

Battery-Sim-Agent: Leveraging LLM-Agent for Inverse Battery Parameter Estimation

Downloaded from: https://research.chalmers.se, 2026-11-03 03:00 UTC

Citation for the original published paper (version of record):

Chen, J., Gui, X., Fang, S. et al (2026). Battery-Sim-Agent: Leveraging LLM-Agent for Inverse Battery Parameter Estimation. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2-B: 10601-10612.
http://dx.doi.org/10.1145/3770855.3818856

N.B. When citing this work, cite the original published paper.

research.chalmers.se offers the possibility of retrieving research publications produced at Chalmers University of Technology. It covers all kind of research output: articles, dissertations, conference papers, reports etc. since 2004. research.chalmers.se is administrated and maintained by Chalmers Library

(article starts on next page)

Battery-Sim-Agent: Leveraging LLM-Agent for Inverse Battery Parameter Estimation

Jiawei Chen*
Peking University
Beijing, China

Xiaofan Gui†
Microsoft Research
Beijing, China

Shikai Fang†
Zhejiang University
Hangzhou, Zhejiang, China

Shengyu Tao
Chalmers University of Technology
Gothenburg, Sweden

Shun Zheng
Microsoft Research
Beijing, China

Weiqing Liu†
Microsoft Research
Beijing, China

Jiang Bian
Microsoft Research
Beijing, China

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. Project website: Battery-Sim-Agent.

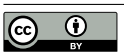
CCS Concepts

• **Applied computing** → *Chemistry*; • **Computing methodologies** → **Intelligent agents**.

Keywords

Large Language Models, AI Agents, Battery Parameter Estimation, AI for Science, Inverse Problems, Digital Twins

*Work done during internship at Microsoft Research.
†Corresponding author.



This work is licensed under a Creative Commons Attribution 4.0 International License.
KDD 2026, Jeju Island, Republic of Korea.
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2259-2/2026/08
https://doi.org/10.1145/3770855.3818856

ACM Reference Format:
Jiawei Chen, Xiaofan Gui, Shikai Fang, Shengyu Tao, Shun Zheng, Weiqing Liu, and Jiang Bian. 2026. Battery-Sim-Agent: Leveraging LLM-Agent for Inverse Battery Parameter Estimation. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD 2026), August 9–13, 2026, Jeju Island, Republic of Korea.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3770855.3818856

1 Introduction

The transition to a sustainable energy future is intrinsically linked to advancements in battery technology [10]. From electrifying transportation to stabilizing power grids, next-generation batteries are a critical need [10]. However, the physical development and testing of these batteries is a major bottleneck [2, 27]. Characterizing a battery's performance and degradation over its lifetime can require thousands of hours of continuous cycling [1, 28]. A promising alternative is to build *digital twins*—high-fidelity virtual replicas instantiated in physics-based simulators such as PyBaMM [30]. 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 [9, 26, 29].

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 [14, 32] or genetic algorithms [4, 16, 38] 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 [33]. These agents have shown success in solving inverse problems in diverse fields like materials science [34] and solid mechanics [19]. 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 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 an 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 [29]. 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 [9, 26].

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 [3, 24]).

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:

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

Here, d is a distance metric that can compare multiple trajectories, w_p are weights for each protocol used to balance different scales, 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 [13]. This has led to the emergence of "agentic science" where LLM agents take on the role of the human scientist [33]. These agents have shown success across diverse domains: molecular design [34], inverse problems in solid mechanics [19], and galaxy observation interpretation [31]. 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. (1), our overall goal is to find parameters $\theta^\star$ that minimize the composite loss $\mathcal{L}(\theta)$. However, unlike traditional methods that 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:

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

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.

By disaggregating the objective, we enable the agent to perform *causal attribution*. The LLM can map specific feature mismatches (symptoms) to specific parameter subsets (causes), effectively navigating the high-dimensional parameter space by decomposing the problem into physically meaningful sub-problems.

3.2 The Hypothesis-Driven Reasoning Loop

The agent's workflow mimics a human scientist, proceeding in three steps within each iteration. Direct mapping from observation to parameters using LLMs can lead to hallucinations, therefore, we design a structured reasoning loop that enforces a Chain-of-Thought (CoT) process.

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"]
}
```

This translation bridges the modality gap. While LLMs struggle to interpret raw floating-point arrays, they excel at reasoning with semantic descriptions of trends and shapes.

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."
```

This intermediate reasoning step serves as a "cognitive check." By forcing an explicit hypothesis, we ground the agent's actions in physical laws, reducing the likelihood of proposing physically implausible parameters.

