Problem Chosen	2026 MCM/ICM Summary Sheet	Team Control Number
A		2602253

A Component-Wise Model for Smartphone Battery Drain

Summary

The smartphone is ubiquitous due to its ability to connect people to others. Whether the task is for entertainment or profession, conscious consumers expect reliable information allowing them to keep their phone active and may want to act in a way that preserves their battery life. We establish a model that determines the effect of various factors on smartphone battery life.

Model 1: Independent Component Power Consumption. We establish an independent model for power consumed by a smartphone's central processing unit (CPU), the background activity, the network activity and GPS activity, and the screen activity and brightness. These independent models are all combined to establish total power consumption of the device, which, when combined with the subsequent models allows us to estimate time-to-empty (TTE).

Model 2: Coulomb Counting. We implement a model for calculating the **state of charge**(SOC) based on the current draw from the phone's various applications and background processes, including screen brightness, network activity, and app notifications. This model involves integrating the total current draw at each time step to obtain charge lost, which we subtract from the battery capacity.

Model 3: Equivalent Circuit Model (ECM). The physical electronic processes of Lithium-Ion (Li-Ion) batteries can be modeled using an Equivalent Circuit Model (ECM). This model comprises physical circuit components (e.g. capacitors and resistors) that reflect the battery's internal resistance and chemical polarization processes.

Model 4. Battery State-Space Model and Kalman Filtering. We combine the Coulomb-Counting and ECM models into a single state-space system of ordinary differential equations (ODEs). As we update this system in time, we employ a Kalman Filter to refine our results. **Coulomb counting** is susceptible to drift due to error accumulation for long time periods. The Kalman filter serves to refine the Coulomb Counting SOC predictions.

Empirical Analysis. We validate our model against the NASA Li-Ion Battery Aging Dataset [15], which contain time-series voltage and current data for various Li-Ion batteries at different temperatures. We also simulate and analyze the time-to-empty (TTE) of phone batteries subject to various parameter conditions with the purpose of analyzing battery-life-savings due to different usage habits.

Sensitivity Analysis. We determine the effect of isolating a single component power and varying it on the average time-to-empty. Specifically, we demonstrate the isolated effectiveness of limiting brightness, network interaction, and CPU usage on increasing time-to-empty. Our models found that CPU usage and network activity consume the most power, while temperature surprisingly doesn't play a major role in our TTE estimate. We also found that the state of health (SOH) of the battery is reduced to approximately 82.5% after two years of charge-discharge cycles. This allows us to recommend battery-saving actions that improve battery life for the consumer.

Keywords: State of Charge (SOC), **Equivalent Circuit Model** (ECM), **Coulomb Counting, Kalman Filter, Nominal Region, Polarization**

Contents

1 Introduction

1.1 Background

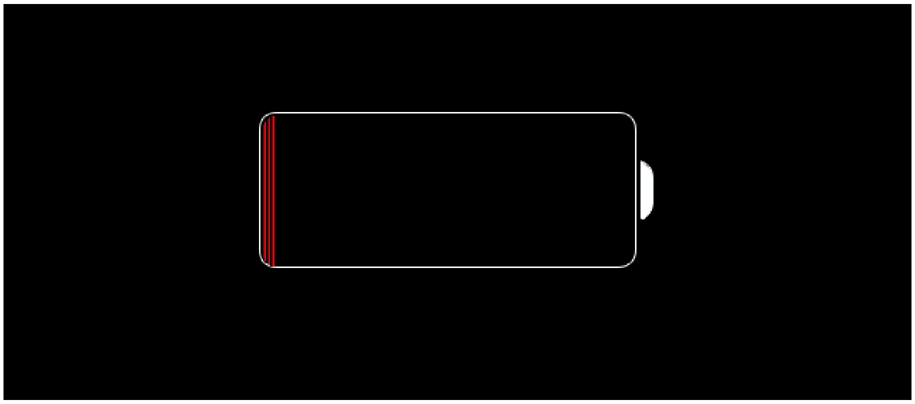


Figure 1: Drained Battery Symbol [9]

The history of smartphones follows a transformation from luxury into indispensability for communication, work, and entertainment. Sustaining the daily smartphone use by the average person has demanded substantial improvements in battery technology, motivating improvements in energy density, charging speed, and overall battery lifespan.

As smartphones become integral to daily life, accurately tracking battery life has become increasingly important. Consumers rely on real-time information about remaining battery capacity to manage their usage and avoid unexpected shutdowns. Understanding and estimating battery state-of-charge (SOC) not only enhances user experience but also informs design decisions for more efficient and reliable devices. In this context, mathematical models and estimation algorithms play a crucial role in providing accurate battery monitoring, ensuring that users are informed in an era where dependence on smartphones is nearly ubiquitous.

1.2 Problem Statement and Analysis

To achieve the objectives outlines in Section 1.1 and better understand the specifc factors that impact battery consumption, we must address the following questions:

1. **Establishing Models for the Power Draw by Phone Components:** In order to determine the effects of modulating the use of phone components on phone battery usage, we create models for several important components including the smartphone's central processing unit (CPU), the background activity, the network activity and GPS acticity, and the screen activity and brightness.

2. **Using Power to determine Battery Loss** We develop a predictive model for the state of charge (SOC) of the phone based on its current state and current draw to the cell phone components.

3. **Empirical Results** We test our model in various scenarios of usage, determining the approximate time a consumer can expect to enjoy on thier phone from different usage styles.

4. **Sensitivity analysis** We compare the effects of the modeled components on time-to-empty over numerous simulations.

5. **Recommendation** We summarize our findings and recommend effective methods for mitigating battery loss.

1.3 Our Work

Figure 2 presents a flowchart of our Model.

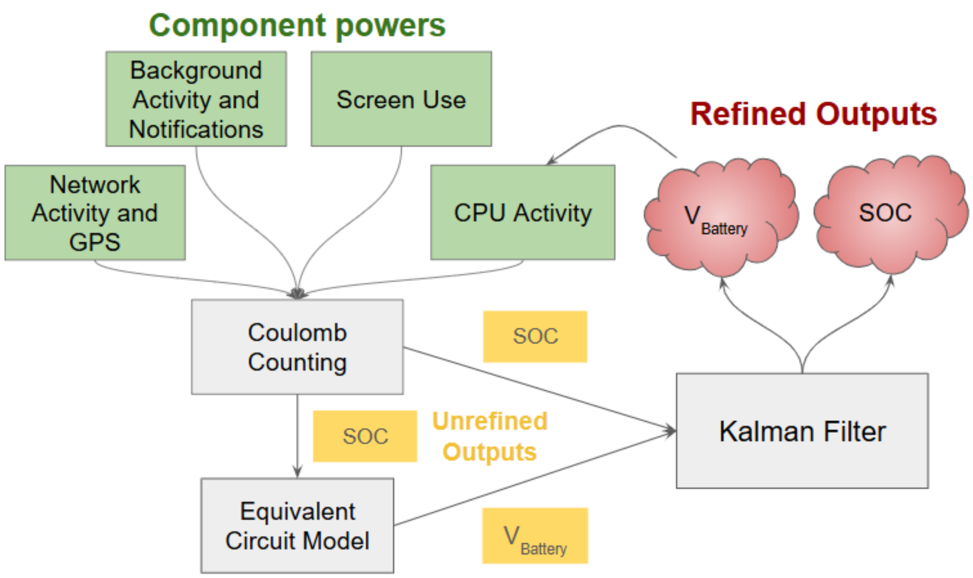


Figure 2: Diagram of Model Inputs and Outputs

2 Preparation of the Models

2.1 Data Source

We gathered data from charging and discharging trials of lithium-ion batteries from the Prognostics Center of Excellence Repository, located within the Intelligent Systems Division at NASA [15].

