

Summary

Every smartphone user has encountered the feeling of questioning whether or not their phone will last throughout the day, or how much more time they have until their phone dies completely. The battery percentages displayed to the user can be unpredictable, unreliable and difficult to accurately understand. Our model develops a mathematical framework that considers this uncertainty and accounts for it.

We formulate smartphone battery loss as a piecewise-deterministic Markov process (PDMP), acknowledging deterministic energy depletion with discrete random transitions between usage states. The state of charge C_t evolves according to the stochastic differential equation. $\frac{\partial C_t}{\partial t} = v(X_t)(1)$ where v maps the current state X_t , to its related power draw. The usage states are modeled as a continuous-time Markov chain with transition rate matrix Q , capturing the random nature of phone activity.

The theoretical foundation of our model draws largely from results in two papers published in the 1980s. Davis's 1984 formalization of PDMPs provides the mathematical concept for the continuous-discrete system[4]. More closely, we adapted the Anick-Mitra-Sondi fluid queue model[1], developed to analyze data buffers in telecommunications, to address capacity drain in smartphone lithium-ion batteries.

We derive the governing dynamics by considering probability flow over infinitesimal time steps, converting our stochastic differential equation into a partial differential equation which gives the density of charge as a function of time, $f(c, t)$. $\frac{\partial f}{\partial t} + \frac{\partial f}{\partial c} V = fQ^T(2)$

The key output is not a single value of the battery time remaining, but rather a probability distribution. The cumulative probability of the battery being dead by time t emerges elegantly as $Pr(T_{death} < t) = \int_{-\infty}^0 f(c, t)dc(3)$. We simply integrate the density that has "flowed past" zero charge to obtain a CDF.

To implement our model we limited the possible usage states to suspended, active, and heavy using power measurements from Carroll and Heiser's component-level smartphone analysis [3], as well as sources that indicate relevant phone usage metrics. Because complete usage state transition statistics were not available through our research, we used time averaged probabilities combined with certain rate statistics to construct a transition rate matrix. We solve (2) with numerical methods to obtain a density describing phone charge as a continuous function of time.

Contents

1	Introduction	3
1.1	Problem	3
1.2	Definitions	3
1.3	Assumptions	3
2	Model: Continuous-time Markov Modulated Battery Capacity	5
2.1	Model Structure	5
2.2	Core Model	5
2.3	Finding Time-to-Empty	7
3	Model Application	7
4	Limitations, Future Work, and Error Analysis	10
4.1	Limitations	10
4.2	Future Work	11
4.3	Error/Sensitivity Analysis	12
5	Conclusions/Recommendations	12
6	Bibliography	13
7	AI disclaimer	14
7.1	AI Usage Report	14
7.1.1	1. OpenAI ChatGPT (February 1, 2026, 10:00 AM, GPT-4)	14
7.1.2	2. Google Gemini (February 1, 2026, 1:45 PM, Gemini 1.5 Flash)	15

1 Introduction

1.1 Problem

The problem asks us to develop a continuous-time mathematical model of a smartphone's battery, under the assumption that the phone has a lithium-ion battery, which returns the state of charge as a function of time under realistic usage conditions. Using this model, we should be able to predict the remaining time-to-empty under different conditions and compare these predictions to observed or plausible behavior.

1.2 Definitions

1. **User(s)** – One or more humans that interact with a phone.
2. **State of Charge (SOC)** – A measure of how much energy remains in a battery compared to its full capacity, expressed as a percentage.
3. **Time-to-Empty** – The estimated amount of time remaining before a battery is completely discharged.
4. **Phone Activity** – The workload placed on the device's subsystems at any given time. This includes factors such as processor load, network activity, display states, and peripheral states.
5. **Usage State** – One of finitely many states or modes that classify phone activity at any given time. These states combine what the user is doing with what the hardware of the phone is consuming. Usage states are mutually exclusive.

1.3 Assumptions

In any model, assumptions must be made. These assumptions simplify reality, and guide the direction, applicability, and effectiveness of the model. A model with bad assumptions is a bad model. In this section, we discuss the justification of each assumption made.

1. **Realistic usage conditions imply both user and external interaction with a phone are non-deterministic.**

The first assumption we make is randomness. A deterministic model supposes that all variables influencing the system are observable and measurable. However, phone activity is influenced by an immense number of variables, such as human behavior, location, etc. This prohibits us from knowing the full state of the system at any given time. Therefore, we must project the higher dimensional space containing all of these variables onto one which is manageable. The influence of the variables in higher dimensional space appears as randomness in our manageable space.

- 2. Under realistic usage conditions, phone activity can be classified into a finite number of usage states.**

The phone is always in exactly one usage state at any time. Our second assumption is aided by the fact that hardware usage does not slide smoothly from low to high, but rather, jumps between different, discrete, performance levels to assist the task at hand. Thus, we can conclude that this assumption would yield an accurate model, simplifying complex user and system behavior into a finite state space. This finite-state abstraction in our model is heavily influenced by Aaron Carroll and Gernot Heiser, who show that smartphone power consumption clusters into distinct operating regimes corresponding to different hardware and usage configurations. [3]

- 3. Phone activity is independent of the phone's state of charge.**

We assume activity independence. The primary reasoning for this is that user interactions with a phone are task driven rather than power driven. A user will use their phone in response to a call, or a need to check a map, etc, rather than a battery level. It is important to note that users and processes do react to SOC in the real world. A user may use their phone less when they have low battery, or they may turn on a power saving mode, or reduce their brightness. However, for the vast majority of the SOC discharge cycle, independent behavior is the norm. Thus we have chosen to make this assumption.

