BATTERY-SIM-AGENT: LEVERAGING LLM-AGENT FOR INVERSE BATTERY PARAMETER ESTIMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Parameterizing high-fidelity "digital twins" of batteries is a critical yet challenging inverse problem that hinders the pace of battery innovation. Prevailing methods formulate this as a black-box optimization (BBO) task, employing algorithms that are sample-inefficient and blind to the underlying physics. In this work, we introduce a new paradigm that reframes the inverse problem as a reasoning task, and present BATTERY-SIM-AGENT, the first framework to deploy a Large Language Model (LLM) agent in a closed loop with a high-fidelity battery simulator. The agent mimics a human scientist's workflow: it interprets rich, multi-modal feedback from the simulator, forms physically-grounded hypotheses to explain discrepancies, and proposes structured parameter updates. On a systematically constructed benchmark suite spanning diverse battery chemistries, operating conditions, and difficulty levels, our agent significantly outperforms strong BBO baselines like Bayesian optimization in identifying accurate parameters. We further demonstrate the framework's capability in complex long-horizon degradation fitting tasks and validate its practical applicability on real-world battery datasets. Our results highlight the promise of LLM-agents as reasoning-based optimizers for scientific discovery and battery parameter estimation.

1 INTRODUCTION

The transition to a sustainable energy future is intrinsically linked to advancements in battery technology. From electrifying transportation to stabilizing power grids, next-generation batteries are a critical need. However, the physical development and testing of these batteries is a major bottleneck. Characterizing a battery's performance and degradation over its lifetime can require thousands of hours of continuous cycling. A promising alternative is to build *digital twins*—high-fidelity virtual replicas instantiated in physics-based simulators such as PYBAMM (Sulzer et al., 2021). Yet, realizing this vision hinges on solving a fundamental *inverse problem*: the simulators require microscopic parameters that cannot be directly measured, while only macroscopic data are available. Accurately identifying these parameters is a long-standing challenge in battery engineering (Subramanian & Braatz, 2013; Prasad et al., 2015; Gopinath et al., 2016).

Traditionally, this inverse problem is formulated as a black-box optimization (BBO) task. As detailed in Table 1, researchers have long employed algorithms like Bayesian optimization (Wang & Jiang, 2023; Jiang et al., 2022) or genetic algorithms (Zhang et al., 2014; Magnor & Sauer, 2016; Blaifi et al., 2016) to iteratively query the simulator and minimize the mismatch between simulated and observed data. While flexible, these methods are inherently *blind*: they treat the simulator as an opaque oracle and lack physical intuition. This often leads to high sample complexity and convergence to implausible local minima.

The limitations of blind search motivate a paradigm shift. With the advent of Large Language Models (LLMs) as powerful *reasoning engines*, a new wave of "agentic science" is emerging, where LLM-powered agents automate complex scientific discovery workflows (Wei et al., 2025). These agents have shown success in solving inverse problems in diverse fields like materials science (Wu et al., 2025) and solid mechanics (Ni & Buehler, 2024). This inspires us to ask a central question: *can the inverse problem of battery parameter estimation be reframed not as a brute-force search, but as a reasoning-driven scientific workflow guided by an LLM-agent?*

We answer this question affirmatively by introducing **Battery-Sim-Agent**, a framework that pioneers the use of an LLM-agent in a *simulator-in-the-loop* configuration to solve the inverse problem in battery science. Our agent acts as an AI scientist: in each iteration, it is presented with rich, multi-modal feedback that compares the current simulation against experimental data. This includes not only quantitative error metrics but also visual overlays of voltage curves, allowing it to identify qualitative discrepancies like misaligned plateaus or incorrect slopes. Based on this evidence, the agent formulates a physical hypothesis (e.g., "premature voltage drop suggests an electrolyte transport limitation") and proposes a targeted parameter update in a structured JSON format. To ensure stability and long-term planning, the agent is equipped with a persistent memory of its past actions and their outcomes. We validate this framework through a comprehensive experimental suite spanning diverse battery chemistries, operating conditions, and difficulty levels, demonstrating that our agent consistently achieves 67-95% reduction in curve-matching error compared to traditional black-box optimization baselines. We further showcase the framework's capability in complex long-horizon degradation fitting tasks and validate its practical applicability on real-world battery datasets.

Our main contributions can be summarized as follows:

1. We introduce a novel agentic framework that reframes the battery inverse problem from a blind mathematical search into an interpretable, hypothesis-driven scientific workflow, pioneering the use of a simulator-in-the-loop LLM-agent in this domain.
2. We architect a suite of principled modules specifically designed for this workflow, including a multi-modal feedback system that translates complex simulation data into actionable insights for the agent, and a persistent memory to enable robust, long-horizon reasoning.
3. We provide a comprehensive experimental validation of our framework, demonstrating on extensive simulated benchmarks spanning diverse chemistries and difficulty levels, as well as real-world battery datasets, that our reasoning-based approach achieves 67-95% reduction in parameter estimation error compared to traditional black-box optimization methods.

Aspect	**Traditional BBO**	**Battery-Sim-Agent (Ours)**
Search Paradigm	Blind Search	Hypothesis-Driven
Feedback Signal	Scalar Loss	Rich & Multi-modal
Interpretability	Low	High
Efficiency	Sample-Inefficient	Guided & Efficient

Table 1: Comparison of Traditional Black-Box Optimization and Battery-Sim-Agent.

2 Background

2.1 The Challenge of Parameterizing Battery Digital Twins

A central goal in battery science is to create high-fidelity "digital twins" that can accurately predict a battery's performance and long-term degradation. This is a critical yet challenging task. The degradation of a battery is a slow process, often requiring hundreds or thousands of charge-discharge cycles to observe significant capacity fade. While macroscopic data from these cycles—such as terminal voltage, current, and total capacity—are readily available, they are merely symptoms of underlying microscopic processes.

The true drivers of battery behavior are a set of internal, microscopic physical and chemical parameters. These include properties like the porosity of the electrodes, the diffusion coefficients of lithium ions in the solid and electrolyte phases, and kinetic reaction rates. These parameters, collectively denoted as a vector θ, govern the complex system of coupled partial differential equations (PDEs) that form the core of high-fidelity electrochemical models like the Doyle-Fuller-Newman (DFN) model (Subramanian & Braatz, 2013). However, directly measuring these parameters is often prohibitively expensive, requires specialized laboratory equipment, or is even physically impossible without destroying the battery cell. This creates a fundamental gap between what we can easily observe (macroscopic data) and what we need to know to build an accurate model (microscopic parameters). The task of bridging this gap of inferring the hidden parameters θ from observable data is known as the inverse problem of parameter estimation in battery science (Prasad et al., 2015; Gopinath et al., 2016).

2.2 Formulation as a Black-Box Optimization Problem

Traditionally, the inverse problem is formulated as a black-box optimization (BBO) task. The goal is to find a parameter vector $\theta^\star$ that minimizes a loss function, $\mathcal{L}(\theta)$, which quantifies the discrepancy between the simulator's outputs and the experimentally observed data. To overcome the ill-posedness of the problem, this matching must be performed across a set of diverse experimental protocols $\mathcal{P}$ (e.g., different charge/discharge C-rates (Balog & Davoudi, 2013; Pantoja et al., 2022)).