2.2 Notations

We use subscripts for component reference, and in discretized models we denote time steps with superscripts. For example, $V_{C_1}^{k+1}$ refers to the voltage across capacitor 1 at timestep $k + 1$

3 Model 1: Independent Component Power Consumption

3.1 Central Processing Unit (CPU)

The power consumption by the CPU of a smartphone is modeled as the composite of two factors: the power expended due to processes demanded of it (dynamic power) and the leakage current, which is expended under no extra process (static power). We use equations (1) and (2) to compute the dynamic and static power consumption over time [7]. Summing equations (1) and (2) we obtain the total power consumption of the CPU over time (3).

$$P_{dynamic}(t) = U \times C \times V(t)^2 \times f_{max} \tag{1}$$

$$P_{static}(t) = a \times e^{-\frac{b+c\sqrt{V(t)}}{T(t)}} V(t)^2 \tag{2}$$

$$P_{total}(t) = P_{dynamic}(t) + P_{static}(t) \tag{3}$$

Here, $V(t)$ denotes the voltage supplied to the CPU, f_{max} is the maximum frequency the CPU can reach while operating, and $T(t)$ represents the temperature of the device over time (measured in Kelvin). The measured constants a, b, and c are obtained through [7]. Due to limited available data, we utilized the reported parameters from [7]. This model provided us with the power consumption caused by the CPU usage at each time step, which will help to make time-to-empty estimations.

Table 1: CPU Parameters for Power Modeling used in our calculations

Parameter	Description
$f_{\max}$	Maximum frequency (8×10^8 Hz)
C	Effective Capacitance (1.28×10^{-9} F)
U	CPU Utilization (0 to 1)
$T(t)$	Temperature of battery
a	1525.07 constant
b	1884.1 constant
c	525.556 constant

3.2 Screen Size and Brightness

To model the power consumed by the smartphone display, we model the power consumption of the total display as the sum of the contributions of all the pixels. This will allow us to take into account larger screens that have more pixels or smaller screens that have fewer pixels. Each pixel consists of three subpixels: red, green, and blue (RGB). The power consumption of each pixel is determined by the sum of the intensities of its red, green, and blue (RGB) subpixels. RGB pixel intensity is measured on a scale from $0 - 255$ where 0 is off, and 255 is maximum intensity. Each color consumes a different amount of energy based on its intensity. Power consumption function for each subpixel was obtained from [4] and defined as follows:

$$f(R_i) = 11 \times 10^{-6} R_i - 1.4 \times 10^{-6} \tag{4}$$

$$h(G_i) = 11 \times 10^{-6}\, G_i - 1.4 \times 10^{-6} \tag{5}$$

$$k(B_i) = 21 \times 10^{-6}\, B_i - 0.4 \times 10^{-6} \tag{6}$$

For simplicity, we set the intensity of each subpixel at 125, giving us constant power consumption for each pixel.

We note from [10] that the screen's power consumption at the given time is the idle power plus the sum over each pixel's RGB subpixels, multiplied by the brightness percentage (7).

$$P_{\text{consumption}}(t) = C + \left(\sum_{i=1}^{n} \left[f(R_i) + h(G_i) + k(B_i) \right] \right) \cdot \text{Brightness}(\%) \tag{7}$$

Table 2: Screen Size and brightness Parameters

Parameter	**Description**
C	Idle Power Consumption of screen (W)
$f(R_i)$	Red subpixel power consumption function
$h(G_i)$	Green subpixel power consumption function
$k(B_i)$	Blue subpixel power consumption function
Brightness	% of full intensity employed
n	Number of Pixels

Here, n is the total number of pixels, which allows us to scale for varying screen sizes. C is the static power of the display module, which can be obtained by measuring the power consumption when the screen is black. To incorporate brightness, we multiply the sum of the power consumptions of each pixel by the brightness percentage.

3.3 Network Activity

To model the power consumption caused by network activity, we will consider the three states a device can be in when connected to the network, described in [1]. In this structure, a smartphone transitions between distinct network states depending on activity. Each state has a specific power consumption. We employ a finite-state machine to determine power consumption depending on the amount of data activity over time. The three network states are:

1. **Idle State:** In the idle state, the smartphone is connected to the network, but no data is shared through the network. In [1], the idle power for a smartphone was measured to be approximately 0.05 W.

2. **Active State:** In the active state, the smartphone is turned on, connected to the network, and actively sending or receiving data. The power consumed P in Watts is determined using a linear model that depends on the volume of data transmitted x in bytes and a constant factor of power that is applied for every data transfer. This model and the corresponding parameters were obtained from [1].

$$P(x) = 0.025x + 3.5 \tag{8}$$

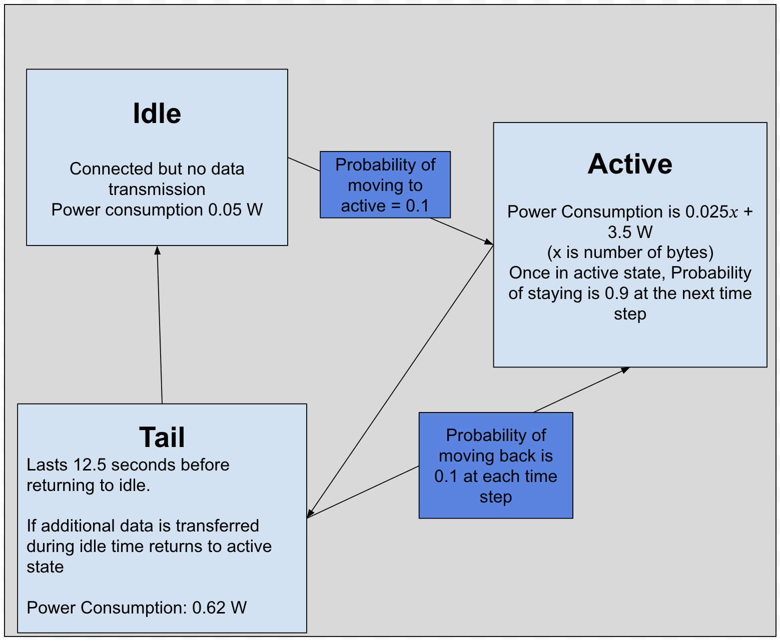


Figure 3: Network process state machine with example probabilities

3. **Tail State:** Following network activity, there is a fixed period where the device actively awaits more data transmission. During this period, which [1] determined to last 12.5 seconds for most devices, the network consumes 0.62 W

State transitions between the idle, active, and tail states are determined by a probabilistic state machine, illustrated in Figure 3. We run varying simulations with different probabilities of network activity occurring at a time step, which would then cause a transition from the idle state to the active state. Again, we will also run simulations with varying probabilities that there will be network activity in the next time period. If there is no network activity in the next time period, a 12.5 second timer starts before we transition back to the idle state, as long as there is no further network activity in that period. Overall, we return the network power consumption at each time step to feed into our SOC model to make the time-to-empty predictions.

In Figure 3, we see a visual of the network activity state machine. As an example, we see that the probability of moving from idle to active is 0.1. If we are in the active state, then the probability we will remain at the active state at the next time step is 0.9. Once in the tail state, the probability of going back to the active state is 0.1. After 12.5 seconds, if we have not moved back to the active state, we move to the idle state.

3.3.1 GPS

Similarly to network activity, we employed the exact same fixed-state machine with different power consumption parameters for the three states related to GPS consumption. In [11], the idle power is determined to be 0.02 W. The active power and tail power are 0.22 W and 0.075 W, respectively. We also use varying probabilities as outlined in the network activity state machine above.

3.4 Background Activity

The current drawn by background applications on a smartphone can be modeled as a discrete-time stochastic process. Let N_{apps} denote the number of background applications, each drawing a baseline current I_{baseline} when idle. Notifications generate additional current I_{notif}, and occur randomly with probability p_{notif} per app per timestep. The duration of a notification, τ_{notif}, is modeled as a Gaussian-distributed random variable with a minimum duration of one timestep.