- 4. The probability that the phone transitions out of its current usage state depends only on the current usage state.**

We assume that usage state transitions are memoryless. Phone activity is reactive and immediate, rather than dependent on past actions. It is important to note that this is false in general. For example, the probability that the user turns their phone on is dependent on how long it has been since they had last checked it. If the user hasn't checked their phone for a long time, they may be more likely to check it. However, while the user's urge to check their phone may grow over time, the process of checking the phone is ultimately triggered by external factors such as notifications, calls, etc. These external factors are memoryless and dominate user urges.

- 5. The probability the phone transitions to or from a usage state does not change with time.**

We assume time homogeneity. During the average day, phone activity is independent of time. Again this is generally not true, a user is much more likely to interact with their phone when they aren't asleep. However, we attempt to get around by restricting the domain of our model to active usage, meaning the user of the phone is not incapacitated for the duration of the domain.

- 6. Given the phone is in a usage state, power consumption within that state is deterministic and constant.**

In reality, power consumption in a phone changes at microsecond rates. However, given the

generalized usage states that we assume, as well as the law of large numbers, we can assume a constant discharge rate for each state, yielding Deterministic Power within States.

7. Linear Discharge

The rate of change of the charge in the battery with respect to time is linearly proportional to the instantaneous power consumed by the phone at any time. The rate of change of the charge is never positive.

2 Model: Continuous-time Markov Modulated Battery Capacity

2.1 Model Structure

The SOC of a smartphone evolves through the combination of two fundamentally different processes: continuous energy drain of the battery and discrete changes in device usage. Aspects such as cpu load and network shift abruptly as the users interact with their phones while the battery depletes at a rate specific to the phone's current usage state. This behavior requires a framework that can effectively consider both the discrete and continuous components. Hence, we model the power usage as a piecewise-deterministic Markov process (PDMP). This is justified by assumption 4,5,6. A PDMP is a process with deterministic motion between random jumps modeled with a Markov structure. This, as mentioned above, correlates to the randomness of an open system. This approach, formalized by Davis [4], provides the mathematical methodology to track how probability distributions over SOC progress when the system switches unpredictably between distinct usage states.

2.2 Core Model

Let C_t be a random process representing the energy remaining in the battery at time t . C_t 's sample space is $(-\infty, c_0]$ where c_0 is the initial energy in the battery. We say that the SOC is 0, or the phone is dead if C_t takes a non positive value. Let X_t be a continuous-time Markov chain (CTMC) which draws a value from the usage states. Let v be a function which maps the usage state to the power used at that given usage state. Our model is as follows:

$$\frac{\partial C_t}{\partial t} = v(X_t) \tag{1}$$

Charge drains at the rate the phone is consuming power and power consumption jumps stochastically between discrete levels. Transitions between usage states occur at random times represented by a rate matrix Q , where q_{ij} is the rate of jumping from mode i to mode j . In order to extract practical results from our model, we must convert the stochastic differential equation into a partial differential equation that gives a probability density as its solution. In order to find the desired partial differential equation, we modified the system laid out in [1]. Our goal was to generalize

the reasoning used in the Anick-Mitra-Sondhi buffer model and apply similar reasoning to address the capacity flow of a lithium ion battery. (7.1.1).

Let $v(i)$ be a function mapping the state i to the power draw in state i .

Let Q be the transition rate matrix defined by the CTMC X_t .

Let $f_i(t, c)$ be defined as the probability density that at a given time $X_t = i$ and $C_t = c$, formally $P(X_t = i, C_t \in [c, c + dc]) \approx f_i(t, c)dc$.

Consider a small update in time Δt . We observe how this step effects f_i . Observe the following relation to be true,

$$\underbrace{f_i(c, t + \Delta t)}_{\text{new density}} = \underbrace{f_i(c - v(i)\Delta t, t)}_{\text{drift term}} \underbrace{(1 - q_{ii}\Delta t)}_{\text{prob. stay in state } i} + \underbrace{\sum_{j \neq i} f_j(c, t)q_{ji}\Delta t}_{\text{contribution of other states}}$$

We can expand the drift term with a Taylor expansion,

$$f_i(c - v(i)\Delta t, t) \approx f_i(c, t) - v(i)\Delta t \frac{\partial f_i}{\partial c}$$

We substitute,

$$f_i(c, t + \Delta t) = \left(f_i(c, t) - v(i)\Delta t \frac{\partial f_i}{\partial c} \right) (1 - q_{ii}\Delta t) + \sum_{j \neq i} f_j(c, t)q_{ji}\Delta t$$

