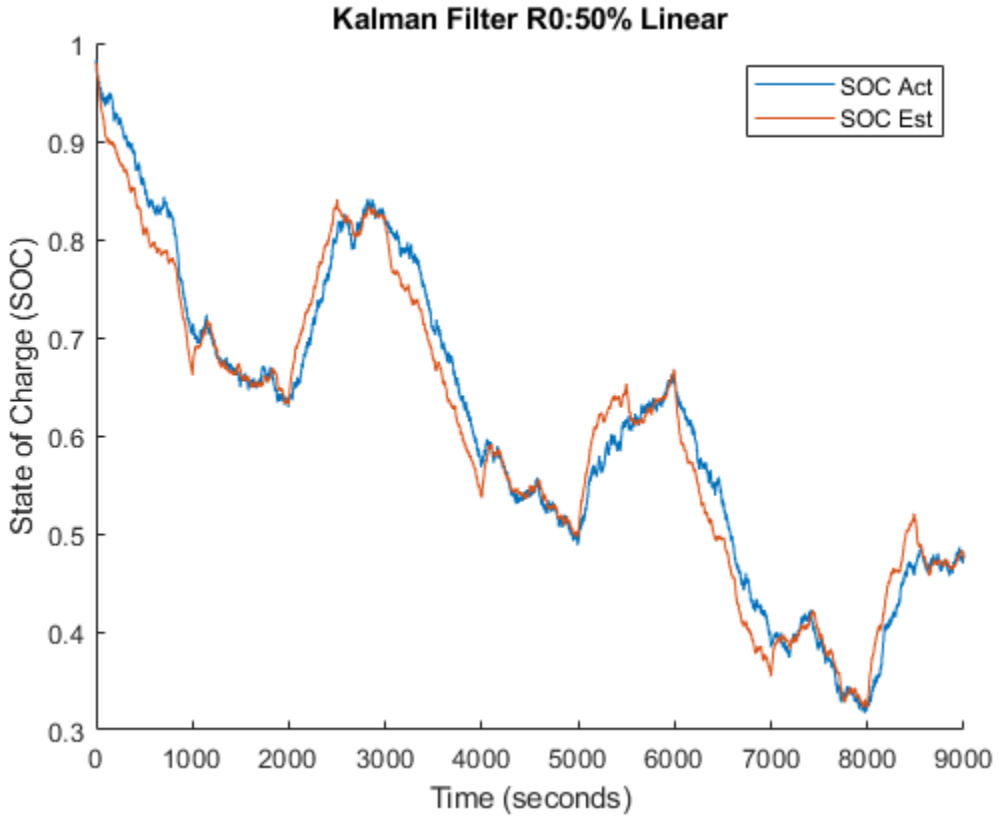
```
KALMAN_ACTUAL(:,i)= SOC_act;
KALMAN_ESTIMATED(:,i)=x1_hat;

figure();
hold on
plot(t,SOC_act)
plot(t,x1_hat)
title('Kalman Filter '+ NameList(i));
xlabel('Time (seconds)');
ylabel('State of Charge (SOC)');
legend('SOC Act','SOC Est');

end
```