Let $A(t)$ denote the number of active notifications at timestep k, and $I_{\text{bg}}(t)$ the total background current. The process is then:

$$I_{\text{bg}}(t) = N_{\text{apps}}\, I_{\text{baseline}} + A(t)\, I_{\text{notif}} \tag{9}$$

where K is the total number of time steps. Notifications are treated as independent events across apps, so the probability that at least one new notification occurs in a timestep is

$$P(\text{new notification at } t) = 1 - (1 - p_{\text{notif}})^{N_{\text{apps}}}. \tag{10}$$

The number of active notifications $(A(t))$ can be approximated as a discrete-time linear system:

$$A(t+1) = \begin{cases} A(t) + P(\text{new notification at } t), & \text{No notification has lapsed} \\ A(t) + P(\text{new notification at } t) - 1, & \text{Notification has lapsed} \end{cases} \tag{11}$$

The duration of each notification was set to approximately 10 seconds, with a random offset drawn from a normal distribution centered at 0 seconds added to model variability in notification lengths.

which leads to an approximate expected background current

$$I_{\text{bg}}(t) = N_{\text{apps}}\, I_{\text{baseline}} + I_{\text{notif}}\, A(t). \tag{12}$$

Table 3: Background App Current Model Parameters

Parameter	Description	Estimated Reasonable Range	Units
N_{apps}	Number of background applications	$[0, 40]$	-
I_{baseline}	Baseline current per app	$[0.01, 0.2]$	Amps
I_{notif}	Additional current per active notification	~ 0.1	Amps
p_{notif}	Probability of app notification per timestep	$[0.01, 0.1]$	-
τ_{notif}	Duration of a notification	$[10, 60]$	s

4 Model 2: Coulomb Counting

In order to compute the State of Charge (SOC) of our battery, we must leverage available and computable data about the outputs of the battery and components of the phone. We must then relate these outputs to a measure of the available capacity of the phone battery, namely charge Q. Given the demands on each of the phone components, we can calculate the current drawn and use elementary physics to define

$$\frac{dQ}{dt} = I \tag{13}$$

We notice that if we integrate current or power output over time and normalize over the maximum charge Q_{max} in Coulombs or maximum energy E_{max} in Joules, we have a unitless measure of the change in capacity of the battery. We can thus determine the effect on the SOC over a select time step. We consider the equivalent functions below.

$$\mathrm{SOC}(t+\Delta t) = \mathrm{SOC}(t) - \frac{1}{Q_{max}} \int_t^{t+\Delta t} I(\tau)d\tau \tag{14}$$

Where Q_{max} represents the maximum charge and held by the battery. This classical process in SOC determination is known as Coulomb Counting [16].

4.1 Coulomb Counting for Battery Health

To test the effect of battery history on battery life, we incorporated a measure of battery capacity known as the Standard of Health SOH. The SOH was calculated by coulomb counting as follows

$$\mathrm{SOH(t+1)} = \mathrm{SOH}(t) - \frac{1}{Q_{EOL}} \int_t^{t+\Delta t} I(\tau)d\tau \tag{15}$$

Where Q_{EOL} is the capacity of the battery at the end of its life (when the battery capacity reaches 80%) of its original value.

5 Model 3: Equivalent Circuit Model (ECM)

Equivalent Circuit Models allow complex component arrangements to be pared down into a computable format. The phone battery is typically modeled as a series of parallel resistors and capacitors (RCs). The battery is represented by an open-circuit voltage source $OCV(\mathrm{SOC})$, a series resistance R_{series} accounting for ohmic losses, and two parallel resistor–capacitor (RC) branches (R_1, C_1) and (R_2, C_2) connected in series, capturing fast and slow polarization effects, respectively. [3,6] Increasing the number of these parallel RC blocks increases computation with diminishing returns. Balancing efficiency with accuracy, a 2nd-order RC circuit, shown in Fig. 4, is chosen to model our phone battery.

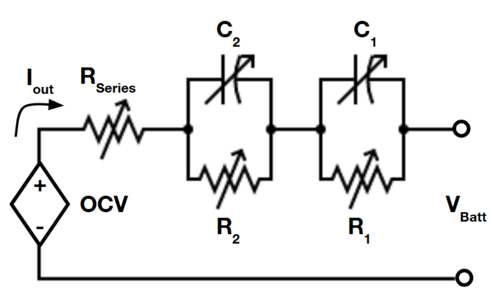


Figure 4: Phone Battery ECM Model

Note that we denote I_{Batt} as positive during charging and negative during discharging.

This circuit model allows us to derive equations for the voltage supplied by the battery in terms of the SOC. Applying Kirchhoff's Voltage and Current Laws and Ohm's Law around the outer loop of the circuit yields the terminal voltage equation

$$V_{\text{Batt}}(t) = OCV(\text{SOC}(t)) - I_{Batt}R_{series} - V_{C_1}(t) - V_{C_2}(t). \tag{16}$$

The dynamic behavior of the battery is governed by the evolution of the capacitor voltages V_{C_1} and V_{C_2}, which model the effects of polarization. In lithium-ion batteries, polarization arises primarily from two sources: the finite time required for electrochemical reactions at the electrodes, known as **electrochemical polarization**, and the limited transport of lithium ions, known as **concentration polarization**. In the equivalent circuit, these effects are represented by V_{C_1} and V_{C_2}, respectively.

Consider one RC branch, consisting of a resistor R_1 and capacitor C_1 in parallel. By Kirchhoff's Current Law, the battery current splits between the resistor and the capacitor, giving

$$I_{\text{Batt}}(t) = I_{R_1}(t) + I_{C_1}(t). \tag{17}$$

Using the constitutive relations for the resistor and capacitor,

$$I_{R_1}(t) = \frac{V_{C_1}(t)}{R_1}, \qquad I_{C_1}(t) = C_1\frac{dV_{C_1}(t)}{dt}, \tag{18}$$

The current balance becomes

$$C_1\frac{dV_{C_1}(t)}{dt} + \frac{V_{C_1}(t)}{R_1} = I_{\text{Batt}}(t). \tag{19}$$

Rearranging yields a first-order linear differential equation,

$$\frac{dV_{C_1}(t)}{dt} = -\frac{1}{R_1C_1}V_{C_1}(t) + \frac{1}{C_1}I_{\text{Batt}}(t). \tag{20}$$

Assuming the battery current is constant over one time step of Δt, this differential equation can be solved exactly. Evaluating the solution at the end of the interval results in the discrete-time update equation

$$V_{C_1}^{k+1} = V_{C_1}^{k}e^{-\Delta t/(R_1C_1)} + R_1I_{\text{Batt}}^{k}\left(1 - e^{-\Delta t/(R_1C_1)}\right). \tag{21}$$

An identical derivation applies to the second RC branch, yielding

$$V_{C_2}^{k+1} = V_{C_2}^{k}e^{-\Delta t/(R_2C_2)} + R_2I_{\text{Batt}}^{k}\left(1 - e^{-\Delta t/(R_2C_2)}\right). \tag{22}$$

5.1 Fitting RC Parameters

To accurately model the battery dynamics using the ECM, it is necessary to fit the parameters R_{series}, R_1, R_2, C_1, and C_2 to the specific battery we are modeling. We must also determine the parameters α and β to linearize the OCV − SOC equation. Parameters are fit using MATLAB's `polyfit` and `fit` functions.

When the battery begins charging, there is an immediate sharp voltage and current spike over which the diffusive and polarizing battery effects can be considered negligible [14]. Thus, we can ignore the effects from the RC blocks to find R_{series}. To find R_{series}, we can use Ohm's Law to obtain $R_{series} = \frac{\Delta V}{\Delta I}$ from the changes in voltage and current over this small instant (see segment (1) in Figure 5).

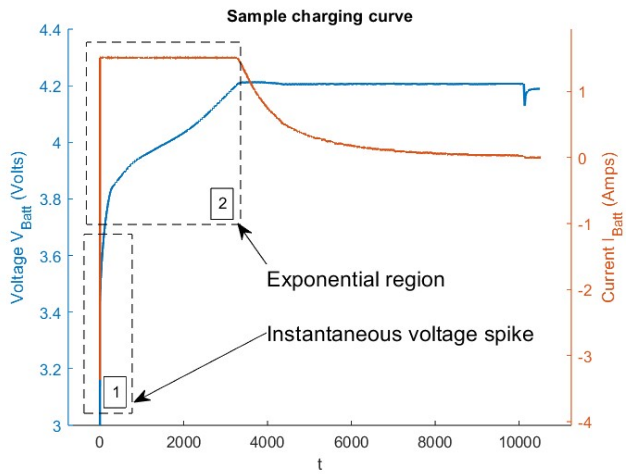


Figure 5: Example charging curve used to fit $R_{series},\ R_1,\ R_2,\ C_1,\ C_2$

Next, we need to model the nonlinear relationship between OCV and SOC, exemplified in Figure 6 below: This curve can be accurately characterized by 2 linear models: one in the steep discharge region ($SOC \in [0.0, 0.1]$) and one in the nominal region ($SOC \in [0.1, 0.9]$) [16]. Thus, we can fit 2 linear models $SOC \approx \alpha_1(SOC) + \beta_1,\ SOC \in [0.0, 0.1]$ and $SOC \approx \alpha_2(SOC) + \beta_2,\ SOC \in [0.1, 1]$ to this experimental data.

