# Cascading Failure Amplification in Multi-Step LLM Reasoning: Operational Definitions, Trace-Level Diagnostics, and Intervention Analysis

**Anonymous Authors**[1]

## Abstract

Chain-of-thought prompting enables multi-step reasoning in large language models, yet errors at intermediate steps can propagate and amplify. We introduce the *Amplification Factor* (AF), a risk-ratio metric quantifying this cascading failure. Across three model families on GSM8K (14,558 step-level records, 2,119 traces), we measure $\mathrm{AF} = 7.79$ for GPT-4o-mini [95% CI: 5.46, 11.24], $\mathrm{AF} = 3.29$ for Llama-3.1-8B, and $\mathrm{AF} = 7.01$ for Qwen-2.5-72B, establishing cascading amplification as a general phenomenon: roughly half of all post-error steps contain new errors regardless of model. A controlled error-injection experiment (504 trials, 2 models) reveals that positional vulnerability *increases* with step position—late-step damage reaches 0.93 vs. 0.59 for earliest-step errors (Spearman $\rho = 0.387$, $p < 0.001$). Among four inference-time interventions (80 problems), cross-verification reduces the cascade rate from 24% to 9% ($p = 0.023$), while step-level self-checking is harmful, reducing accuracy from 52% to 36%. These results provide operational failure-mode definitions and actionable guidance for allocating verification compute.

## 1. Introduction

Chain-of-thought (CoT) prompting (Wei et al., 2022) decomposes complex problems into sequential intermediate steps, enabling strong multi-step reasoning. This sequential dependency introduces a vulnerability: an error at any step can corrupt all subsequent steps, transforming a local mistake into a global failure.

The intuition that errors "snowball" is well established. Zhang et al. (2023) showed that LLM hallucinations compound across chains; Dziri et al. (2023) demonstrated multiplicative accuracy decay with compositional depth; and Gan et al. (2025) formalized the relationship to reasoning correctness. Process reward models (Lightman et al., 2024; Uesato et al., 2022; Wang et al., 2024; Khalifa et al., 2025) provide step-level supervision, implicitly acknowledging that intermediate errors matter. Yet no prior work has provided a precise measurement of how much more likely an error becomes *after* a preceding error has occurred, nor systematically compared interventions through the lens of cascading failure rates.

This gap matters because: (1) without an operational metric, claims about error propagation remain non-reproducible; (2) verification compute allocation (Snell et al., 2024) requires knowing *where* errors cascade; and (3) CoT deployment in agentic systems amplifies consequences of undetected failures.

We make three contributions:

1. **Operational metric.** We define the *Amplification Factor* (AF), a conditional risk ratio quantifying cascading amplification in a model- and dataset-agnostic manner (§3).
2. **Multi-model measurement.** On GSM8K, we establish $\mathrm{AF} = 7.79$ for GPT-4o-mini, $\mathrm{AF} = 3.29$ for Llama-3.1-8B, and $\mathrm{AF} = 7.01$ for Qwen-2.5-72B (§4.1).
3. **Positional vulnerability and intervention analysis.** Controlled injection shows late-step errors are more damaging (§4.2); cross-verification significantly reduces cascading while self-checking is harmful (§4.3).

## 2. Related Work

**Chain-of-thought failure modes.** CoT prompting (Wei et al., 2022) is susceptible to unfaithful explanations (Turpin et al., 2023; Lanham et al., 2023; Arcuschin et al., 2025), greedy error compounding (Saparov & He, 2023), and systematic degradation under chained evaluation (Miner et al.,

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

2024).

**Error propagation and compounding.** Zhang et al. (2023) characterized snowballing; Dziri et al. (2023) showed $\sim p^n$ accuracy decay; Gan et al. (2025) formalized the connection to reasoning correctness. This is theoretically grounded in the DAgger framework (Ross et al., 2011), predicting $O(T^2)$ error accumulation. Concurrent work on positional fragility—including ASCoT-style position-aware correction, algebraic stress-test frameworks revealing chain-length bottlenecks, and quantization-induced process failures—complements structured perturbation taxonomies and adversarial step-injection methods. Our Experiment 2 provides independent causal evidence, while AF offers a diagnostic consonant with uncertainty-propagation formalisms. AF can be situated within survival analysis or HMM frameworks ($c_k$ as a "contaminated" state, AF as the state-dependent hazard ratio); history-compression methods (e.g., Markov Chain-of-Thought) could be evaluated through this lens.

**Process supervision and verification.** PRMs (Uesato et al., 2022; Lightman et al., 2024; Wang et al., 2024; Khalifa et al., 2025) and premise-augmented chains (Mukherjee et al., 2025) allocate verification resources across steps. The broader landscape—best-of-$N$ selection, calibrated PRMs (including MCNIG-based scalable supervisors), compact cross-model verifiers (e.g., UHeads-style lightweight step judges), structured verification grounded in external knowledge, and retrospective correction (e.g., DiffCoT-style step revision)—provides rich methods, though LLM-based verifiers degrade on long chains, suggesting AF estimates from such judges may be conservative. Without quantifying how cascading varies with position, allocation cannot be optimally targeted.