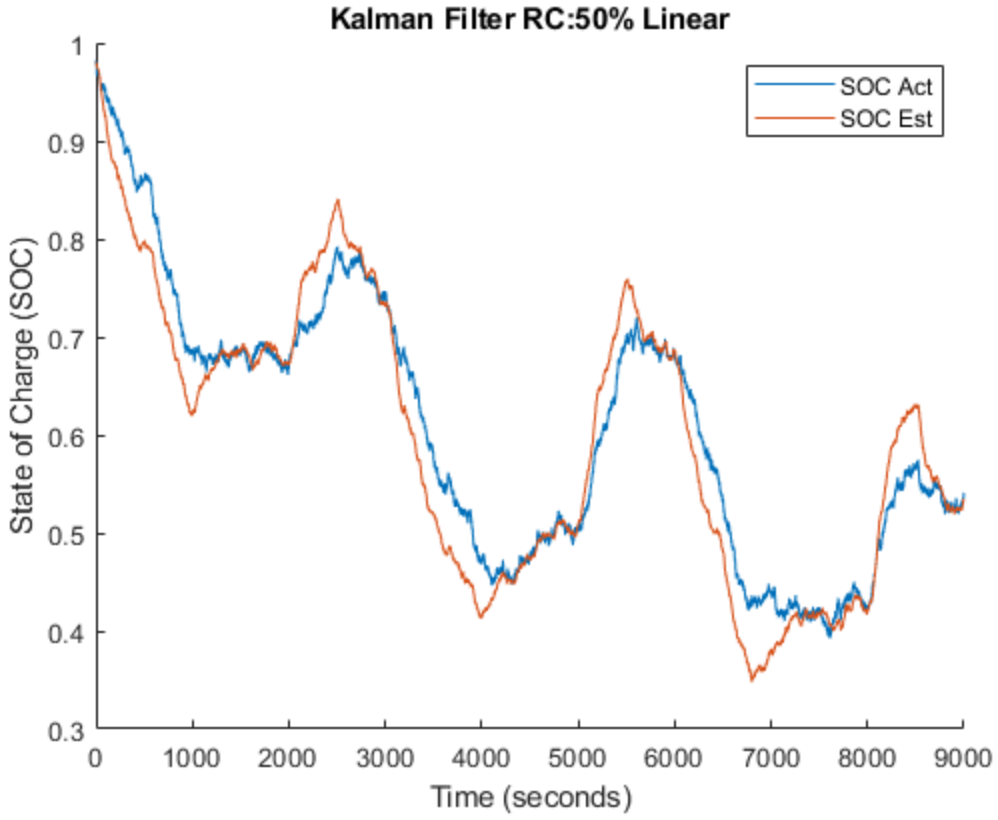


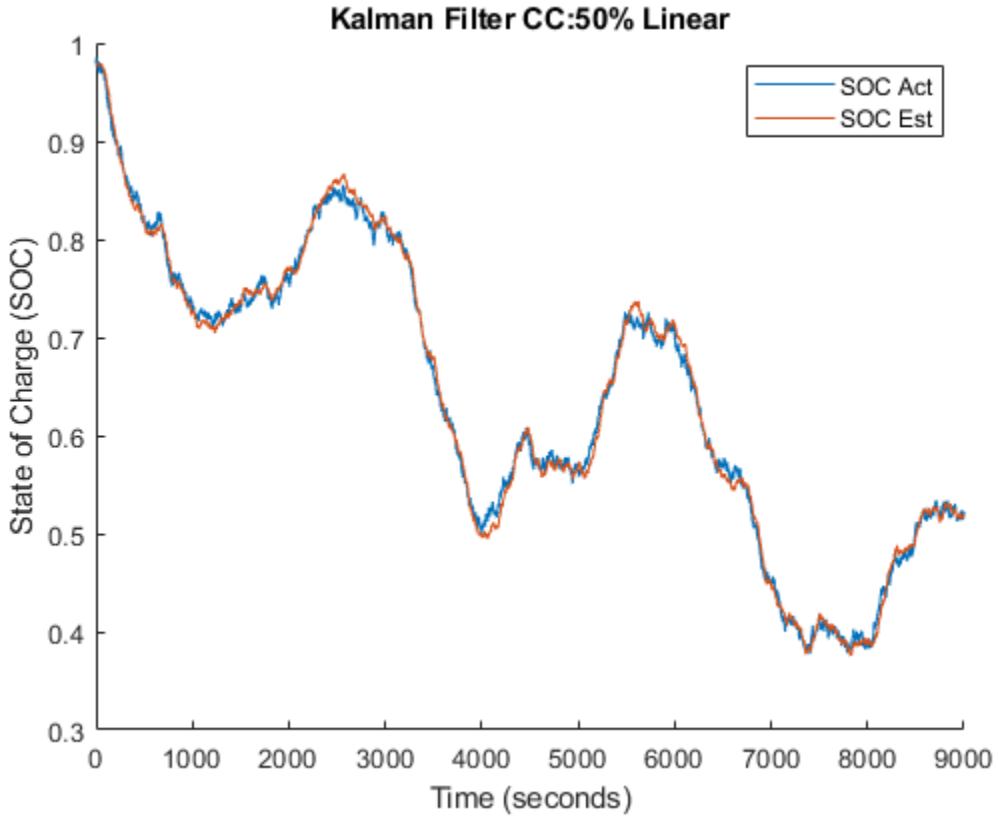

Kalman Filter R0:25% Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

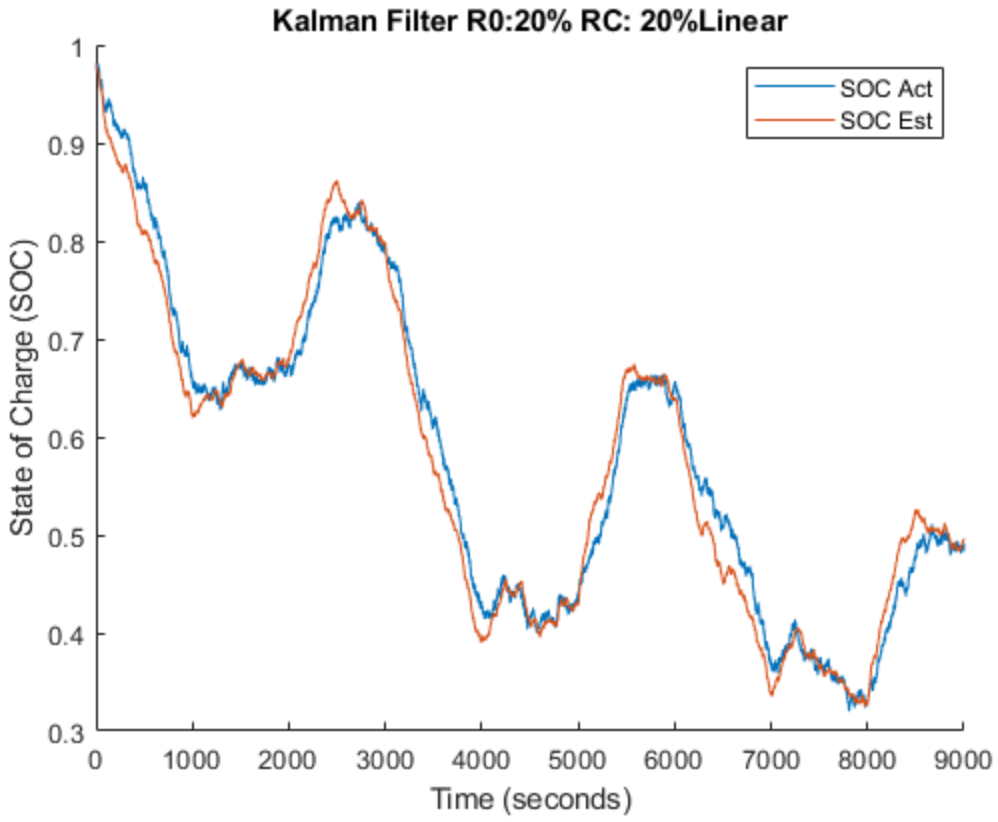

Kalman Filter RC:25% Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

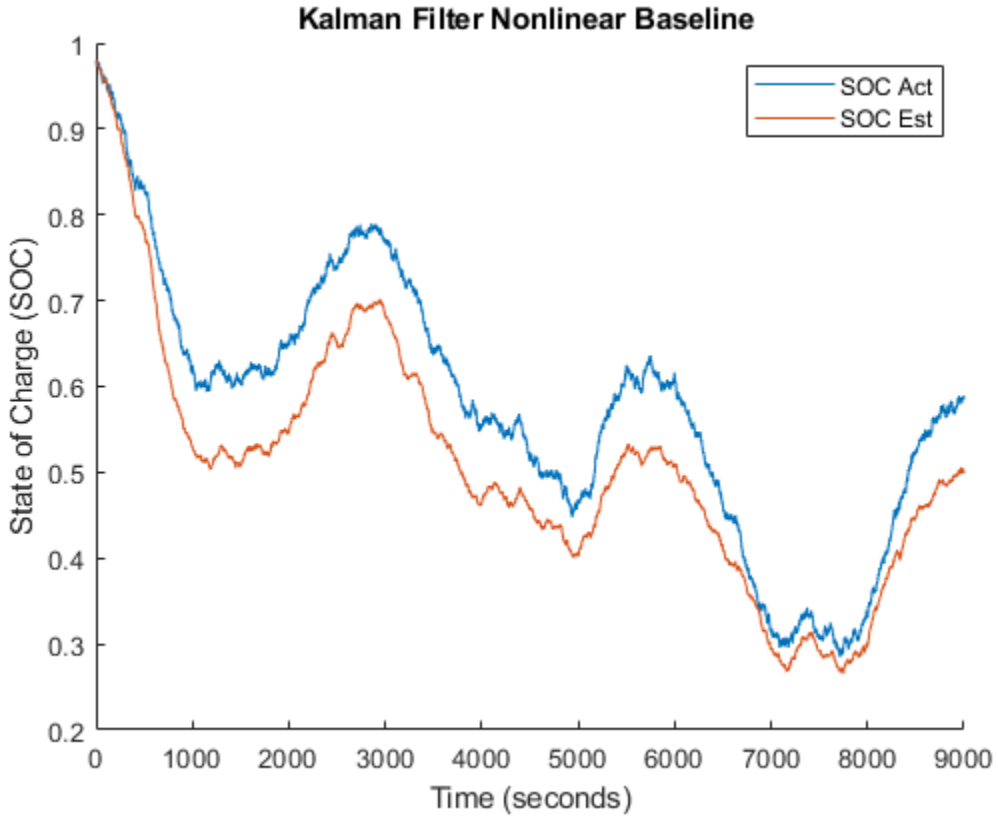

Kalman Filter CC:25% Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

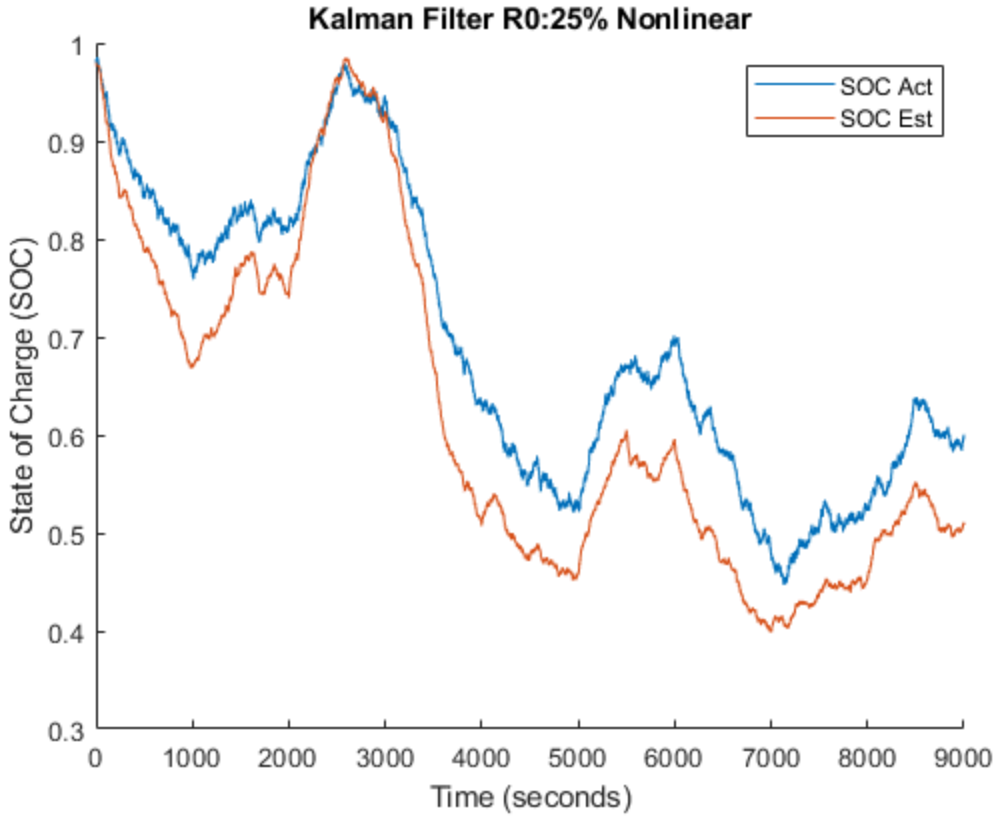

Kalman Filter R0:50% Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

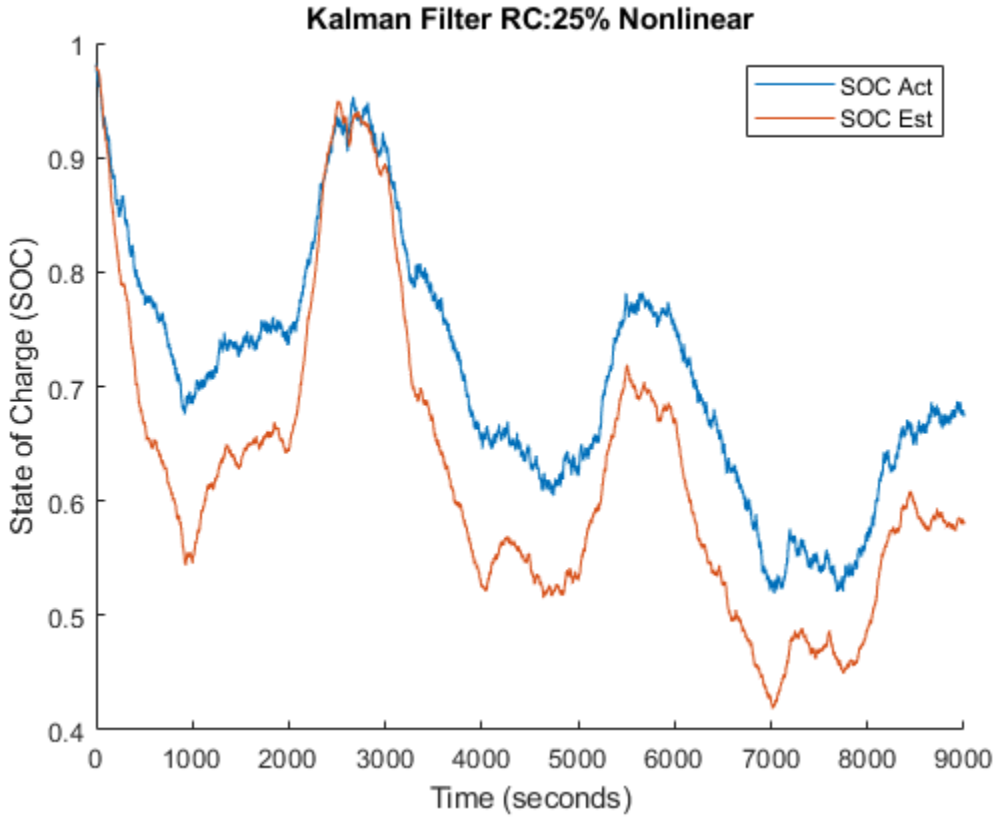