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."
}
```

Battery-Sim-Agent performs targeted local adjustments based on the specific hypothesis derived in the previous step, rather than relying on global exploration of the parameter space.

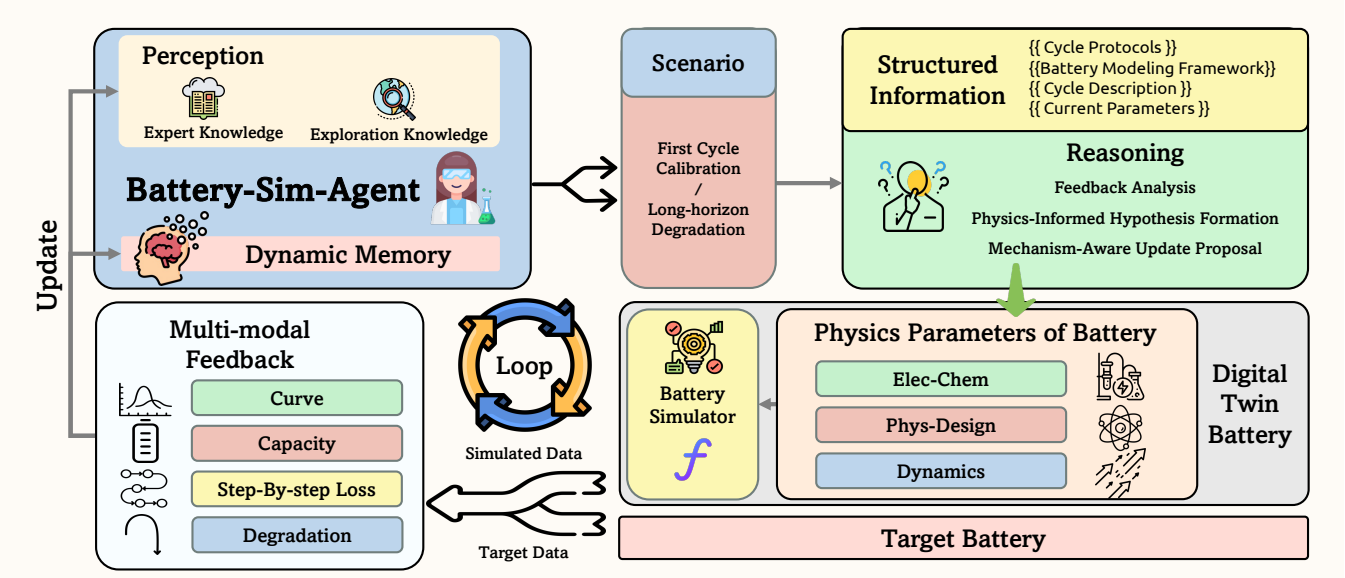


Figure 1: The closed-loop workflow of Battery-Sim-Agent. The agent proposes parameters for the PyBaMM simulator. The simulator's output is then compared against target data to generate structured, multi-modal feedback (Sec. 3.2), which the agent analyzes using its dynamic memory (Sec. 3.3) to reason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

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"). Since we cannot compute the gradient $\nabla_\theta \mathcal{L}$ directly, this phase effectively allows the agent to build an "internal mental model" of the local sensitivity landscape. 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 [29, 30]. The problem is notoriously ill-posed, with many parameter combinations yielding similar macroscopic outputs [9, 26]. Historically, this challenge has been addressed using classical black-box optimization (BBO) methods, such as Bayesian optimization or evolutionary algorithms [14, 32, 38]. While versatile, these methods are fundamentally "blind" optimizers; they treat the simulator as an opaque oracle and lack physical

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
  $Y^{\text{sim}} \leftarrow \textsc{Simulate}(\theta_0 + \delta_k)$
  $F_k \leftarrow \textsc{BuildFeedback}(Y^{\text{sim}}, Y^{\text{obs}})$
  $\mathcal{M}_k \leftarrow \textsc{UpdateMemory}(\mathcal{M}_{k-1}, F_k, \text{“Summarize sensitivity”})$
end for
// **Phase 2: Main Optimization Loop**
for $t = 0$ to $T - 1$ **do**
  $Y^{\text{sim}} \leftarrow \textsc{Simulate}(\theta_t)$
  $F_t \leftarrow \textsc{BuildFeedback}(Y^{\text{sim}}, Y^{\text{obs}})$
  $\Delta\theta_t, \text{rationale}_t \leftarrow \textsc{QueryLLM}(F_t, \mathcal{M}_{N_w+t-1})$
  $\theta_{t+1} \leftarrow \Pi_{[\ell,u]}(\theta_t + \eta_t \, \Delta\theta_t)$
  $\mathcal{M}_{N_w+t} \leftarrow \textsc{UpdateMemory}(\mathcal{M}_{N_w+t-1}, F_t, \Delta\theta_t, \text{rationale}_t)$
  if converged **then**
    break
  end if
end for
return $\theta_{t^\star}$

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 [13, 39]. 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” [33]. 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 [34], solid mechanics [19], astrophysics [31], and hyperparameter optimization [15].

Human-AI collaborative optimization frameworks further integrate expert knowledge into the search loop. COBOL [36] augments Bayesian Optimization with human “accept/reject” feedback and provides theoretical no-harm and handover guarantees. In contrast, our Battery-Sim-Agent treats the LLM not as a verifier but as a generative reasoner proposing continuous parameter updates. While this offers richer semantic guidance grounded in physical intuition, it lacks the formal regret bounds available in COBOL which is an opportunity for future hybrid approaches.

Most relevant to our work is the emerging use of LLMs for parameter inference in physical systems. SimLM [18] demonstrated simulator-in-the-loop reasoning on kinematic problems, though performance degraded on more complex dynamics. Our work extends this paradigm to high-fidelity engineering systems: battery parameter estimation requires navigating coupled PDEs, high dimensional parameter spaces, and pronounced equifinality—far beyond the low-dimensional settings addressed in prior work.

These works demonstrate the potential of LLM-agents to navigate complex search spaces more intelligently than traditional algorithms. Building upon these foundations, 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. Specifically, our experimental design explores a new paradigm for addressing the inverse problem of battery parameter estimation through physics-grounded reasoning. We hypothesize that an agent capable of forming and testing causal hypotheses can navigate the parameter landscape with greater efficiency and robustness. To assess this, we structure our experiments across three progressively challenging tiers: (1) **Controlled Benchmarks** to rigorously quantify parameter accuracy against ground truth; (2) **Complex Dynamics** involving long-horizon degradation to test reasoning over time; and (3) **Practical Validation** on real-world data to evaluate applicability in noisy, uncertain environments.

5.1 Experimental Setup

Benchmark Test Suite. We construct a diverse benchmark suite using the high-fidelity Doyle-Fuller-Newman (DFN) model [6] in PyBaMM [30]. To address the challenge of defining a consistent evaluation metric across heterogeneous systems, our construction follows a strict “Base-Perturbation-Filter” pipeline. This ensures that every task represents a realistic inverse problem where the agent starts with a known prior (θ_{init}) and must recover an unknown ground truth (θ^*):