**Self-correction and inference-time interventions.** LLMs cannot self-correct without external feedback (Huang et al., 2024; Kamoi et al., 2024). Self-consistency (Wang et al., 2023) marginalizes over reasoning paths; tree-of-thoughts (Yao et al., 2023) explores branching structures; and scaling test-time compute can outperform scaling parameters (Snell et al., 2024). Li et al. (2024) evaluated mathematical error identification. Our analysis connects these approaches to cascading failure rates via a unified metric. Compared to hop-based failure taxonomies, AF offers a compact scalar focusing on conditional post-error risk; computing AF per latent state or quantization setting could isolate where cascade risk spikes.

# 3. Method

## 3.1. The Amplification Factor

Consider a reasoning trace $\tau = (s_1, \ldots, s_T)$ for problem $q$. Let $e_k \in \{0, 1\}$ indicate whether step $s_k$ contains an error, and define the *prior error indicator* $c_k = \mathbf{1}[\exists\, j < k : e_j = 1]$. The *Amplification Factor* is:

$$\mathrm{AF} = \frac{P(e_k = 1 \mid c_k = 1)}{P(e_k = 1 \mid c_k = 0)} = \frac{p_{\text{post}}}{p_{\text{clean}}}, \tag{1}$$

where probabilities are over all step records with $k > 0$.

**Design considerations.** Because $c_k$ is monotonically non-decreasing, pooling across positions introduces compositional confounding. We address this via: (1) a GEE regression including step index $k$ as a covariate, and (2) position-stratified $\mathrm{AF}_k$ estimates (Appendix F). Alternative conditioning on $e_{k-1}$ or $\sum_{j<k} e_j$ yields qualitatively similar results ($\mathrm{AF} > 5$).

**Labeling semantics.** Our verifier assesses *local* step correctness: whether the computation is correct given its stated inputs. A step applying correct arithmetic to a previously corrupted value is labeled "correct"—the error in its input is attributed to the earlier step. AF thus measures the rate at which *new* local errors emerge after prior contamination, not the propagation of corrupted quantities. A global-correctness criterion would yield different (likely higher) AF values (§6).

## 3.2. Experiment 1: Observational Cascading Analysis

From the GSM8K test set (Cobbe et al., 2021), we generated up to 8 independent CoT traces per problem for 100 problems across three models: GPT-4o-mini, Llama-3.1-8B-Instruct, and Qwen-2.5-72B-Instruct. After filtering to traces that produced parseable multi-step chains (i.e., traces with clearly delineated sequential reasoning steps and a final numerical answer), we retained 2,119 traces and 14,558 step-level records. The number of retained traces per problem ranged from 1 to 8 (GPT-4o-mini: 589 traces across 94 qualifying problems; Llama-3.1-8B: 799/100; Qwen-2.5-72B: 731/100). This filtering may introduce selection bias by excluding terse or unstructured correct solutions; we note this in §6.

Each step was verified by a separate GPT-4o-mini call using the local-correctness criterion. We note that using the same model family for generation and verification risks correlated errors; we discuss this and the need for cross-family or human calibration in §6. Final answers were checked against GSM8K gold answers. The verifier achieved 94.2% agreement on a derived ground-truth subset (steps within fully correct traces), though this is not a full reliability study. Chain lengths range from 2 to 20 steps (mean 7.0 for GPT-4o-mini), distributed roughly evenly across short (2–4 steps,

31%), medium (5–7, 31%), and long (≥8, 38%) chains (see Appendix A for chain-length distributions).

We fit a GEE logistic regression with both $c_k$ and step index $k$ as covariates, exchangeable working correlation, and clustering by problem. We cluster at the problem level (traces per problem share underlying difficulty); a nested problem→trace design or trace-level block bootstrap would better capture within-trace serial dependence (§6).

### 3.3. Experiment 2: Controlled Error Injection

We injected a single arithmetic error at controlled position $p \in \{0, 1, 2, 3, 4\}$ (absolute step index) into verified-correct reference solutions for 60 problems. Reference solutions were obtained by generating CoT traces and retaining only those whose final answer matched the GSM8K gold answer. Injection replaced the numerical result of step $p$ with a value differing by a fixed perturbation ($\pm$10–50% of the correct value), preserving the verbal structure. Two models were tested (GPT-4o-mini and Llama-3.1-8B-Instruct), yielding 504 trials (252 per model). Per-step damage $D(p)$ is the fraction of incorrect continuation steps after injection at position $p$, where continuation steps are in the model's own output and correctness is judged by the same local verifier. We note two scope limitations: (1) only numerical errors are injected—structural or logical errors may differ; (2) injection positions are absolute indices (0–4), not relative within the chain, so results primarily characterize the first five steps rather than proportional positions in longer chains (§6).

### 3.4. Experiment 3: Intervention Comparison

We evaluated four strategies on 80 GSM8K problems using GPT-4o-mini:

1. **Baseline CoT**: Single generation + verification.
2. **Step self-check**: After each step, the model is prompted: "*Review the step above. Is it correct? If not, fix it and continue.*"
3. **Self-consistency** ($K = 5$): Five independent generations with majority voting (Wang et al., 2023).
4. **Cross-verification**: One generation, then a separate call verifies each step against the problem statement (58 problems completed).