Kalman Filter RC:50% Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

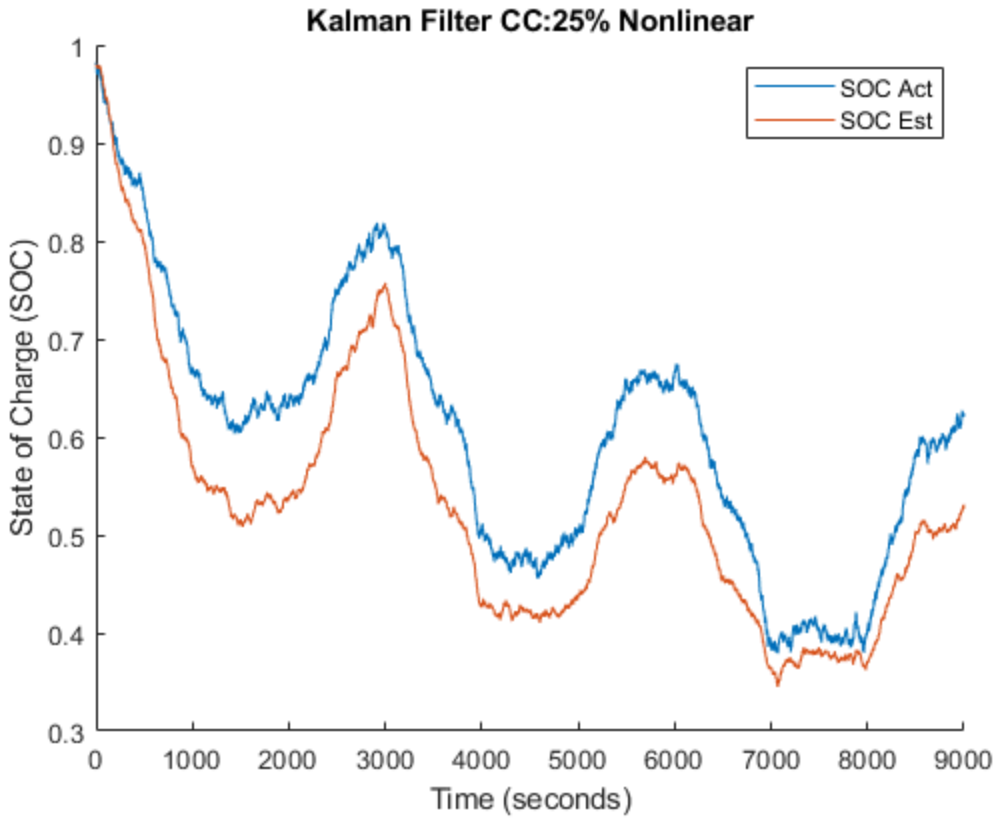

Kalman Filter CC:50% Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

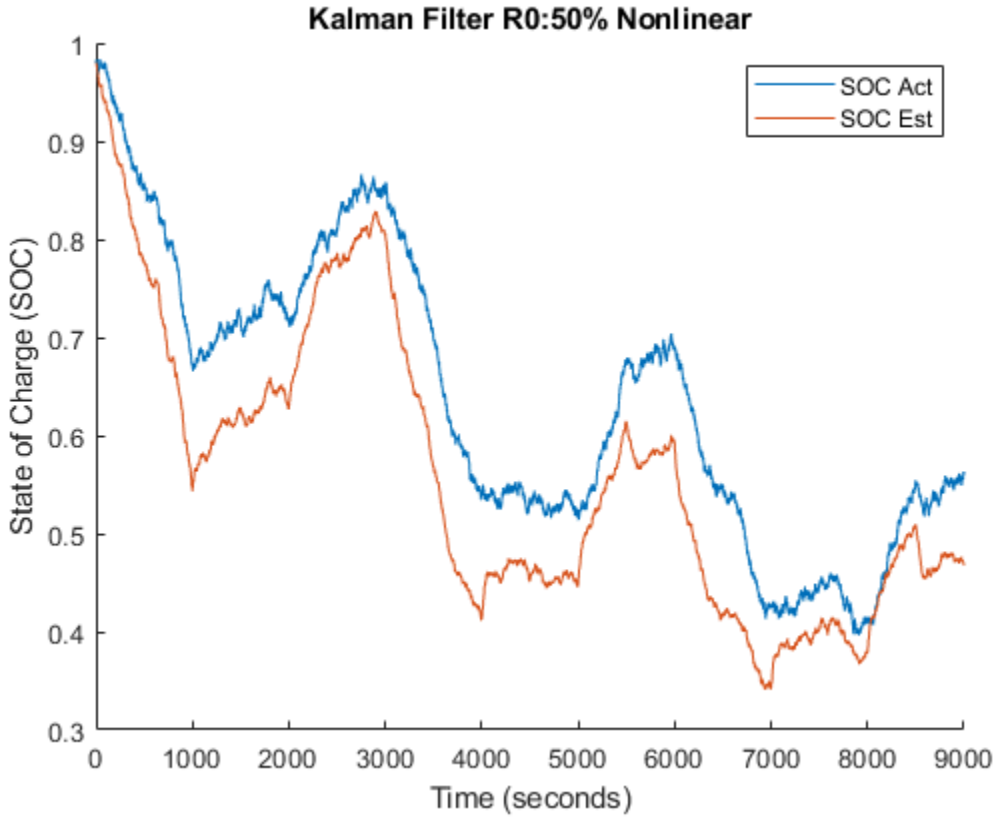

Kalman Filter R0:20% RC: 20%Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

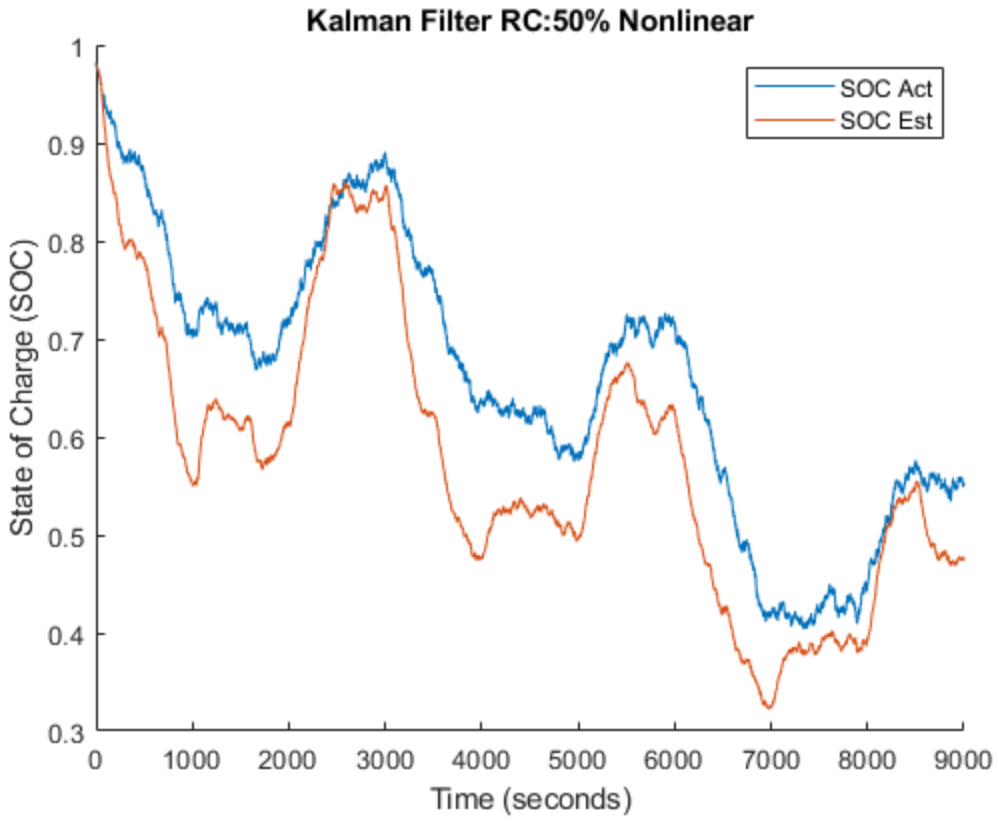

Kalman Filter Nonlinear Baseline
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

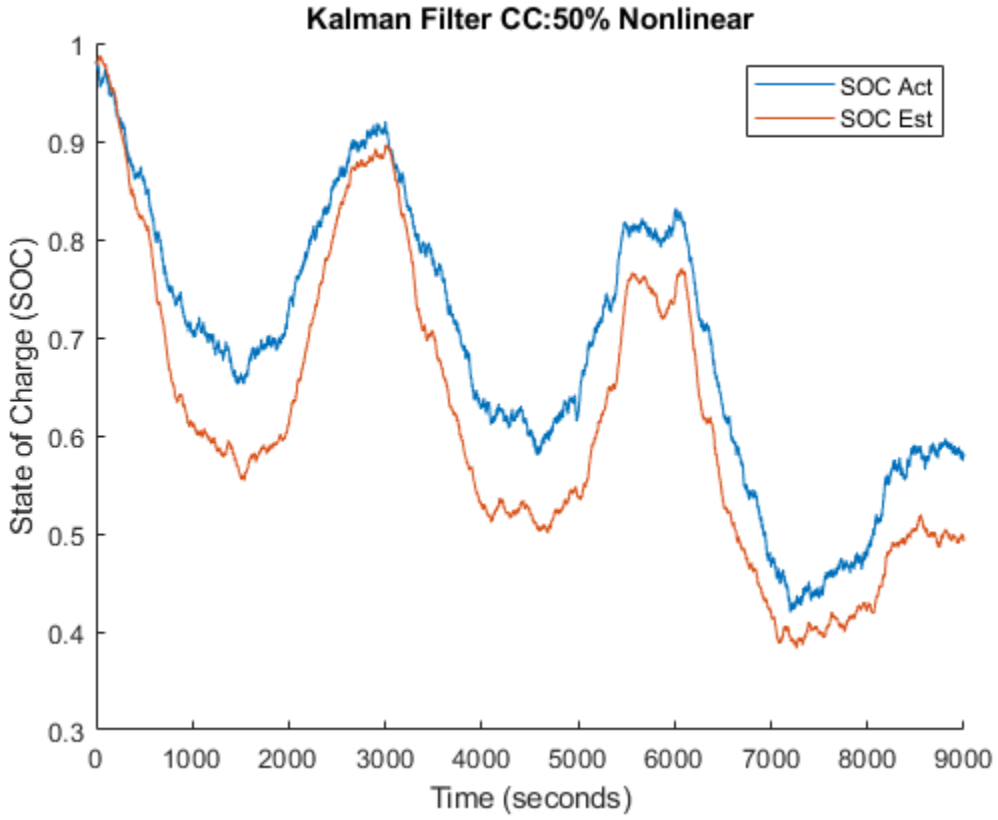

Kalman Filter R0:25% Nonlinear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