For each protocol $p \in \mathcal{P}$, we collect a set of observed macroscopic trajectories, Y_p^{obs}, which can include terminal voltage $V(t)$, current $I(t)$, and cycle capacity Q. The simulator, given parameters θ, produces corresponding simulated trajectories $Y_p^{\text{sim}}(\theta)$. The overall objective is to minimize a composite loss function, typically a weighted sum over all protocols:

$$\theta^\star = \arg\min_\theta \mathcal{L}(\theta), \quad \text{where} \quad \mathcal{L}(\theta) \;=\; \sum_{p\in\mathcal{P}} w_p \cdot d\big(Y_p^{\text{sim}}(\theta),\, Y_p^{\text{obs}}\big) \;+\; \lambda R(\theta). \tag{1}$$

Here, d is a distance metric that can compare multiple trajectories, w_p are weights for each protocol, and $R(\theta)$ is a regularization term. This optimization is notoriously difficult for three main reasons:

- **Expensive, Non-Differentiable Black-Box:** Each evaluation of $\mathcal{L}(\theta)$ requires a full, computationally costly simulation, and the gradients $\nabla_\theta \mathcal{L}$ are typically unavailable.
- **Ill-Posedness:** The problem is ill-posed, meaning many different parameter sets θ can produce nearly identical output trajectories (a phenomenon known as equifinality), making the minimum of the loss landscape difficult to identify uniquely.
- **High Dimensionality:** The parameter vector θ can be high-dimensional, making a brute-force search of the parameter space intractable.

2.3 Simulator-in-the-Loop and Agentic Science

The limitations of treating the simulator as an opaque oracle have motivated a shift towards more interactive paradigms. A common approach in computational science is the "simulator-in-the-loop" model, where a human expert iteratively adjusts parameters based on simulation outputs. Recently, the rise of Large Language Models (LLMs) as powerful reasoning engines has opened the door to automating this process at scale (Hu et al., 2025). This has led to the emergence of "agentic science" where LLM agents take on the role of the human scientist (Wei et al., 2025). These agents have shown success across diverse domains: molecular design (Wu et al., 2025), inverse problems in solid mechanics (Ni & Buehler, 2024), and galaxy observation interpretation (Sun et al., 2024). Instead of being guided by a single scalar loss value, LLM agents can interpret rich, structured feedback from simulators—including full data trajectories, visual plots, and diagnostic error messages. This allows agents to reason about physical causes of discrepancies and formulate targeted hypotheses, reframing optimization from a blind search into an intelligent, hypothesis-driven workflow. This emerging paradigm provides the direct motivation for our work.

3 Method

To address the complex, multi-objective, and heterogeneous optimization challenge formulated in Sec. 2.2, we introduce Battery-Sim-Agent. The core innovation of our framework is to replace the conventional "blind" numerical search of traditional BBO with a reasoning engine that can interpret and act upon the rich, structured information produced by a physics-based simulator. An LLM-agent, acting as an AI scientist, can handle the multi-objective nature of the problem by reasoning about qualitative trade-offs, and navigate the heterogeneous parameter space by proposing targeted, mechanism-aware updates. This allows us to reframe the inverse problem as an interpretable, hypothesis-driven workflow.

3.1 Agent-Driven Optimization Formulation

Aligned with the optimization objective formulated in Eq. equation 1, our overall goal is to find parameters $\theta^\star$ that minimize the composite loss $\mathcal{L}(\theta)$. However, unlike traditional methods that

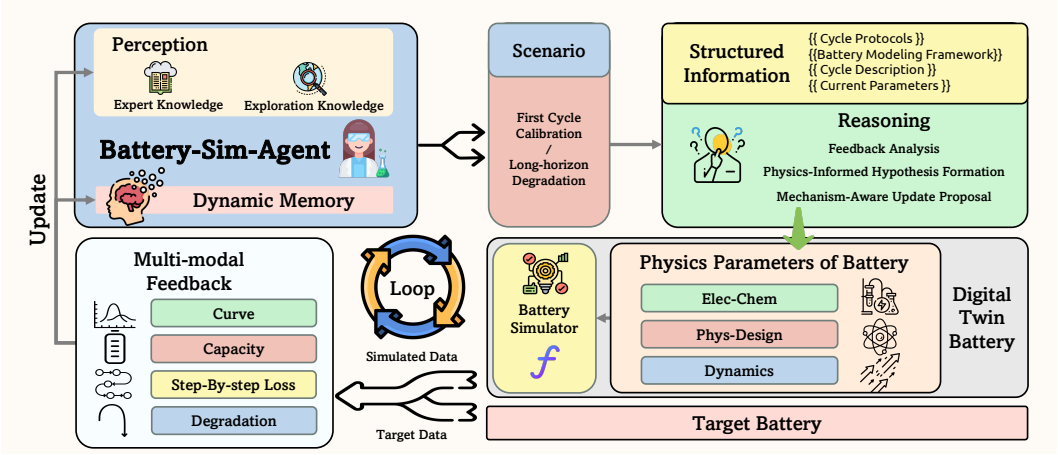


Figure 1: The closed-loop workflow of BATTERY-SIM-AGENT. The agent proposes parameters for the PYBAMM simulator. The simulator's output is then compared against target data to generate structured, multi-modal feedback (Sec. 3.2), which the agent analyzes using its dynamic memory (Sec. 3.3) to reason about the next parameter update.

aggregate multiple objectives into a single scalar, our agent operates on a disaggregated set of objectives. The target is not a single value, but a set of discrepancies across various physical quantities:

$$\mathcal{L}(\theta) = \{d_V(V_{\text{sim}}, V_{\text{obs}}), d_Q(Q_{\text{sim}}, Q_{\text{obs}}), \dots\}. \tag{2}$$

The regularization $R(\theta)$ is also enforced implicitly by the agent's reasoning, guided by the physical priors stored in its memory $\mathcal{M}_t$. The agent-driven framework bypasses manual loss weighting by receiving a structured feedback signal F_t containing the individual discrepancy components and proposing an update $\Delta\theta_t = \Phi_{\text{LLM}}(F_t, \mathcal{M}_t)$ to jointly improve the objectives. The agent function Φ_{LLM} is realized by querying a Large Language Model with a structured prompt that encapsulates the feedback F_t and relevant knowledge from memory $\mathcal{M}_t$. The iterative update rule is:

$$\theta_{t+1} = \Pi_{[\ell,u]}(\theta_t + \eta_t \Delta\theta_t), \tag{3}$$

where Π is a projection to enforce physical bounds and η_t is an adaptive step size.

3.2 THE HYPOTHESIS-DRIVEN REASONING LOOP

The agent's workflow mimics a human scientist, proceeding in three steps within each iteration.

Step 1: Analyze Feedback. The agent receives a multi-modal feedback package F_t in a structured JSON format. This contains not just overall error metrics, but also fine-grained, feature-space residuals that a human expert would examine:

```
{
  "residuals": { "capacity_mape": 0.08, "voltage_rmse": 0.05 },
  "features": { "cc_charge_time_mismatch_s": -120.5, "plateau_shift_v": -0.02 },
  "visual": "path/to/voltage_curve_overlay.png",
  "events": ["simulation_success"]
}
```

Step 2: Reason and Hypothesize. Guided by its memory $\mathcal{M}_t$, the agent analyzes this rich feedback to form a causal hypothesis. The prompt encourages a scientific reasoning process:

```
"Given the feedback, especially the short CC charge time and the low
 voltage plateau, what is the most likely physical cause? Formulate a
 hypothesis and decide on a corrective strategy."
```

Step 3: Propose a Structured Update. Finally, the agent is prompted to translate its hypothesis into a concrete, machine-actionable update, which it returns in a strict JSON format, ensuring reliability and interpretability:

```
"Based on your hypothesis, propose a targeted parameter update:"
{
  "updated_params": { "Positive electrode reaction rate [s^-1]": "*1.2" },
  "rationale": "Increasing the positive reaction rate by 20% should
               raise the voltage plateau and extend the CC charge time."
}
```

3.3 Dynamic Memory with Knowledge Warm-up

The agent's ability to reason effectively relies on its memory, $\mathcal{M}_t$, which dynamically incorporates both expert knowledge and empirical findings.

Initial Knowledge Injection. We initialize the memory $\mathcal{M}_0$ with human expert knowledge from the literature and our own domain expertise. This includes fundamental parameter information (e.g., physical bounds) and a set of fuzzy, qualitative rules-of-thumb.