Cascade rate is the fraction of problems where the final answer is wrong *and* at least one intermediate error precedes the final error. API calls are an imperfect compute proxy; token budgets are in Appendix F.

*Table 1.* Cascading amplification across three model families on GSM8K. All statistics computed on step records with $k > 0$.

| | **GPT-4o -mini** | **Llama-3.1 -8B** | **Qwen-2.5 -72B** |
|---|---|---|---|
| Traces | 589 | 799 | 731 |
| Step records | 4,138 | 5,502 | 4,918 |
| $p_{\text{post}}$ | 0.498 | 0.485 | 0.542 |
| $p_{\text{clean}}$ | 0.064 | 0.147 | 0.077 |
| AF | **7.79** | **3.29** | **7.01** |
| $\Delta$ | 0.434 | 0.338 | 0.465 |
| $\chi^2$ | 878.2 | 635.0 | 1059.2 |
| OR | 14.51 | 5.45 | 14.13 |
| Cascade ratio | 35.7% | 48.7% | 24.4% |

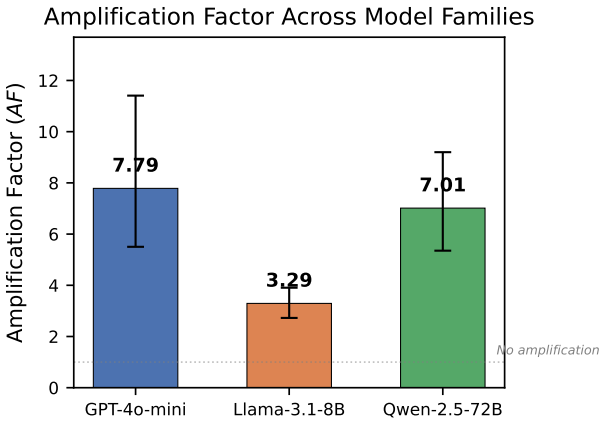


*Figure 1.* Amplification Factor (AF) with 95% bootstrap CIs across three model families. All models show AF $\gg 1$, confirming cascading amplification as a general phenomenon.

## 4. Results

### 4.1. Experiment 1: Cascading Amplification Across Models

Table 1 shows that cascading amplification is consistent across all three model families: AF ranges from 3.29 to 7.79 (all $\chi^2 > 600$, $p < 10^{-100}$). Strikingly, $p_{\text{post}} \approx 0.5$ across all models—roughly half of all post-error steps contain new errors regardless of model family or scale (Figure 2). The lower AF for Llama-3.1-8B reflects its higher baseline error rate ($p_{\text{clean}} = 0.147$), not weaker cascading; the absolute risk difference remains large ($\Delta = 0.338$).

For GPT-4o-mini, the GEE logistic regression with step-index covariate yields position-adjusted OR $= 14.51$ [95% CI: 10.49, 20.08]; bootstrap 95% CI for AF: [5.46, 11.24] (Figure 1). The OR (logistic link) exceeds AF (risk ratio) because the outcome is not rare ($p_{\text{post}} = 0.498$); we report AF as the primary metric and OR for regression comparability.

**Stratified analyses.** The pooled AF is a corpus-level sum-

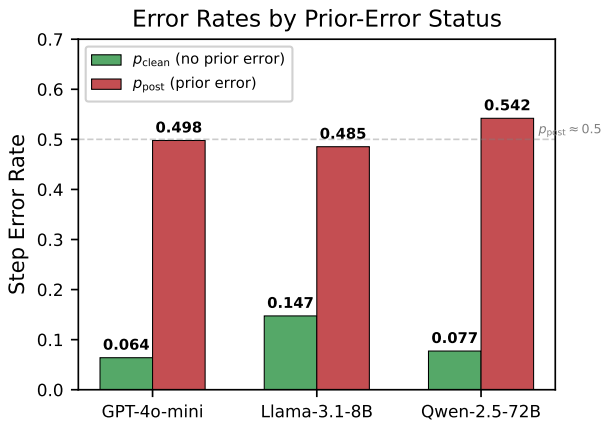


*Figure 2.* Step error rates conditioned on prior-error status. $p_{\text{post}} \approx 0.5$ is an invariant across all three models, while $p_{\text{clean}}$ varies with model capability.

*Table 2.* Per-step damage $D(p)$ by injection position (504 trials, 2 models, 60 problems). $\pm$ denotes standard error.

| **Position** $p$ | **GPT-4o-mini** | **Llama-3.1-8B** |
|---|---|---|
| 0 (earliest) | $0.48 \pm 0.05$ | $0.71 \pm 0.04$ |
| 1 | $0.54 \pm 0.05$ | $0.76 \pm 0.04$ |
| 2 | $0.68 \pm 0.04$ | $0.89 \pm 0.03$ |
| 3 | $0.85 \pm 0.04$ | $0.93 \pm 0.03$ |
| 4 (latest) | $0.86 \pm 0.04$ | $1.00 \pm 0.00$ |