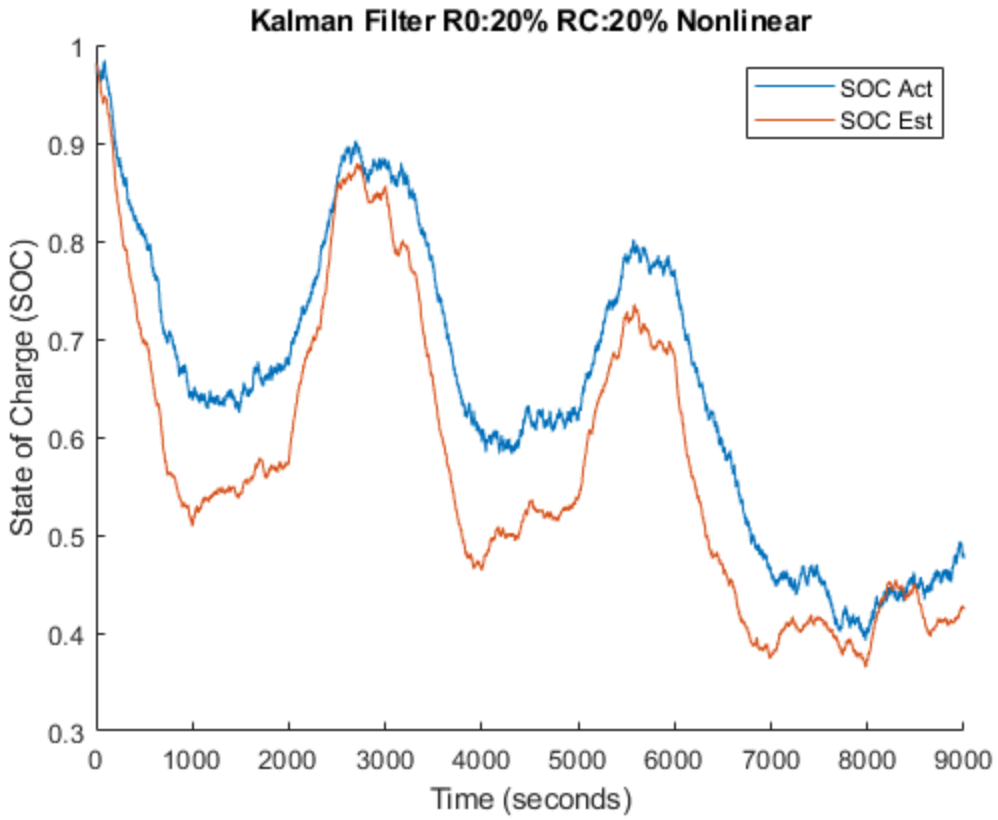

Kalman Filter RC:25% Nonlinear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0
1000
2000
3000
4000
5000
6000
7000
8000
9000


Kalman Filter CC:25% Nonlinear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000


Kalman Filter R0:50% Nonlinear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000


Kalman Filter RC:50% Nonlinear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000


Kalman Filter CC:50% Nonlinear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

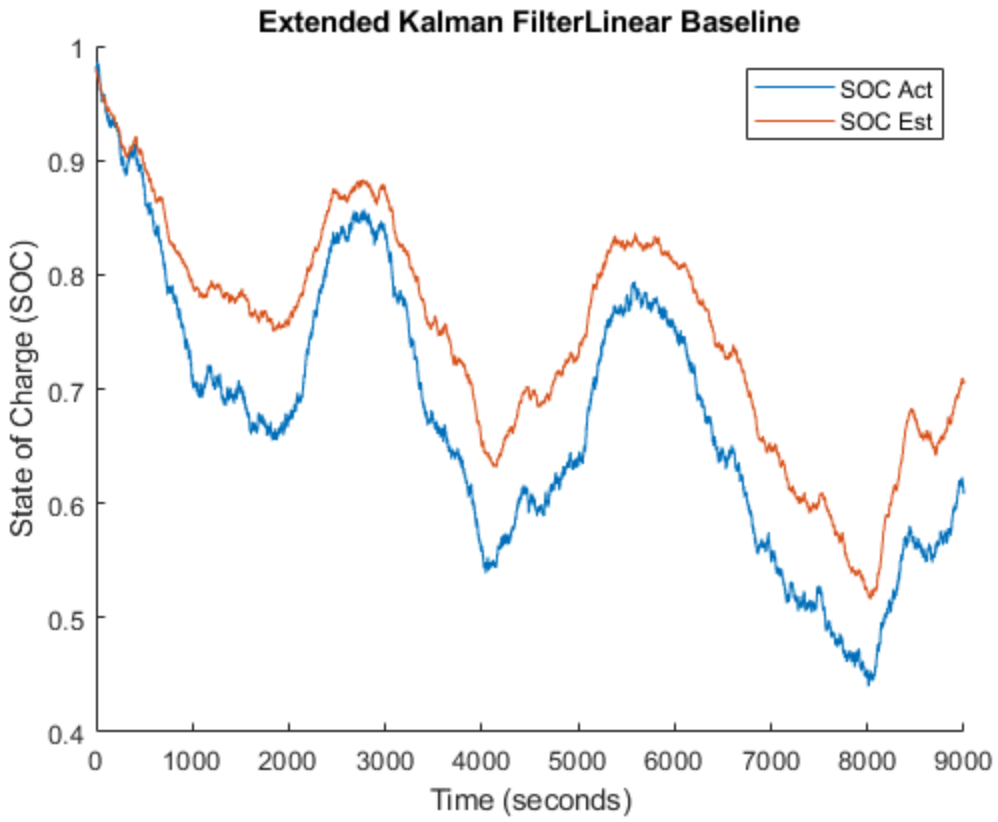


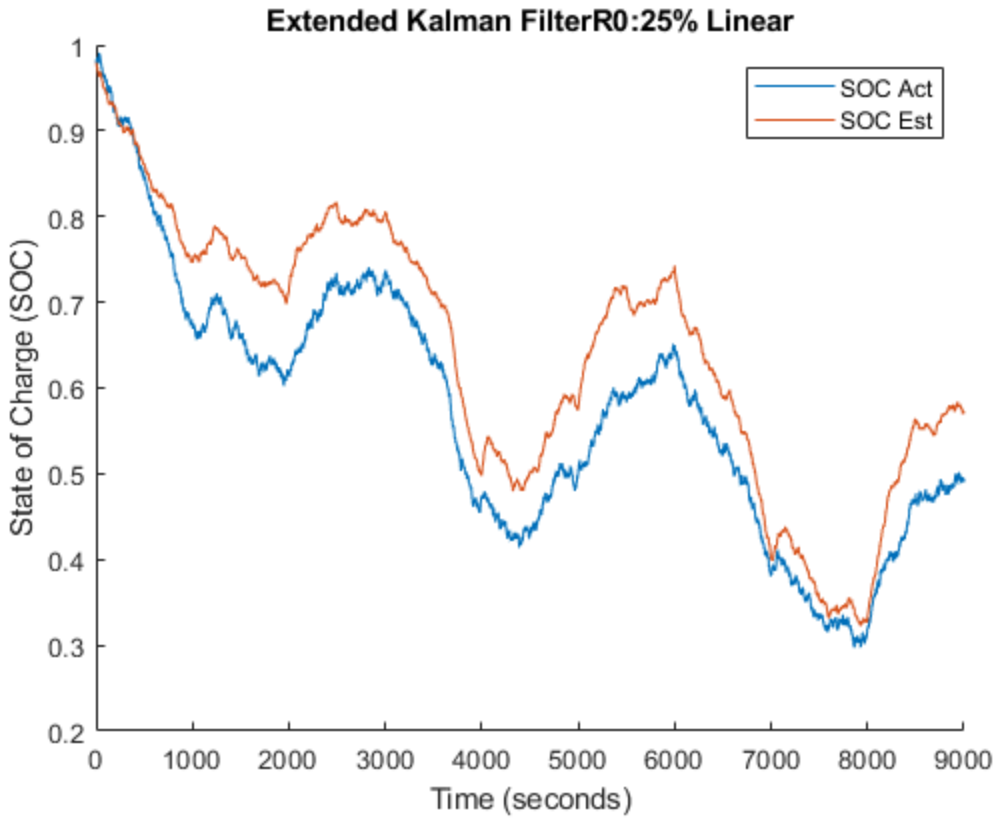
Extended Kalman Filter

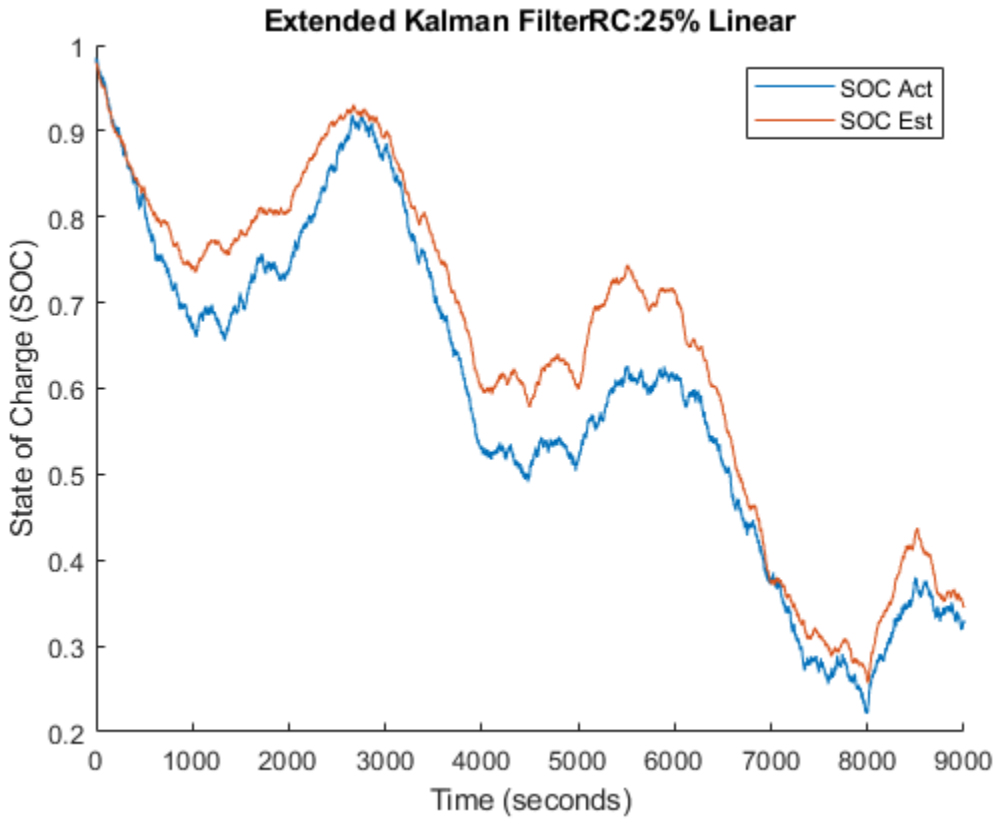
```
List =
 ["FirstOrderTruth_BASELINE_linear.mat","FirstOrderTruth_R0_25_linear.mat","FirstO
NameList = ["Linear Baseline", "R0:25% Linear", "RC:25%
 Linear","CC:25% Linear","R0:50% Linear","RC:50% Linear","CC:50%
 Linear","R0:20% RC: 20%Linear","Nonlinear Baseline ","R0:25%
 Nonlinear","RC:25% Nonlinear","CC:25% Nonlinear","R0:50%
 Nonlinear","RC:50% Nonlinear","CC:50% Nonlinear", "R0:20% RC:20%
 Nonlinear" ];


for i = 1:length(List)

    load(List(i));

R0 = 0.01;
Rc = 0.015;
Cc = 2400;
Cbat = (5*3600);
alpha = .65;
Voc_0 = 3.435;
SOC0 = 1;

dt = .1;
```