Finally, we need to fit the RC parameters. The exponential region (segment (2) in Figure 5) is characterized by the diffusive and polarizing battery effects modeled by the 2 RC blocks. Thus, we can fit the equation $V_{C_1} + C_{C_2} = I_{Batt}\Big(R_1(1 - e^{-\frac{t}{R_1C_1}}) + R_2(1 - e^{-\frac{t}{R_2C_2}})\Big)$ to the measured $OCV(SOC) - V_{Batt} - I_{Batt}R_{series}$ according to equation (16). We use our parameters $\alpha_1, \alpha_2, \beta_1, \beta_2$ in order to obtain $OCV(SOC)$ in both the discharge and nominal regions.

Table 4: ECM parameters fitted in the steep discharge region

Temp (°C)	α_1	β_1	R_{series}	$R_{1,1}$	$R_{2,1}$	$C_{1,1}$	$C_{2,1}$
4	5.825	3.345	0.188	1.000	1.000	1.469×10^3	1.468×10^3
24	0.735	3.774	0.149	1.000	0.484	7.198×10^3	8.173×10^4
44	11.621	2.936	0.212	1.000	1.000	4.213×10^2	4.251×10^2

Table 5: ECM parameters fitted in the nominal region

Temp (°C)	α_2	β_2	$R_{1,2}$	$R_{2,2}$	$C_{1,2}$	$C_{2,2}$
4	0.434	3.760	0.632	0.197	1.575×10^4	5.387×10^4
24	0.329	3.813	0.821	0.044	1.145×10^4	2.496×10^6
44	0.673	3.497	0.938	0.150	1.016×10^4	1.103×10^6

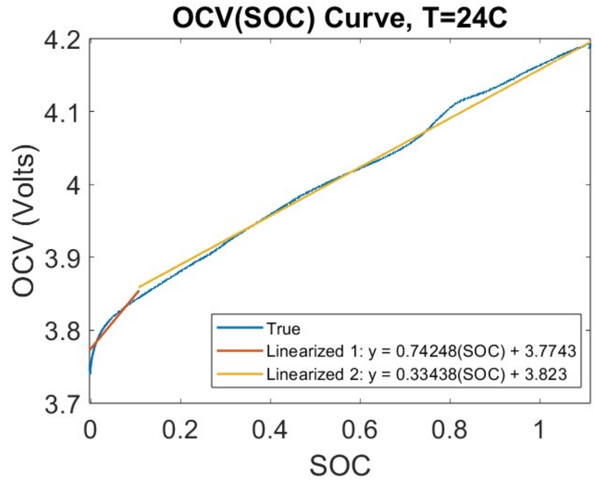


Figure 6: Sample OCV-SOC curve depicted for room temperature ($24°C$), generated from NASA Li-Ion Battery Aging Dataset.

6 Model 4: Battery State-Space Model

In the case of smartphone battery SOC prediction, we can assume that we have access to the current draw I_{Batt} at each time step, as well as the OCV-SOC curve typically given by battery manufacturers. Thus, we can obtain estimates for SOC using Coulomb integration (equation (14)). However, Coulomb integration is highly susceptible to error accumulation and is considered too inaccurate to use as a direct estimate for SOC. To achieve a robust model that accurately models the physical properties of batteries (i.e. **internal resistance, electrochemical polarization, and concentrative polarization**), we combine the Coulomb-Counting and ECM models. We simultaneously mitigate accumulated error from measurement and process noise using a Kalman filter.

Combining equations (14), (21), (22), we get a system of ordinary differential equations (ODEs). The evolution of the state is governed by a linear discrete-time approximation of the continuous-time dynamics at time t^k with time-step size Δt.

$$\begin{aligned}
\mathrm{SOC}^{k+1} &= \mathrm{SOC}^k + \frac{\Delta t}{Q_{max}} I_{Batt}^k \\
V_{C_1}^{k+1} &= V_{C_1}^k e^{-\Delta t/(R_1 C_1)} + R_1 I_{\mathrm{Batt}}^k \left(1 - e^{-\Delta t/(R_1 C_1)}\right) \\
V_{C_2}^{k+1} &= V_{C_2}^k e^{-\Delta t/(R_2 C_2)} + R_2 I_{\mathrm{Batt}}^k \left(1 - e^{-\Delta t/(R_2 C_2)}\right)
\end{aligned} \tag{23}$$

The system's dynamics depend on the current draw I_{Batt}, maximum battery capacity Q_{max}, and RC parameters R_1, R_2, C_1, C_2. We can write this system in state-space, using the state vector $\mathbf{x}$ and input $\mathbf{u}$ defined as

$$\mathbf{x}(t) = \begin{bmatrix} \mathrm{SOC}(t) \\ V_{C_1}(t) \\ V_{C_2}(t) \end{bmatrix}, \ \mathbf{u}(t) = \begin{bmatrix} \mathrm{I}_{\mathrm{Batt}}(t) \\ 1 \end{bmatrix} \tag{24}$$

where $\mathrm{SOC}(t)$ denotes the state of charge, and $V_{C_1}(t)$, $V_{C_2}(t)$ are the voltages across the fast and slow polarization capacitors, respectively, at time t. Rewriting equation (23), we obtain the state-space equation

$$\mathbf{x}^{k+1} = \mathbf{A}\mathbf{x}^k + \mathbf{B}\mathbf{u}^k \tag{25}$$

where

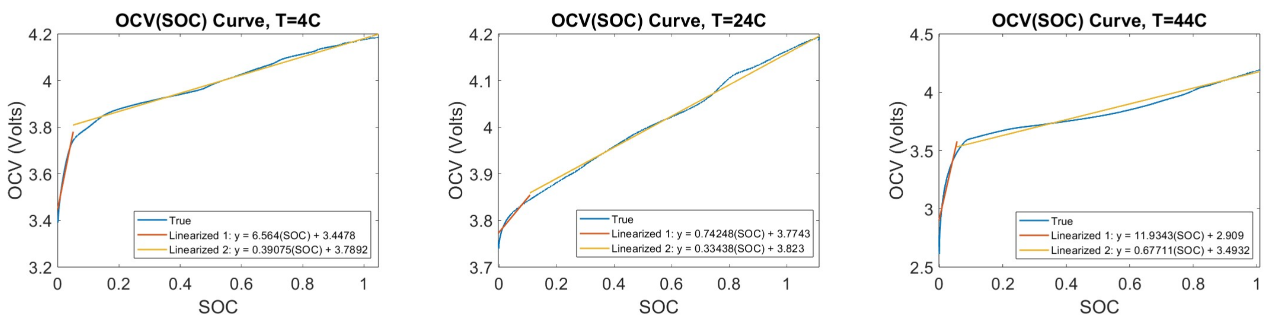
$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 - \frac{\Delta t}{R_1 C_1} & 0 \\ 0 & 0 & 1 - \frac{\Delta t}{R_2 C_2} \end{bmatrix}, \ \mathbf{B} = \begin{bmatrix} -\frac{\Delta t}{Q_{max}} \\ \frac{\Delta t}{C_1} \\ \frac{\Delta t}{C_2} \end{bmatrix} \tag{26}$$

We can relate the internal state $\mathbf{x}^k$ to the observable quantities $\mathbf{y}^k = \left[\mathrm{SOC}^k,\ V^k_{Batt}\right]^T$ that we want from our model. Recall that the battery terminal voltage is expressed as

$$V_{\mathrm{Batt}}(t) = OCV(\mathrm{SOC}(t)) - V_{C_1}(t) - V_{C_2}(t) - R_{\mathrm{series}} I(t), \tag{27}$$

where the open-circuit voltage is approximated by a linear function of SOC.