mary; stratification reveals important structure. By *chain length*, AF increases monotonically for GPT-4o-mini: 4.95 for short chains (2–4 steps), 7.34 for medium (5–7), and 8.14 for long ($\geq$8), consistent with multiplicative error accumulation under the $O(T^2)$ DAgger bound (Ross et al., 2011) (see Appendix B for all three models). Under a simple Markov assumption where each step independently errs with probability $p$, the expected fraction of contaminated steps grows as $1-(1-p)^k$; the observed $\text{AF}_k$ pattern exceeds this baseline, suggesting non-trivial error-state dependence beyond memoryless compounding. By *step position*, $\text{AF}_k$ increases from $\sim$10 at step 1 to $>$20 by steps 7–9 (Figure 3; bootstrap CIs in Appendix F). This occurs because $p_{\text{clean}}$ decreases at later positions (surviving clean contexts are from easier sub-problems) while $p_{\text{post}}$ remains $\sim$0.5. We caution that the position-stratified estimates have wide CIs at later steps and that claims of monotone increase across strata are descriptive rather than formally tested for trend.

The cascade ratio—fraction of final-answer errors preceded by intermediate errors—is 35.7% for GPT-4o-mini, 48.7% for Llama-3.1-8B, and 24.4% for Qwen-2.5-72B.

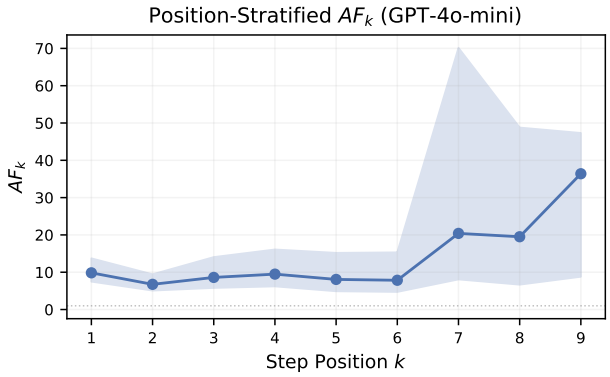


*Figure 3.* Position-stratified $\text{AF}_k$ for GPT-4o-mini with 95% bootstrap CIs. Amplification intensifies at later step positions because $p_{\text{clean}}$ shrinks while $p_{\text{post}} \approx 0.5$.

*Table 3.* Intervention comparison (GPT-4o-mini on GSM8K). Brackets denote 95% Wilson CIs. $N$ varies because cross-verification completed on 58/80 problems. $p_{\text{adj}}$: two-sided Fisher vs. baseline, Holm–Bonferroni corrected.

| **Intervention** | $N$ | **Acc (%)** | **Casc. (%)** | $p_{\text{adj}}$ |
|---|---|---|---|---|
| Baseline CoT | 80 | 52 [42, 63] | 24 [16, 34] | — |
| Step self-check | 80 | 36 [27, 47] | 30 [21, 41] | 0.95 |
| Self-consistency | 80 | 44 [33, 55] | 25 [17, 35] | 1.00 |
| Cross-verification | 58 | 45 [33, 58] | **9** [4, 19] | **0.070** |

### 4.2. Experiment 2: Late-Step Errors Are More Damaging

Per-step damage increases monotonically with injection position for both models (Table 2, Figure 4; overall Spearman $\rho = 0.387$, $p < 0.001$). We note that the Spearman test is computed over individual trials (not the five position means), providing adequate power despite the small number of discrete positions; a regression-based trend test (e.g., ordinal logistic regression of $D$ on $p$) would be a more principled alternative and is consistent with the monotone pattern. Llama-3.1-8B shows consistently higher damage at every position and reaches $D = 1.00$ at position 4 (complete failure to recover), while GPT-4o-mini retains some recovery capacity even at late positions. This suggests that larger or more capable models may have greater implicit error-correction ability, though both exhibit the same monotonic positional vulnerability pattern.

### 4.3. Experiment 3: Cross-Verification Reduces Cascading; Self-Check is Harmful

The intervention results (Table 3) reveal that cross-verification is the most effective cascade-reduction strategy, lowering the cascade rate from 24% to 9% (raw Fisher $p = 0.023$; Holm–Bonferroni adjusted $p = 0.070$,

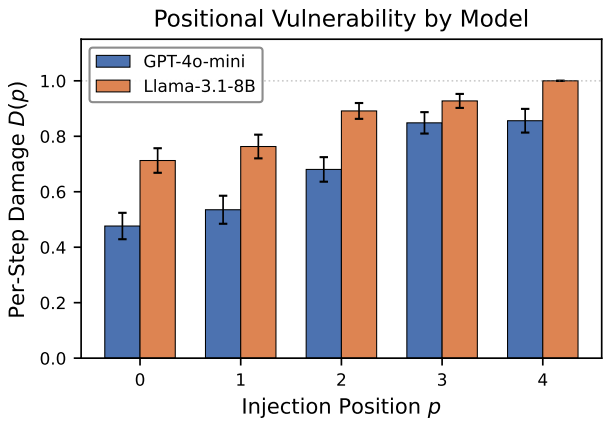


*Figure 4.* Per-step damage $D(p)$ by injection position for both models. Late-step errors are consistently more damaging, and Llama-3.1-8B shows higher damage at every position. Error bars denote standard error.

