Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.075939

ARTICLE

Large Language Model-Enabled Constitutive Modeling for Rate-Dependent Plasticity and Automatic UMAT Subroutine Generation

Yuchuan Gu[1,2], Lusheng Wang[1,*], Jun Ding[1], Yanhong Peng[1], Changgeng Li[3,*] and Shaojie Gu[4,5]

[1]College of Mechanical Engineering, Chongqing University of Technology, Chongqing, China
[2]Institute of Agricultural Machinery, Chongqing Academy of Agricultural Sciences, Chongqing, China
[3]School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan, China
[4]Magnesium Research Center, Kumamoto University, Kumamoto, Japan
[5]Faculty of Advanced Science and Technology, Kumamoto University, Kumamoto, Japan

*Corresponding Authors: Lusheng Wang. Email: wangls@cqut.edu.cn; Changgeng Li. Email: d202480392@hust.edu.cn

Received: 11 November 2025; Accepted: 03 February 2026; Published: 12 March 2026

ABSTRACT: In materials science and engineering design, high-fidelity and high-efficiency numerical simulation has become a driving force for innovation and practical implementation. To address longstanding bottlenecks in the development of conventional material constitutive models—such as lengthy modeling cycles and difficulties in numerical implementation—this study proposes an intelligent modeling and code generation approach powered by large language models. A structured knowledge base integrating constitutive theory, numerical algorithms, and UMAT (User Material) interface specifications is constructed, and a retrieval-augmented generation strategy is employed to establish an end-to-end workflow spanning experimental data parsing, constitutive model formulation, and automatic UMAT subroutine generation. Experimental results show that the method achieves high accuracy for both a classical Johnson–Cook model and a physics-informed neural network (PINN) model, with key parameter identification errors below 5%. Moreover, the automatically generated UMAT subroutines yield finite element simulation results in Abaqus that are highly consistent with theoretical predictions (coefficient of determination $R^2 > 0.98$) while maintaining good numerical stability. This framework is currently focused on the automatic construction of rate-dependent elastoplastic material models, and its core method also provides a clear path for extending to other constitutive categories such as hyperelasticity and viscoelasticity. This work provides an effective technical route for the rapid development and reliable numerical implementation of material constitutive models, significantly advancing the intelligence level of computational mechanics research and improving engineering application efficiency.

KEYWORDS: Large language model; constitutive model; UMAT subroutine

1 Introduction

As a core methodology in computer-aided engineering (CAE), the predictive accuracy and numerical stability of finite element analysis (FEA) fundamentally depend on constitutive models, which serve as key components of the simulation framework [1]. Constitutive models mathematically describe the mechanical response of materials under complex thermo-mechanical loading, thereby linking microstructural mechanisms to macroscopic structural behavior and providing a bridge from material characterization to structural performance prediction and design optimization [2]. As application domains continue to expand toward extreme service conditions, multiscale interactions, and multiphysics coupling, the demand for model interpretability, stable extrapolation capability, and computational robustness has increased substantially.

This trend also imposes more stringent requirements on the standardization and engineering deployment of trustworthy scientific machine learning [3].

The pathway from experimental data to constitutive implementations suitable for engineering simulation typically faces two tightly coupled challenges. The first is model development itself. Traditional physics-based models (e.g., the Johnson–Cook model and Arrhenius-type formulations) offer clear physical interpretations; however, their development often relies heavily on expert knowledge, requires long calibration cycles, and may be inadequate for representing highly nonlinear behavior involving multiple interacting mechanisms. The second challenge is reliable numerical implementation. Even when a satisfactory constitutive formulation is available, translating it into a stable and efficient user material subroutine (UMAT) in finite element software (e.g., Abaqus) requires substantial expertise in computational mechanics and programming. This process entails the implementation of complex stress-update algorithms, derivation of the consistent tangent modulus, and rigorous management of state variables; it is therefore prone to errors and difficult to verify, becoming a major bottleneck that hinders the engineering adoption of advanced constitutive models.

2 Related Work

To address the aforementioned challenges, recent research has progressed along two complementary directions: automation of constitutive model implementation and intelligent constitutive model development.

With respect to implementation automation, the central objective is to translate mathematical models into numerical code in a reliable and efficient manner. Eisenträger et al. [4,5] systematically proposed a "template-based" implementation strategy for isotropic hyperelastic models and further enabled automatic MATLAB code generation from strain-energy density functions to key constitutive tensors (stress and elasticity tensors), thereby significantly simplifying finite element integration for this class of models. More broadly, automated finite element frameworks such as FEniCS [6] and AceGen [7] leverage high-level programming abstractions and symbolic computation to automatically derive executable code from weak-form equations.

Regarding intelligent model development, machine learning (ML) techniques have been introduced to directly discover or calibrate constitutive relations from data. This goes beyond conventional parameter fitting and moves toward the selection (or discovery) of model forms. Linka and Kuhl [8] proposed a neural-network-based strategy for "optimal model discovery". Peirlinck et al. [9,10] further advanced this concept toward engineering deployment by developing a general-purpose material subroutine that can be integrated into the commercial software Abaqus. Of particular interest, physics-augmented or physics-constrained neural networks (e.g., Physics-Augmented Neural Networks, PANNs) embed physical laws as built-in constraints or through specialized network architectures. Compared with standard physics-informed neural networks (PINNs), such approaches can improve extrapolation performance while maintaining stronger physical consistency, and have therefore emerged as a prominent research frontier. For example, Schommartz et al. [11] systematically proposed a physics-augmented neural network framework for finite-deformation hyperelastic constitutive modeling. Kästner, Kalina, and their collaborators [12] developed constitutive artificial neural networks (CANNs) for elastoplasticity in which thermodynamic constraints are rigorously embedded. Aldakheel et al. [13] extensively investigated physics-constrained machine learning approaches in data-driven computational mechanics and multiphysics fracture modeling. The Kochmann group [14] has also made important contributions to multiscale constitutive modeling by leveraging microstructural information and machine-learning assistance. Collectively, these studies have driven the use of AI in constitutive modeling toward a higher level of physical consistency.

Despite these advances, seamlessly integrating "data-driven intelligent modeling" with "high-reliability automated implementation" into an end-to-end engineering workflow remains challenging. Existing intelligent approaches primarily focus on model discovery or parameter identification, and the resulting models typically still require manual translation into numerical code. In contrast, automation tools for implementation usually assume that an accurate mathematical model is provided a priori.