1. Base Chemistries (The Priors): We employ five classic, well-established parameter sets from the literature: Chen2020[5] (NMC811/SiOx-Graphite), ORegan2022[23] (NMC811/SiOx-Graphite), Prada2013[25] (LFP/graphite), Ecker2015[7, 8] ($Li(Ni_{0.4}Co_{0.6})O_2$/graphite), and Marquis2019[17] (LCO/graphite). These serve as the initial parameter guess θ_{init} for the agent in each task. **2. Target Generation (The Ground Truth):** To generate the “unknown” target data Y_{obs}, we apply controlled perturbations to the base parameters to create a ground truth vector θ^*. We define two difficulty modes:

- **Regular Mode (Multi-Parameter):** We apply 12 expert-designed, physically-plausible multi-parameter perturbations that represent realistic manufacturing variations or design choices (e.g., simultaneously altering electrode thickness and porosity). These combinations are carefully crafted to maintain physical plausibility while creating meaningful optimization challenges.
- **Extreme Mode (Single-Parameter):** We apply large perturbations (0.5× to 2.0×) to one of 9 key parameters (particle radiation, electrode thicknesses, porosities, Bruggeman coefficients, separator thickness), creating challenging cases that often push the simulator to its stability limits.

3. 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.

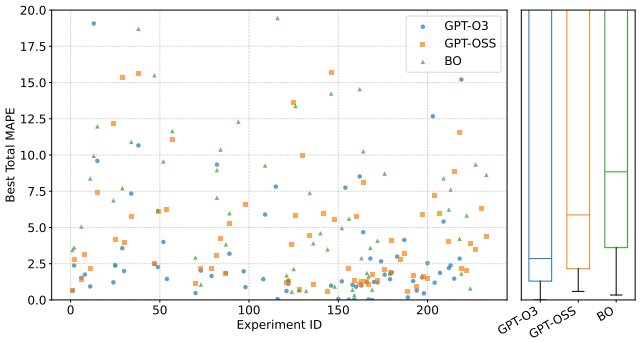
Data Generation and Filtering Process. Our systematic data generation follows a rigorous multi-stage process. We iterate through all combinations of base parameter sets, C-rates, and perturbation rules, then apply a two-stage filtering process: (1) We discard parameter combinations that result in simulation 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. In each task, the agent is initialized at θ_{init} and must recover the hidden θ^* by minimizing the discrepancy with Y_{obs}. 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:

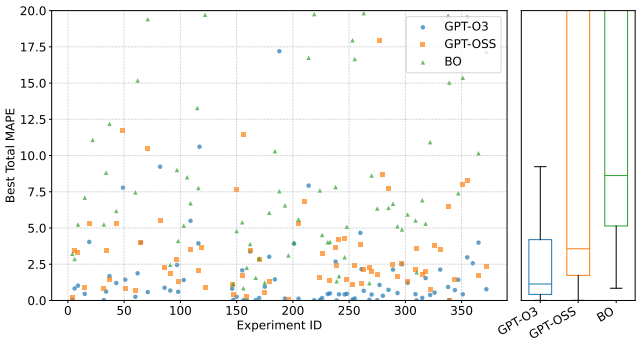
- **Battery-Sim-Agent-O3:** The full agent powered by GPT-O3 [21], incorporating our complete reasoning workflow with hypothesis generation, iterative refinement, and multi-objective optimization capabilities.
- **Battery-Sim-Agent-OSS:** An ablation using GPT-OSS [22], a powerful 120B parameter open-source 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 [20], representing state-of-the-art black-box optimization methods commonly used in parameter estimation.

We also experimented with other evolutionary algorithms including CMA-ES [11], 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. Because the inverse problem is generally non-identifiable from a single protocol, trajectory error alone does not *a priori* certify parameter recovery; we therefore validate, on the synthetic benchmark where the ground-truth θ^* is known, that trajectory error and parameter error decrease together within every test case (mean within-case Pearson $r = 0.963$, Spearman $\rho = 1.000$, monotonic co-decrease in 100% of cases over 50 test cases spanning 5 chemistries $\times$ 2 modes). We also verify that recovered parameters remain valid under held-out protocols (e.g., for the 5% noise regime, trajectory MAPE stays below 2% across unseen 0.2C/1C/2C CCCV conditions). Full procedures and tables are reported in Appendix C.



(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.

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 58–97% compared to BO across all five chemistries (and lowest RMSE on four of five), with 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 picture is more nuanced: the agent reduces

MAPE on Chen2020, ORegan2022 and Ecker2015 substantially, but Bayesian Optimization remains competitive and is actually better on Prada2013 and Marquis2019 in both MAPE and RMSE. We attribute this to a low-headroom regime where the literature defaults are already close to the perturbed target, leaving little room for the agent's structured exploration to add value beyond a well-tuned acquisition function. We therefore calibrate the headline claim: the agent's advantage is most pronounced in regular-mode multi-parameter calibration and in the more difficult settings discussed in Sec. 5.3, while extreme single-parameter shifts in low-headroom chemistries remain competitive for classical BO. A failure-case discussion under aggressive 2C-CCCV simulation (ORegan2022) is provided in Appendix H.

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 reasoning. 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.

Real-World Validation. We validate Battery-Sim-Agent on 7 real battery tasks, using data from the CALCE[12, 35] dataset obtained from public repositories [37], 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.

5.4 Attribution: Scaffold vs. Backbone Capability