Trial-and-Error Warm-up Phase. Before the main optimization loop, the agent undergoes a "warm-up" phase to build a preliminary causal model of parameter effects. It generates random perturbations around θ_0 and executes simulations. The resulting feedback is *not* for optimization, but is processed by the LLM to enrich its memory. The agent is prompted to summarize the outcomes into learned sensitivity rules (e.g., "Observed: perturbing 'Negative electrode thickness' by +10% strongly increases capacity but causes simulation failure at high C-rates"). This learned knowledge makes the subsequent optimization search significantly more targeted and robust.

3.4 Instantiated Pipelines for Key Scientific Scenarios

The following two pipelines showcase the flexibility of our framework in tackling both a short-horizon, high-fidelity matching task and a long-horizon, dynamic tracking task.

First-Cycle Calibration. This scenario focuses on matching the detailed voltage curve of the initial cycles. It relies heavily on multi-modal feedback and the agent's ability to perform protocol-aware staged matching. For a standard CC-CV protocol, the agent is prompted to analyze the CC and CV phases separately, attributing mismatches to different physical phenomena (e.g., kinetics vs. transport limitations), a nuanced strategy that is difficult to encode in a simple loss function.

Long-Horizon Degradation Fitting. This scenario aims to capture capacity fade over hundreds of cycles by fitting SEI-related degradation parameters. To handle the vast amount of data, we employ a **dynamic cycle indexing** mechanism. Instead of analyzing all cycles, the agent is shown the full degradation curve and is prompted to select a small, informative subset of cycle indices (e.g., start, end, points of maximum curvature) for detailed feedback. This ensures the feedback is both compact and highly relevant for capturing the long-term degradation dynamics.

4 Related Work

Our work is positioned at the intersection of battery science and the emerging field of AI-driven scientific discovery. The inverse problem of identifying microscopic parameters for high-fidelity electrochemical models, such as the Doyle-Fuller-Newman (DFN) model implemented in simulators like PyBaMM, is a long-standing challenge in battery engineering (Sulzer et al., 2021; Subramanian & Braatz, 2013). The problem is notoriously ill-posed, with many parameter combinations yielding similar macroscopic outputs (Gopinath et al., 2016; Prasad et al., 2015). Historically, this challenge has been addressed using classical black-box optimization (BBO) methods, such as Bayesian optimization or evolutionary algorithms (Wang & Jiang, 2023; Jiang et al., 2022; Zhang et al., 2014). While versatile, these methods are fundamentally "blind" optimizers; they treat the

Algorithm 1 The Two-Phase Workflow of BATTERY-SIM-AGENT

Input: Target data Y^{obs}, parameter bounds $[\ell, u]$, budget T, warm-up steps N_w
Initialize memory $\mathcal{M}_0$ with human knowledge
// **Phase 1: Trial-and-Error Warm-up**
for $k = 1$ to N_w **do**
    Generate a random perturbation δ_k around θ_0
    $Y^{\text{sim}} \leftarrow \text{SIMULATE}(\theta_0 + \delta_k)$
    $F_k \leftarrow \text{BUILDFEEDBACK}(Y^{\text{sim}}, Y^{\text{obs}})$
    $\mathcal{M}_k \leftarrow \text{UPDATEMEMORY}(\mathcal{M}_{k-1}, F_k, \text{“Summarize sensitivity”})$
end for
// **Phase 2: Main Optimization Loop**
for $t = 0$ to $T - 1$ **do**
    $Y^{\text{sim}} \leftarrow \text{SIMULATE}(\theta_t)$
    $F_t \leftarrow \text{BUILDFEEDBACK}(Y^{\text{sim}}, Y^{\text{obs}})$
    $\Delta\theta_t, \text{rationale}_t \leftarrow \text{QUERYLLM}(F_t, \mathcal{M}_{N_w+t-1})$
    $\theta_{t+1} \leftarrow \Pi_{[\ell,u]}\big(\theta_t + \eta_t \, \Delta\theta_t\big)$
    $\mathcal{M}_{N_w+t} \leftarrow \text{UPDATEMEMORY}(\mathcal{M}_{N_w+t-1}, F_t, \Delta\theta_t, \text{rationale}_t)$
    if converged **then**
        break
    end if
end for
return $\theta_{t^\star}$

simulator as an opaque oracle and lack physical intuition, often resulting in high sample complexity and convergence to implausible solutions.

Concurrently, a paradigm shift is underway in how AI is applied to science, moving from data analysis to autonomous discovery. Large Language Models (LLMs) are increasingly used as “cognitive partners” for tasks like hypothesis generation and literature synthesis (Zuo et al., 2025; Hu et al., 2025). More powerfully, they are being deployed as the core reasoning engine in autonomous agents that can interact with external tools in a closed loop, a trend often referred to as “agentic science” (Wei et al., 2025). This agent-based approach has already shown significant promise in solving complex parameter tuning and design problems in diverse scientific and engineering domains, such as materials science (Wu et al., 2025), solid mechanics (Ni & Buehler, 2024), astrophysics (Sun et al., 2024), and hyperparameter optimization (Liu et al., 2024). These works demonstrate the potential of LLM-agents to navigate complex search spaces more intelligently than traditional algorithms. Our work is the first to bridge these two domains. We introduce an LLM-agent as a reasoning-based optimizer to specifically tackle the challenging inverse problem in battery science.

5 EXPERIMENTS

We conduct comprehensive experiments on simulated benchmarks and real-world data to evaluate BATTERY-SIM-AGENT. Our evaluation demonstrates the superiority of the reasoning-based approach across diverse battery chemistries, operating conditions, and difficulty levels.

5.1 EXPERIMENTAL SETUP

Benchmark Test Suite Design. We construct a diverse benchmark suite using the high-fidelity Doyle-Fuller-Newman (DFN) model (Doyle et al., 1993) in PYBAMM (Sulzer et al., 2021). Our systematic design covers three key axes of variation to ensure comprehensive evaluation:

Diverse Chemistries: We employ five classic, well-established parameter sets from the literature: Chen2020 (Chen et al., 2020) (NMC811/graphite), ORegan2022 (O’Regan et al., 2022) (NMC532/graphite), Prada2013 (Prada et al., 2013) (LFP/graphite), Ecker2015 (Ecker et al., 2015b;a) (NMC111/graphite), and Marquis2019 (Marquis et al., 2019) (NMC622/graphite). These represent diverse battery chemistries with varying internal properties, electrode materials, and electrochemical behaviors.

Varied Operating Conditions: For each chemistry, we generate ground-truth data under three different charge/discharge protocols (0.2C, 1C, and 2C), simulating a range of operational severities from gentle to aggressive cycling conditions.

Two Difficulty Modes: We create two distinct test scenarios with different perturbation strategies:

- **Regular Mode:** We apply 12 expert-designed, physically-plausible multi-parameter perturbations that represent realistic manufacturing variations or design choices. These combinations are carefully crafted to maintain physical plausibility while creating meaningful optimization challenges.
- **Extreme Mode:** We apply large perturbations to one of 9 key parameters (particle radii, electrode thicknesses, porosities, Bruggeman coefficients, separator thickness), creating challenging cases that often push the simulator to its stability limits.

Data Generation Process. Our systematic data generation follows a rigorous multi-stage process. We iterate through all combinations of base parameter sets, C-rates, and perturbation rules, then apply a two-stage filtering process: (1) We discard parameter combinations that result in simulation failures in PyBaMM, ensuring numerical stability; (2) We filter out cases where the resulting capacity change is less than 1% compared to baseline, ensuring each test case presents a meaningful, non-trivial challenge. This process results in 233 valid combinations for extreme mode and 373 for regular mode, from which we randomly sample 100 cases each to form our final evaluation suite of 200 unique tasks. Detailed generation rules and examples are provided in Appendix B.