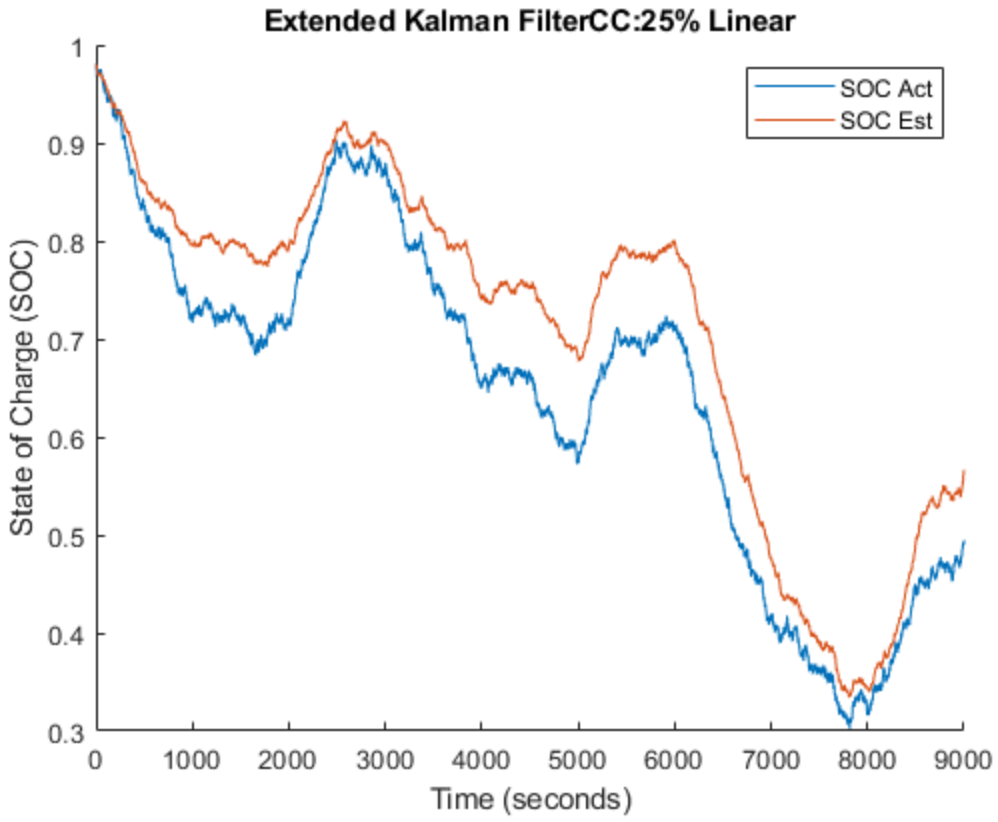
```
wk_mean = 0;
Q = 2.5*10^-7;
vk_mean = 0;
R = 1*10^-4;
P0 = 0;

A_ek = 1 ;

% function val = C_ek(SOC_hat)
%  val = intep1(soc_intpts_OCV_slope,OCV_slope_intpts,SOC_hat);
% end
E_ek = 1;
F_ek = 1;

Ak = 1;
Bk = -dt/Cbat;

load('OCV_table.mat')
load('OCV_slope_table.mat')
% load('IV_data_nonlinear')

R = 1*10^-4;
Q = 2.5*10^-9;

% Set Initial Conditions
P(1) = P0;
x1(1) = .98; % SOC
x2(1) = 0;  % Vc
x1_hat(1) = .98;

for k = 2:length(t)
    x1(k) = x1(k-1)-(dt/Cbat)*I(k-1); %+normrnd(0,Q);  % SOC - Coulomb
 Counting
    x2(k) = (1-(dt/(Rc*Cc)))*x2(k-1)+(dt/Cc)*I(k-1);  % Vc


    C_ek =
 interp1(soc_intpts_OCV_slope' ,OCV_slope_intpts,x1_hat(k-1));

    % Model Prediction:
    x1_hat_prev = Ak*x1_hat(k-1)+Bk*I(k-1);
    P_prev = A_ek*P(k-1)*A_ek'+E_ek*Q*E_ek';

    % Measurement Update:
    V_hat = interp1(soc_intpts_OCV' ,OCV_intpts,x1_hat(k-1)) -I(k)*R0-
x2(k);
    L = P_prev*C_ek'*inv(C_ek*P_prev*C_ek'+F_ek*R*F_ek');

    x1_hat(k) = x1_hat_prev + L*(V(k)-V_hat);
    P(k) = P_prev -L*C_ek*P_prev;

end
```

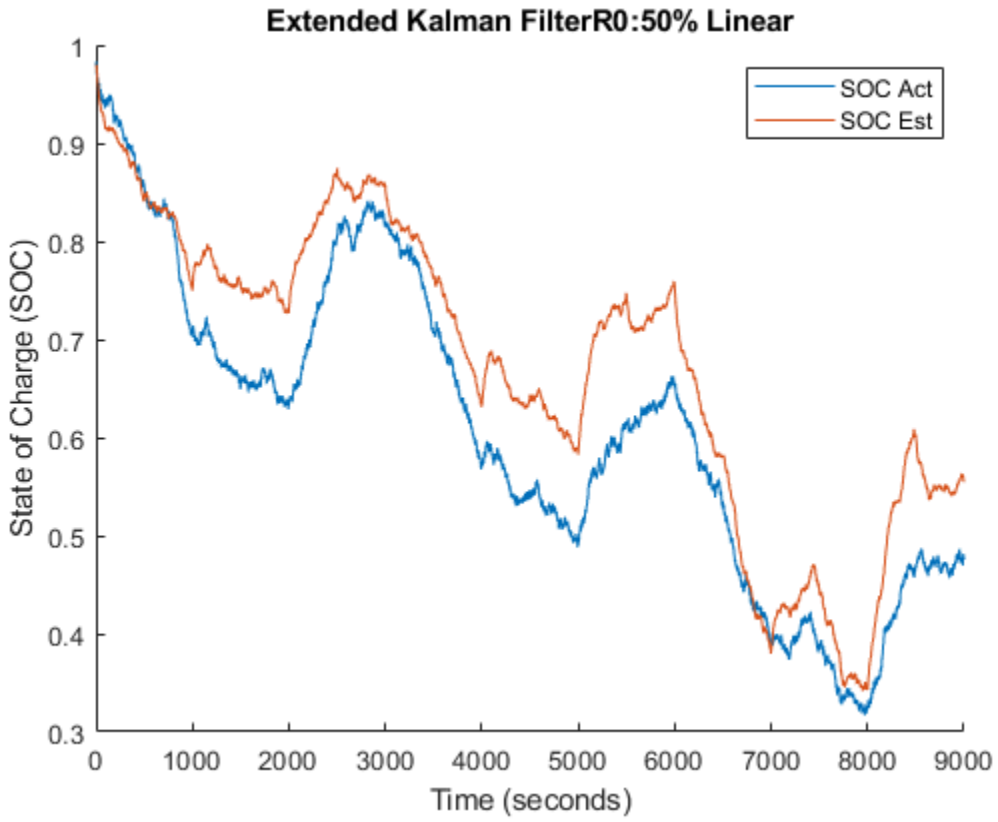
```
EXTENDED_KALMAN_ACTUAL(:,i)= SOC_act;
EXTENDED_KALMAN_ESTIMATED(:,i)=x1_hat;

figure();
hold on
plot(t,SOC_act)
plot(t,x1_hat)
title('Extended Kalman Filter' + NameList(i));
xlabel('Time (seconds)');
ylabel('State of Charge (SOC)');

legend('SOC Act','SOC Est');


end
```