A natural question is whether the agent's gains come from the reasoning scaffold itself or from the choice of a strong proprietary backbone (GPT-O3). To disentangle these effects, we run two controlled studies on a fixed 8-case subset of Chen2020 with 15 rounds per case. First, a **same-family Qwen2.5 scaling study** under the *full* framework yields average final MAPE of 30.08% (7B), 4.02% (14B), and 3.33% (32B), showing that backend capability matters strongly: 7B is too weak, while 14B and 32B both reach low final errors. Second, a **fixed-backbone ablation** on Qwen2.5-32B isolates scaffold components: the complete scaffold reaches 3.33% MAPE, scalar-only feedback degrades to 6.64%, removing domain knowledge gives 5.44%, and removing the per-round memory collapses to 32.94%—a 9.9× degradation that identifies the iterative memory mechanism as the dominant component of the scaffold. We further tested whether *richer feedback alone*, fed directly to a numerical optimizer, recovers the gain: weighted multi-objective DE reaches 3.68% vs. scalar DE's 1.00% on the same case, and qNEHVI fails catastrophically (current MAPE $\approx$ 132,091.98%). Richer feedback is therefore not sufficient on its own; the gain emerges from combining iterative memory-based scaffolding, structured diagnostic feedback, lightweight domain priors, and sufficient model capability. Full tables are reported in Appendix D.

5.5 Computational Cost

We add a matched outer-loop runtime benchmark on Chen2020. Under identical 20-step budgets, our agent (API-served o3) makes 20 simulator evaluations and 20 LLM calls with cumulative simulator time 58.1s, cumulative LLM-API time 248.9s, and end-to-end wall-clock 322.4s; the matched BO run takes 541.0s. We further observe a different growth pattern: our per-round wall-clock stays nearly flat (13.85s vs. 13.82s between the first and last five rounds), whereas BO's per-trial time rises sharply from 44.3s to 477.3s, reflecting accumulated GP-solver overhead. We therefore calibrate the cost claim carefully: the agent does add LLM inference, but under this matched benchmark its closed-loop runtime remains practical and is lower than BO on this case. Full numbers are reported in Appendix E.

6 Conclusion and Limitations

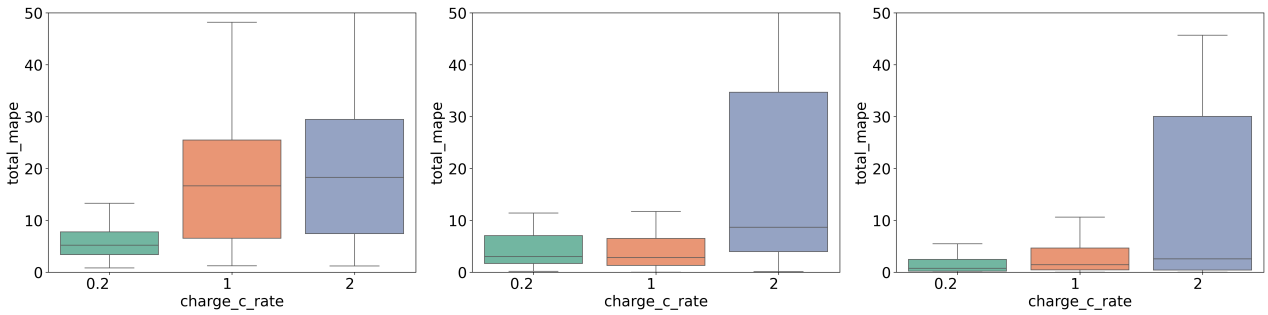
We introduced Battery-Sim-Agent, a novel framework that reframes the challenging inverse problem of parameterizing battery digital twins as a reasoning task. By deploying an LLM-agent in the loop with a high-fidelity simulator, we demonstrated a new paradigm for scientific optimization that mimics human expert workflows. Our experiments showed that this reasoning-based approach outperforms traditional black-box optimizers on a diverse benchmark suite under most regular-mode settings and remains competitive in the more challenging extreme regime, while also extending credibly to long-horizon degradation fitting and real-world CALCE cells.

Calibrated scope of the claims. On our benchmark, trajectory fit serves as a *validated proxy* for parameter recovery—trajectory error and parameter error decrease monotonically together within every test case (mean within-case Pearson $r = 0.963$; Appendix C)—and recovered parameters remain valid under held-out protocols. We therefore frame Battery-Sim-Agent as a strong simulator-based calibration / digital-twin method on this benchmark, rather than as a universal solver for every inverse battery setting. In particular, classical inverse problems can be non-identifiable when only a limited protocol is observed, and our framework does not claim to resolve identifiability in that adversarial regime.

Where the agent is most useful. The agent's advantage is most pronounced in harder calibration regimes that require structured iterative reasoning: extreme parameter shifts, long-horizon degradation, and noisy real-world cells. Conversely, in low-headroom

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

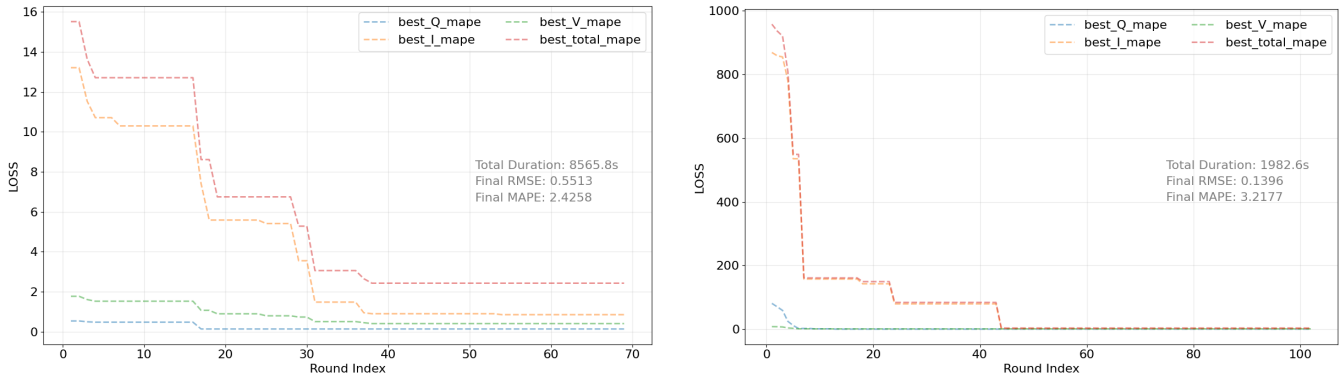


(a) BO **(b) Battery-Sim-Agent-OSS** **(c) Battery-Sim-Agent-O3**

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.

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**



(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.

chemistries where the literature defaults are already near the ground truth (e.g., Prada2013 and Marquis2019 in extreme mode), classical Bayesian Optimization remains competitive and can outperform the agent on RMSE—we observe and discuss these cases explicitly in Sec. 5.2 and Appendix H. Rare simulator-edge regimes such as aggressive ORegan2022 2C-CCCV remain difficult for all methods because the forward PyBaMM/DFN model itself becomes numerically unstable.