Baselines and Comparison Strategy. We compare our full agent against strong baselines and an ablation to isolate the benefits of different components:

- **Battery-Sim-Agent-O3:** The full agent powered by GPT-O3 (OpenAI, 2025), incorporating our complete reasoning workflow with hypothesis generation, iterative refinement, and multi-objective optimization capabilities.
- **Battery-Sim-Agent-OSS:** An ablation using GPT-OSS (OpenAI et al., 2025), a powerful 120B parameter open-source model, but without the chain-of-thought reasoning capabilities of our full agent. This isolates the benefit of the reasoning workflow itself.
- **Bayesian Optimization (BO):** We use standard Bayesian Optimization implemented by Meta's Ax platform (Olson et al., 2025), representing state-of-the-art black-box optimization methods commonly used in parameter estimation.

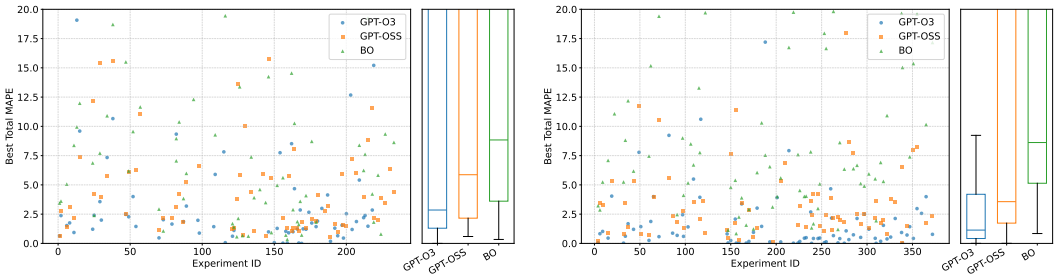
We also experimented with other evolutionary algorithms including CMA-ES (Hansen et al., 2019), but found that these methods generally failed to converge on our challenging parameter estimation tasks. We also present results of **Default Parameters**, which includes the original parameter values from each literature source as a naive baseline, representing the performance when using published parameters without optimization.

Evaluation Metrics. We evaluate performance using comprehensive error metrics between predicted and ground-truth voltage/capacity curves: Mean Absolute Percentage Error (MAPE) and Root Mean Squared Error (RMSE). These metrics capture both relative and absolute deviations, providing a thorough assessment of parameter identification accuracy.

5.2 Results on First-Cycle Calibration

Figure 2 and Table 2 present our comprehensive results for first-cycle calibration. The findings clearly demonstrate the superiority of our reasoning-based approach across all evaluation scenarios. Specifically, **Battery-Sim-Agent-O3** consistently and significantly outperforms all other methods across both regular and extreme modes. As shown in Fig. 2, our agent achieves not only substantially lower median error but also dramatically reduced variance, indicating more reliable and stable performance. The ablation (OSS) performs better than BO methods but is clearly inferior to our full agent, confirming that the agent's explicit reasoning capabilities are critical to its success.

The quantitative results in Table 2 reveal the magnitude of our improvements. In regular mode, our agent achieves MAPE reductions of 67-95% compared to BO across different chemistries, with



(a) Regular Mode (b) Extreme Mode

Figure 2: **Main results on first-cycle calibration.** Our reasoning-based agent (GPT-O3) consistently outperforms its ablation (GPT-OSS) and Bayesian Optimization across both difficulty modes, achieving lower median error and significantly reduced variance.

Table 2: Detailed MAPE and RMSE results for first-cycle calibration across modes and chemistries.

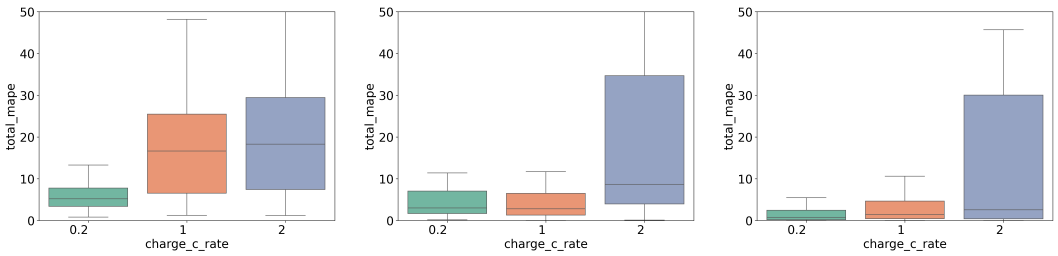
Mode	Methods	Chen2020	ORegan2022	Ecker2015	Prada2013	Marquis2019
		MAPE				
Regular	Default	159.09±118.6	160.47±119.0	108.42±65.1	79.16±56.1	122.60±79.2
	BO	211.97±404.4	81.73±224.0	27.37±80.3	56.31±222.2	13.54±9.3
	BatterySimAgent-OSS	38.05±100.0	46.34±53.3	7.63±17.3	23.74±44.7	9.55±20.3
	BatterySimAgent-O3	**12.60±24.5**	**34.18±48.2**	**0.77±1.2**	**5.97±12.2**	**1.27±1.1**
Extreme	Default	119.32±110.4	181.95±181.2	100.43±115.3	150.65±87.9	108.00±89.7
	BO	159.41±362.4	84.55±213.7	137.31±278.6	**17.05±25.3**	**8.42±6.3**
	BatterySimAgent-OSS	23.66±42.2	50.88±61.5	45.47±92.2	23.96±42.3	19.50±39.1
	BatterySimAgent-O3	**23.38±52.4**	**19.44±24.2**	**27.85±79.5**	59.14±62.3	48.34±90.4
		RMSE				
Regular	Default	5.47±2.7	6.47±3.0	1.30±0.4	2.68±1.5	1.64±0.5
	BO	2.50±3.4	**2.27±1.7**	0.21±0.1	0.57±0.4	0.26±0.1
	BatterySimAgent-OSS	1.87±2.5	3.07±2.6	0.26±0.2	1.22±1.5	0.43±0.4
	BatterySimAgent-O3	**1.18±1.9**	2.41±2.4	**0.06±0.1**	**0.32±0.4**	**0.19±0.1**
Extreme	Default	4.23±2.3	6.11±3.0	1.74±1.2	5.21±2.8	1.48±0.8
	BO	1.63±2.9	2.10±2.1	0.77±2.5	**0.60±0.5**	**0.26±0.2**
	BatterySimAgent-OSS	1.91±2.2	3.04±2.8	0.69±1.0	1.23±1.3	0.47±0.5
	BatterySimAgent-O3	**1.48±2.6**	**1.53±1.5**	**0.43±0.8**	2.50±2.3	0.59±0.8

particularly impressive performance on Ecker2015 (0.77% vs 27.37% MAPE) and Marquis2019 (1.27% vs 13.54% MAPE). The ablation study demonstrates that while GPT-OSS provides some benefit over traditional optimization, our full reasoning workflow delivers substantial additional improvements. In **extreme mode**, where single parameters are dramatically perturbed, the performance of baseline optimizers degrades significantly due to the highly non-convex optimization landscape. In contrast, our agent's reasoning capabilities allow it to maintain robust performance by systematically exploring the parameter space and adapting its search strategy based on intermediate results.

C-rate Performance Analysis. Figure 3 shows performance across different charge/discharge protocols. Our agent maintains superior performance across all C-rates, with particularly notable improvements at higher rates where traditional optimization methods struggle with the increased complexity of the electrochemical dynamics.