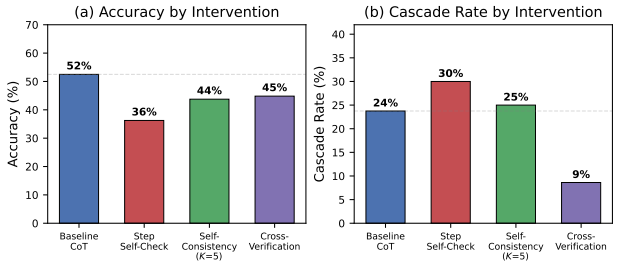


*Figure 5.* Intervention comparison. **(a)** Accuracy: self-checking reduces accuracy below baseline. **(b)** Cascade rate: cross-verification achieves the lowest cascade rate (9% vs. 24% baseline). Dashed lines indicate baseline levels.

marginally significant). Among the 32 problems where cross-verification produced an incorrect answer, only 4 (12.5%) exhibited cascading, compared to 8/38 (21.1%) for the baseline—indicating that the external verification step catches intermediate errors before they propagate. Notably, cross-verification reduces cascading without improving overall accuracy (45% vs. 52%), suggesting that cascade reduction and accuracy improvement are partially independent objectives; cross-verification's benefit is specifically in error-propagation prevention.

Self-consistency ($K = 5$) does not significantly reduce cascading (25% vs. 24%, $p = 1.00$; Figure 5), though it provides a modest accuracy improvement (44% vs. 52%, not significant). This contrasts with the expectation that majority voting would marginalize out cascade chains; evidently, when problems are sufficiently difficult to produce consistent wrong answers across multiple traces, cascading persists in the voted output. Among the 45 problems where SC produced an incorrect voted answer, 12 (26.7%) exhibited

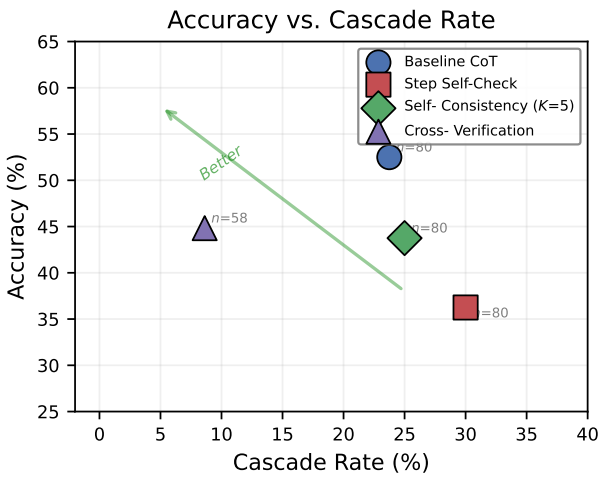


*Figure 6.* Accuracy vs. cascade rate for each intervention. The ideal direction (lower cascade, higher accuracy) is indicated by the green arrow. Cross-verification uniquely reduces cascade rate.

cascading in the trace corresponding to the voted answer. We note that this measures cascade in the *voted-answer trace* only; computing AF over all $K$ individual traces (irrespective of voting outcome) would provide a more direct per-step analysis of whether ensembling reduces conditional error rates, but requires step-level labels for all $K$ traces, which our current pipeline does not retain.

The most striking negative result is that step-level self-checking is *harmful*: accuracy drops from 52% to 36% ($p = 0.056$, marginal) and the cascade rate increases from 24% to 30%. The self-check prompt induces the model to second-guess correct steps, introducing new errors that cascade—a *correction-that-corrupts* failure mode consistent with Huang et al. (2024). The tradeoff between accuracy and cascade rate across interventions is visualized in Figure 6: cross-verification occupies the Pareto-optimal region (lowest cascade rate), while self-checking occupies the worst region (lowest accuracy, highest cascade rate).

## 5. Discussion

**Cascading amplification is strong, general, and scale-dependent.** The measured AF of 7.79 (GPT-4o-mini) establishes cascading as a dominant failure mechanism: $p_{\text{post}} \approx 0.5$ across all three models, meaning half of all post-error steps are themselves erroneous. The lower AF for Llama-3.1-8B (3.29) reflects its higher baseline error rate ($p_{\text{clean}} = 0.147$), not weaker cascading—the absolute risk difference ($\Delta = 0.338$) remains large. AF increases with chain length (4.95 for short chains, 8.14 for long), consistent with the DAgger $O(T^2)$ error accumulation bound (Ross et al., 2011). The observed $\text{AF}_k$ profile exceeds what a sim-

ple memoryless (Markov) model predicts, suggesting that error states create non-trivial dependencies—the model's reasoning quality degrades beyond the independent per-step failure rate once contamination is present.

**Differential recoverability.** Late-step errors are more damaging per step than early-step errors, and this effect is stronger for the smaller model: Llama-3.1-8B reaches $D = 1.00$ at position 4, while GPT-4o-mini retains partial recovery capacity ($D = 0.86$). This suggests that larger models have greater implicit error-correction ability—they can re-derive needed quantities from the problem statement rather than relying exclusively on corrupted prior steps—but this capacity diminishes as the chain progresses and commitment to the solution trajectory increases. An important extension would decompose $\mathrm{AF}$ by error type (arithmetic vs. reasoning vs. formatting) and by step function (setup vs. computation vs. conclusion) to identify which error categories most readily spawn downstream failures and guide targeted verifier design.