$$OCV(\mathrm{SOC}) \approx \alpha\, \mathrm{SOC} + \beta. \tag{28}$$



(a) $4°C$ (b) $24°C$ (c) $44°C$

Figure 7: OCV(SOC) Curves for temperatures $4°C, 24°C, 44°C$

A piecewise linear approximation preserves high accuracy, especially in the nominal region, allowing us to maintain the linear structure of our state-space model, which we define as

$$\mathbf{y}^{k+1} = \mathbf{C}\mathbf{x}^k + \mathbf{D}\mathbf{u}^k \tag{29}$$

where

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \\ \alpha & -1 & -1 \end{bmatrix}, \ \mathbf{D} = \begin{bmatrix} 0 & 0 \\ -R_{series} & \beta \end{bmatrix} \tag{30}$$

6.1 Kalman Filtering

A Kalman Filter is an algorithm that takes in system parameters with noisy measurements and refines them into less noisy and more accurate system parameters. At each time-step, we obtain estimates for $\mathbf{y}^k$ using Coulomb integration (equation (23)) and equation (27). However, Coulomb integration is highly susceptible to error accumulation and is considered too inaccurate to use as a direct estimate for

SOC. Thus, it is advantageous to use a Kalman filter to account for variances between the estimated and actual SOC values.

Our Kalman filter is implemented similarly to the work of [12, 13]. For a more in-depth explanation of the Kalman filter, we refer the reader to [2]. We expect the process noise Q to be zero-mean, normally distributed, and independent, inducing natural random error in the SOC and V_{Batt} variables over time. Similarly, we assume the measurement error R is large enough to account for the errors accumulated from the Coulomb integration.

At each time step t^k, the Kalman filter performs a prediction step using equation (25) to obtain an a priori estimate of the state $\hat{\mathbf{x}}^{k,-}$ and its corresponding observable prediction $\hat{\mathbf{y}}^{,-}$. From this, we compute the a priori residual $\hat{\mathbf{r}}^k = \mathbf{y}^k - \hat{\mathbf{y}}^{k,-}$, where the measurement reference $\mathbf{y}^{k+1} = [SOC^{k+1}_{measured}, V^{k+1}_{Batt,measured}]^T$ corresponds to the Coulomb-integrated SOC and measured V_{Batt}. The a priori covariance is found according to the mean squared error between the a priori state estimate and the "true" state

$$\mathbf{P}^{k,-} = \mathbf{A}\mathbf{P}^{k-1}\mathbf{A}^T + Q \tag{31}$$

where $\mathbf{P}^{k-1}$ is the covariance at the previous step.

$$\mathbf{K}^k = (\mathbf{P}^{k,-}\mathbf{C}^T)(\mathbf{C}\mathbf{P}^{k,-}\mathbf{C}^T + R)^{-1} \tag{32}$$

We use the Kalman gain $\mathbf{K}^k$ and a priori residual $\hat{\mathbf{r}}^k$ to update our a priori estimation to the a posteriori estimation $\hat{\mathbf{x}}^{k,+}$:

$$\hat{\mathbf{x}}^{k,+} = \hat{\mathbf{x}}^{k,-} + \mathbf{K}^k(\hat{\mathbf{r}}^k) \tag{33}$$

Finally, we compute the a posteriori covariance to use in the following time step.

$$\mathbf{P}^k = (\mathbf{I} - \mathbf{K}^{k,-}\mathbf{C})\mathbf{P}^{k,-} \tag{34}$$

7 Results

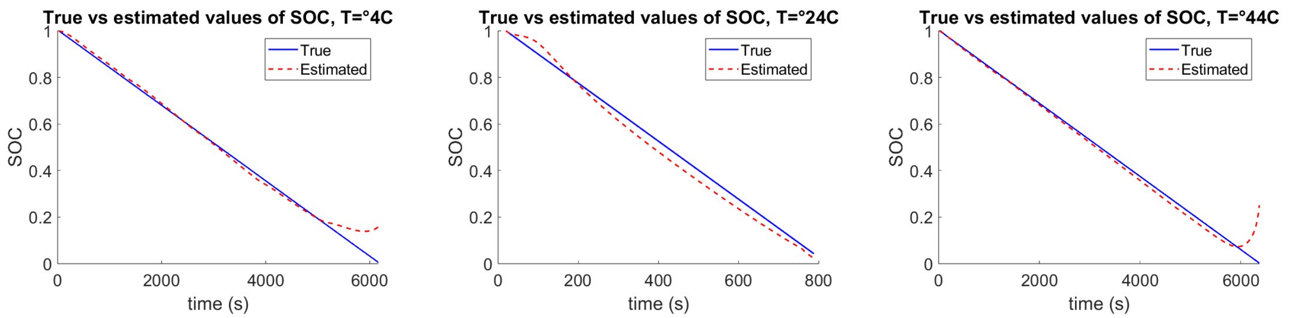
7.1 NASA Battery Test

We first tested our model on Li-Ion battery data provided by NASA [15]. In this test, batteries at $4°C, 24°C$, and $44°C$ were drained at constant $1.5A$ current, and their voltages were simultaneously measured. We calculate SOC references (shown in blue) using Coulomb integration and compare these references to our model's predictions (in red) in Figure 8. Similarly, we compare the measured reference battery voltages (in blue) to our model's prediction voltage V_{Batt} in Figure 9. In these tests, we fit ECM parameters $\alpha, \beta, R_{series}, R_1, R_2, C_1, C_2$ as described in Section 5.1. We observe agreement between these measured references and predictions at all three temperatures, as shown in table 6, implying low uncertainty contributed by this model.

Table 6: Sum of Squared Errors for RC and OCV models at Different Temperatures

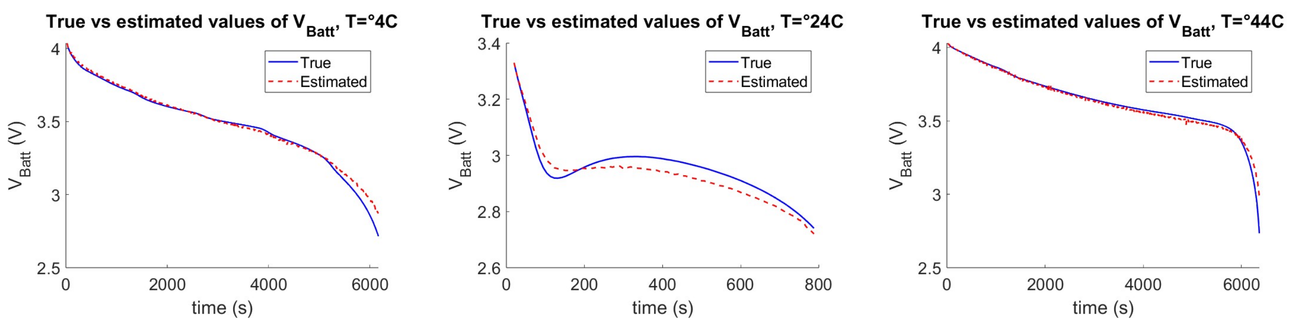
Temperature (°C)	RC SSE	OCV SSE
4	8.58×10^{-5}	0.2214
24	3.59×10^{-4}	0.1160
44	0.0189	1.5671

Because the NASA dataset provides charging/discharging data for the same batteries at both $24°C$ and $44°C$, we compare their SOC and V_{Batt} time-to-empty predictions in plots (b) and (c) of Figures 8 and 9. We observe that at the higher temperature, the battery is more efficient (indeed its measured maximum charge Q_{max} is a full $1Ah$ higher at $44°C$ than at $24°C$, which is why it lasts significantly longer ($\sim 6000s$ compared to $\sim 800s$) when the battery's ambient temperature is higher. The dataset, however, does not provide $4°C$ data for batteries also measured at $24°C$ and $44°C$. Thus, we include a battery with a similar maximum charge to the battery at $44°C$, and observe a similar time-to-empty, though notably a larger voltage drop than the $44°C$ battery.