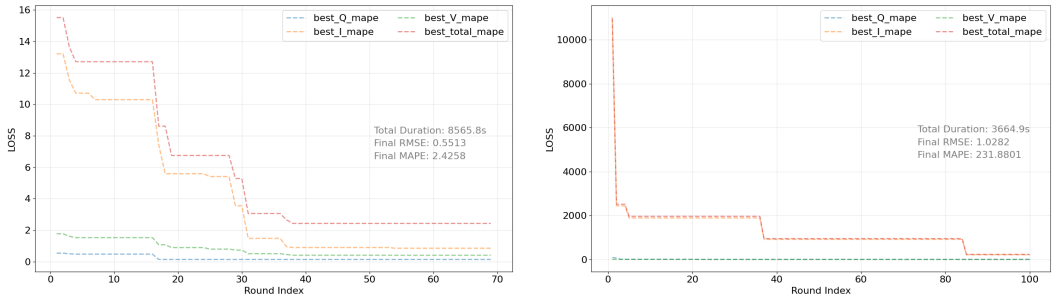
5.3 Advanced Applications

Long-Horizon Degradation Fitting. We extend our evaluation to degradation scenarios requiring simultaneous fitting of electrochemical and SEI parameters, representing a significantly more challenging optimization problem. Table 3 demonstrates that Battery-Sim-Agent framework successfully handles this complex task across both model variants. Interestingly, BatterySimAgent-OSS achieves superior performance in degradation fitting (1.37% vs 1.77% Total MAPE), suggesting that the reasoning complexity should match task characteristics, for smooth, long-horizon degradation trends, OSS's more direct optimization approach proves more effective than O3's sophisticated rea-



(a) BO (b) Battery-Sim-Agent-OSS (c) Battery-Sim-Agent-O3

Figure 3: **Performance across C-rates.** Comparison of different methods across various charge/discharge protocols. Each subplot shows MAPE distribution for different C-rate protocols.



(a) Degradation Fitting (b) Real-World Data

Figure 4: **Convergence analysis.** Evolution of error metrics over optimization iterations for GPT-O3 on degradation fitting (left) and real-world battery data (right), demonstrating systematic convergence in complex scenarios where traditional methods fail.

soning. Both agent variants substantially outperform traditional methods, as Bayesian Optimization fails to converge on this challenging task due to the high-dimensional parameter space and complex objective landscape, highlighting the fundamental advantage of reasoning-based approaches over blind optimization in complex battery parameter estimation scenarios.

Table 3: Performance on long-horizon degradation fitting and real-world battery tasks. BO failed to converge and is *excluded* from comparison.

Method	**Degradation**				**Real Battery**			
	Total MAPE	**Q_MAPE**	**I_MAPE**	**V_MAPE**	**Total MAPE**	**Q_MAPE**	**I_MAPE**	**V_MAPE**
BatterySimAgent-OSS	1.3674	0.5148	0.6595	0.1931	8.7489	0.9027	6.5803	1.2659
BatterySimAgent-O3 (Ours)	1.7705	0.6711	0.8501	0.2494	**3.4591**	**0.6020**	**1.8136**	**1.0436**

Real-World Validation. We validate BATTERY-SIM-AGENT on 7 real battery tasks, using data from the CALCE(He et al., 2011; Xing et al., 2013) dataset obtained from public repositories (Zhang et al., 2024), demonstrating practical applicability. Figure 4 shows convergence behavior for both degradation fitting and real-world data, revealing robust optimization even with noisy experimental data and unknown ground-truth parameters.

6 CONCLUSION

We introduced BATTERY-SIM-AGENT, a novel framework that reframes the challenging inverse problem of parameterizing battery digital twins as a reasoning task. By deploying an LLM-agent in a closed loop with a high-fidelity simulator, we demonstrated a new paradigm for scientific optimization that mimics human expert workflows. Our comprehensive experiments showed that this reasoning-based approach systematically outperforms traditional black-box optimizers on a diverse suite of simulated benchmarks.

REFERENCES

Robert S. Balog and Ali Davoudi. Batteries, Battery Management , and Battery Charging Technology. In *Transportation Technologies for Sustainability*, pp. 122–157. Springer, New York, NY, 2013. ISBN 978-1-4614-5844-9. doi: 10.1007/978-1-4614-5844-9_822. URL https://link.springer.com/rwe/10.1007/978-1-4614-5844-9_822.

S Blaifi, S Moulahoum, I Colak, and W Merrouche. An enhanced dynamic model of battery using genetic algorithm suitable for photovoltaic applications. *Applied Energy*, 169:888–898, 2016.

Chang-Hui Chen, Ferran Brosa Planella, Kieran O'Regan, Dominika Gastol, W. Dhammika Widanage, and Emma Kendrick. Development of experimental techniques for parameterization of multi-scale lithium-ion battery models. *Journal of The Electrochemical Society*, 167(8):080534, may 2020. doi: 10.1149/1945-7111/ab9050. URL https://dx.doi.org/10.1149/1945-7111/ab9050.

Marc Doyle, Thomas F Fuller, and John Newman. Modeling of galvanostatic charge and discharge of the lithium/polymer/insertion cell. *Journal of the Electrochemical society*, 140(6):1526, 1993.

Madeleine Ecker, Stefan Käbitz, Izaro Laresgoiti, and Dirk Uwe Sauer. Parameterization of a Physico-Chemical Model of a Lithium-Ion Battery: II. Model Validation. *Journal of The Electrochemical Society*, 162(9):A1849, June 2015a. ISSN 1945-7111. doi: 10.1149/2.0541509jes.

Madeleine Ecker, Thi Kim Dung Tran, Philipp Dechent, Stefan Käbitz, Alexander Warnecke, and Dirk Uwe Sauer. Parameterization of a Physico-Chemical Model of a Lithium-Ion Battery: I. Determination of Parameters. *Journal of The Electrochemical Society*, 162(9):A1836, June 2015b. ISSN 1945-7111. doi: 10.1149/2.0551509jes.

R. Gopinath, S. Santhanagopalan, and Richard D. Braatz. An inverse method for estimating the electrochemical parameters of lithium-ion batteries. *Journal of The Electrochemical Society*, 163 (14):A3045–A3054, 2016.

Nikolaus Hansen, Youhei Akimoto, and Petr Baudis. CMA-ES/pycma on Github. Zenodo, DOI:10.5281/zenodo.2559634, February 2019. URL https://doi.org/10.5281/zenodo.2559634.

Wei He, Nicholas Williard, Michael Osterman, and Michael Pecht. Prognostics of lithium-ion batteries based on Dempster–Shafer theory and the Bayesian Monte Carlo method. *Journal of Power Sources*, 196(23):10314–10321, December 2011. ISSN 0378-7753. doi: 10.1016/j.jpowsour.2011.08.040.

Ming Hu, Chenglong Ma, Wei Li, Wanghan Xu, Jiamin Wu, Jucheng Hu, Tianbin Li, Guohang Zhuang, Jiaqi Liu, Yingzhou Lu, Ying Chen, Chaoyang Zhang, Cheng Tan, Jie Ying, Guocheng Wu, Shujian Gao, Pengcheng Chen, Jiashi Lin, Haitao Wu, Lulu Chen, Fengxiang Wang, Yuanyuan Zhang, Xiangyu Zhao, Feilong Tang, Encheng Su, Junzhi Ning, Xinyao Liu, Ye Du, Changkai Ji, Cheng Tang, Huihui Xu, Ziyang Chen, Ziyan Huang, Jiyao Liu, Pengfei Jiang, Yizhou Wang, Chen Tang, Jianyu Wu, Yuchen Ren, Siyuan Yan, Zhonghua Wang, Zhongxing Xu, Shiyan Su, Shangquan Sun, Runkai Zhao, Zhisheng Zhang, Yu Liu, Fudi Wang, Yuanfeng Ji, Yanzhou Su, Hongming Shan, Chunmei Feng, Jiahao Xu, Jiangtao Yan, Wenhao Tang, Diping Song, Lihao Liu, Yanyan Huang, Lequan Yu, Bin Fu, Shujun Wang, Xiaomeng Li, Xiaowei Hu, Yun Gu, Ben Fei, Zhongying Deng, Benyou Wang, Yuewen Cao, Minjie Shen, Haodong Duan, Jie Xu, Yirong Chen, Fang Yan, Hongxia Hao, Jielan Li, Jiajun Du, Yanbo Wang, Imran Razzak, Chi Zhang, Lijun Wu, Conghui He, Zhaohui Lu, Jinhai Huang, Yihao Liu, Fenghua Ling, Yuqiang Li, Aoran Wang, Qihao Zheng, Nanqing Dong, Tianfan Fu, Dongzhan Zhou, Yan Lu, Wenlong Zhang, Jin Ye, Jianfei Cai, Wanli Ouyang, Yu Qiao, Zongyuan Ge, Shixiang Tang, Junjun He, Chunfeng Song, Lei Bai, and Bowen Zhou. A survey of scientific large language models: From data foundations to agent frontiers, 2025. URL https://arxiv.org/abs/2508.21148.