**Why cross-verification works and self-checking fails.** Cross-verification succeeds because an independent verification call, given the original problem statement as ground truth, can identify and flag intermediate errors before they propagate. Self-checking fails because LLMs lack metacognitive capacity to distinguish correct from incorrect steps (Huang et al., 2024; Kamoi et al., 2024), and the self-check prompt biases toward unnecessary modifications. Self-consistency's failure to reduce cascading suggests that when problems are hard enough to produce consistent errors across traces, majority voting cannot rescue the failure mode. A natural next step is comparing cross-verification against PRM-guided best-of-$N$ under matched token budgets. A compute-normalized tradeoff analysis—plotting cascade rate and accuracy jointly against total tokens consumed—would clarify whether cross-verification's cascade benefit justifies its compute overhead relative to the simpler baseline.

**Implications for verification design.** Current PRMs (Lightman et al., 2024; Wang et al., 2024; Khalifa et al., 2025) apply uniform step verification. Our results suggest non-uniform allocation: later steps, where $\mathrm{AF}_k$ is highest and recovery is hardest, warrant disproportionate verification depth. The $\mathrm{AF}$ metric could serve as a monitoring signal in deployed systems, triggering additional verification when cascade risk exceeds a threshold. A head-to-head comparison of $\mathrm{AF}$ to PRM-score-based or survival-hazard diagnostics on the same data would clarify whether AF offers distinct cascade sensitivity beyond existing per-step scores.

## 6. Limitations

**Verification reliability and semantics.** Our LLM-as-judge verifier uses local correctness and the same model family (GPT-4o-mini) for generation and verification, risking correlated errors and differential misclassification—the verifier may perform differently in "clean" vs. "contaminated" contexts, biasing the AF risk ratio. $\mathrm{AF}$ measures new-error emergence, not semantic contamination; global correctness would likely yield higher $\mathrm{AF}$. The 94.2% agreement assesses precision only; recall is unquantified. Cross-model verification or human annotation stratified by contamination status would better bound bias.

**Domain and model coverage.** All experiments use GSM8K; cascading dynamics may differ in code, commonsense, or scientific reasoning. While we test three model families, reasoning-specialized models (e.g., with tool use) remain untested.

**Statistical scope.** The GEE controls for step position but not explicit difficulty proxies (e.g., chain length or per-problem baseline accuracy as covariates). We use an exchangeable working correlation structure; the estimated within-problem correlation is moderate, supporting this choice, but an autoregressive structure might better capture sequential dependencies. The cascade rate differences in Experiment 3, while consistent in direction, are marginally significant after multiple-comparison correction ($p_{\text{adj}} = 0.070$ for cross-verification), warranting replication at larger scale.

**Selection bias and sensitivity to generation parameters.** The parseable-chain filter excludes terse or unstructured traces, potentially biasing $\mathrm{AF}$. We used a single temperature, prompt template, and step-segmentation strategy; stability of $\mathrm{AF}$ under varying temperature, sampling, and step-boundary definitions is untested. Computing $\mathrm{AF}$ per trace and meta-analyzing across units would mitigate within-trace dependence more directly than our pooled approach.

**Injection scope.** Experiment 2 injects only numerical errors at absolute positions 0–4; structural, logical, or unit errors may produce different positional profiles. Using relative positions (e.g., last 20% of steps) and testing on chains longer than 5 steps would improve external validity. Extending the injection to non-math domains (code, multi-hop QA) is critical for establishing whether the late-step vulnerability pattern generalizes beyond arithmetic reasoning.

## 7. Conclusion

We introduced the Amplification Factor ($\mathrm{AF}$), a reproducible metric for cascading failure amplification in multi-step reasoning. Measurement across three model families ($\mathrm{AF} = 3.29$–$7.79$) confirms that prior errors increase subsequent error probability by 3–8$\times$, with the invariant that

$p_{\text{post}} \approx 0.5$ regardless of model. Positional vulnerability increases with step index, more so for smaller models. Among interventions, cross-verification is the most effective cascade reducer, while self-checking is harmful. Future work should extend AF to non-mathematical domains, compare interventions under matched token budgets, and develop position-aware PRMs exploiting the non-uniform vulnerability profile. We will release prompts, verification scripts, and step-labeled traces for replication.

## References


Arcuschin, I., Janiak, J., Krzyzanowski, R., Rajamanoharan, S., Nanda, N., and Conmy, A. Chain-of-thought reasoning in the wild is not always faithful. In *ICLR Workshop on Reasoning and Planning for LLMs*, 2025.

Cobbe, K., Kosaraju, V., Bavarian, M., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Dziri, N., Lu, X., Sclar, M., et al. Faith and fate: Limits of transformers on compositionality. In *NeurIPS*, 2023.

Gan, Z., Liao, Y., and Liu, Y. Rethinking external slow-thinking: From snowball errors to probability of correct reasoning. *ICML*, 2025.

Huang, J., Chen, X., Mishra, S., Zheng, H. S., Yu, A. W., Song, X., and Zhou, D. Large language models cannot self-correct reasoning yet. In *ICLR*, 2024.