In recent years, large language models (LLMs) have demonstrated remarkable capabilities in code generation, logical reasoning, and multi-step task planning [15–17], offering a new opportunity to address the aforementioned challenge of seamlessly integrating intelligent model development with automated, reliable implementation. In the specific context of constitutive modeling, parameter identification, and UMAT development, LLMs have the potential to add value at several critical stages: automatically recommending or composing suitable constitutive frameworks based on salient features of experimental data [18]; translating constitutive relations into standardized mathematical representations [19]; deriving the consistent tangent modulus required by stress-update algorithms; and generating high-quality code compliant with prescribed interface specifications [20]. By constructing a domain knowledge base covering yield criteria, flow rules, and hardening laws, and by designing verification mechanisms that enforce numerical consistency and physical constraints, the reliability and engineering applicability of the generated outputs can be substantially improved.

This work aims to bridge the gap between "intelligent modeling" and "automated implementation". We propose an integrated framework based on a large language model (LLM) and retrieval-augmented generation (RAG). The key innovation lies in leveraging the natural-language understanding and code-generation capabilities of LLM to establish an end-to-end workflow that maps raw experimental data directly to a compilable and verifiable UMAT subroutine, with a particular focus on rate-dependent plastic constitutive models.

3 Methodology

The full pipeline, encompassing experimental data processing, constitutive model development, UMAT implementation, and verification within Abaqus, was established through an end-to-end framework grounded in LLM and RAG. As illustrated in Fig. 1, the framework centers on specification-driven prompt engineering. With RAG support, authoritative physical priors (yield functions, plastic flow rules, hardening laws, internal variable evolution equations) and interface templates are dynamically injected into the LLM's reasoning process. This enables a closed loop from semantic understanding of data, through model recommendation and synthesis, to automatic code generation and self-verification of the consistent tangent. The overall workflow comprises two stages: Stage I focuses on constructing a trustworthy constitutive model (analytical or neural network-based); Stage II concentrates on UMAT generation and finite element simulation verification.

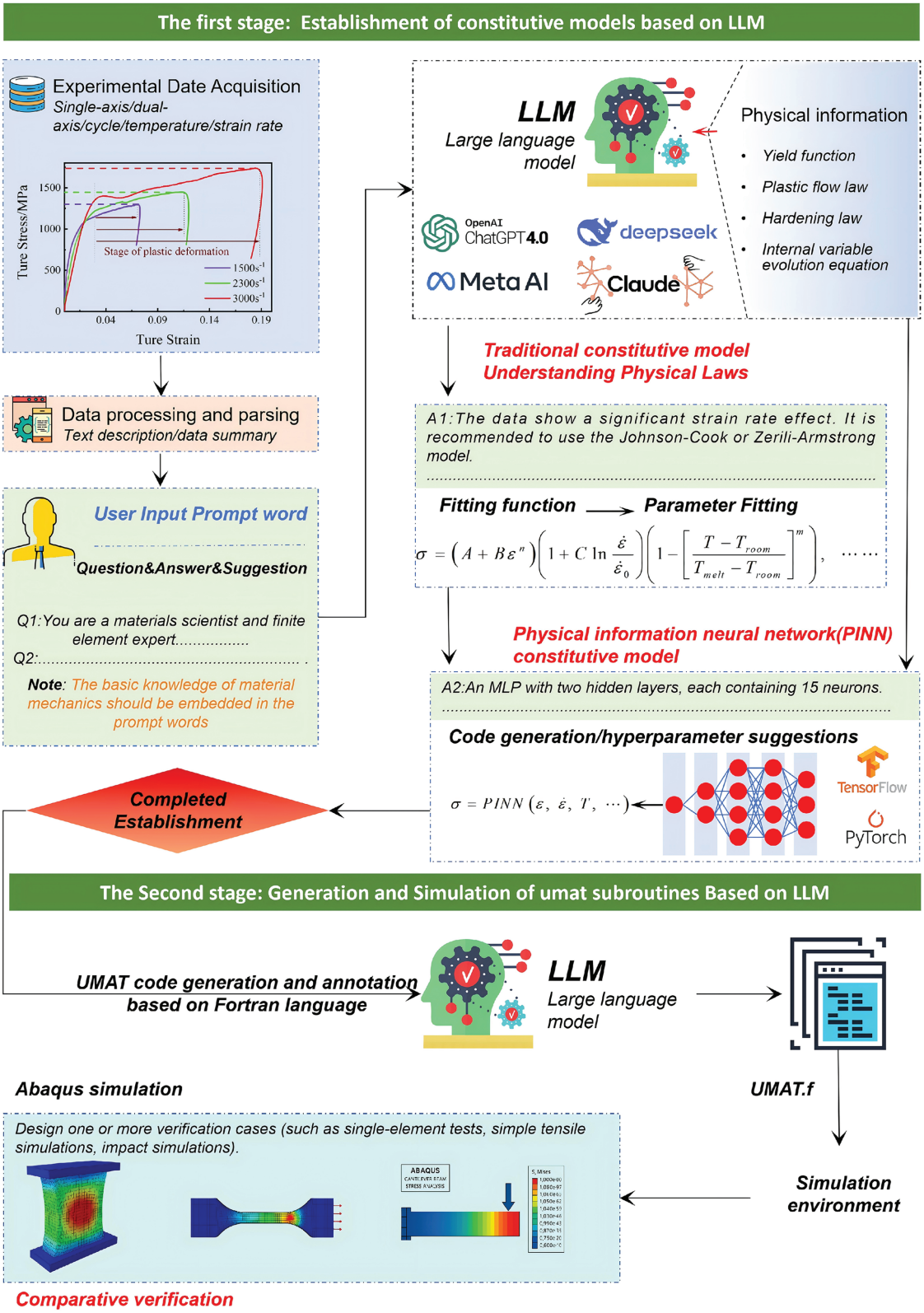

Figure 1: A schematic diagram of the process for constructing a constitutive model and UMAT subroutine based on LLM.