Limitations and future directions. Several aspects merit further investigation. First, in contrast to classical optimization techniques or Bayesian methods with formal guarantees, the LLM-agent's behavior is inherently probabilistic, and tighter theoretical characterization of its convergence remains an open question; our empirical evidence consists of step-wise error reduction on synthetic and real tasks (Fig. 4, Appendix Fig. 5), a fixed-model ablation identifying memory as the dominant component, and a same-family Qwen2.5 scaling study (Appendix D). Second, the agent's effectiveness depends on the underlying LLM's reasoning capability; exploring lighter-weight or domain-fine-tuned backbones is a natural next step. Third, the closed-loop cost of repeated simulator–agent interactions, while practical in our matched-budget benchmark (Appendix E), suggests opportunities for efficiency improvements through model reduction or hybrid numerical–agent strategies that delegate local refinement to fast solvers and global reasoning to the agent. Finally, the framework is simulator-agnostic in principle, but its empirical validation in this paper is restricted to PyBaMM/DFN; extending it to domains without high-fidelity digital twins likely requires coupling with learned surrogates. Overall, Battery-Sim-Agent provides a first step toward reasoning-driven autonomous scientific discovery in battery research.

Ethical Considerations

Battery-Sim-Agent targets a scientific control and engineering optimization task, calibrating physics-based digital twins of lithium-ion cells, where improved sample efficiency can reduce computational cost and the number of physical experiments required for cell characterization.

Data and human subjects. This study does not involve human subjects, personal data, or private information. All synthetic benchmarks are generated from open-source electrochemical parameter sets distributed with PyBaMM [30] (Chen2020, ORegan2022, Prada2013, Ecker2015, and Marquis2019), and the real-world validation uses cycling data from the publicly released CALCE[12, 35] dataset. No proprietary, sensitive, or personally identifiable information was used at any stage of training, calibration, or evaluation.

Dual-use considerations. The agent operates on physics-based forward models of commercial lithium-ion cells and recovers parameters such as electrode thickness, porosity, and active-material volume fractions. These quantities are already widely documented in the public literature, and the recovered values do not provide a meaningful pathway for malicious uplift or circumvention of safety controls in deployed battery systems. The framework is methodological in nature: it accelerates inverse calibration of digital twins rather than enabling the design of physically novel or dangerous chemistries. As with any optimization tool, downstream users should ensure that recovered parameters are interpreted within the validated operating envelope of the underlying simulator and the manufacturer's safety limits, and should not be extrapolated to off-nominal regimes (e.g., thermal runaway, over-charge, mechanical abuse) without independent physical verification.

Responsible deployment. Any deployment of Battery-Sim-Agent for industrial battery design, second-life screening, or grid-scale energy-storage applications should comply with applicable safety regulations, institutional review processes, and domain-specific certification standards. We recommend keeping a human-in-the-loop review of the agent's natural-language hypotheses and structured parameter updates—particularly for safety-critical parameters—and treating the agent as a calibration assistant rather than an autonomous decision-maker. The recorded chain-of-thought, JSON rationale fields, and per-round memory provide an auditable trail that supports such oversight.

References

[1] 2026. iMOE: Prediction of Second-Life Battery Degradation Trajectory Using Interpretable Mixture of Experts. *Nature Communications* (2026). https://www.nature.com/articles/s41467-026-69369-1

[2] Peter M. Attia, Eric Moch, and Patrick K. Herring. 2025. Challenges and Opportunities for High-Quality Battery Production at Scale. 16, 1 (2025), 611. doi:10.1038/s41467-025-55861-7

[3] Robert S. Balog and Ali Davoudi. 2013. Batteries, Battery Management , and Battery Charging Technology. In *Transportation Technologies for Sustainability*. Springer, New York, NY, 122–157. doi:10.1007/978-1-4614-5844-9_822

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

[5] Chang-Hui Chen, Ferran Brosa Planella, Kieran O'Regan, Dominika Gastol, W. Dhammika Widanage, and Emma Kendrick. 2020. Development of Experimental Techniques for Parameterization of Multi-scale Lithium-ion Battery Models. *Journal of The Electrochemical Society* 167, 8 (may 2020), 080534. doi:10.1149/1945-7111/ab9050

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

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

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

[9] R. Gopinath, S. Santhanagopalan, and Richard D. Braatz. 2016. An Inverse Method for Estimating the Electrochemical Parameters of Lithium-Ion Batteries. *Journal of The Electrochemical Society* 163, 14 (2016), A3045–A3054.