Benben Jiang, Marc D Berliner, Kun Lai, Patrick A Asinger, Hongbo Zhao, Patrick K Herring, Martin Z Bazant, and Richard D Braatz. Fast charging design for lithium-ion batteries via bayesian optimization. *Applied Energy*, 307:118244, 2022.

Siyi Liu, Chen Gao, and Yong Li. Large language model agent for hyper-parameter optimization. *arXiv preprint arXiv:2402.01881*, 2024.

Dirk Magnor and Dirk Uwe Sauer. Optimization of pv battery systems using genetic algorithms. *Energy Procedia*, 99:332–340, 2016.

Scott G. Marquis, Valentin Sulzer, Robert Timms, Colin P. Please, and S. Jon Chapman. An Asymptotic Derivation of a Single Particle Model with Electrolyte. *Journal of The Electrochemical Society*, 166(15):A3693, November 2019. ISSN 1945-7111. doi: 10.1149/2.0341915jes.

Bo Ni and Markus J Buehler. Mechagents: Large language model multi-agent collaborations can solve mechanics problems, generate new data, and integrate knowledge. *Extreme Mechanics Letters*, 67:102131, 2024.

Miles Olson, Elizabeth Santorella, Louis C. Tiao, Sait Cakmak, David Eriksson, Mia Garrard, Sam Daulton, Maximilian Balandat, Eytan Bakshy, Elena Kashtelyan, Zhiyuan Jerry Lin, Sebastian Ament, Bernard Beckerman, Eric Onofrey, Paschal Igusti, Cristian Lara, Benjamin Letham, Cesar Cardoso, Shiyun Sunny Shen, Andy Chenyuan Lin, and Matthew Grange. Ax: A Platform for Adaptive Experimentation. In *AutoML 2025 ABCD Track*, 2025.

OpenAI. Openai o3 and o4-mini system card, April 2025. URL https://cdn.openai.com/pdf/2221c875-02dc-4789-800b-e7758f3722c1/o3-and-o4-mini-system-card.pdf.

OpenAI, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Haiming Bao, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sebastien Bubeck, Che Chang, Kai Chen, Mark Chen, Enoch Cheung, Aidan Clark, Dan Cook, Marat Dukhan, Casey Dvorak, Kevin Fives, Vlad Fomenko, Timur Garipov, Kristian Georgiev, Mia Glaese, Tarun Gogineni, Adam Goucher, Lukas Gross, Katia Gil Guzman, John Hallman, Jackie Hehir, Johannes Heidecke, Alec Helyar, Haitang Hu, Romain Huet, Jacob Huh, Saachi Jain, Zach Johnson, Chris Koch, Irina Kofman, Dominik Kundel, Jason Kwon, Volodymyr Kyrylov, Elaine Ya Le, Guillaume Leclerc, James Park Lennon, Scott Lessans, Mario Lezcano-Casado, Yuanzhi Li, Zhuohan Li, Ji Lin, Jordan Liss, Lily, Liu, Jiancheng Liu, Kevin Lu, Chris Lu, Zoran Martinovic, Lindsay McCallum, Josh McGrath, Scott McKinney, Aidan McLaughlin, Song Mei, Steve Mostovoy, Tong Mu, Gideon Myles, Alexander Neitz, Alex Nichol, Jakub Pachocki, Alex Paino, Dana Palmie, Ashley Pantuliano, Giambattista Parascandolo, Jongsoo Park, Leher Pathak, Carolina Paz, Ludovic Peran, Dmitry Pimenov, Michelle Pokrass, Elizabeth Proehl, Huida Qiu, Gaby Raila, Filippo Raso, Hongyu Ren, Kimmy Richardson, David Robinson, Bob Rotsted, Hadi Salman, Suvansh Sanjeev, Max Schwarzer, D. Sculley, Harshit Sikchi, Kendal Simon, Karan Singhal, Yang Song, Dane Stuckey, Zhiqing Sun, Philippe Tillet, Sam Toizer, Foivos Tsimpourlas, Nikhil Vyas, Eric Wallace, Xin Wang, Miles Wang, Olivia Watkins, Kevin Weil, Amy Wendling, Kevin Whinnery, Cedric Whitney, Hannah Wong, Lin Yang, Yu Yang, Michihiro Yasunaga, Kristen Ying, Wojciech Zaremba, Wenting Zhan, Cyril Zhang, Brian Zhang, Eddie Zhang, and Shengjia Zhao. Gpt-oss-120b & gpt-oss-20b Model Card, August 2025.

Kieran O'Regan, Ferran Brosa Planella, W. Dhammika Widanage, and Emma Kendrick. Thermal-electrochemical parameters of a high energy lithium-ion cylindrical battery. *Electrochimica Acta*, 425:140700, 2022. ISSN 0013-4686. doi: 10.1016/j.electacta.2022.140700.

Wendy Pantoja, Jaime Andres Perez-Taborda, and Alba Avila. Tug-of-War in the Selection of Materials for Battery Technologies. *Batteries*, 8(9):105, September 2022. ISSN 2313-0105. doi: 10.3390/batteries8090105.

E. Prada, D. Di Domenico, Y. Creff, J. Bernard, V. Sauvant-Moynot, and F. Huet. A Simplified Electrochemical and Thermal Aging Model of LiFePO4-Graphite Li-ion Batteries: Power and Capacity Fade Simulations. *Journal of The Electrochemical Society*, 160(4):A616, February 2013. ISSN 1945-7111. doi: 10.1149/2.053304jes.

K. Prasad, A. Rahimian, and M. Fowler. Inverse parameter determination in the development of an optimized lithium iron phosphate–graphite battery discharge model. *Journal of Power Sources*, 273:1348–1359, 2015.

Venkat R. Subramanian and Richard D. Braatz. Modeling and simulation of lithium-ion batteries from a systems engineering perspective. *Journal of The Electrochemical Society*, 160(4):R93–R108, 2013.

Valentin Sulzer, Scott G. Marquis, Robert Timms, Martin Robinson, and S. Jon Chapman. PyBaMM: Python battery mathematical modelling. *Journal of Open Research Software*, 9(1):14, 2021.

Zechang Sun, Yuan-Sen Ting, Yaobo Liang, Nan Duan, Song Huang, and Zheng Cai. Interpreting multi-band galaxy observations with large language model-based agents. *arXiv preprint arXiv:2409.14807*, 2024.

Xizhe Wang and Benben Jiang. Multi-objective optimization for fast charging design of lithium-ion batteries using constrained bayesian optimization. *Journal of Power Sources*, 584:233602, 2023.

Jiaqi Wei, Yuejin Yang, Xiang Zhang, Yuhan Chen, Xiang Zhuang, Zhangyang Gao, Dongzhan Zhou, Guangshuai Wang, Zhiqiang Gao, Juntai Cao, et al. From ai for science to agentic science: A survey on autonomous scientific discovery. *arXiv preprint arXiv:2508.14111*, 2025.

Mengsong Wu, YaFei Wang, Yidong Ming, Yuqi An, Yuwei Wan, Wenliang Chen, Binbin Lin, Yuqiang Li, Tong Xie, and Dongzhan Zhou. Chemagent: Enhancing llms for chemistry and materials science through tree-search based tool learning. *arXiv preprint arXiv:2506.07551*, 2025.

Yinjiao Xing, Eden W. M. Ma, Kwok-Leung Tsui, and Michael Pecht. An ensemble model for predicting the remaining useful performance of lithium-ion batteries. *Microelectronics Reliability*, 53(6):811–820, June 2013. ISSN 0026-2714. doi: 10.1016/j.microrel.2012.12.003.

