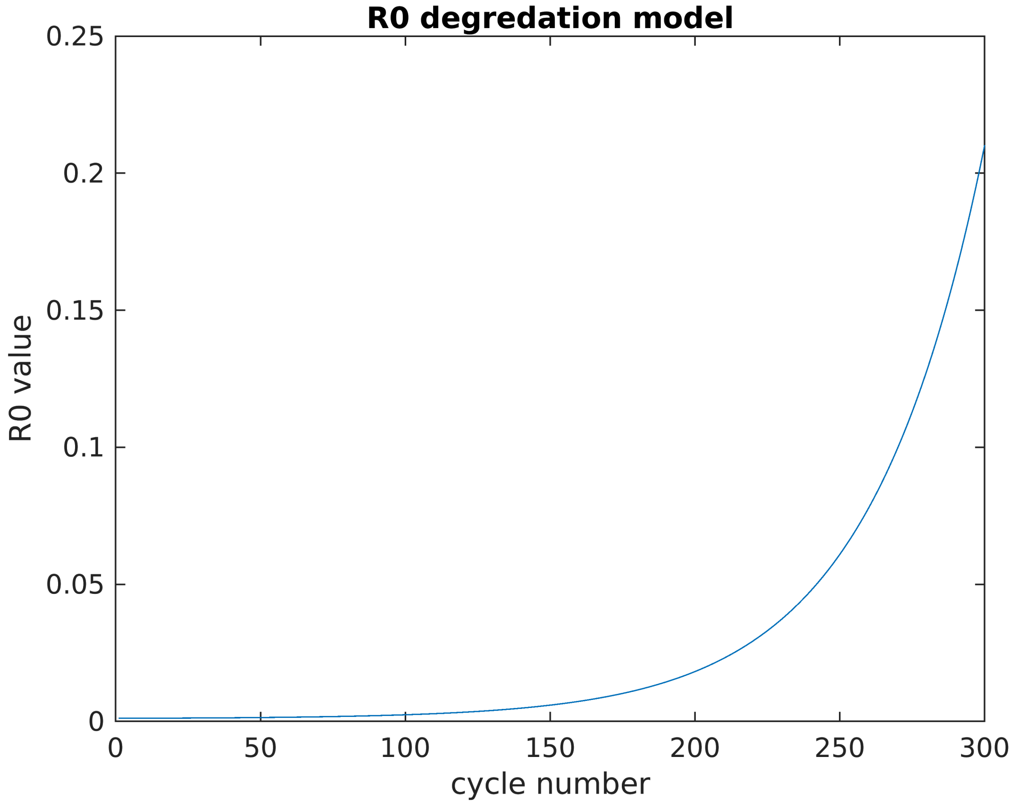
# Battery Degradation Models

```
load batteryParams.mat;
load ukfBatteryParams.mat;
global batteryParams;
```

## R0

- degrade from .0011368 to .212 exponentially

```
R0 = zeros(300,1);
R0(1) = .0011368;
for i = 2:300
    R0(i) = degradeR0(R0(i-1), i);
end
f1 = figure(1); clf;
plot(R0);
title("Resistance Degredation Model");
ylabel("R0 value");
xlabel("cycle number");
```



```
disp(R0(1));
```

```
0.0011
```

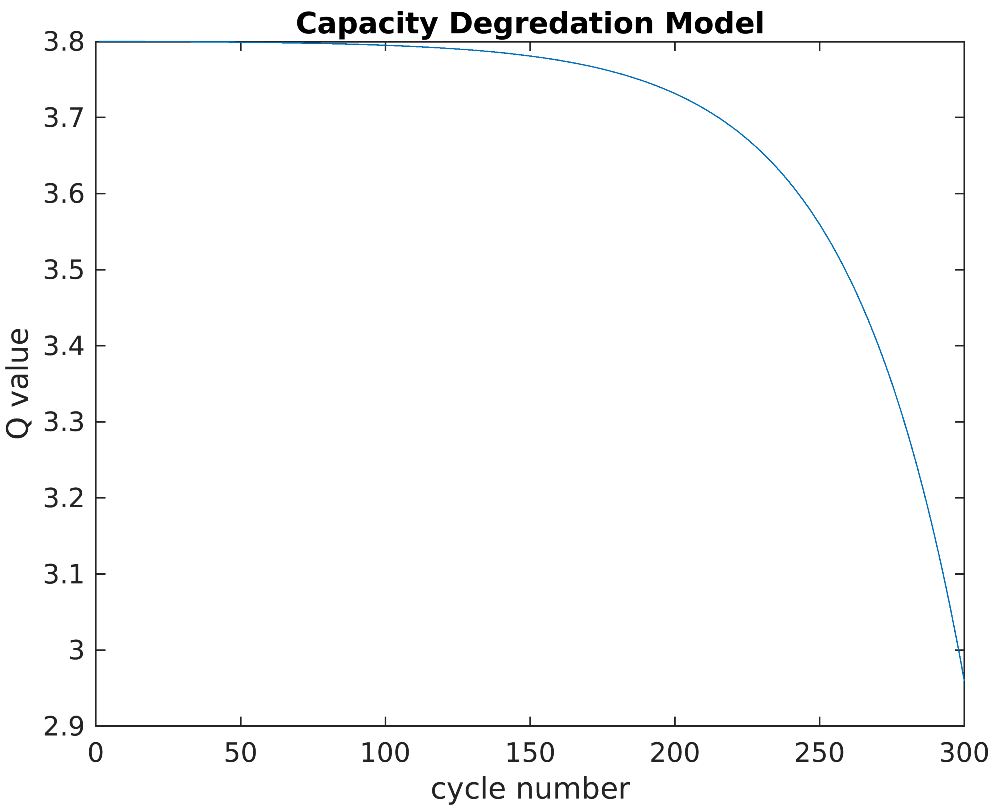
```
disp(R0(300));
```

```
0.2102
```

## Q

- degrade from 3.8 to 2.95 exponentially

```
Q = zeros(300,1);
Q(1) = 3.8;
for i = 2:300
    Q(i) = degradeQ(Q(i-1), i);
end
f2 = figure(2); clf;
plot(Q);
title("Capacity Degredation Model");
ylabel("Q value");
xlabel("cycle number");
```



```
disp(Q(1));
```

```
3.8000
```

```
disp(Q(300));
```

```
2.9588
```

## sim the model function

- TODO: update for prognostics experiment

```
function [voltages, socs, paramVals, batteryParams] = getSimResults(batteryParams, para
    % placeolder for dynamic variable creation
    names = ['a', 'b', 'c'];
    for i = 1:3
        simout = sim('batteryMine');
        voltages.(names(i)) = simout.voltage.Data;
        socs.(names(i)) = simout.soc.Data;
        paramVals.(names(i)) = batteryParams.(param);
        batteryParams.(param) = batteryParams.(param) * factor;
        save('batteryParams.mat', 'batteryParams');
    end
end
```

## R0 Degradation

```
function R0 = degradeR0(R0, cycle)
    R0 = R0 + (exp(.025*cycle) / (350000));
end
```

## Q Degradation

```
function Q = degradeQ(Q, cycle)
     Q = Q - (exp(.025*cycle) / (87000));
end
```