[10] Ahmad Hamdan, Cosmas Daudu, Adefunke Fabuyide, Emmanuel Etukudoh, and Sedat Sonko. 2024. Next-Generation Batteries and U.S. Energy Storage: A Comprehensive Review: Scrutinizing Advancements in Battery Technology, Their Role in Renewable Energy, and Grid Stability. 21 (2024), 1984–1998. doi:10.30574/wjarr.2024.21.1.0256

[11] Nikolaus Hansen, Youhei Akimoto, and Petr Baudis. 2019. CMA-ES/pycma on Github. Zenodo, DOI:10.5281/zenodo.2559634. doi:10.5281/zenodo.2559634

[12] Wei He, Nicholas Williard, Michael Osterman, and Michael Pecht. 2011. Prognostics of Lithium-Ion Batteries Based on Dempster–Shafer Theory and the Bayesian Monte Carlo Method. *Journal of Power Sources* 196, 23 (Dec. 2011), 10314–10321. doi:10.1016/j.jpowsour.2011.08.040

[13] 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. 2025. A Survey of Scientific Large Language Models: From Data Foundations to Agent Frontiers. arXiv:2508.21148 [cs.CL] https://arxiv.org/abs/2508.21148

[14] Benben Jiang, Marc D Berliner, Kun Lai, Patrick A Asinger, Hongbo Zhao, Patrick K Herring, Martin Z Bazant, and Richard D Braatz. 2022. Fast charging design for Lithium-ion batteries via Bayesian optimization. *Applied Energy* 307 (2022), 118244.

[15] Siyi Liu, Chen Gao, and Yong Li. 2024. Large language model agent for hyperparameter optimization. *arXiv preprint arXiv:2402.01881* (2024).

[16] Dirk Magnor and Dirk Uwe Sauer. 2016. Optimization of PV battery systems using genetic algorithms. *Energy Procedia* 99 (2016), 332–340.

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

[18] Sean Memery, Mirella Lapata, and Kartic Subr. 2024. *SimLM: Can Language Models Infer Parameters of Physical Systems?* arXiv:2312.14215 [cs] doi:10.48550/arXiv.2312.14215

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

[20] 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. 2025. Ax: A Platform for Adaptive Experimentation. In *AutoML 2025 ABCD Track*.

[21] OpenAI. 2025. OpenAI o3 and o4-mini System Card. https://cdn.openai.com/pdf/2221c875-02dc-4789-800b-e7758f3722c1/o3-and-o4-mini-system-card.pdf

[22] 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. 2025. Gpt-Oss-120b & Gpt-Oss-20b Model Card. arXiv:2508.10925 [cs] doi:10.48550/arXiv.2508.10925

[23] Kieran O'Regan, Ferran Brosa Planella, W. Dhammika Widanage, and Emma Kendrick. 2022. Thermal-Electrochemical Parameters of a High Energy Lithium-Ion Cylindrical Battery. *Electrochimica Acta* 425 (2022), 140700. doi:10.1016/j.electacta.2022.140700

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

[25] E. Prada, D. Di Domenico, Y. Creff, J. Bernard, V. Sauvant-Moynot, and F. Huet. 2013. 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 (Feb. 2013), A616. doi:10.1149/2.053304jes

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

[27] L. A. Román-Ramírez and J. Marco. 2022. Design of Experiments Applied to Lithium-Ion Batteries: A Literature Review. 320 (2022), 119305. doi:10.1016/j.apenergy.2022.119305

[28] Ana-Irina Stroe, Daniel-Loan Stroe, Vaclav Knap, Maciej Swierczynski, and Remus Teodorescu. 2018. Accelerated Lifetime Testing of High Power Lithium Titanate Oxide Batteries. In *2018 IEEE Energy Conversion Congress and Exposition (ECCE)*. 3857–3863. doi:10.1109/ECCE.2018.8557416

[29] Venkat R. Subramanian and Richard D. Braatz. 2013. Modeling and Simulation of Lithium-Ion Batteries from a Systems Engineering Perspective. *Journal of The Electrochemical Society* 160, 4 (2013), R93–R108.

[30] Valentin Sulzer, Scott G. Marquis, Robert Timms, Martin Robinson, and S. Jon Chapman. 2021. PyBaMM: Python Battery Mathematical Modelling. *Journal of Open Research Software* 9, 1 (2021), 14.

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

[32] Xizhe Wang and Benben Jiang. 2023. Multi-objective optimization for fast charging design of lithium-ion batteries using constrained Bayesian optimization. *Journal of Power Sources* 584 (2023), 233602.

[33] Jiaqi Wei, Yuejin Yang, Xiang Zhang, Yuhan Chen, Xiang Zhuang, Zhangyang Gao, Dongzhan Zhou, Guangshuai Wang, Zhiqiang Gao, Juntai Cao, et al. 2025. From AI for Science to Agentic Science: A Survey on Autonomous Scientific Discovery. *arXiv preprint arXiv:2508.14111* (2025).

[34] Mengsong Wu, YaFei Wang, Yidong Ming, Yuqi An, Yuwei Wan, Wenliang Chen, Binbin Lin, Yuqiang Li, Tong Xie, and Dongzhan Zhou. 2025. ChemAgent: Enhancing LLMs for Chemistry and Materials Science through Tree-Search Based Tool Learning. *arXiv preprint arXiv:2506.07551* (2025).

[35] Yinjiao Xing, Eden W. M. Ma, Kwok-Leung Tsui, and Michael Pecht. 2013. An Ensemble Model for Predicting the Remaining Useful Performance of Lithium-Ion Batteries. *Microelectronics Reliability* 53, 6 (June 2013), 811–820. doi:10.1016/j.microrel.2012.12.003
[36] Wenjie Xu, Masaki Adachi, Colin N. Jones, and Michael A. Osborne. 2024. *Principled Bayesian Optimisation in Collaboration with Human Experts.* arXiv:2410.10452 [cs] doi:10.48550/arXiv.2410.10452
[37] Han Zhang, Xiaofan Gui, Shun Zheng, Ziheng Lu, Yuqi Li, and Jiang Bian. 2024. BatteryML: An Open-source Platform for Machine Learning on Battery Degradation. In *The Twelfth International Conference on Learning Representations.*