Han Zhang, Xiaofan Gui, Shun Zheng, Ziheng Lu, Yuqi Li, and Jiang Bian. BatteryML: An open-source platform for machine learning on battery degradation. In *The Twelfth International Conference on Learning Representations*, 2024.

Liqiang Zhang, Lixin Wang, Gareth Hinds, Chao Lyu, Jun Zheng, and Junfu Li. Multi-objective optimization of lithium-ion battery model using genetic algorithm approach. *Journal of Power Sources*, 270:367–378, 2014.

Wenhua Zuo, Huihuo Zheng, Tanjin He, Venkatram Vishwanath, Maria KY Chan, Rick L Stevens, Khalil Amine, and Gui-Liang Xu. Large language models for batteries. *Joule*, 9(8), 2025.

A USE OF LARGE LANGUAGE MODELS (LLMS)

In preparing this manuscript, we employed a Large Language Model (LLM) as a general-purpose writing assistant. Specifically, the LLM was used to polish the language, improve clarity and flow, and enhance the presentation of the text. All technical content, experimental design, data analysis, and model development were performed independently by the authors. The LLM was not used to generate any novel scientific ideas, experimental results, or interpretations.

B BENCHMARK GENERATION DETAILS

This appendix provides the detailed rules for generating the simulated benchmark suite used in our experiments, as described in Section 5.1. All ground-truth data is generated using the Doyle-Fuller-Newman (DFN) model in PYBAMM with the SEI model enabled ("reaction limited").

B.1 SINGLE-PARAMETER VARIATIONS (EXTREME MODE)

In this mode, we start from one of the five base parameter sets (Chen2020, ORegan2022, etc.) and perturb a single critical parameter. This is designed to create challenging, often physically extreme scenarios. The nine parameters and their perturbation rules are listed in Table 4.

Table 4: Parameter perturbation rules for the *extreme mode* benchmark.

Parameter Name	**Perturbation Rule**
Negative particle radius [m]	base $\times$ 0.5, base $\times$ 2.0
Positive particle radius [m]	base $\times$ 0.5, base $\times$ 2.0
Negative electrode thickness [m]	base $\times$ 0.75, base $\times$ 1.5
Positive electrode thickness [m]	base $\times$ 0.75, base $\times$ 1.5
Negative electrode porosity	base - 0.05, base + 0.05
Positive electrode porosity	base - 0.05, base + 0.05
Negative electrode Bruggeman coefficient	1.5, 2.0, 2.5
Positive electrode Bruggeman coefficient	1.3, 1.8, 2.3
Separator thickness [m]	base $\times$ 0.7, base $\times$ 1.3

B.2 MULTI-PARAMETER COMBINATIONS (REGULAR MODE)

In this mode, we apply expert-designed combinations of perturbations to multiple parameters simultaneously. These combinations are designed to be physically plausible and represent realistic manufacturing variations or design choices. The twelve predefined combinations are detailed in Table 5.

Table 5: Predefined multi-parameter combinations for the *regular mode* benchmark.

ID	**Description and Parameter Overrides**
1	**Max-power, manufacturing-plausible:** Neg./Pos. particle radius $\times 0.7$, Neg./Pos. electrode thickness $\times 0.85/0.9$, etc.
2	**Energy-leaning but realistic:** Neg./Pos. electrode thickness $\times 1.10/1.25$ (maintaining N/P ratio), porosity $-0.02/-0.03$.
3	**Electrolyte-limited cathode:** Pos. electrode thickness $\times 1.25$, Pos. porosity -0.05, Pos. Bruggeman coeff. to 2.0.
4	**Solid-diffusion-limited (both electrodes):** Neg./Pos. particle radius $\times 1.5$.
5	**Anode-biased diffusion limit:** Neg. particle radius $\times 1.8$, Neg. electrode thickness $\times 1.15$.
6	**Cathode-biased diffusion limit:** Pos. particle radius $\times 1.8$, Pos. electrode thickness $\times 1.15$.
7	**High-ε / low-tortuosity (ionic-friendly):** Neg./Pos. porosity $+0.06$, Neg./Pos. Bruggeman coeff. to 1.5.
8	**Low-ε / high-tortuosity (ionic bottleneck):** Neg./Pos. porosity -0.06, Neg./Pos. Bruggeman coeff. to 2.0.
9	**Asymmetric particles (fast anode / slow cathode):** Neg. radius $\times 0.7$, Pos. radius $\times 1.4$.
10	**Asymmetric particles (slow anode / fast cathode):** Neg. radius $\times 1.4$, Pos. radius $\times 0.7$.
11	**Thin separator + thick electrodes:** Separator thickness $\times 0.85$, Neg./Pos. electrode thickness $\times 1.20/1.25$.
12	**Thick separator + low-ε (ionic choke):** Separator thickness $\times 1.5$, Neg./Pos. porosity -0.04.

B.3 Final Selection Process

For both modes, we iterate through all combinations of base parameter sets, C-rates, and perturbation rules. We then apply a two-stage filtering process to curate the final test set:

1. We first discard any parameter combination that results in a simulation failure in PyBaMM.
2. We then filter out cases where the resulting change in discharge capacity is less than 1% compared to the baseline, ensuring that each test case presents a meaningful, non-trivial challenge.

Finally, from the remaining pool of valid and meaningful cases (233 for extreme mode, 373 for regular mode), we randomly sample 100 cases for each mode to form our final evaluation suite of 200 distinct tasks.

C Additional Experiment S

C.1 Bayesian Optimization Experiment Setup

Table 6: Key Hyperparameter Settings

Parameter	Value
Random Seed	`1234`
Warmup Round	`number of parameters * 2`

C.2 Covariance matrix adaptation evolution strategy Experiment Setup

Table 7: Key Hyperparameter Settings

Parameter	Value
Random Seed	`1234`
bounds	`[x0_lower_bounds, x0_upper_bounds]`
maxiter	`generations`
popsize	`number of parameters + 1`
verb_disp	`1`
tolfun	`0`
tolx	`0`
tolfunhist	`0`
tolstagnation	`0`
tolflatfitness	`0`

D Additional Experimental Results

D.1 Detailed Degradation Experiment Setup

For the long-horizon degradation fitting experiments, we select 5 representative parameter sets from our benchmark suite and enable SEI modeling with the "reaction limited" mechanism in PyBaMM. Each simulation runs for 200 cycles to capture capacity fade behavior. The optimization task involves fitting both base electrochemical parameters and SEI degradation parameters (SEI kinetic rate constant, SEI conductivity, etc.) to match the observed capacity degradation curve.

D.2 Real-World Data Validation Details

We apply Battery-Sim-Agent-O3 to 7 real battery datasets from public repositories, including NASA and CALCE battery datasets. These datasets contain charge/discharge cycles from actual

lithium-ion batteries under various operating conditions. For each dataset, we use the first few cycles to infer battery parameters and validate against remaining cycles. The convergence analysis demonstrates robust optimization behavior even with noisy experimental data.

D.3 Additional Performance Analysis

Robustness Analysis. Our agent's advantage is particularly pronounced in challenging scenarios. In extreme mode, baseline optimizers degrade significantly while our agent remains robust. At higher C-rates (2C), where dynamics are more complex, the performance gap widens further.

Convergence Behavior. The convergence analysis reveals that our agent maintains stable optimization behavior even in challenging high-dimensional parameter spaces where traditional optimization methods struggle to converge. This is particularly evident in the degradation fitting task, where BO completely fails to converge.

E Prompt Design of Battery-Sim-Agent

First-Cycle Calibration Prompt

System prompt:

You are a battery parameter expert with extensive experience and expertise in adjusting battery parameters and are proficient in the PyBaMM simulation tool. You can adjust battery parameters based on the actual battery capacity degradation to ensure that simulation results match actual results.