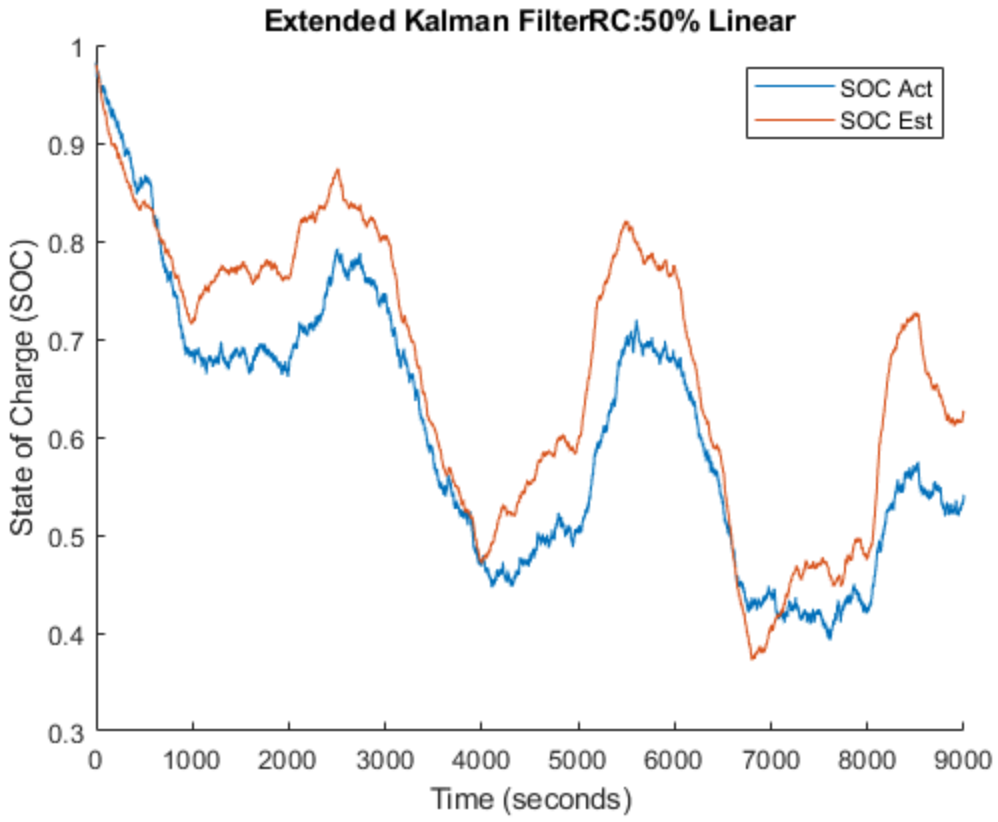


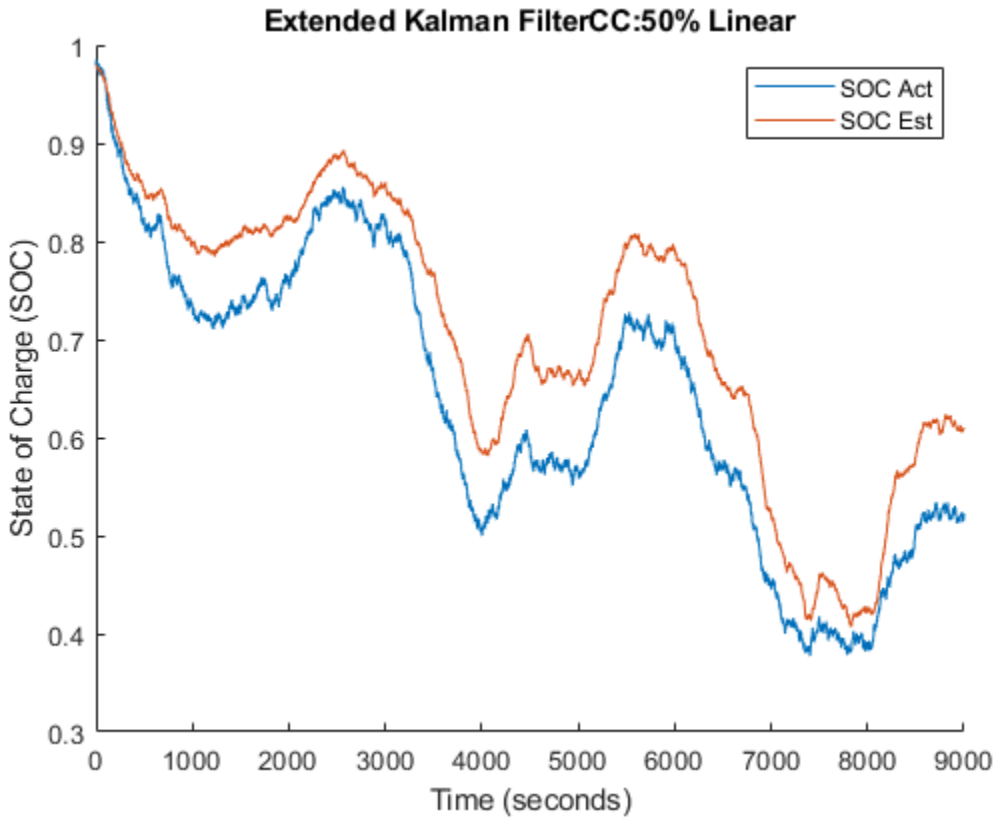

Extended Kalman FilterR0:25% Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

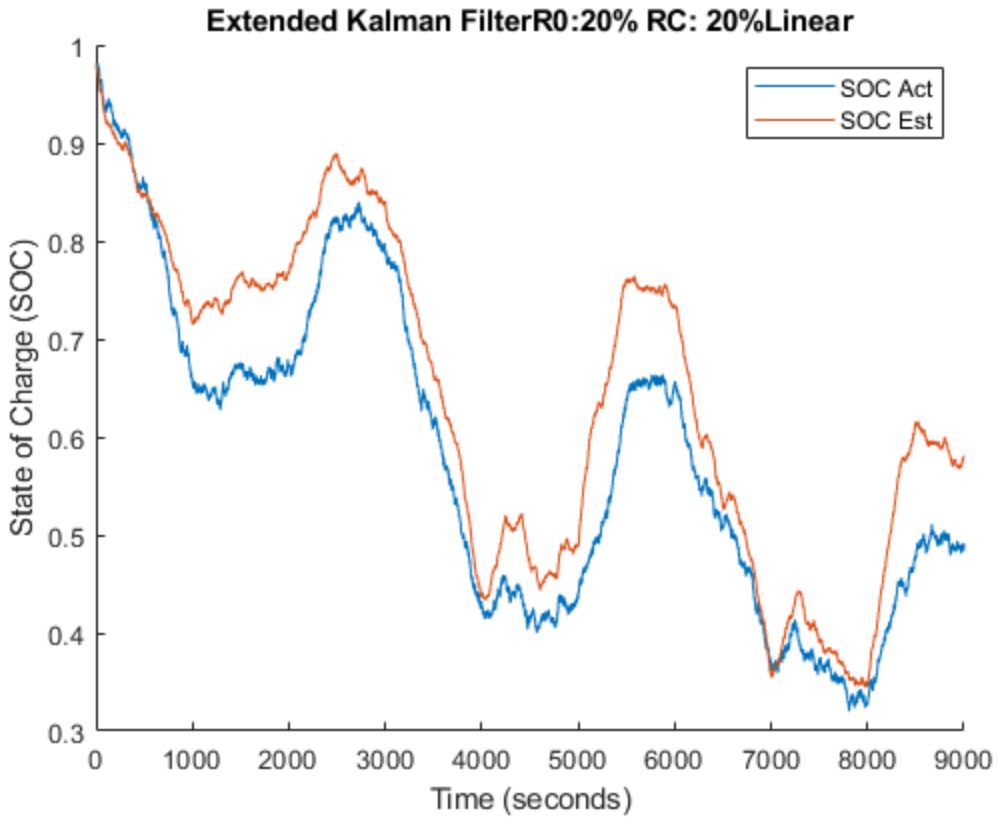

Extended Kalman FilterRC:25% Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

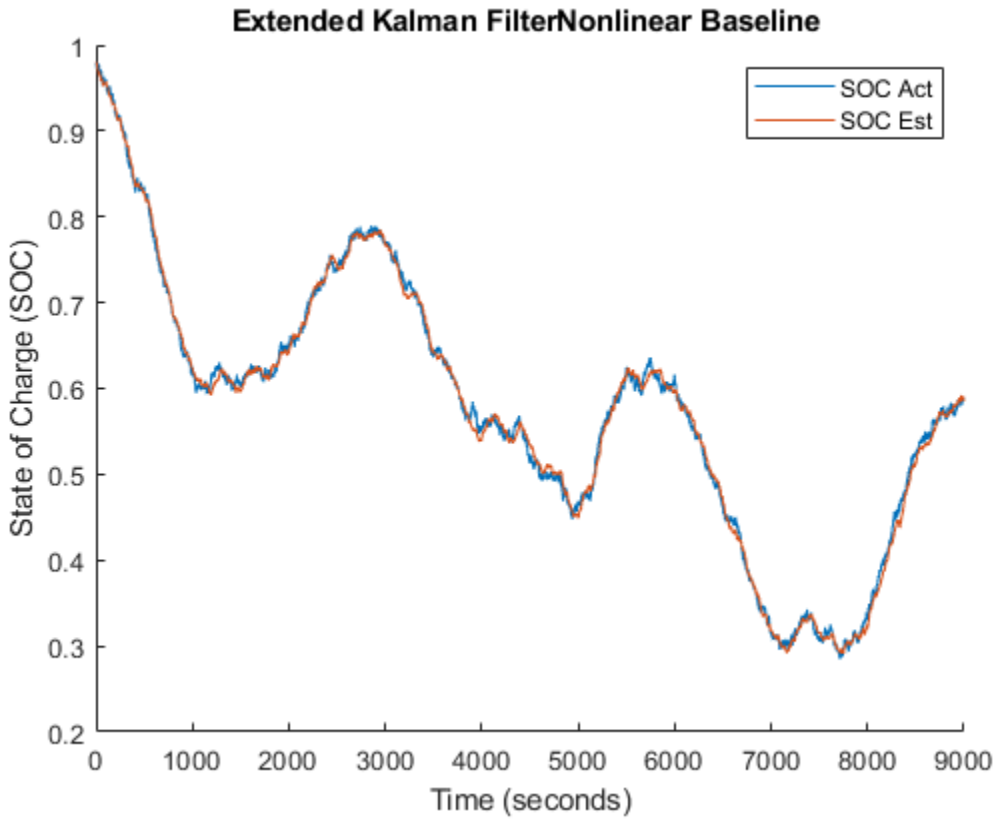

Extended Kalman FilterCC:25% Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

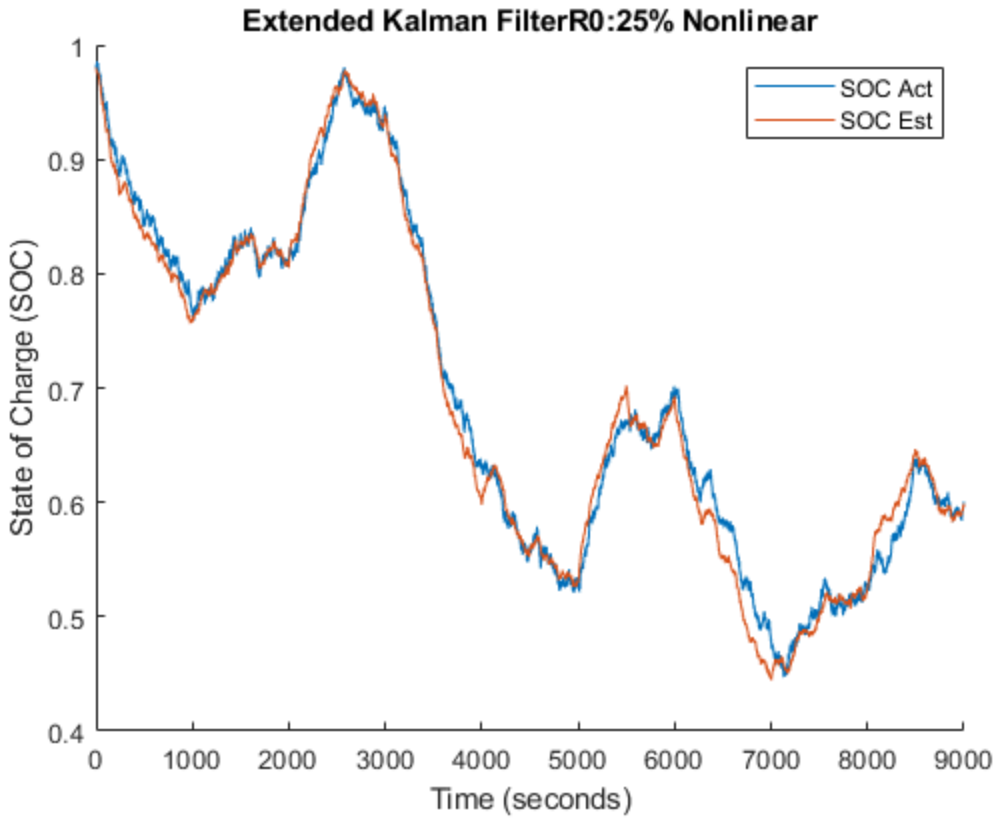

Extended Kalman FilterR0:50% Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