Removing second order terms gives,

$$f_i(c, t + \Delta t) - f_i(c, t) = -v(i)\Delta t \frac{\partial f_i}{\partial c} + -f_i(c, t)q_{ii}\Delta t + \sum_{j \neq i} f_j(c, t)q_{ji}\Delta t$$

Now we divide by Δt , take the limit as $\Delta t \rightarrow 0$, and combine the sum,

$$\frac{\partial f_i}{\partial t} = -v(i) \frac{\partial f_i}{\partial c} + \sum_j f_j(c, t)q_{ji}$$

Now suppose $f = [f_1(c, t), f_2(c, t), \dots]$ is a $1 \times n$ vector of all f_i . V is a diagonalized version of $v(i)$. The resulting vector equation is

$$\frac{\partial f}{\partial t} + \frac{\partial f}{\partial c} V = f Q^T. \quad (2)$$

The equation above is much more applicable than equation (1). It gives a vector of densities contributed from each usage state. To find the overall density $f(c, t)$ we sum the vector. In order to obtain a solution for (2) we impose the condition that $f(c, 0) = \delta(c - c_0)$ where c_0 is the initial

capacity. This means that the probability that the phone is charged to c_0 is 1. Laplace methods may help solve (2), however in this paper we simply use code (7.1.2) to solve for f numerically.

2.3 Finding Time-to-Empty

Rather than enforcing a strict boundary condition that says a $C_t \geq 0$, we defined C_t as having a sample space $(-\infty, c_0]$. This means that the resulting density $f(c, t)$ is defined in the same range. We could use some sort of flow conditions to bound the c axis, but we chose a less intensive solution. We leave c unbounded, then we claim that probability that the phone had died (T_{death}) before time t is as follows:

$$Pr(T_{death} < t) = \int_{-\infty}^0 f(c, t) dc \quad (3)$$

This happens to be the CDF, the probability that the phone has died before time t . Given any conditions we can obtain the density of the phone having died at time t just by integrating the function obtained from equation (2). This is where the power of a probabilistic model is shown. We don't just obtain a single number for phone battery life, but a whole distribution.

3 Model Application

To demonstrate our model we have chosen an example smartphone with available power data and statistics, the Openmoko Neo Freerunner Smartphone [3]. This phone is from 2008, so it is rather outdated, but can still help us demonstrate an effective model. The phone contains a 1200 mAh lithium ion battery, and according to it's datasheet[6] supports a talk time of approximately 3-4 hours. In addition to statistics about the phone, we collected usage data from various sources [2], [9], [7], [8]. This data is shown in the table below.

Table 1: Usage State Power Consumption for Openmoko Neo Freerunner Smartphone

Usage State	Power Usage(mW)	Current (mA)	Time in State (Min)	Prob. in State
Suspended	68.6	18.5	1027	0.7132
Active	268.8	72.6	325	0.2257
Heavy Use	1054.3	284.9	88	0.0611

From assumption 7 we claim that the change in the charge is proportional to the power usage, but in order to make units match we must scale the power usage by voltage to obtain the current. However, we know that voltage changes as the charge in the battery depletes but we choose to ignore this and take the voltage to be a constant 3.7 volts, a standard value for a lithium ion battery.

To demonstrate a simple model, we have chosen 3 states from the state space outlined in [3]. This means that our CTMC will draw from these three states, so the Q and V matrices will be 3×3 . Before we explore how we create a Q matrix with steady state probabilities, consider the following

small-scale example. Suppose that the phone only stayed in it's heavy state. Then according to datasheet, the battery should last approximately 3 – 4 hours. Let's verify this in our model. We can construct a Q matrix such that the probability of transitioning into the heavy state is very likely, and the probability of transitioning out of the heavy state is very unlikely. In this case, we should expect to see only the contribution of the heavy use state on the CDF, shown below:

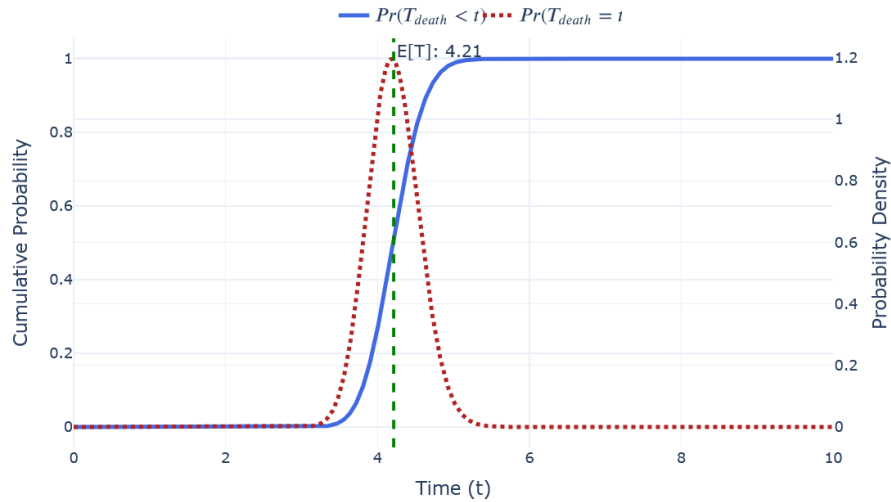


Figure 1: $Pr(T_{death} < t)$ with $E(T_{death}) = 4.21$ hours

We achieved a value of 4.21 hours for battery life, this closely matches the 3-4 hour prediction laid out in the phones datasheet. Now let's use the model with realistic transition rates.