(a) $4°C$ (b) $24°C$ (c) $44°C$

Figure 8: SOC prediction curves validated on NASA Li-ion Battery Aging Dataset with constant $1.5A$ current drainage until empty.



(a) $4°C$ (b) $24°C$ (c) $44°C$

Figure 9: V_{Batt} prediction curves validated on NASA Li-ion Battery Aging Dataset with constant charge $1.5A$ current drainage until empty.

Table 7: Estimation Error Statistics at Different Temperatures

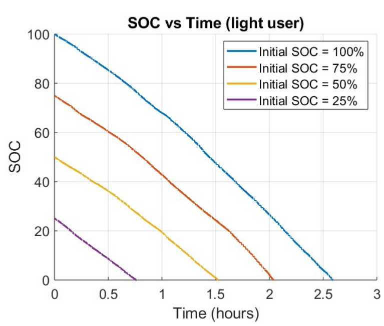
Temperature (°C)	Mean Error	Std. Dev.
4	-0.0108	0.0307
24	-0.0591	0.0342
44	0.0056	0.0295

7.2 Time-to-Empty predictions

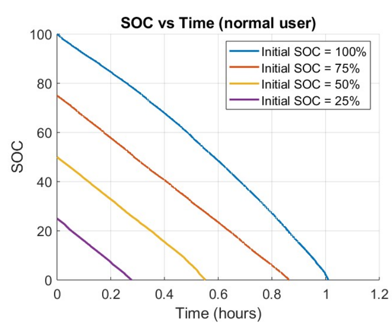
We tested our model on several different classes of users with different input parameters. These users were designed to demonstrate the amount of battery life an individual could expect to expend based on general smartphone use at varying degrees. We modulated the brightness, CPU usage, and size of network interaction. We determined the time to empty based on various starting SOCs. All trials were ran at $T = 24°C$ with a $Q_{max} = 4.5Ah$ battery.

Parameter	**Light User (Battery-Saver)**	**Normal User**	**Heavy User**
Screen brightness	10%	40%	70%
# background apps	2	5	10
Notification probability	1%	5%	10%
CPU usage	10%	20%	50%
Data size per transfer	20 bytes	50 bytes	100 bytes
Data transfer/GPS sampling probability	1%	5%	10%
Screen size	720×1600 pixels	1080×2400 pixels	1080×2400 pixels

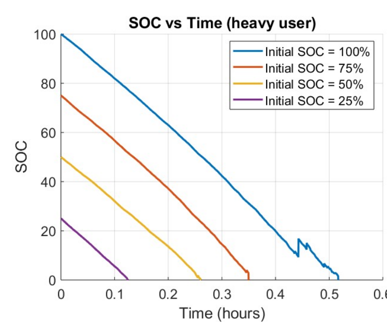
Table 8: User activity definitions



(a) Light user (battery-saver mode)



(b) Normal user



(c) Heavy user

Figure 10: TTE simulation for light, normal, and heavy battery user profiles. $T = 24°C$, $Q_{max} = 4.5Ah$

Table 9: Time-to-Empty (TTE) Results for Light, Normal, and Heavy User Profiles

TTE (hh:mm:ss)		
Light User	**Normal User**	**Heavy User**
02:35:31	01:00:35	00:31:01
02:02:14	00:51:53	00:21:02
01:31:32	00:33:09	00:15:40
00:45:40	00:16:39	00:07:31

7.3 Effect of Battery History on Battery Life

We ran a trial of our model calculating the value for the Standard of Health (SOH) of the battery.

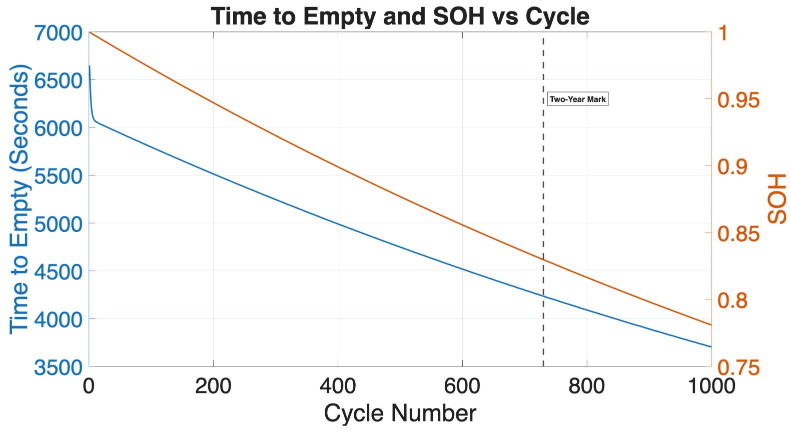


Figure 11: State of Health vs Full discharge Cycles

Our model computed realistic values for SOH. We see that after approximately 2 years of use with 1 full discharge per day, the phone has reached 80% of its original capacity.

8 Sensitivity Analysis

To determine the sensitivity of our model to variations in power drawn from the various components (e.g., network traffic, screen brightness, etc.), we isolated each component and let it be the only active contributor (although background power drains such as $P_{CPU,static}$ remained). We isolated components by removing the other components' additions to the current draw, and ran 100 trials varying the isolated condition. The Average Time-to-Empty (TTE) was recorded for each of variation of the isolated condition.

8.1 Brightness

We vary brightness from 0% to 100% by 10% increments.

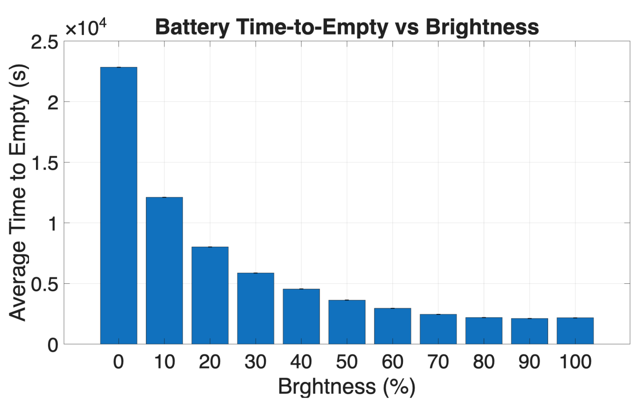


Figure 12: Average Time-to-Empty vs. Brightness

We saw a dramatic decrease in TTE, especially between lower brightness levels. This was expected because the screen consumes a relatively large portion of current available to the various components.

8.2 Background Activity

We incremented the background activity by single apps up to 100.

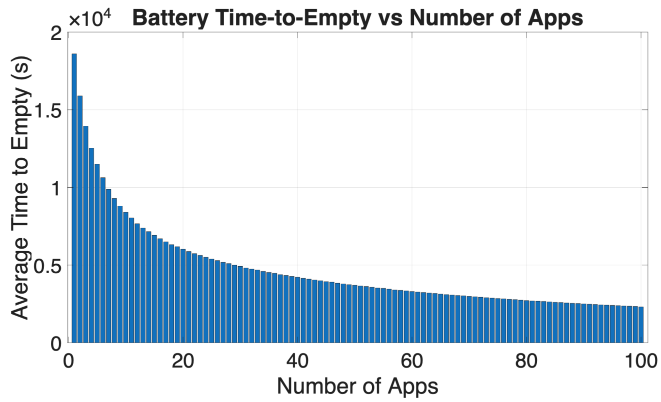


Figure 13: Average Time-to-Empty vs Background Activity

We observed a gradual decrease in battery life due to the number of apps open in the background. Single apps had a diminishing effect. After having one or two apps in the background, adding more does little to significantly impact battery life.

8.3 CPU Utilization

We incremented CPU usage from 0% to 100% by 10% increments.