3.1 Main Framework

Stage I focuses on constitutive model construction driven by LLM–RAG (retrieval-augmented generation) synergy. Raw experimental stress–strain datasets spanning uni-/biaxial loading, cyclic paths, and coupled strain-rate/temperature conditions are first standardized (unit normalization, denoising) and structurally analyzed to extract salient features: yield onset, hardening/softening slopes, inflection points, and rate–temperature sensitivity indicators. These statistics are compressed into a structured textual abstract that conditions specification-driven prompt engineering. Under RAG, authoritative priors (yield criteria, plastic flow rules, hardening laws, internal variable evolution templates) are dynamically injected into the LLM reasoning chain, enabling mechanism attribution and adaptive model pathway selection. Two branches result: (i) analytical formulations (e.g., Johnson–Cook, Zerilli–Armstrong) for which the LLM generates parameter-fitting scripts, calibrates coefficients, and outputs a validated parameter set; (ii) physics-informed neural architectures (PINN) for which it specifies topology, hyperparameters, physical regularizers (non-negative dissipation, rotational invariance), and produces complete training code (PyTorch/TensorFlow) together with a vetted model card and weights.

Stage II translates the finalized constitutive representation into an Abaqus-compliant Fortran UMAT and performs hierarchical verification. Analytical models are hard-coded with stress update logic, return-mapping algorithms, and consistent tangent derivations; neural network models embed preprocessing, weight matrices, activation functions, and provide analytical, semi-analytical, or numerically reconciled Jacobians. The generated code includes explicit STATE Variable (STATEV) layout, comprehensive inline documentation, static interface/dimension checks, and a self-test comparing algorithmic tangents to finite-difference approximations to flag discrepancies. After compilation and integration, element-level and simplified structural benchmarks (tension, shear, impact) are executed. Outputs are quantitatively compared against original experimental data and a manually implemented UMAT baseline using accuracy (R^2, MAE) and convergence metrics (Newton iterations, step-size reductions), forming a reproducible closed loop that substantiates physical fidelity and numerical robustness.

3.2 Construction of Domain Knowledge Base

To enable accurate LLM outputs for constitutive model development and UMAT implementation, we constructed a structured domain knowledge base (Table 1) through three tightly linked stages: data collection, data processing, and hybrid knowledge base assembly. In data collection, multi-source datasets were aggregated: (i) mechanical response data for representative engineering materials (metals, polymers, composites) under uniaxial tension, cyclic loading, creep, and stress relaxation across varied strain rates and temperatures; (ii) UMAT (and equivalent user material subroutine) specifications from major finite element platforms (Abaqus, ANSYS) [21], including interface argument standards, parameter passing rules, state variable management conventions, and consistent tangent modulus requirements; and (iii) canonical constitutive formulations (e.g., Johnson–Cook, Arrhenius/Zener–Hollomon type equations) together with their mathematical expressions and parameter calibration records.

Table 1: Knowledge base construction process.

Data Collection	Data Processing	Knowledge Base Construction
Interface Standard Collection	Mechanical Property Data Processing	Structured DB Building
Constitutive Model Collection	Interface Standard Data Processing	Unstructured DB Building
Material Property Data Collection	Constitutive Model Data Processing	Knowledge Index Construction

In the data processing stage, a multi-step quality control workflow was applied. Mechanical datasets underwent cleaning to remove anomalous tests, unit and format standardization, and feature extraction to identify yield points, hardening/softening slopes, creep/relaxation characteristics, and rate–temperature sensitivity indicators as structured descriptors. UMAT specifications were parsed from official documentation and validated examples to extract argument data types, dimensions, storage conventions, and interface invariants, forming a complete schema. Constitutive model equations were verified for mathematical completeness, and parameter sensitivity analyses were conducted to assess physical interpretability, influence ranges, and potential identifiability issues—supporting robust downstream model recommendation and calibration.

The final knowledge base adopts a hybrid storage architecture integrating a relational database for structured entities (constitutive parameters, UMAT interface schemas, material properties) and both a document store and a vector database for unstructured or semi-structured assets (code templates, technical manuals, simulation logs, extended test reports). A unified indexing and cross-reference layer links tabular records with embedded semantic representations, enabling cross-modal retrieval (formula → code template → interface specification → example dataset). This design supplies comprehensive, reliable domain knowledge to constrain LLM reasoning, enhance physical consistency, and support evidence-driven generation of constitutive models and executable UMAT implementations.

3.3 Retrieval Augmented Generation

To ensure that large language model (LLM) outputs for high-precision tasks such as constitutive modeling and UMAT implementation are both reliable and engineering-ready, we introduce a deeply optimized RAG framework (Fig. 2) [22]. Its core principle is to constrain the LLM's generative process in an evidence-driven manner. Given an input consisting of a stress–strain data summary, applicable strain-rate and temperature ranges, and platform constraints, the system first performs targeted retrieval against a structured domain knowledge base (D). This knowledge base spans yield functions, plastic flow rules, hardening laws, return-mapping algorithm templates, consistent tangent modulus derivations, Abaqus/UMAT interface specifications, neural network weight export procedures, and numerical robustness strategies [23].

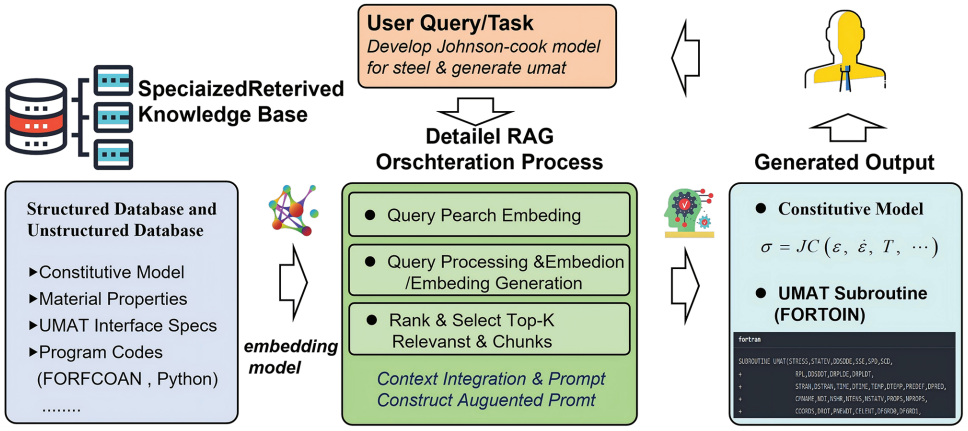

Figure 2: Principle of retrieval augmented generation.