The average person picks up there phone 113 times per day[2]. If we assume that each time the user picks up the phone it transitions from the suspended state to the active state, then the rate at which the user transition between suspended and active $r_{sus,act} = 113$ transitions/day = 4.708 transitions/hr.

You may notice that we don't have statistics for the transition rates between all states. Due to limitations in available data, we choose to construct the rate matrix Q from the steady state probabilities in Table 1. The goal is to find Q given the steady state probabilities of being in each state. From the properties of a Markov chain, let π be a vector of steady state probabilities for each state, then $\pi Q = 0$. However this does not constrain Q to a single "correct solution." In fact, if we find a matrix K that satisfies $\pi K = 0$, we can scale that matrix and maintain the relationship. We can further constrain Q in the following way.

1. Find a matrix K . We must assume that every transition is possible, and we must assume balance $\pi_i q_{ij} = \pi_j q_{ji}$. Then we can use the metropolis rule and a corresponding algorithm to find a K .
2. Scale K by alpha to match a target rate. If we know a single rate R_{ij} , the transition rate from i to j , then the following relationship must be true: $R_{ij} = \pi_i q_{ij} = \pi_i b k_{ij}$ where b is the factor by which we scale K .

3. Then $Q = bK$.

$$Q = \begin{bmatrix} -10.4444 & 7.8333 & 2.6111 \\ 15.6667 & -20.8889 & 5.2222 \\ 15.6667 & 15.6667 & -31.3333 \end{bmatrix}$$

Figure 2 shows the results of utilizing the above matrix in our model.

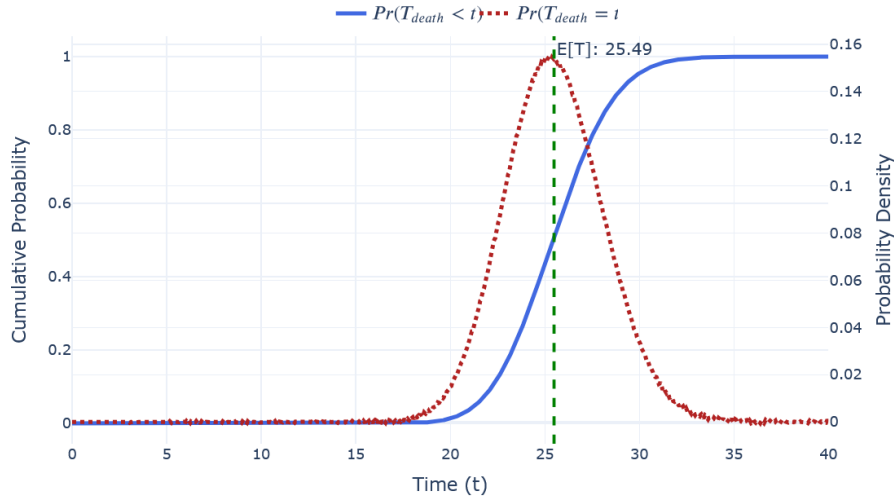
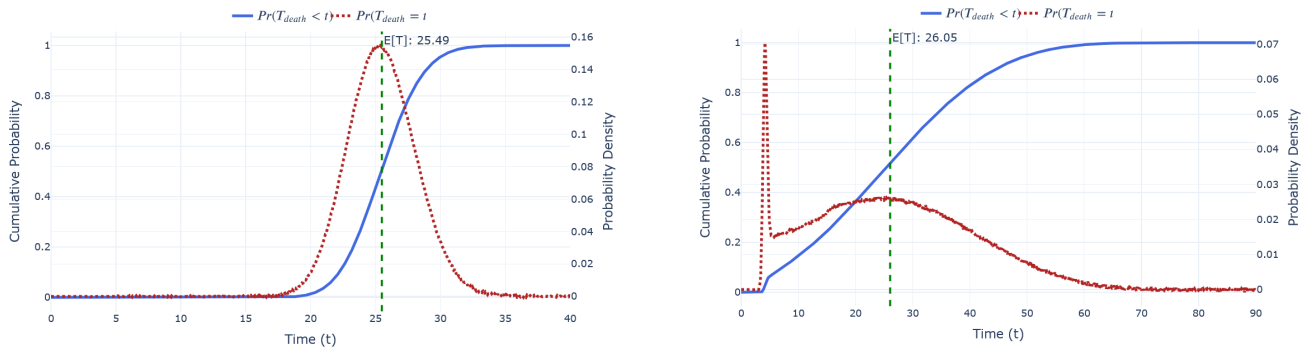


Figure 2: $Pr(T_{death} < t)$ with $E(T_{death}) = 25.49$ hours

One justification for the effectiveness of our model is confirmation that the different states, and rates between them affect this density directly. Figure 3 shows a comparison of two different transition rates. In Figure 3b, we can see the effect of the heavy use state on the CDF. With low transition rates, the user is more likely to be "stuck" in the heavy use state. This state's contribution is observed as a spike around 4 hours. Notice the variance increases with lower rates, this is because the user activity isn't "averaged" by switching between usage states.