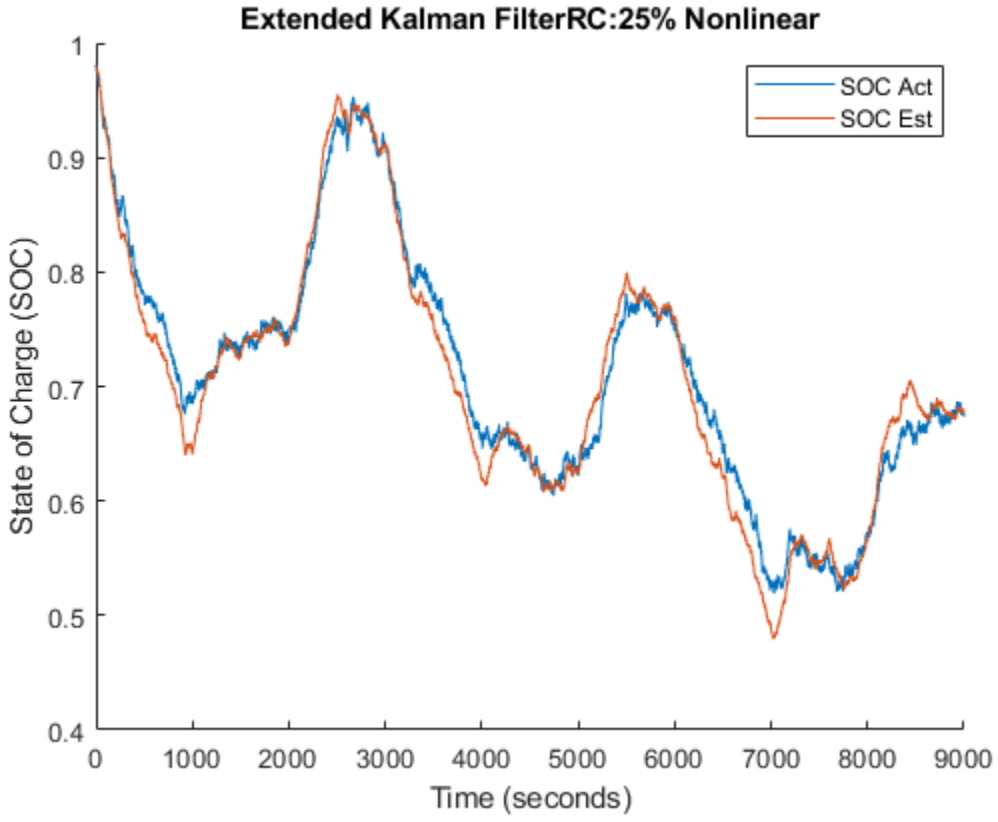

Extended Kalman FilterRC:50% Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

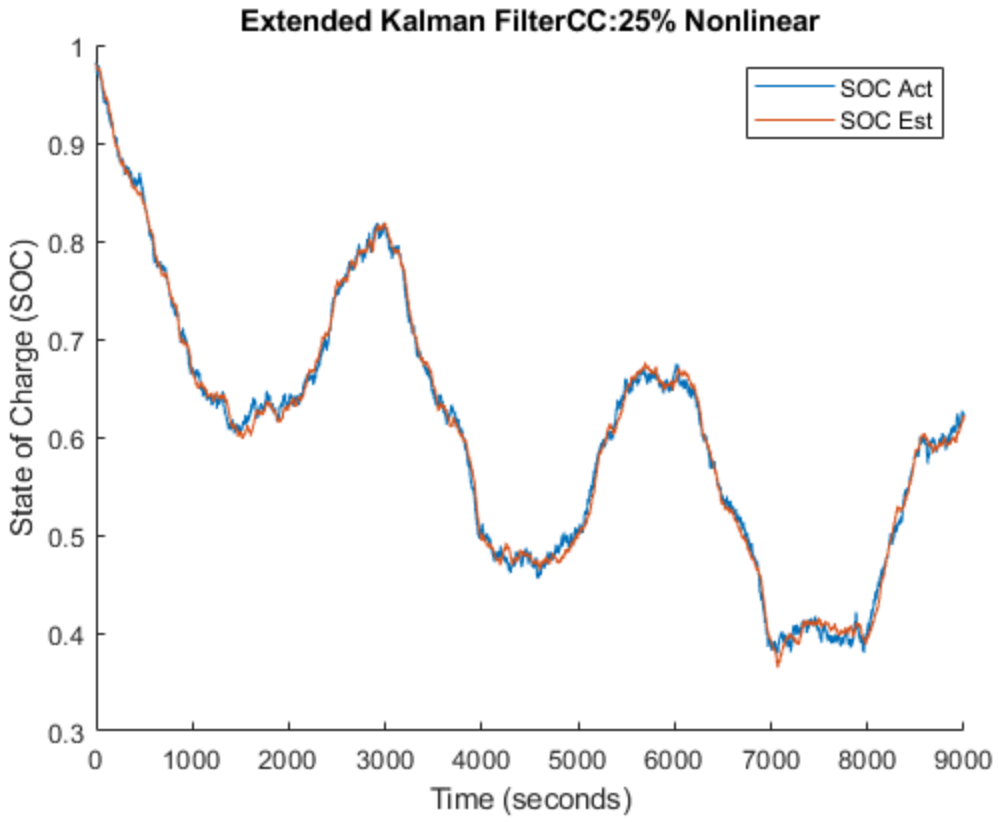

Extended Kalman FilterCC:50% Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

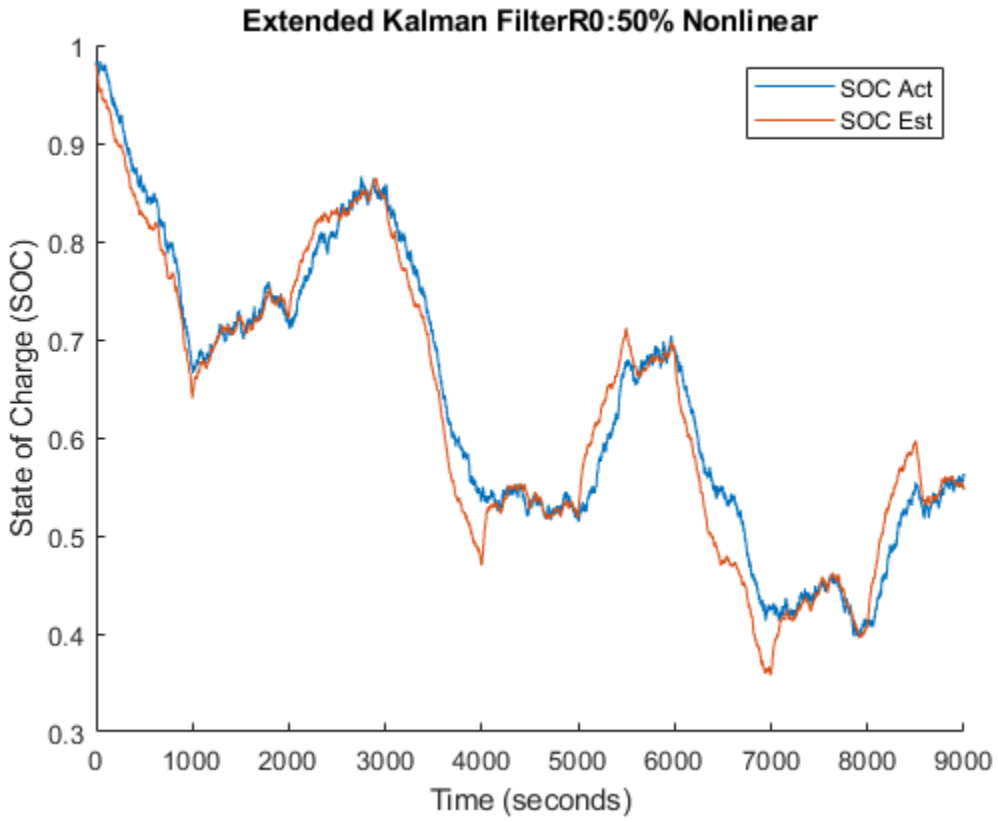

Extended Kalman FilterR0:20% RC: 20%Linear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

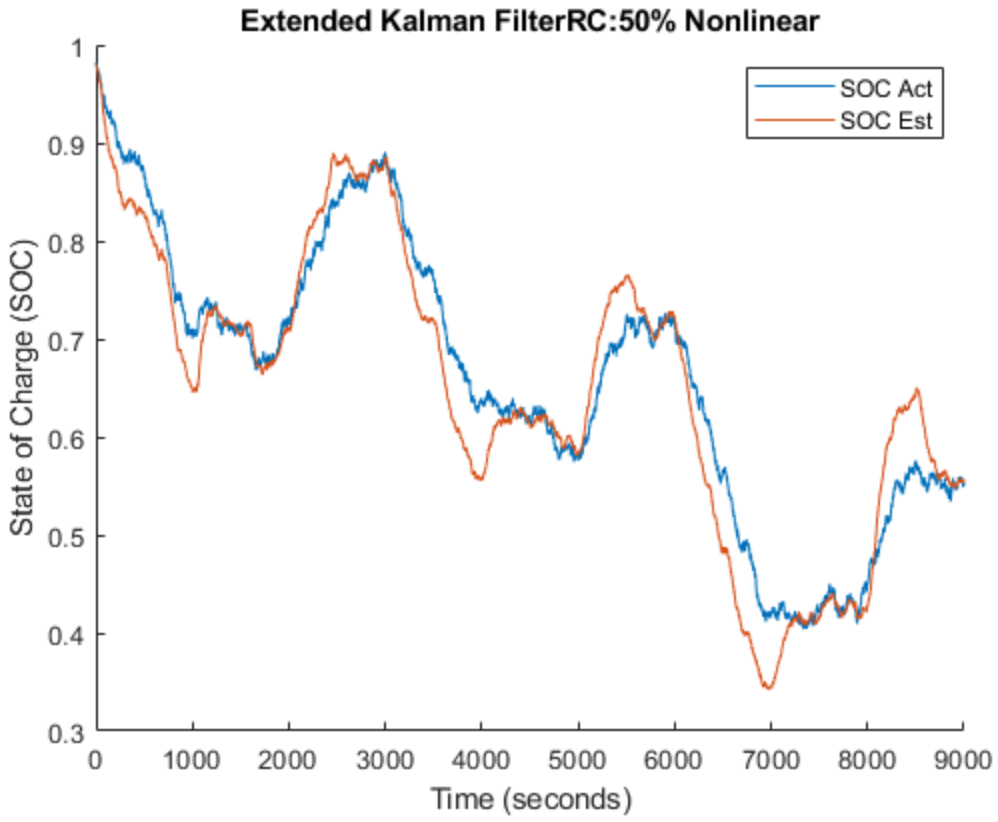

Extended Kalman FilterNonlinear Baseline
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