[38] Liqiang Zhang, Lixin Wang, Gareth Hinds, Chao Lyu, Jun Zheng, and Junfu Li. 2014. Multi-objective optimization of lithium-ion battery model using genetic algorithm approach. *Journal of Power Sources* 270 (2014), 367–378.
[39] Wenhua Zuo, Huihuo Zheng, Tanjin He, Venkatram Vishwanath, Maria KY Chan, Rick L Stevens, Khalil Amine, and Gui-Liang Xu. 2025. Large language models for batteries. *Joule* 9, 8 (2025).

A Reproducibility

All experiments use fixed random seeds, and key experiments were repeated to verify consistency. The source code and related resources have been released at https://github.com/opqrst-chen/Battery-Sim-Agent. Key hyperparameters used in the headline experiments are summarised in Table 4.

Table 4: Key hyperparameters of the LLM-agent and BO/CMA-ES baselines.

LLM-agent	
Warm-up rounds N_w / total budget T	20 / 80
Backbone (main results)	GPT-O3 (API)
Bayesian Optimization	
Platform	Meta's Ax (v1.1.0)
Init. / optimisation	Sobol / GPEI
Surrogate / acquisition	SingleTaskGP (Matérn) / LogNEI
Warm-up trials	2× number of parameters
Random seed	1234
CMA-ES	
Population size	number of parameters + 1
Bounds / max iter.	$[x_0^{\text{lower}}, x_0^{\text{upper}}]$ / matched budget
Random seed	1234

B Benchmark Generation Details

We construct the benchmark via a strict "Base–Perturbation–Filter" pipeline. Starting from five literature parameter sets θ_{init} (Chen2020, ORegan2022, Prada2013, Ecker2015, Marquis2019), we apply controlled perturbations to define each ground-truth θ^* and forward-simulate the corresponding observation Y_{obs} with PyBaMM/DFN under 0.2C, 1C and 2C CCCV protocols. Each task then asks the agent to recover θ^* starting from θ_{init}.

Extreme mode. We perturb a single parameter at a time with large multiplicative factors, producing highly non-convex objectives: particle radii (anode/cathode) $\times\{0.5, 2.0\}$; electrode thicknesses (anode/cathode) $\times\{0.75, 1.5\}$; electrode porosities ± 0.05; Bruggeman coefficients $\in \{1.5, 2.0, 2.5\}$ (anode) / $\{1.3, 1.8, 2.3\}$ (cathode); separator thickness $\times\{0.7, 1.3\}$.

Regular mode. We apply 12 expert-designed multi-parameter perturbations representing physically plausible manufacturing variations and design trade-offs (e.g., max-power profiles with reduced particle radii and electrode thicknesses; energy-leaning profiles with co-varying porosity and thickness; high-/low-porosity ionic regimes; asymmetric anode/cathode diffusion limits; thin-separator / thick-electrode combinations).

Filtering and final selection. We retain only combinations that (i) converge in the PyBaMM IDAKLU DAE solver and (ii) induce $\geq 1\%$ capacity change relative to baseline. From the resulting 233 (extreme) / 373 (regular) valid combinations, we sample 100 per mode for the 200-task evaluation suite. Simulation "failures" refer to non-convergence due to physical infeasibility (e.g., negative surface concentration under extreme low-diffusion / high-C-rate combinations), not numerical precision; we treat these cases as invalid parameter sets.

Long-horizon degradation and real-world tasks. For the degradation experiment we select 5 representative parameter sets from the benchmark, enable the "reaction limited" SEI mechanism in PyBaMM, and run each simulation for 200 cycles; the agent jointly fits the base electrochemical parameters together with SEI degradation parameters (kinetic rate constant, conductivity, etc.) to match the observed capacity-fade curve. For the real-world experiment we apply the agent to 7 CALCE cells, using the first few cycles to infer parameters and the remaining cycles for held-out evaluation.

C Parameter Recovery and Held-Out Protocol Validation

Because the inverse battery problem is generally non-identifiable from a single protocol, trajectory error alone does not certify that the recovered parameters are the true latent parameters. On the synthetic benchmark where θ^* is known by construction, we validate that trajectory error and parameter error track each other tightly.

Within-case rank correlation. For each of 50 test cases (5 chemistries $\times$ 2 modes, evenly sampled across C-rates), we record the per-round trajectory MAPE and parameter error $\|\hat{\theta} - \theta^*\|$. Across all cases: mean within-case Pearson $r = 0.963$, Spearman $\rho = 1.000$, and 100% of cases exhibit monotonic co-decrease, so trajectory fit is a validated proxy for parameter recovery on this benchmark.

Held-out protocol validation. We apply parameters recovered on a fitting protocol to unseen CCCV protocols at 0.2C/1C/2C (Table 5). With 5%-noise parameter recovery, held-out MAPE stays below 2% across all C-rates; the unfitted literature default incurs 6.85–28.78%, growing with C-rate. This supports physically meaningful recovery on this benchmark.

D Scaffold vs. Backbone Capability

To disentangle the scaffold from backbone capability we run two controlled studies on a fixed 8-case Chen2020 subset (15 rounds per case).