(a) High Transition Rate: $Pr(T_{death} < t)$

(b) Low Transition Rate: $Pr(T_{death} < t)$

Figure 3: Comparing Transition Rates

To further understand what is happening with the model, we show a visual of the density function for Low Transition Rates in Figure 4. The density clearly shows different paths of usage states. This visual shows the initial condition with a delta function.

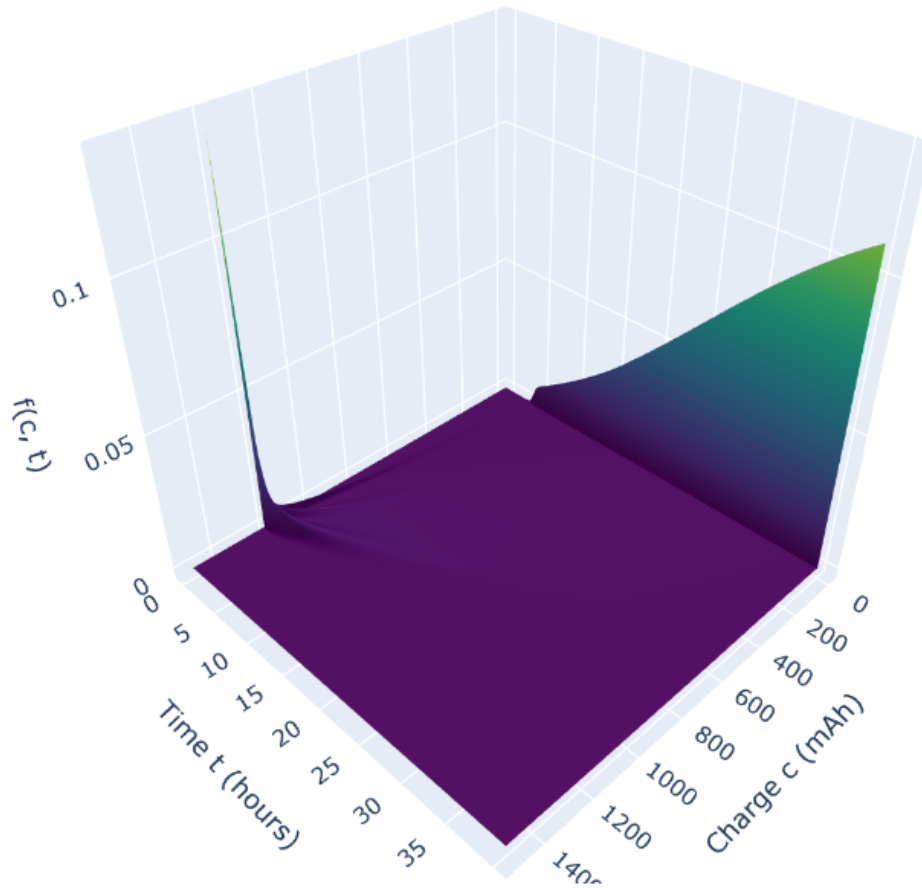


Figure 4: Low Transition Rates: $f(c, t)$ hours

Figure 4 shows how $f(c, t)$ evolves in 3D space. Specifically, the different “ripples” that can be observed on the time-charge plane can be interpreted as the user being stuck in a given usage state. For example, the heavy use path visualized as the spike in Figure 4 causes the probability spike we see at 4 hours in Figure 3b.

4 Limitations, Future Work, and Error Analysis

4.1 Limitations

- **Assumption 5 (Time Homogeneity):** Due to the fact that we built our model around a PDMP, this forced us to assume that the transition rates displayed in our Q matrix were constant. This assumption, however, creates issues in a 24-hour time frame as we

can reasonably assume that these rates would sharply decrease in the users' hours of sleep, yielding potential inaccuracies in our model.

- **Simplification of Linear Discharge:** In our model, we assume that the rate of charge is proportional to power consumption. Lithium-Ion batteries, however, do not tend to display linear discharge curves, and as a result, high current draw states may cause the battery to deplete faster than our assumption predicts.
- **State Generalization:** Our model generalizes user and system activities into a set of broad usage states. The limited number of states we used (suspended, active, heavy use) can lead to inaccuracies as user interactions, triggering different system activities, can potentially be better represented by a state which we have not yet defined.
- **Temperature/Environmental Effects:** Temperature can play a big role in the depletion of state of charge. Our model, however, emphasizes user behavior patterns as a means of cycling between states which do not take temperature into account, leading to potential inaccuracies.