3.3.1 Offline Retrieval Stage

To convert dispersed constitutive knowledge from manuals, papers, and historical experiments into an immediately accessible external memory, the system first performs unified segmentation of text, formulas,

and code, producing fragments of 256 tokens each. A hybrid "text–formula–code" encoder—implemented by parallel Language-Bert Sentence Encoder (LaBSE) and Code-Bidirectional Encoder Representations from Transformers (CodeBERT) streams, followed by linear projection—generates 1024-dimensional embeddings for every fragment, after which L2 normalization is applied to enforce metric consistency. An approximate nearest neighbor index is then built using FAISS (Facebook AI Similarity Search)-IVFPQ (Inverted File with Product Quantization), enabling sub-millisecond similarity queries at scale. For each fragment, structured metadata (material family, applicable temperature–strain rate range, knowledge category: yield criterion/flow rule/UMAT interface) is recorded in an auxiliary table to support multi-level filtering. This offline corpus indexing establishes the foundation for precise, low-latency semantic matching during subsequent online retrieval and prompt assembly.

3.3.2 Online Retrieval Stage

The system extracts standardized mechanical features (yield stress, hardening exponent, rate/temperature sensitivity) to form a compact query vector. A dual recall pipeline combines (i) vector Maximum Inner Product Search (MIPS) over the full embedding index (top-N semantic candidates) and (ii) rule-based filtering with hierarchical scoring on material family, temperature, and strain-rate coverage. Merged candidates undergo Maximal Marginal Relevance (MMR) de-redundancy; the top k = 5 physically consistent, semantically relevant evidence fragments are retained to condition downstream generation.

3.3.3 The Stage of Evidence-Conditioned Generation

To enable the model to generate executable constitutive equations and UMAT code under open-book (RAG) conditions, we fuse the user query and retrieved evidence into a specification-locked instruction prompt. The generated UMAT code is automatically validated by a rule-based checking module. This module: (1) parses the code to verify the presence and correct declaration of all required Abaqus UMAT arguments; (2) for analytical models, performs a symbolic check on the extracted yield/flow equations to ensure non-negative plastic dissipation; (3) examines the tangent-matrix computation for explicit symmetry enforcement. Any violation triggers a re-generation cycle where the specific error is fed back to the LLM with instructions for correction, iterating until all checks pass (max 3 iterations).

$$P\left(y\left|x\right.\right)=\sum_{z\in\ Top-k,\left(P\left(\cdot\left|x\right.\right)\right)}P_q\left(Z\left|x\right.\right)\prod_{i=1}^{N}P_\theta\left(y_i\left|x\right.,Z,y_{1:i-1}\right),\tag{1}$$

where $P(y|x)$ represents the conditional probability of the sequence y given the input x. The summation extends over the set z within the *Top-k* scoring latent variables according to $P(\bullet|x)$. $P_q(Z|x)$ is the probability of selecting the latent variable Z given the input x. The product runs from $i = 1$ to N, iterating through the sequence of tokens in y. $P_\theta(y_i|x, Z, y_{1:i-1})$ is the probability of generating token y_i conditioned on the input x, the latent variable Z, and all preceding tokens $y_{1:i-1}$.

3.3.4 Post-Event Self-Inspection Stage

To verify both numerical and physical correctness of generated results, the system performs an automated "dual closed-loop" self-check. In the numerical loop, the freshly generated UMAT is mounted on a single eight-node cubic element (single-element model) and subjected to five representative loading paths; two successive stress updates are executed, and a numerical consistent tangent D_{num} is computed via central differencing, then compared against the analytical tangent D_{alg} using the relative Frobenius norm error:

$$\delta = \left\| D_{a\lg} - D_{num} \right\|_F / \left\| D_{num} \right\|_F + \varepsilon \le \tau, \tag{2}$$

If the error is greater than 1%, the determination fails. In the physical loop, each increment is examined for non-negative dissipation power and bounded rotational invariance deviation; any violation triggers a "re-retrieve → re-generate" cycle, with at most N = 3 iterations. This dual closed-loop procedure operationalizes the static RAG knowledge base into a verifiable, correctable evidence chain, enabling the large model to assimilate fresh experimental data in real time while preserving thermodynamic consistency and numerical robustness, thereby providing a trustworthy technical foundation for automated constitutive model generation and subsequent Abaqus verification.

3.4 Framework Testing

3.4.1 Experimental Setup

This study employs OpenAI's GPT-4 model (gpt-4-turbo-2024-04-09) as the core computational agent, accessed via its official API. The model demonstrates exceptional capabilities in complex logical reasoning, code comprehension and generation, and multi-step task planning, making it particularly suitable for end-to-end tasks ranging from constitutive theory derivation to UMAT code implementation [24]. During the code generation phase, we designed a structured system prompt that explicitly defines the role, task, and constraints for the LLM. For instance, the prompt begins with: "You are an expert in computational solid mechanics and Abaqus UMAT development. Your task is to generate a UMAT subroutine for a given constitutive model. The code must be in Fortran syntax, include stress update logic, a consistent tangent matrix (DDSDDE), and manage state variables (STATEV). Thereby ensuring accuracy in professional terminology usage, mathematical model formulation, and programming conventions. During the code generation phase, we designed a structured system prompt that explicitly defines the role, task, and constraints for the LLM. For instance, the prompt begins with:

Role: *You are a senior expert in computational solid mechanics, material constitutive modeling, and Abaqus UMAT subroutine development.*
Task: *Generate a complete, compilable Abaqus UMAT subroutine in Fortran for a given constitutive model.*
Instructions:
1. The UMAT must comply with Abaqus UMAT interface specifications.
2. Required input/output variables: STRESS, STATEV, DDSDDE, SSE, SPD, SCD, etc.
3. Implement a stress update algorithm with implicit integration (return mapping).
4. Derive and implement the consistent tangent modulus (Jacobian matrix).
5. Include clear comments for each section: variable definitions, stress update, tangent calculation, and state variable management.
6. Ensure numerical stability: handle large strains, material nonlinearity, and convergence issues.
7. Output only the Fortran code, without explanatory text.
Provide the corresponding UMAT code.