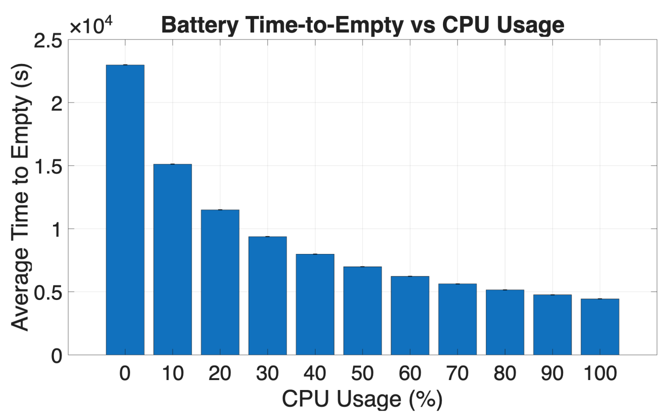
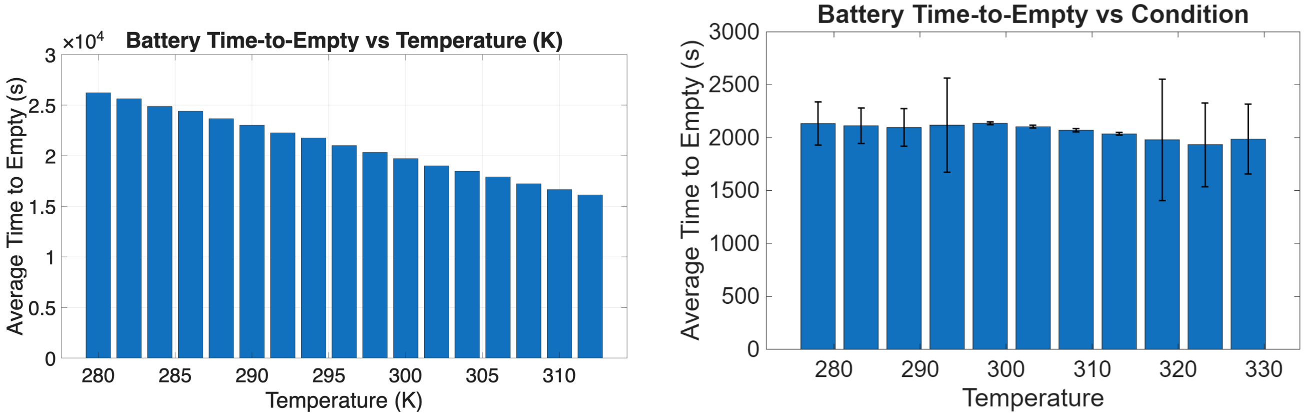


Figure 14: Average Time-to-Empty vs CPU Utilization

Though increasing the number of background apps did decrease the average time to empty, reducing the battery life by half, as occurred between 0% and 20% brightness, did not occur until there were 16 apps open in the background. This points to the lesser effects of open background apps in comparison.

8.4 Temperature

In the case of temperature, we incremented through temperatures ranging from $13°C$ to $42°C$ by $3°C$.



(a) Average Time-to-Empty vs Temperature

(b) Modified ECM: Average TTE vs Temperature

Figure 15: Comparison of baseline and modified ECM time-to-empty behavior across temperature

In isolation, increasing temperature decreases battery life by increasing the static power draw from the CPU, as seen in equation (2) and demonstrated in Figure 15a. The CPU is a significant source of power drain for real-world smartphone batteries, so we expected to see large effects. However, higher temperatures are known to increase the efficiency of the internal components of the battery. To consider the effects of temperature on efficiencies, we modeled ECM circuit components $R_1, R_2, C_1, C_2,$ and R_{series} as functions of temperature in accordance with measured temperature and component data (Figure 15b) [15]. This increase in component efficiency nullifies the decrease depicted in Figure 15a and results in an ultimately non-correlated relationship shown in Figure 15b.

The result shows that the effect of temperature is negligible when efficiency of the battery is considered. The error bars represent one standard deviation of the runs taken at each temperature. This aligns with previous research showing that the effects of including temperature dependence on small scale are negligible [5].

8.5 Network Activity and GPS

We increment interaction with the network and GPS by 10 bytes up to 500 bytes.

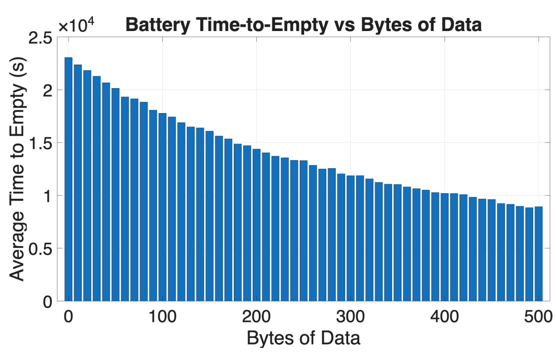


Figure 16: Average Time-to-Empty

We see that increasing the interaction with the internet does play a meaningful role in decreasing battery life. Sending large images or videos regularly would even further decrease the smartphone battery time-to-empty.

8.6 Summary of battery-saving effects

To compare the most effective battery-saving methods, we combine and plot the results of the sensitivity analyses for brightness, background activity, CPU usage, temperature, and network activity on a normalized plot (Figure 17). We use the data from the isolated sensitivity analyses to obtain average TTE values. This plot normalizes these TTEs into battery-life savings by dividing each savings by the maximum TTE over all parameters. The plot also normalizes the x-axis to range from 0 to 1, representing the normalized variation in the parameters between reasonable values. The absolute ranges are shown in the legend.

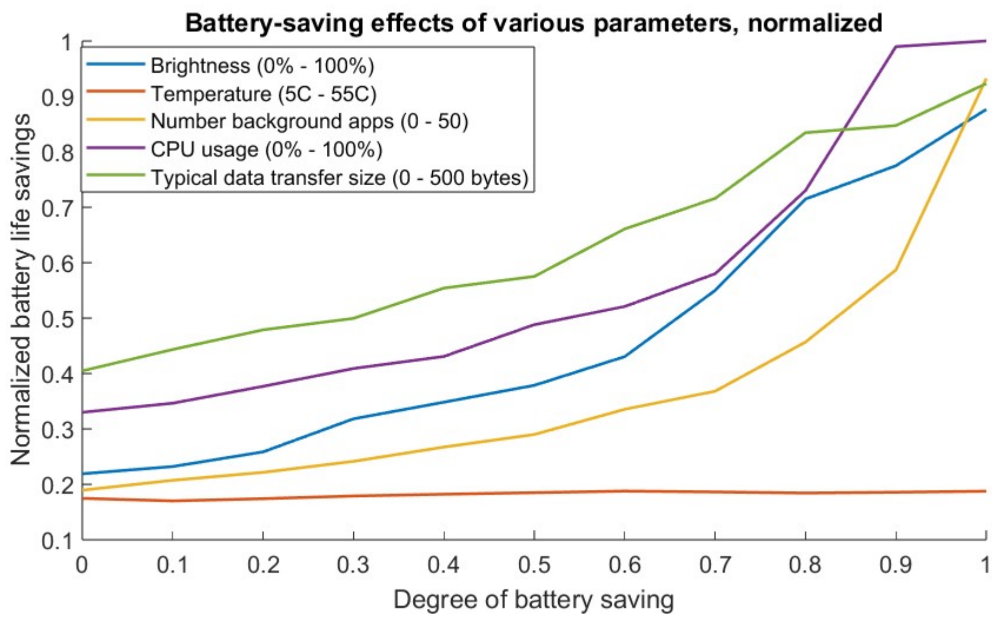


Figure 17: Combined, normalized battery saving effects

9 Conclusion

We determined that reducing data transfer size and CPU usage were the greatest actionable determinants of overall battery life. We saw that reducing brightness had significant effects as well, and that closing background apps has a significant effect only at low total background app counts. We saw that increased temperature decreased the battery life when its effects on the ECM model of the circuit were not considered. However, when considering ECM model components as functions of temperature, the earlier effects of temperature on time-to-empty were absorbed. We saw that cycling the battery once a day reduces battery state-of-health to $\sim$ 82.5% after 2 years of once-per-day full charging and discharging, due to large contributions to output current, resulting in a corresponding 36% drop in time-to-empty, as illustrated in Figure 11. Our model likely underestimates time-to-empty for all three user classes, particularly heavy users, due to simplifying assumptions such as continuous screen usage and constant CPU load without idle periods. Despite linearizing the OCV–SOC relationship, the model closely matched recorded phone usage data up to the ending discharge region, where the strongly nonlinear behavior of the OCV–SOC curve dominates. Our results suggest that user behavior dominates environmental effects on time-to-empty, indicating clear, actionable strategies for extending battery life.