User prompt:

First Round:

- I want to simulate a real battery using Pybamm, and I plan to adjust the parameters so that the current and voltage curves look consistent.
- The charge and discharge protocol I use is as follows: {{ protocols }}
- I hope you can use your existing knowledge about these parameters and summarize how adjusting these parameters will change the capacity and how the current-voltage curve will change.
- I hope you can adjust the parameters based on your knowledge and these rules, as well as the capacity and curve of the battery under the current parameters, so that the simulated curve is closer to the real one. These are the current parameters.
- You need to adjust these parameters to make the curve of the first circle close. If necessary, you can also change other parameters.
- params = {{ current_params }} And other parameter would follow {{ parameter_set }} parameter set values.
- The upper picture shows the current changing with time curve, and the lower picture shows the voltage changing with time curve. The yellow one is the real battery curve, and the blue one is the curve generated by the {{ model_name }} model with the current parameters. Please first describe the difference between the blue and yellow curves in the figure, and summarize the direction in which the parameters need to be optimized, then adjust the parameters, and return the parameters and values that need to be adjusted.
- {{ cycle_description }}
- We need to ensure that the capacity and the time of different steps (such as constant current charging) are the same between the simulated data and the real data.

User prompt:

TEXT KNOWLEDGE:
And here are some patterns that test by experiment by us:

- For Electrode Width [m], If the value is increased, the corresponding battery capacity will increase, and if the value is decreased, the corresponding battery capacity will decrease.
- For Negative Electrode Active Material Volume Fraction, If the value is increased, the corresponding battery capacity will increase, and if the value is decreased, the corresponding battery capacity will decrease.
- At the same time, decreasing the value will increase the relative proportion of the constant voltage (CV) stage in the charge stage, while the relative proportion of the constant current (CC) stage will decrease.
- Note that the Negative Electrode Active Material Volume Fraction should be larger than the Positive Electrode Active Material Volume Fraction.
- For Positive Electrode Active Material Volume Fraction, If the value is increased and decreased, the corresponding battery capacity will not change significantly.
- At the same time, decreasing the value will increase the relative proportion of the constant voltage (CV) stage in the charge stage, while the relative proportion of the constant current (CC) stage will decrease.
- For Negative Electrode Thickness [m], If the value is increased, the corresponding battery capacity will increase, and if the value is decreased, the corresponding battery capacity will decrease.
- At the same time, decreasing the value will increase the relative proportion of the constant voltage (CV) stage in the charge stage, while the relative proportion of the constant current (CC) stage will decrease significantly. Note that if the value is too large, it will cause errors.
- For Positive Electrode Thickness [m], If the value is increased and decreased, the corresponding battery capacity will not change significantly.
- At the same time, decreasing the value will decrease the relative proportion of the constant voltage (CV) stage in the charge stage, while the relative proportion of the constant current (CC) stage will not change significantly.
- For Maximum Concentration in Negative Electrode [mol.mˆ-3], If the value is increased and decreased, the corresponding battery capacity will not change significantly.
- At the same time, decreasing the value will increase the relative proportion of the constant voltage (CV) stage in the charge stage, while the relative proportion of the constant current (CC) stage will decrease. Note that the maximum concentration must be greater than the initial concentration.
- For Maximum Concentration in Positive Electrode [mol.mˆ-3], If the value is increased, the corresponding battery capacity will increase, and if the value is decreased, the corresponding battery capacity will decrease.
- At the same time, increasing the value will decrease the relative proportion of the constant voltage (CV) stage in the charge stage, while the relative proportion of the constant current (CC) stage will decrease.
- Note that the maximum concentration must be greater than the initial concentration, and slight adjustments may cause errors.
- For Initial Concentration in Negative Electrode [mol.mˆ-3] If the value is increased, the corresponding battery capacity will increase, and if the value is decreased, the corresponding battery capacity will decrease.

- At the same time, decreasing the value will decrease the relative proportion of the constant voltage (CV) stage in the charge stage, while the relative proportion of the constant current (CC) stage will decrease.
- For Initial Concentration in Positive Electrode [mol.mˆ-3], If the value is increased, the corresponding battery capacity will decrease, and if the value is decreased, the corresponding battery capacity will increase.
- At the same time, decreasing the value will decrease the relative proportion of the constant voltage (CV) stage in the charge stage, while the relative proportion of the constant current (CC) stage will decrease.

The params should just in {{ search_keys }}, I hope you can summarize the above results and suggest the next 1 updated params group with new values as dict (without name) in JSON format.
SEARCH KNOWLEDGE:
I want to explore and gather the knowledge from first 20 groups results. You can only modify some parameters in this list {{ search_keys }}. Give me a series of parameter adjustment (about 20 groups) in JSON format for me to execute using pybamm first. Please do not add any invalid comments for JSON.
OTHER ROUNDPROMPT:
Here are the results:
{{ cycle_description }}
The params should just in {{ search_keys }}, I hope you can summarize the above results and suggest the next 1 updated params group with new values as dict (without name) in JSON format.

Long-Horizon Degradation Fitting Prompt

System prompt:

You are a battery parameter expert with extensive experience and expertise in adjusting battery parameters and are proficient in the PyBaMM simulation tool. You can adjust battery parameters based on the actual battery capacity degradation to ensure that simulation results match actual results.

User prompt:

First Round:

- I want to simulate a real battery degradation using Pybamm, and I plan to adjust the SEI parameters so that the current and voltage curves of every cycle look consistent.
- The initial settings are the same, so the first cycle of real and simulated data are the same. From cycle 2, we want to adjust SEI params to keep real and simulated data look same. I will provide the corresponding cycle number {{ cycle_idxs }} and the corresponding real and simulated information.
- The charge and discharge protocol I use is as follows: {{ protocols }}
- I hope you can use your existing knowledge about these parameters {{ search_keys }} and summarize how adjusting these parameters will change the degradation capacity and how the current-voltage curve will change.
- # I hope you can adjust the parameters based on your knowledge and these rules, as well as the capacity and curve of the battery under the current parameters, so that the simulated curve is closer to the real one.
- You need to adjust these parameters to make the curve of the {{ cycle_idxs }} close.
- curent params = {{ current_params }} and other parameter would follow {{ parameter_set }} parameter set values.
- {{ cycle_description }}

- We need to ensure that the capacity and the time of different steps (such as constant current charging) are the same between the simulated data and the real data of each cycle.

User prompt:

TEXT KNOWLEDGE:
And here are some patterns that test by experiment by us:

- Higher solvent concentration (bulk_solvent_concentration_mol_m-3) accelerates side reactions like the SEI, leading to greater degradation.
- A higher lithium-to-SEI molar ratio (ratio_of_lithium_moles_to_SEI_moles) increases the active lithium consumption efficiency and accelerates capacity degradation.
- Increasing the initial EC concentration in the electrolyte (EC_initial_concentration_in_electrolyte_mol_m-3) generally results in larger initial capacity and impedance decay, with a downward-convex curve.
- A higher SEI solvent diffusivity (SEI_solvent_diffusivity_m2_s-1) increases the degradation rate and magnitude.
- A higher EC diffusivity (EC_diffusivity_m2_s-1) accelerates the degradation rate and results in a downward-convex curve.
- A higher initial SEI thickness (initial_SEI_thickness_m) slows the degradation rate and minimizes the degradation. The larger the SEI partial molar volume (SEI_partial_molar_volume_m3_mol-1), the slower and larger the degradation.

  The params should just in {{ search_keys }}, I hope you can summarize the above results and suggest the next 1 updated params group with new values as dict (without name) in JSON format.

The params should just in {{ search_keys }}, I hope you can summarize the above results and suggest the next 1 updated params group with new values as dict (without name) in JSON format.
SEARCH KNOWLEDGE:
I want to explore and gather the knowledge from first 10 groups results. You can only modify some parameters in this list {{ search_keys }}. Give me a series of parameter adjustment (about 10 groups) in JSON format for me to execute using pybamm first. Please do not add any invalid comments for JSON.
OTHER ROUNDPROMPT:
Here are the results:
{{ cycle_description }}
The params should just in {{ search_keys }}, I hope you can summarize the above results and suggest the next 1 updated params group with new values as dict (without name) in JSON format.