Kamoi, R., Zhang, Y., Zhang, N., Han, J., and Zhang, R. When can LLMs actually correct their own mistakes? A critical survey of self-correction of LLMs. *TACL*, 12: 1417–1440, 2024.

Khalifa, M., Agarwal, R., Logeswaran, L., et al. Process reward models that think. *arXiv preprint arXiv:2504.16828*, 2025.

Lanham, T., Chen, A., Radhakrishnan, A., et al. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*, 2023.

Li, X., Wang, W., Li, M., Guo, J., Zhang, Y., and Feng, F. Evaluating mathematical reasoning of large language models: A focus on error identification and correction. In *ACL Findings*, 2024.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. In *ICLR*, 2024.

Miner, S., Takashima, Y., Han, S., Kouteili, S., Erata, F., Piskac, R., and Shapiro, S. J. Scheherazade: Evaluating chain-of-thought math reasoning in LLMs with chain-of-problems. *arXiv preprint arXiv:2410.00151*, 2024.

Mukherjee, S., Chinta, A., Kim, T., Sharma, T. A., and Hakkani-Tür, D. Premise-augmented reasoning chains improve error identification in math reasoning with LLMs. In *ICML*, 2025.

Ross, S., Gordon, G., and Bagnell, J. A. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, 2011.

Saparov, A. and He, H. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *ICLR*, 2023.

Snell, C., Lee, J., Xu, K., and Kumar, A. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Turpin, M., Michael, J., Perez, E., and Bowman, S. R. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *NeurIPS*, 2023.

Uesato, J., Kushman, N., Kumar, R., et al. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Wang, P., Li, L., Shao, Z., et al. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *ACL*, 2024.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models. In *ICLR*, 2023.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. Tree of thoughts: Deliberate problem solving with large language models. In *NeurIPS*, 2023.

Zhang, M., Press, O., Merrill, W., Liu, A., and Smith, N. A. How language model hallucinations can snowball. *arXiv preprint arXiv:2305.13534*, 2023.

## A. Chain-Length Distribution

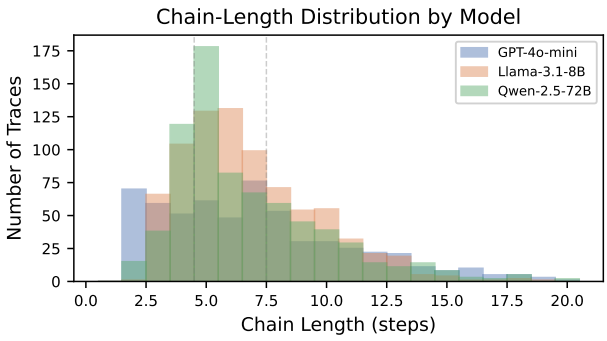


*Figure 7.* Chain-length distribution across the three model families. Dashed lines demarcate short (2–4), medium (5–7), and long ($\geq$8) categories used for stratified AF analysis.

## B. AF by Chain Length

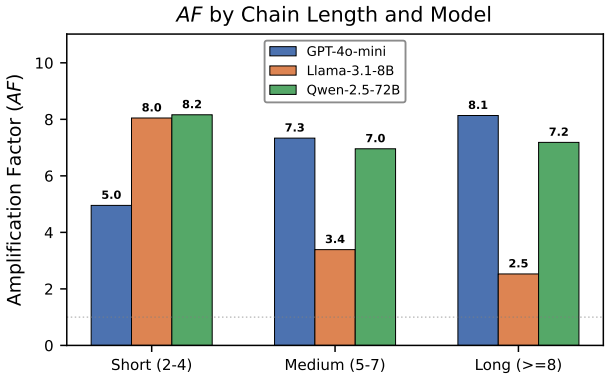


*Figure 8.* AF by chain-length category for all three models. Amplification increases monotonically with chain length, consistent with $O(T^2)$ error accumulation.

## C. Error-Rate Heatmap

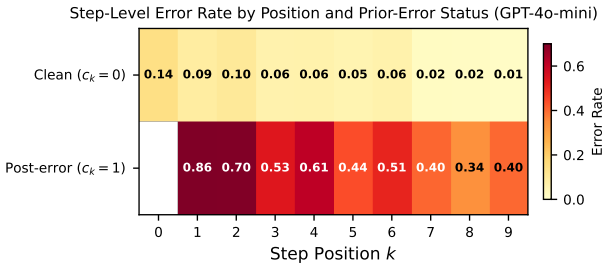


*Figure 9.* Step-level error rate heatmap by position and prior-error status (GPT-4o-mini). The contrast between clean and post-error rows visualizes the AF gap at each step position.

## D. Cascade Ratio Breakdown

Among GPT-4o-mini traces, 35.7% of final-answer errors were preceded by intermediate errors (cascading), while 64.3% arose independently. Cascading traces constitute 32.1% of all 589 traces. Figure 10 shows the trace-outcome breakdown across all three models: the cascade ratio varies substantially (24.4% for Qwen-2.5-72B to 48.7% for Llama-3.1-8B), indicating that while $p_{\text{post}} \approx 0.5$ is invariant, the fraction of traces entering the contaminated state differs by model.