3.4.2 Parameter Identification and Fitting

The LLM first extracts mechanical characteristics from the preprocessed experimental data, identifying key features such as the yield stress, hardening rate, strain-rate sensitivity coefficient, and temperature sensitivity indicators. It then retrieves, from the knowledge base, statistical distributions and plausible ranges of Johnson–Cook parameters for similar metallic materials (e.g., 304 stainless steel), and proposes physically reasonable initial intervals. Based on feature-matching scores, the LLM selects an initial parameter set for the first iteration. Given the constitutive form of the Johnson–Cook model, the framework automatically generates a Python fitting script and formulates a parameter-optimization problem, using the root-mean-square error (RMSE) between the predicted and measured stresses as the objective function.

Physics-informed neural network (PINN) constitutive models. For this branch, the framework automatically designs a fully-connected multilayer perceptron to map the mechanical state (strain, strain rate, temperature) to stress. The default architecture comprises 5 hidden layers (128, 128, 64, 64, 32 neurons) with Swish activation functions. To ensure physical admissibility, thermodynamic constraints are embedded as penalty terms in the loss function $L = L_{data} + \lambda L_{physics}$. The model is trained using the AdamW optimizer (initial learning rate 1×10^{-3}).

3.4.3 Test Results

This study successfully accomplished end-to-end, systematic verification of the proposed LLM-based framework for material constitutive modeling and UMAT generation. The evaluation employed a dynamic mechanical performance dataset of 304 stainless steel [25] under varying strain rates and temperatures as input, requiring the framework to execute in parallel two core tasks with markedly different characteristics: (i) parameter calibration and optimization of a traditional mechanism-based Johnson–Cook constitutive model, and (ii) architecture design and training of a data-driven physics-informed neural network (PINN) model. This dual-task design aimed to comprehensively assess the framework's adaptability to heterogeneous classes of constitutive representations, while also testing whether accurate UMAT user subroutines could be generated from the constructed models.

As shown in Fig. 3, the verification results demonstrate that the framework stably and reliably realizes a fully automated pipeline from experimental data preprocessing to final executable code generation. The system successfully completed the following key stages: multidimensional semantic parsing and mechanical feature extraction from experimental stress–strain data; intelligent model recommendation and inverse parameter identification; automatic embedding and verification of physics-based constraints; and generation of UMAT subroutines conforming to the stringent interface specifications of Abaqus. Ultimately, the framework produced two complete, compilable UMAT code files corresponding to the Johnson–Cook model and the PINN-based model, respectively, thereby substantiating the practicality and robustness of the proposed method in complex engineering contexts. Quantitative analyses of constitutive parameter fitting accuracy, generalization capability, and the numerical fidelity of the generated UMAT implementations will be presented and discussed in subsequent sections.

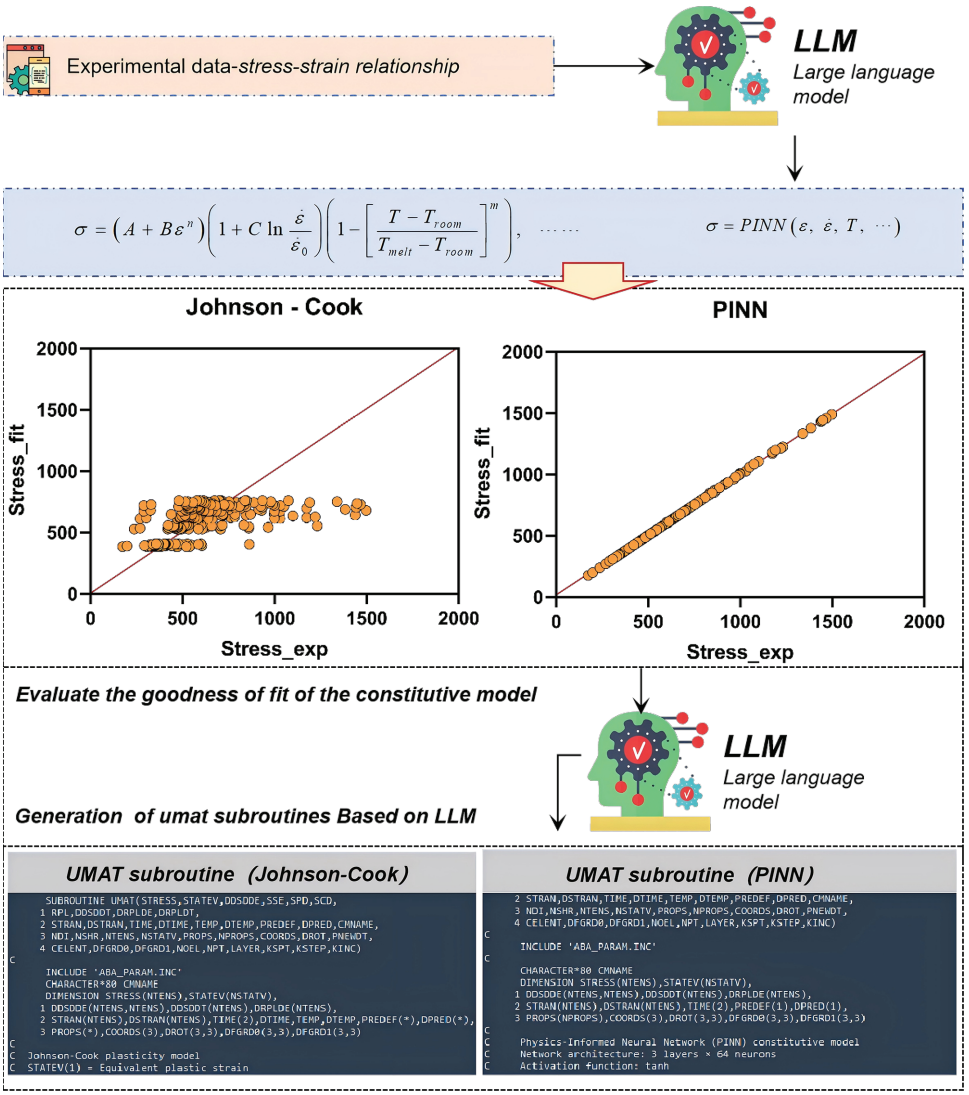

Figure 3: From experimental data to the full-process test result graph generated by UMAT.