4.2 Future Work

- **State of Charge Scaling:** In this model, the state of charge is a physical constant displaying the number of coulombs in the battery. Realistically we would have to scale the state of charge to accurately reflect it as a percentage between 0 and 100 that the phone would reflect.
- **Battery Age Effects:** Battery aging primarily affects the effective capacity of the battery rather than the instantaneous rate of repletion. For an older battery, even though the phone may reflect it is charging to 100%, its capacity likely has dropped to only 80% of its original capacity.
- **Non-Homogenous Markov Chains:** Implementing transition rates for our Q matrix that take time into account would allow our model to make more accurate predictions, as transition rates will drop significantly for the part of the day in which the user sleeps.
- **Temperature Variables:** Including temperature as a variable which affects the state of charge would improve our models predictions.
- **Modeling Different Users with Accurate Transition Rates:** Ideally our model would be most efficient in comparing the difference in battery capacity drain between different types of users. With access to specific user data that accurately reflects the transition rates of the Q matrix, the model would be effectively more efficient and applicable.

4.3 Error/Sensitivity Analysis

Sensitivity Analysis: To perform a sensitivity analysis, we could vary both transition and consumption rates by a factor of 10% and calculate the percent change in the time it takes the battery to deplete. In our model, we would expect this to change the expected value, however, we believe our distribution around our expected value remains the same.

Error Analysis: In terms of error analysis, we can compare our PDE results to a stochastic simulation. Specifically, we could run thousands of simulated phones through a randomized walk-through based on our Q matrix. We can then plot both the histograms of the phones' death times against the probability density curve $f(0, t)$. Finally, we can employ the root mean square error as a metric which quantifies how accurately our PDE approximates the stochastic process.

5 Conclusions/Recommendations

The results reinforce commonly understood factors that play a role in a phone's state of charge. Our model matches real user experience. Even if two independent days start at the same state of charge they can end very differently because the phone will likely reside in different usage states for unpredictable periods of time.

This model performs best when phone activity can be effectively partitioned into a set of clear usage states with approximately stable average power consumption for each state. Under these conditions the PDMP clearly separates deterministic drain for each state from the stochastic switching between these states. The current model does not address other drivers such as temperature, day/night effects on the model, and brightness, however, these factors can be explored through future work and expansion of the model.

As mentioned above our model supports recommendations that are often found online. To maintain a longer battery life avoid longterm heavy use as well as leaving processes running in the background when unneeded. To keep things concise, being mindful of usage patterns allows users to better estimate how long their phone is likely to last.

6 Bibliography

References

- [1] Anick, Daniel, Debasis Mitra, and Mukul M. Sondhi. 1982. "Stochastic Theory of a Data-Handling System with Multiple Sources." *Bell System Technical Journal* 61 (8): 1871-1894.
- [2] Bradley, Abigail H. M., and Andrea L. Howard. 2023. "Stress and Mood Associations With Smartphone Use in University Students: A 12-Week Longitudinal Study." *Clinical Psychological Science* 11 (5): 921-41.
- [3] Carroll, Aaron, and Gernot Heiser. 2010. "An Analysis of Power Consumption in a Smartphone." In *Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference*, 21-21. Berkeley, CA: USENIX Association.
- [4] Davis, M. H. A. 1984. "Piecewise-Deterministic Markov Processes: A General Class of Non-Diffusion Stochastic Models." *Journal of the Royal Statistical Society: Series B (Methodological)* 46 (3): 353-388.
- [5] Vassilis Kostakos, Denzil Ferreira, Jorge Gonçalves, and Simo Hosio, "Modelling Smartphone Usage: A Markov State Transition Model," *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16)*, Heidelberg, Germany, September 12-16, 2016.
- [6] OpenMoko. n.d. "Neo FreeRunner Hardware." *OpenMoko Wiki*. https://wiki.openmoko.org/wiki/Neo_FreeRunner_Hardware. (Accessed February 2, 2026).
- [7] Statista. 2025. "Worldwide Game Apps Time Spent Daily by Age." *Statista*. (Accessed February 1, 2026).
- [8] Statista. 2025. "U.S. Daily Time Spent on Social Media Platforms." *Statista*. <https://www.statista.com/statistics/1301075/us-daily-time-spent-social-media-platforms/>. (Accessed February 1, 2026).
- [9] YouGov. 2024. "How Long Do Americans Talk on Calls in a Day?" *YouGov* (July 1, 2024).

7 AI disclaimer

Our team used artificial intelligence tools in three capacities: as an educational aid, a literature review assistant, and a coding support resource.

The most significant use was bridging differences in technical background among team members. One member has prior experience with mathematical modeling and an education in engineering proving very valuable for our model; the others were new to this context. AI provided concise explanations of unfamiliar concepts and assisted in interpreting the partial differential equation governing our model as it evolved from its initial stochastic formulation. As the model became more complex than expected, we deliberately continued pursuing this direction because it produced the best model. AI was used as a supportive tool to deepen our understanding when needed. This supplemented, rather than replaced, our engagement with academic sources.

AI also helped identify relevant literature within time constraints by assessing whether sources pertained to our modeling approach. All incorporated information was independently verified and cited by the team. During coding, AI assisted with syntax clarification, debugging, and structuring numerical simulations. All modeling decisions, assumptions, and interpretations were made by the team; the mathematical formulation was developed independently prior to implementation. Additionally, AI assisted in refining this disclaimer for clarity, and Grammarly Pro was used for grammar and spelling checks throughout the paper.