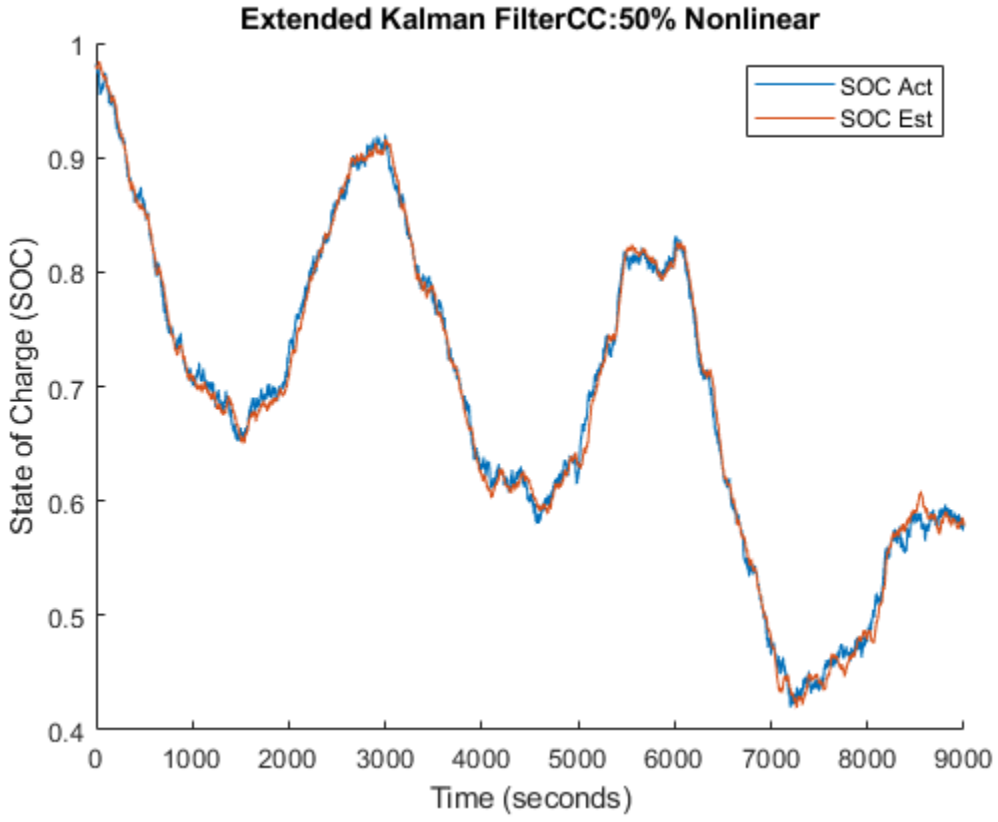

Extended Kalman FilterR0:25% Nonlinear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

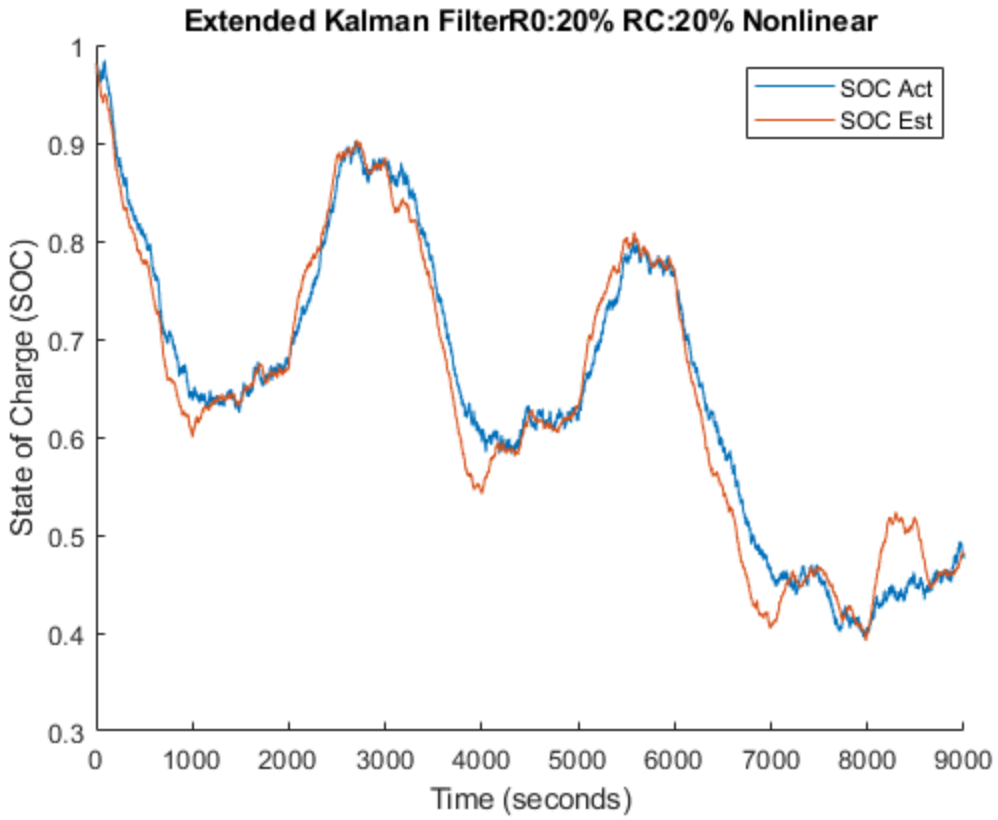

Extended Kalman FilterRC:25% Nonlinear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0
1000
2000
3000
4000
5000
6000
7000
8000
9000


Extended Kalman FilterCC:25% Nonlinear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000


Extended Kalman FilterR0:50% Nonlinear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000


Extended Kalman FilterRC:50% Nonlinear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
1000
2000
3000
4000
5000
6000
7000
8000
9000

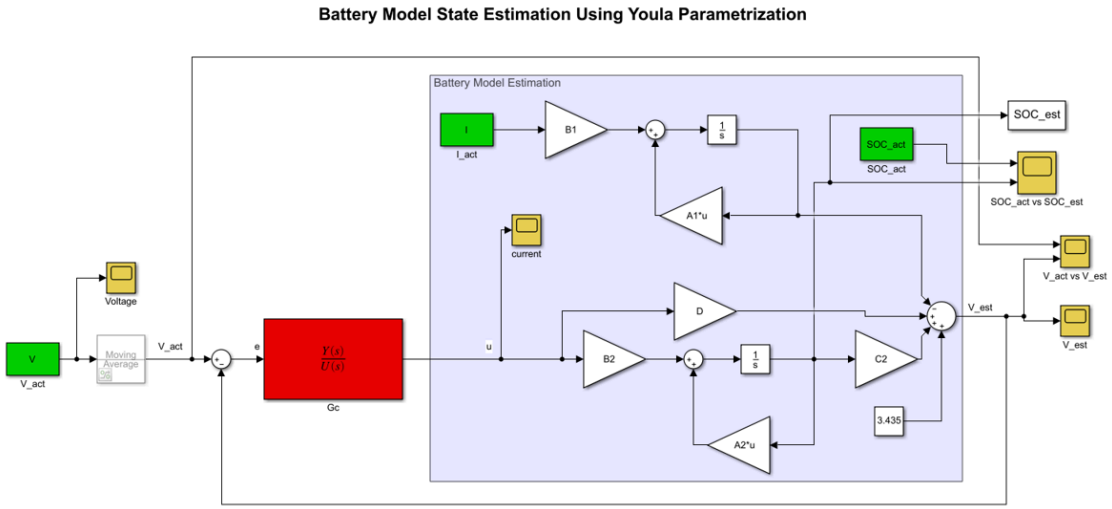

Extended Kalman FilterCC:50% Nonlinear
SOC Act
SOC Est
State of Charge (SOC)
Time (seconds)
1
0.9
0.8
0.7
0.6
0.5
0.4
0
1000
2000
3000
4000
5000
6000
7000
8000
9000



RMS Tabulation

```
NameList = ["Linear Baseline", "R0:25% Linear", "RC:25%
 Linear","CC:25% Linear","R0:50% Linear","RC:50% Linear","CC:50%
 Linear","R0:20% RC: 20%Linear","Nonlinear Baseline ","R0:25%
 Nonlinear","RC:25% Nonlinear","CC:25% Nonlinear","R0:50%
 Nonlinear","RC:50% Nonlinear","CC:50% Nonlinear", "R0:20% RC:20%
 Nonlinear" ];


for i = 1:length(List)


Youla_act_rms(1,i) = rms(YOULA_ACTUAL(:,i));
Youla_est_rms(1,i) = rms(YOULA_ESTIMATED(:,i));

Kalman_act_rms(1,i) = rms(KALMAN_ACTUAL(:,i));
Kalman_est_rms(1,i) = rms(KALMAN_ESTIMATED(:,i));

EKF_act_rms(1,i) = rms(EXTENDED_KALMAN_ACTUAL(:,1));
EKF_est_rms(1,i) = rms(EXTENDED_KALMAN_ESTIMATED(:,1));
end

table = [ " ", NameList;...
         "Youla Actual RMS", Youla_act_rms;...
```

```
            "Youla Estimated RMS", Youla_est_rms;...
            "Youla RMS Error", (Youla_act_rms-Youla_est_rms);...

            "Kalman Filter Actual RMS", Kalman_act_rms;...
            "Kalman Filter Estimated RMS",Kalman_est_rms;...
            "Kalman RMS Error", (Kalman_act_rms-Kalman_est_rms);...

            "EKF Actual RMS", EKF_act_rms; ...
            "EKF Estimated RMS", EKF_est_rms;...
            "EKF RMS Error", (EKF_act_rms-EKF_est_rms)];

open_system('Estimator_Simulink_trial1')
```



Published with MATLAB® R2019a