4 Results and Discussion

4.1 The Accuracy of the Constitutive Model

Based on a systematic comparative study of constitutive parameter identification strategies, we constructed the Johnson–Cook (JC) constitutive model using both a traditional stepwise fitting procedure and an LLM-assisted approach, and additionally developed a physics-informed neural network (PINN) constitutive model under LLM guidance. As shown in Table 2, the parameters identified for the Johnson–Cook model via the LLM-assisted method fall within a reasonable range relative to those obtained by the conventional procedure. Specifically, the relative deviations of the key parameters-baseline strength (A), strain hardening modulus (B), strain hardening exponent (n), strain-rate sensitivity coefficient (C), and thermal softening exponent (m)—are all below 2%. This demonstrates that the LLM-assisted workflow can effectively replicate the parameter identification outcome of conventional methods for the Johnson–Cook model. The identified parameters exhibit strong numerical consistency with established fitting results and fall within empirically observed ranges for similar materials, which supports the numerical reliability

of the approach within the given constitutive framework. Collectively, these results validate the practical applicability of the LLM-assisted workflow for identifying material constitutive relations.

Table 2: Comparison of fitting parameters of JC constitutive models.

Parameters	LLM-JC	Traditional	Percentage Difference
A (MPa)	352.46	350	0.698%
B (MPa)	684.73	680	0.691%
C	0.0318	0.032	0.628%
n	0.428	0.43	0.467%
m	0.893	0.88	1.456%

Quantitative error analysis (Table 3) reveals pronounced differences in predictive accuracy among the four approaches. The LLM-PINN model exhibits superior precision: its mean absolute error (MAE), mean relative error (MRE), and Root Mean Square Error (RMSE) are markedly lower than those of the other two methods, and its coefficient of determination reaches 0.998, indicating near-perfect agreement with the experimental data. By contrast, the LLM-JC model and the traditionally stepwise fitted JC model show comparable accuracy, with MAE values of 84.3 and 86.7 and R^2 values of 0.892 and 0.885, respectively. Further examination indicates that the accuracy of the traditional JC model is highly sensitive to temperature and strain-rate variations. As illustrated in Fig. 4, its prediction error is acceptable under low-temperature and low-strain-rate conditions; however, accuracy degrades markedly as temperature and strain rate increase, reflected by a growing mean relative error. This degradation stems from the inability of constant-form parameters in the classical JC formulation to represent complex physical processes such as dynamic recrystallization at elevated temperatures and the coupled interactions of dislocation mechanisms at high strain rates. In contrast, the LLM-PINN model—benefiting from embedded physical constraints—maintains uniformly high accuracy across all operating regimes; notably, its maximum relative error in the high-temperature, high-strain-rate domain does not exceed 1.5%, substantially outperforming both JC-based variants. In summary, the traditional JC model exhibits clear accuracy limitations under extreme conditions; the LLM-JC approach offers moderate improvements in stability; and the LLM-PINN model, through integration of physical mechanisms with data-driven learning, achieves high-fidelity, domain-wide prediction, demonstrating significant methodological advantages. As shown in Table 3 and Fig. 5, the conventional JC model and its LLM-assisted calibrated variant exhibit a pronounced loss of accuracy under extreme loading conditions, corroborating the intrinsic theoretical limitations of the JC formulation. In contrast, the LLM–PINN model compensates for these deficiencies by learning directly from data.

Table 3: A comparison of the accuracy of each model.

Models	MAE (MPa)	MRE (%)	RMSE (MPa)	Coefficient of Determination (R^2)
LLM-JC	84.3	8.2	204.0	0.892
Traditional	86.7	8.9	200.8	0.885
LLM-PINN	3.2	0.6	4.8	0.998

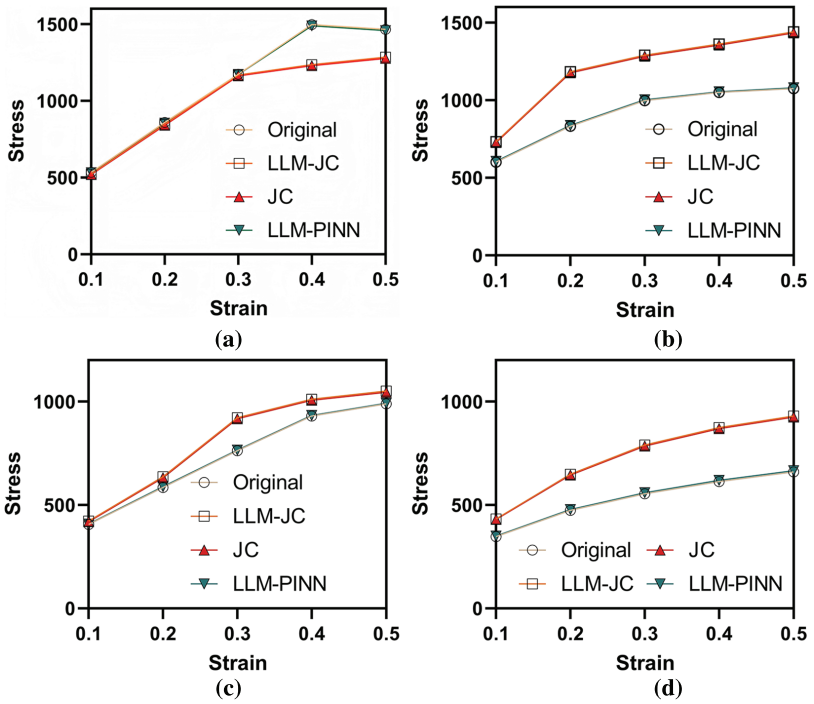

Figure 4: Comparison of the theoretical calculation results of the constitutive model and the actual stress results: (**a**) temperature is 20°C, strain rate is 0.001 s^{-1}; (**b**) temperature: 20°C, strain rate: 10 s^{-1}; (**c**) temperature: 100°C, strain rate: 0.01 s^{-1}; (**d**) temperature: 300°C, strain rate: 0.01 s^{-1}.