7.1 AI Usage Report

7.1.1 1. OpenAI ChatGPT (February 1, 2026, 10:00 AM, GPT-4)

Used to assist in understanding the derivation of the partial differential equation governing our piecewise-deterministic Markov process model.

Query 1a: Given our core model where C_t represents battery energy, X_t is a CTMC over usage states, and $\frac{dC_t}{dt} = v(X_t)$, walk me through the exact steps on how to take the SDE into a PDE.

Output 1a: This is a transition from a Lagrangian view (tracking one path) to an Eulerian view (tracking the distribution). Since the randomness is discrete (the Markov chain), we use the Chapman-Kolmogorov equations.

1. Define the Joint Density: Let $f_i(c, t)dc = P(C_t \in [c, c + dc], X_t = i)$.
2. Conservation Law: Over Δt , the probability $f_i(c, t + \Delta t)$ depends on staying in state i (with shifted energy) or jumping from state j .
3. Infinitesimal Look-Ahead: $f_i(c, t + \Delta t) \approx f_i(c - v(i)\Delta t, t)(1 + q_{ii}\Delta t) + \sum_{j \neq i} f_j(c, t)q_{ji}\Delta t$
4. Taylor Expansion: $f_i(c - v(i)\Delta t, t) \approx f_i(c, t) - v(i)\Delta t \frac{\partial f_i}{\partial c}$.

5. The Limit: Subtract $f_i(c, t)$, divide by Δt , and let $\Delta t \rightarrow 0$ to get: $\frac{\partial f_i}{\partial t} = -v(i)\frac{\partial f_i}{\partial c} + \sum_j q_{ji}f_j$

Query 1b: More details with steps 2 and 3.

Output 1b:

- Step 2 (Conservation): Think of probability as a fluid. It must be accounted for. For energy to be c at $t + \Delta t$, it had to be $c - v(i)\Delta t$ earlier.
- Step 3 (The Math):
 - Staying in i : Probability is $(1 + q_{ii}\Delta t)$. The energy “drifts” backward.
 - Jumping from $j \rightarrow i$: Probability is $q_{ji}\Delta t$. Energy is assumed continuous during the jump.
 - Summation Trick: Since $q_{ii} = -\sum_{j \neq i} q_{ji}$, we group all terms into $\sum_j q_{ji}f_j$.

Query 1c: More help on the input flux and algebraic simplification steps.

Output 1c:

- Step B (Input Flux): When a phone switches from “Gaming” to “Idle,” it carries its current battery level with it. State i “inherits” the density from state j .
- Step D (Algebra): We combine the “Leaving” term ($q_{ii}f_i$) and the “Entering” terms ($\sum_{j \neq i} q_{ji}f_j$) into one sum. In matrix form, this is $Q^T f$.

Query 1d: But isn’t the term $q_{ii}f_i(c - v\Delta t, t)$ and not $q_{ii}f_i(c, t)$, so how can you add it?

Output 1d: Correct! But when we expand $(1 + q_{ii}\Delta t)[f_i(c, t) - v(i)\Delta t\frac{\partial f_i}{\partial c}]$, the product of $q_{ii}\Delta t$ and $v(i)\Delta t\frac{\partial f_i}{\partial c}$ results in a $(\Delta t)^2$ term. As $\Delta t \rightarrow 0$, $(\Delta t)^2$ vanishes. We are left with $q_{ii}\Delta t f_i(c, t)$. This allows us to evaluate everything at c in the summation.

Query 1e: How do you simplify the sum over j to fQ ?

Output 1e: This is matrix-vector multiplication. If f is a column vector, the sum $\sum_j q_{ji}f_j$ is exactly the definition of the i -th row of $Q^T f$...

Query 1f: But for the first two f ’s in the equation, it’s f_i and the last one is f_j .

Output 1f: Exactly. The PDE for a specific state i describes: (1) f_i (Left side): How that specific state’s population moves through energy space. (2) f_j (Right side): How other states feed into state i . The matrix form stacks these equations...

7.1.2 2. Google Gemini (February 1, 2026, 1:45 PM, Gemini 1.5 Flash)

Used to develop a numerical solver, write code to solve and display density function.

Query 2a: Write a python program that find $f(x, t)$ given the vector pde $\frac{\partial f}{\partial t} + \frac{\partial f}{\partial c}V = fQ$. Q is a 3x3 transition rate matrix, V is a diagonal matrix containing the rates. and I have some boundary conditions of f like $f(c, 0)$ is $\delta(c - c_0)$, $f(c, \infty) = \delta(c)$.