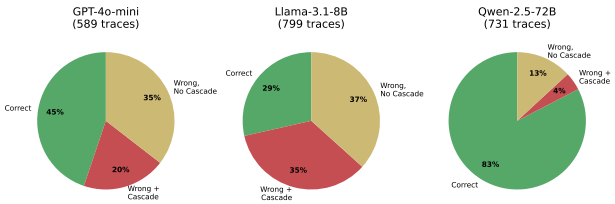


*Figure 10.* Trace outcome breakdown across all three models. Cascading errors account for a substantial fraction of failures, especially for Llama-3.1-8B.

## E. Additional Experimental Figures

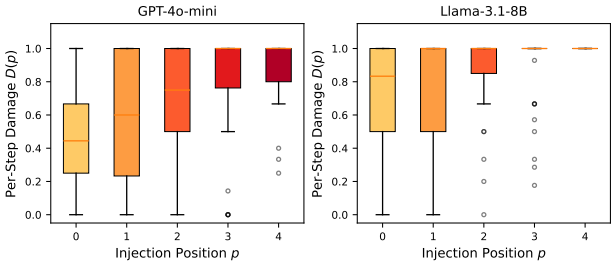


*Figure 11.* Distribution of per-step damage $D(p)$ by injection position for both models. Box plots reveal the full spread of damage outcomes at each position.

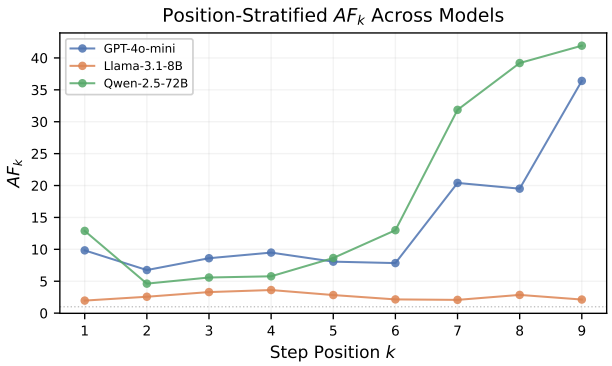


*Figure 12.* Position-stratified $\mathrm{AF}_k$ comparison across all three models. The general pattern of increasing $\mathrm{AF}_k$ with step position holds across model families.

## F. Statistical Details

**GEE.** $\mathrm{logit}(P(e_k{=}1)) = \beta_0{+}\beta_1 c_k{+}\beta_2 k$ with exchangeable correlation, clustering by problem. $\exp(\hat{\beta}_1) = 14.51$ with robust (sandwich) SEs. The estimated exchangeable within-cluster correlation parameter is $\hat{\alpha} \approx 0.08$, indicating moderate within-problem dependence that justifies clustering but does not dominate the between-step variation.

**Bootstrap CI.** AF 95% CI [5.46, 11.24] via 10,000 problem-level cluster bootstrap resamples.

**Position-stratified $\mathrm{AF}_k$ (GPT-4o-mini).** Step 1: $\mathrm{AF}_k = 9.84$ [6.58, 16.49]; Step 3: 8.61 [5.10, 16.20]; Step 5: 8.07 [4.51, 17.78]; Step 7: 20.41 [7.53, 75.88]; Step 9: 36.40 [11.21, 50.98]. CIs widen at later positions due to shrinking clean-context denominators.

**Chain-length stratified AF (GPT-4o-mini).** Short (2–4 steps): $\mathrm{AF} = 4.95$; Medium (5–7): 7.34; Long ($\geq$8): 8.14.

**Fisher tests (Experiment 3).** Two-sided, Holm–Bonferroni corrected. Cross-verify: 5/58 vs. 19/80, raw $p = 0.023$, adj. $p = 0.070$. Self-check: 24/80 vs. 19/80, raw $p = 0.476$, adj. $p = 0.952$. Self-consistency: 20/80 vs. 19/80, raw $p = 1.000$, adj. $p = 1.000$.

**Sensitivity of AF to verifier noise.** Let the verifier have false-positive rate $\alpha$ (labeling correct steps as errors) and false-negative rate $\beta$ (missing true errors). The observed rates become $\hat{p}_{\text{post}} = p_{\text{post}}(1-\beta)+(1-p_{\text{post}})\alpha$ and $\hat{p}_{\text{clean}} = p_{\text{clean}}(1-\beta)+(1-p_{\text{clean}})\alpha$. For our estimated precision ($\alpha \approx 0.06$, from the 94.2% agreement) and a plausible recall of 80% ($\beta = 0.2$), the noise-attenuated AF for GPT-4o-mini would be approximately $\frac{0.498\times0.8+0.502\times0.06}{0.064\times0.8+0.936\times0.06} \approx \frac{0.429}{0.107} \approx 4.0$, a lower bound that remains substantially above 1. Even under pessimistic recall ($\beta = 0.4$), the attenuated $\mathrm{AF} \approx \frac{0.329}{0.095} \approx 3.5$, confirming that cascading amplification is robust to verifier noise.

**Token budgets.** Baseline: $\sim$600 tok/problem. Self-check: $\sim$800. Self-consistency: $\sim$2,500. Cross-verification: $\sim$1,200.