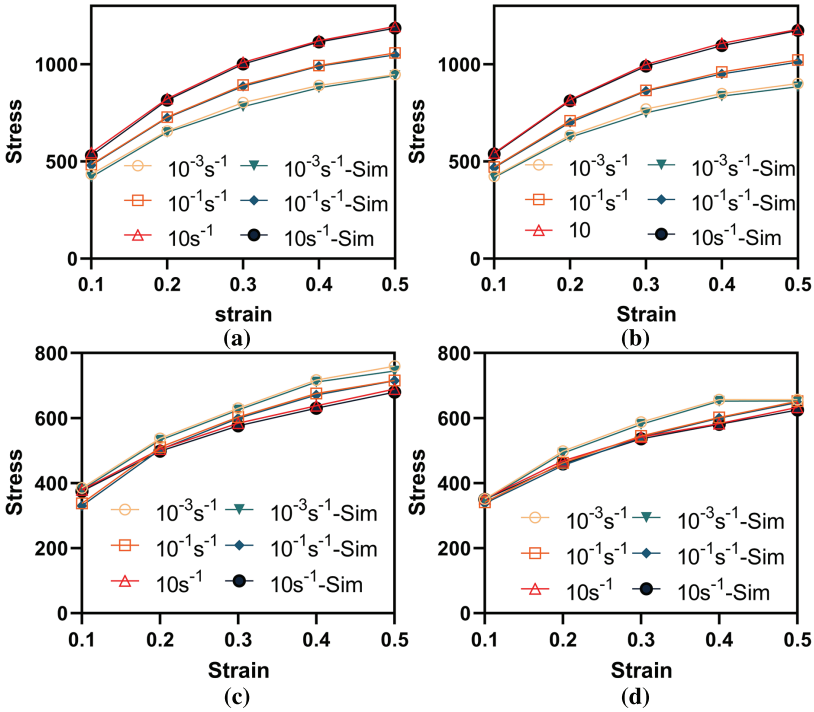

Figure 5: Comparison of abaqus simulation results with theoretical calculation results: (**a**) simulation results of LLM-JC at 200°C; (**b**) simulation results of LLM-JC at 300°C; (**c**) simulation results of LLM-PINN at 300°C; (**d**) simulation results of LLM-PINN at a temperature of 400°C.

4.2 The Accuracy of Umat Subroutine Construction

To quantitatively evaluate the overall effectiveness of the proposed framework, we conducted rigorous finite element simulations immediately after successful UMAT subroutine generation. The objective of this verification was to confirm that the automatically produced UMAT code can accurately reproduce the material's mechanical response under complex stress states by comparing simulation outputs with the predictions of the source constitutive models.

4.2.1 Numerical Setup

The generated UMAT subroutines were validated through single-element finite element simulations in Abaqus/Standard, which is the standard methodology for isolating and verifying constitutive integration algorithms. A single eight-node brick element (C3D8) was subjected to displacement-controlled uniaxial tension and simple shear loading paths that precisely matched the strain histories of the experimental calibration data. All simulations used a Static, General step with automatic time incrementation. The stress and strain at the integration point were extracted and compared point-by-point against the theoretical predictions from the constitutive models under identical strain paths. After obtaining the simulation datasets, comparative verification against the source constitutive models was performed. Finite element stress–strain curves were overlaid with theoretical predictions from the constitutive equations under identical conditions to enable intuitive visual agreement assessment. Building on this, quantitative metrics such as root mean square error (RMSE) and coefficient of determination (R^2) were employed to characterize deviations between simulation results and model predictions, enabling objective evaluation of the performance of different constitutive models.

4.2.2 Accuracy Analysis

The core objective of this study is to validate the accuracy and reliability of UMAT subroutines automatically generated by a large language model (LLM) when deployed in finite element simulations. As shown in Fig. 5, simulation results obtained using the LLM-JC UMAT and the LLM-PINN UMAT are compared against the corresponding theoretical response curves computed directly from their constitutive equations. The visual overlays under a uniaxial tension loading path show near-complete superposition, indicating excellent agreement between simulation predictions and theoretical expectations. To quantify this consistency, we computed the root mean square error (RMSE) and coefficient of determination (R^2) between the simulated stress–strain sequences and the theoretical curves. Both models achieved high accuracy, with RMSE values below 8 MPa and coefficients of determination (R^2) exceeding 0.98. These quantitative metrics confirm that deviations between the finite element predictions and analytical/theoretical responses are minimal, demonstrating the high numerical fidelity of the automatically generated UMAT implementations.

The experimental findings establish that the LLM-generated UMAT subroutines are numerically accurate: each subroutine successfully translates the core mathematical structure of the constitutive model (Johnson–Cook formulation or PINN-based neural representation) into a precise stress update algorithm, preserving the correctness of stress integration within nonlinear finite element iterations and yielding responses that closely track theoretical predictions. Moreover, this result closes the logical loop initiated earlier in the study: we previously verified that the LLM-constructed constitutive equations achieve high fitting accuracy with respect to experimental data; here, we further show that code automatically synthesized from those equations is reliable in simulation practice. Together, these outcomes substantiate the feasibility of using large language models as end-to-end automated modeling tools, bridging "constitutive theory" and "computational implementation", significantly enhancing research efficiency in computational mechanics and the accuracy of production-grade code.

5 Conclusion

This study innovatively proposes an integrated large language model (LLM)-based framework for automatic constitutive modeling and UMAT subroutine generation. The framework operates along an intelligent "data-to-simulation" pipeline that tightly fuses material constitutive theory, machine learning methodologies, and finite element implementation techniques. Through coordinated modular interaction, it achieves full automation from experimental data preprocessing and feature extraction, to constitutive model construction, and finally to UMAT code synthesis. Critically, the LLM-based framework automates the development of both traditional constitutive models and physics-informed neural networks, PINN models offer higher accuracy under extreme conditions (e.g., high temperature/strain rate), while traditional models provide speed and interpretability, users can intelligently select the appropriate model based on their specific simulation needs. The automatically generated UMAT subroutines pass rigorous verification, and their finite element simulation outputs exhibit excellent agreement with theoretical predictions ($R^2 > 0.98$), confirming numerical accuracy and reliability. Overall, the proposed framework alleviates long-standing bottlenecks of traditional workflows-insufficient model adaptability, difficult parameter identification, and the complexity of robust UMAT implementation—thereby providing a practical pathway toward end-to-end, engineering-ready intelligent constitutive modeling.

Acknowledgement: The authors gratefully acknowledge the financial and infrastructural support provided by the National Natural Science Foundation of China, the Chongqing Science and Technology Committee, the Foundation of National Key Laboratoryof Computational Physics, and the Chongqing Municipal Education Commission. We also thank laboratory staff and technical support personnel for assistance with data acquisition and computing resources.