Same-family Qwen2.5 scaling (full framework). Average final MAPE: Qwen2.5-7B 30.08%, Qwen2.5-14B 4.02%, Qwen2.5-32B **3.33**%.

Table 5: Held-out protocol validation: trajectory MAPE (%) on unseen 0.2C/1C/2C CCCV conditions, as a function of recovery quality.

Recovery quality	0.2C	1C	2C
Perfect	0.00	0.00	0.00
5% noise	1.49	1.72	1.97
Default (no fit)	6.85	13.95	28.78

Backbone capability matters strongly: 7B is too weak, while 14B and 32B both reach low final error.

Fixed-backbone ablation (Qwen2.5-32B).. Full framework **3.33**%; scalar-only feedback 6.64%; no domain knowledge 5.44%; no memory 32.94%. Removing the per-round memory collapses performance by ∼ 9.9×, identifying iterative memory-based scaffolding as the dominant component.

Does richer feedback alone suffice? Feeding the same structured residual signals to numerical multi-objective optimizers on the same case: weighted multi-objective DE reaches 3.68% MAPE (worse than scalar DE's 1.00%) and qNEHVI fails catastrophically (∼ 132,091.98% under our default budget). Richer feedback alone is not sufficient; it becomes useful only when coupled with iterative reasoning over accumulated context.

E Matched Runtime Benchmark

We benchmark wall-clock cost at matched 20-step outer-loop budgets on Chen2020 (Table 6). Our agent stays nearly flat per round (13.85s → 13.82s between first and last five rounds), whereas BO's per-trial time rises from 44.3s to 477.3s due to accumulated GP-solver overhead. We do not claim universal runtime dominance over BO, but on this matched benchmark the closed-loop cost of Battery-Sim-Agent is practical and lower than BO even after including LLM inference.

Table 6: Matched 20-step runtime on Chen2020. "Ours" uses the API-served o3 model.

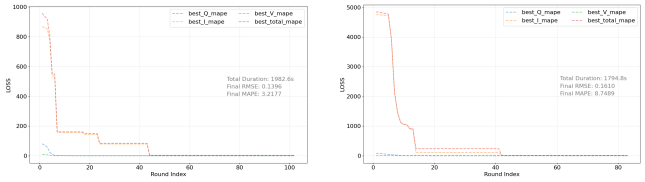
Metric	Ours	BO
Simulator evaluations	20	20
LLM calls	20	0
End-to-end wall-clock	**322.4s**	541.0s
Per-round avg. (first 5 / last 5)	13.85 / 13.82s	44.3 / 477.3s

F Warm-up Strategy: LLM-Driven vs. Random Perturbations

To validate the design choice of using LLM-generated perturbations during the warm-up phase (Phase 1 of Algorithm 1), we compare it against a random fixed-perturbation baseline that draws warm-up candidates uniformly from the same parameter bounds. Across the regular-mode benchmark, the LLM-driven warm-up consistently achieves both lower median trajectory error and a tighter interquartile range than fixed random search: in particular the Total MAPE upper whisker of the fixed strategy extends close to 10, whereas the LLM-driven strategy stays well below that and produces far fewer outliers. This indicates that the warm-up samples are not arbitrary noise but purposeful explorations that adapt to the current problem instance, supplying a higher-quality knowledge memory $\mathcal{M}_{N_w}$ for the subsequent main loop.

G Convergence on Real-World Data

To complement the convergence panel in the main text (Fig. 4, right), Figure 5 reports two additional CALCE cells. Sample A illustrates typical performance: starting from a high initial loss (MAPE > 900), the agent identifies and corrects the dominant error mode in a step-wise fashion (sharp drops near rounds 5 and 24), converging to RMSE = 0.1396 and MAPE = 3.22% by round 44 and remaining stable thereafter. Sample B is a harder case in which the initial parameters yield an extreme initial error (MAPE ≈ 4800); the agent still recovers rapidly, with successive drops at rounds 8, 15, and 42 and a final RMSE = 0.1610 / MAPE = 8.75%. The distinct plateaus between drops are consistent with the agent decoupling the parameter space and refining one mechanism at a time. These step-wise patterns provide the empirical evidence referenced in the convergence-reliability discussion in the Conclusion.



(a) Sample A: final MAPE 3.22%. **(b) Sample B: final MAPE** 8.75%.

Figure 5: Step-wise convergence on two additional CALCE cells. Dashed lines: per-group MAPE for capacity Q, voltage V, and total loss L.

H Failure Case: Poorly Specified Priors

We analyse a representative failure (Experiment ID 134): an aggressive 2C CCCV / 1C discharge protocol paired with the ORegan2022 default prior, which is numerically unstable under the target protocol—the PyBaMM solver terminates around 600 s during the CC charge phase due to stoichiometry/concentration-bound violations. Over 100 rounds, only 3 of the agent's candidate parameter sets produce successful simulations, so the agent receives sparse and largely uninformative feedback and cannot form a meaningful belief over the parameter space. GPT-o3 and BO baselines fail under the same setting for the same reason. This case illustrates a structural limitation: LLM-based agents and acquisition-function optimizers alike require a prior that is at least physically viable for the target protocol; agent reasoning cannot recover from a simulator that refuses to return informative residuals.