Output 2a: Numerical Strategy: Upwind Scheme. Because the V matrix contains rates (speeds), we use an Upwind Scheme to maintain stability.

```

import numpy as np
import matplotlib.pyplot as plt

# — Parameters —
L = 10.0          # Spatial domain length
N_c = 200        # Spatial grid points
dt = 0.01        # Time step
T_final = 2.0    # Total simulation time
c = np.linspace(0, L, N_c)
dc = c[1] - c[0]

# V: Diagonal matrix of rates (speeds for each component of f)
V_diag = np.array([1.5, 1.0, 0.5])
V = np.diag(V_diag)

# Q: 3x3 Transition rate matrix
Q = np.array([[ -0.5,  0.5,  0.0],
              [  0.3, -0.6,  0.3],
              [  0.0,  0.4, -0.4]])

# — Initial Condition: f(c, 0) approximation —
c0 = 2.0
f = np.zeros((3, N_c))
for i in range(3):
    f[i, :] = np.exp(-100 * (c - c0)**2)

# — Solver: Upwind Finite Difference —
steps = int(T_final / dt)
for _ in range(steps):
    f_new = f.copy()
    interaction = f.T @ Q
    for i in range(3):
        df_dc = np.zeros(N_c)
        df_dc[1:] = (f[i, 1:] - f[i, :-1]) / dc
        f_new[i, :] = f[i, :] - dt * (V_diag[i] * df_dc) + dt *
            interaction[:, i]
    f = f_new.copy()

```

Query 2b: the output should be a 3d graph showing the full density as a function of space and

time.

Output 2b: Since f is a vector with three components, the "total density" is represented by the sum of those components, $\sum f_i(c, t)$.

```
# Calculate Total Density: Sum of the 3 components
total_density = np.sum(f_hist , axis=0)

# --- 3D Plotting ---
fig = plt.figure(figsize=(12, 8))
ax = fig.add_subplot(111, projection='3d')
C, T = np.meshgrid(c, t)
surf = ax.plot_surface(C, T, total_density.T, cmap='viridis',
    edgecolor='none')
ax.set_xlabel('Space-(c)')
ax.set_ylabel('Time-(t)')
ax.set_zlabel('Total-Density')
plt.show()
```

Query 2c: ok redo this with a solver from a library. Remember f is a vector function so it must be summed before graphing Make $c_0 = 10$ and the rates be whole numbers around 3.

Output 2c: We use `scipy.integrate.solve_ivp` for robust integration via the Method of Lines.

```
from scipy.integrate import solve_ivp

V_diag = np.array([4, 3, 2])
Q = np.array([[ -3,  2,  1], [ 1, -2,  1], [ 2,  1, -3]])

def pde_system(t, y):
    f = y.reshape((3, N_c))
    dfdt = np.zeros_like(f)
    interaction = (f.T @ Q).T
    for i in range(3):
        df_dc = np.zeros(N_c)
        df_dc[1:] = (f[i, 1:] - f[i, :-1]) / dc
        dfdt[i, :] = -V_diag[i] * df_dc + interaction[i, :]
    return dfdt.flatten()

sol = solve_ivp(pde_system, (0, 4.0), y0, t_eval=np.linspace(0, 4,
    100))
```

Query 2d: the behavior is increasing, but it should be decreasing with time.

Output 2d: Adjusted the Q matrix to be dissipative and flipped the velocity sign to ensure transport moves toward $c = 0$.

Query 2e: incorporate another boundary condition, name $f(x, t)$ $t \rightarrow \infty$ is a delta at 0. Enforce the boundry that $f(0, t)$ is always increasing. You are adding these boundary conditions in the logic rather than in the pde directly.

Output 2e: To enforce the boundary condition at $c = 0$, we define the derivative $\frac{df}{dt}$ at that point to specifically be the arrival flux.

```
# Boundary Condition logic inside the PDE system
for i in range(3):
    v = V_diag[i]
    dfdt[i, 1:-1] = v * (f[i, 2:] - f[i, 1:-1]) / dx
    # Boundary Condition at c = 0 (Left side)
    flux_in = v * (f[i, 1] - f[i, 0]) / dx
    dfdt[i, 0] = flux_in + interaction[i, 0]
```

Query 2f: its not working, remember that his function is a density so the integral of the function with respect to space must always be 1.

Output 2f: To maintain $\int f(c, t)dc = 1$, we normalize the initial condition and ensure strict flux-balance at the boundary.

```
# — Final Refined Solver with Mass Conservation —
f0_raw = np.exp(-2 * (c - c0)**2)
# Normalize so integral over space = 1
f0_normalized = (f0_raw / (np.sum(f0_raw) * dx)) / 3.0
y0 = np.concatenate([f0_normalized for _ in range(3)])

def pde_system(t, y):
    f = y.reshape((3, N_c))
    dfdt = np.zeros_like(f)
    interaction = (f.T @ Q).T
    for i in range(3):
        v = V_diag[i]
        dfdt[i, 1:-1] = v * (f[i, 2:] - f[i, 1:-1]) / dx
        dfdt[i, -1] = v * (0 - f[i, -1]) / dx
        # Formal flux balance at c=0
        flux_in = v * (f[i, 1] - f[i, 0]) / dx
        dfdt[i, 0] = flux_in + interaction[i, 0]
    return dfdt.flatten()
```

Would you like me to help with the formal mathematical derivation of the flux-conserving boundary condition to accompany this code in your report?