10 Strengths & Weaknesses

10.1 Strengths

1. **Equivalent Circuit Model:** The ECM could have been modeled as a 3rd order, 1st order, or a Partnership for a New Generation of Vehicles (PNGV) circuit which resembles the 2nd order ECM with C_2 and R_2 in series rather than in parallel [8]. However, the 2nd-order RC ECM is widely adopted for its mix of computational efficiency and accuracy. [3] [6]

2. **Kalman Filter:** The Kalman Filter is very efficient and allows for real-time optimization of noisy data. It allows us to rein in SOC drifting at minimal computational cost.

3. **Physical relevance:** Our model is heavily motivated by and built upon real-world physical processes. We demonstrated its ability to meaningfully capture accurate physical processes by comparing our model's prediction to the real-world behavior captured in the NASA dataset.

4. **Interpretable:** Unlike traditionally ”black-box” machine-learning models and neural networks, our model has easily extractable and interpretable parameters such as R_1, C_1, R_{series} that have physical meaning and capture the physics of our modeled batteries.

10.2 Weaknesses

1. **Component Power Models:** The major limitation of the component power models is the reliance on published data from varying sources. We attempted to assemble a set of reasonable givens for a smartphone, as many parameters such as chance that an app gives a notification, for example.

2. **Coulomb Counting:** At each step in the coulomb counting process for computing the SOC, error is incurred. Though error is mitigated by the Kalman Filter, it may still build up over long times, especially if there has been no reference, i.e., the phone has not been charged to a full 100%.

3. **Battery State-Space Model:** Linear OCV − SOC Relationship Although the OCV − SOC relationship is approximately linear over the nominal region SOC and exponential region, in the discharge region, the relationship becomes nonlinear. Modeling this nonlinear portion linearly induces some error for the sake of efficiency.

11 Extensions

To extend our model, we propose incorporating an Extended Kalman filter into this model if the user is willing to expend the extra computing power. An extended Kalman filter would allow the model to handle nonlinearity directly by taking a higher-order polynomial fit of the OCV(SOC) curve, which was explored in [13]. Additionally, we would vary the values within our ECM with SOC, as the resistances and capacitances therein often vary with remaining capacity of the battery [3]. Although incorporating SOH for single time-to-empty trials is not very significant, if we commercialized this model, we would take battery use data and incorporate SOH into the calculations for SOC. Having the evolving history of battery use would allow us to reduce the capacity of the battery accordingly, so that

$$Q_{max,usable} = Q_{max,initial} \times \text{SOH} \tag{35}$$

12 Executive Summary

The reliance on smartphones has drastically increased throughout the 21st century. A major side effect of this is the dependence on batteries to keep our phones on and connected. In this paper, we combined specific component power consumption models, a Coulomb Counting model, and a Kalman Filter to obtain time-to-empty estimates for a smartphone battery under various usage scenarios.

Main Results:

- User behavior: We simulated various user profiles and how that impacts the TTE prediction. Our light user (specific parameters defined in Table 8 simulation achieved over 2.5 hours of continued screen usage under the outlined parameters. The Heavy user achieved only 0.5 hours of continuous usage. However, it is important to note that through the simulations in our model, we believe that it potentially underestimates the time-to-empty in the heavy usage scenario, as we see such a drastic decrease in TTE.
- Dominant Factors: Through the sensitivity analysis, we learn that CPU usage and Network have the most significant impact on battery consumption, followed closely by screen brightness.
- Weak Factors: The number of apps has an impact on battery consumption, but as the number of apps increases, there is diminishing marginal power consumption. Temperature surprisingly doesn't seem to make a big difference in power consumption.
- Battery Aging: We simulated full charging cycles, which reduces the state of health (SOH) of the battery. Our Model predicted that after approximately two years of charging and discharging your phone's battery once a day, the SOH of the battery will be at approximately 82.5%. This means that when the battery is "fully charged," it will only be at 82.5% of its original capacity

Insights & Recommendations:

1. The two largest factors that reduce battery life are easily controlled by the user, which is a good thing, as reducing screen brightness and CPU usage is possible, but not ideal if you absolutely need to use your device in a condition that requires a higher brightness and CPU-intensive application.

2. Our main recommendation for prolonging battery life would be to use fewer CPU-intensive apps. We recommend battery-conscious users avoid playing gamesstreaming live videos, or using AI programs for long periods of time.

3. We recommend reducing the number of charge-discharge cycles your battery goes through. This will help maintain better battery health and battery life in the long run. We recommend always keeping your device plugged in, as this will reduce the damage to the battery incurred by charging-discharging cycles.

4. It was interesting to observe that our model predicted diminishing marginal power consumption as the number of apps in the background increased past the initial few. Commonly, people associate having a large number of apps open in the background increases batter drainage. While our results show that battery drain still increases, the incremental loss in battery life, decreases once several background applications are already active.

References

[1] Niranjan Balasubramanian, Aruna Balasubramanian, and Arun Venkataramani. Energy consumption in mobile phones: a measurement study and implications for network applications. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*, IMC '09, page 280–293, New York, NY, USA, 2009. Association for Computing Machinery.

[2] Erik Cheever. The scalar kalman filter. https://cheever.domains.swarthmore.edu.

[3] Min Chen and G.A. Rincon-Mora. Accurate electrical battery model capable of predicting runtime and i-v performance. *IEEE Transactions on Energy Conversion*, 21(2):504–511, 2006.

[4] Mian Dong, Yung-Seok Kevin Choi, and Lin Zhong. Power modeling of graphical user interfaces on oled displays. In *2009 46th ACM/IEEE Design Automation Conference*, pages 652–657, 2009.

[5] O. Erdinc, B. Vural, and M. Uzunoglu. A dynamic lithium-ion battery model considering the effects of temperature and capacity fading. In *2009 International Conference on Clean Electrical Power*, pages 383–386, June 2009.

[6] Gustavo Piske Fenner, Leonardo Weber Stringini, Camilo Alberto Sepulveda Rangel, and Luciane Neves Canha. Comprehensive Model for Real Battery Simulation Responsive to Variable Load. *Energies*, 14(11):3209, 2021.

[7] Bhavishya Goel and Sally A. McKee. A methodology for modeling dynamic and static power consumption for multicore processors. In *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 273–282, 2016.

[8] Yanan Gou and Xuezheng Han. Study of soc dynamic estimation method of power lithium battery. *IOP Conference Series: Materials Science and Engineering*, 452:032057, 12 2018.

[9] iAccessibility.net. ios battery low icon, 2025. Accessed: 2026-01-31.

[10] Young-Jin Kim Jun-Hyeok Park. Accurate power model for mobile amoled displays. *Electronics Letters*, 2015.

[11] Jaemin Kim, Naehyuck Chang, and Donghwa Shin. Mobile gps application design based on system-level power and battery status estimation. *Energies*, 14(17), 2021.

[12] Kuan-Ting Lee, Min-Jhen Dai, and Chiung-Cheng Chuang. Temperature-compensated model for lithium-ion polymer batteries with extended kalman filter state-of-charge estimation for an implantable charger. *IEEE Transactions on Industrial Electronics*, 65(1):589–596, 2017.

[13] Seongjun Lee, Jonghoon Kim, Jaemoon Lee, and Bo H Cho. State-of-charge and capacity estimation of lithium-ion battery using a new open-circuit voltage versus state-of-charge. *Journal of power sources*, 185(2):1367–1373, 2008.

[14] Luigi Mattia, Hamzeh Beiranvand, Walter Zamboni, and Marco Liserre. Lithium-ion battery thermal modelling and characterisation: a comprehensive review. *Journal of Energy Storage*, 129:117114, 2025.

[15] B. Saha and K. Goebel. "battery data set". *NASA Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA*, (2007).

[16] AG Stefanopoulou and Y Kim. System-level management of rechargeable lithium-ion batteries. *Rechargeable Lithium Batteries*, pages 281–302, 2015.