Funding Statement: This research was funded by the National Natural Science Foundation of China, grant number 52405341; Foundation of National Key Laboratoryof Computational Physics, grant number 6142A05QN24012; Chongqing Science and Technology Committee, grant number CSTB2023NSCQ-MSX0363; The Science and Technology Research Program of Chongqing Municipal Education Commission, grant number KJQN202301117.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Lusheng Wang and Yuchuan Gu; methodology, Yuchuan Gu and Changgeng Li; software, Yanhong Peng; verification, Yanhong Peng; formal analysis, Yuchuan Gu; investigation, Lusheng Wang; resources, Lusheng Wang and Jun Ding; writing—original draft preparation, Yuchuan Gu; writing—review and editing, Yuchuan Gu; supervision, Lusheng Wang; project administration, Lusheng Wang and Shaojie Gu; funding acquisition, Lusheng Wang. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: Not applicable.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Simo JC, Taylor RL. Consistent tangent operators for rate-independent elastoplasticity. Comput Meth Appl Mech Eng. 1985;48(1):101–18. doi:10.1016/0045-7825(85)90070-2.
2. Roters F, Eisenlohr P, Hantcherli L, Tjahjanto DD, Bieler TR, Raabe D. Overview of constitutive laws, kinematics, homogenization and multiscale methods in crystal plasticity finite-element modeling: theory, experiments, applications. Acta Mater. 2010;58(4):1152–211. doi:10.1016/j.actamat.2009.10.058.
3. Karniadakis GE, Kevrekidis IG, Lu L, Perdikaris P, Wang S, Yang L. Physics-informed machine learning. Nat Rev Phys. 2021;3(6):422–40. doi:10.1038/s42254-021-00314-5.

4. Eisenträger S, Maurer L, Juhre D, Altenbach H, Eisenträger J. Implementation of isotropic hyperelastic material models: automatic code generation in MATLAB. Acta Mech. 2025;236(6):3413–44. doi:10.1007/s00707-025-04277-x.
5. Eisenträger S, Maurer L, Juhre D, Altenbach H, Eisenträger J. Implementation of isotropic hyperelastic material models: a »template« approach. Acta Mech. 2025;236(3):1899–934. doi:10.1007/s00707-025-04235-7.
6. Logg A, Mardal KA, Wells G. Automated solution of differential equations by the finite element method: the FEniCS book. Berlin/Heidelberg, Germany: Springer; 2012. doi:10.1007/978-3-642-23099-8.
7. Korelc J, Wriggers P. Automation of finite element methods. Cham, Switzerland: Springer International Publishing; 2016. doi:10.1007/978-3-319-39005-5.
8. Linka K, Kuhl E. Best-in-class modeling: a novel strategy to discover constitutive models for soft matter systems. Extreme Mech Lett. 2024;70:102181. doi:10.1016/j.eml.2024.102181.
9. Peirlinck M, Linka K, Hurtado JA, Holzapfel GA, Kuhl E. Democratizing biomedical simulation through automated model discovery and a universal material subroutine. Comput Mech. 2025;75(6):1703–23. doi:10.1007/s00466-024-02515-y.
10. Peirlinck M, Hurtado JA, Rausch MK, Tepole AB, Kuhl E. A universal material model subroutine for soft matter systems. Eng Comput. 2025;41(2):905–27. doi:10.1007/s00366-024-02031-w.
11. Schommartz JO, Klein DK, Alzate Cobo JC, Weeger O. Physics-augmented neural networks for constitutive modeling of hyperelastic geometrically exact beams. Comput Meth Appl Mech Eng. 2025;435(1):117592. doi:10.1016/j.cma.2024.117592.
12. Kalina KA, Brummund J, Kästner M. A physics-augmented neural network framework for finite strain incompressible viscoelasticity. arXiv:2511.02959. 2025.
13. Aldakheel F, Elsayed ES, Zohdi TI, Wriggers P. Efficient multiscale modeling of heterogeneous materials using deep neural networks. Comput Mech. 2023;72(1):155–71. doi:10.1007/s00466-023-02324-9.
14. Le Clézio H, Karapiperis K, Kochmann DM. Nonlinear two-scale beam simulations accelerated by thermodynamics-informed neural networks. Extreme Mech Lett. 2024;73(10):102260. doi:10.1016/j.eml.2024.102260.
15. Gaur H, Khidhir B, Manchiryal RK. Solution of structural mechanic's problems by machine learning. Int J Hydromechatron. 2022;5(1):22. doi:10.1504/ijhm.2022.122459.
16. Ladevèze P, Chamoin L. Data-driven material modeling based on the constitutive relation error. Adv Model Simul Eng Sci. 2024;11(1):23. doi:10.1186/s40323-024-00279-x.
17. Park H, Cho M. Multiscale constitutive model using data-driven yield function. Compos Part B Eng. 2021;216(2):108831. doi:10.1016/j.compositesb.2021.108831.
18. Jia H, Yuan Z. A thermodynamically consistent viscoelastic with rate-dependent damage constitutive model. Int J Solids Struct. 2023;269(2):112198. doi:10.1016/j.ijsolstr.2023.112198.
19. Capuano G, Rimoli JJ. Smart finite elements: a novel machine learning application. Comput Meth Appl Mech Eng. 2019;345(4):363–81. doi:10.1016/j.cma.2018.10.046.
20. Asensio G, Moreno C. Linearization and return mapping algorithms for elastoplasticity models. Int J Numer Methods Eng. 2003;57(7):991–1014. doi:10.1002/nme.718.
21. Liu PP, Shen B. Implementation of double scalar elastic damage constitutive model in UMAT interface. Comput Concr. 2021;27(2):153 doi: 10.1590/1679-78256607.
22. Church KW, Sun J, Yue R, Vickers P, Saba W, Chandrasekar R. Emerging trends: a gentle introduction to RAG. Nat Lang Eng. 2024;30(4):870–81. doi:10.1017/s1351324924000044.
23. Tushar Kumar S, Chakraborty S. Fusion-based constitutive model (FuCe): toward model-data augmentation in constitutive modeling. Int J Mech Syst Dyn. 2025;5(1):86–100. doi:10.1002/msd2.70005.
24. Liu D, Wang C, Gao P, Zhang R, Ma X, Meng Y, et al. 3DAxisPrompt: promoting the 3D grounding and reasoning in GPT-4o. Neurocomputing. 2025;637:130072. doi:10.1016/j.neucom.2025.130072.
25. Venugopal S, Mannan SL, Prasad YVRK. Optimization of cold and warm workability in 304 stainless steel using instability maps. Metall Mater Trans A. 1996;27(1):119–26. doi:10.1007/BF02647752.