Applying Regularized Recursive Self-Improvement (RRSI) to a Banking and Financial Services Landscape

Regularized Recursive Self-Improvement (RRSI) offers a disciplined way to improve AI agents without retraining the language models behind them. Instead of changing model weights, RRSI improves the agent harness, the instructions, safeguards and workflow that guide its actions. In our banking experiment, RRSI achieved 85% accuracy on held-out tasks, compared with 65% for standard Recursive Self-Improvement (RSI) and 35% for the original agent. On out-of-distribution tasks, RRSI scored 55% compared to 40% for RSI and 30% for the original.

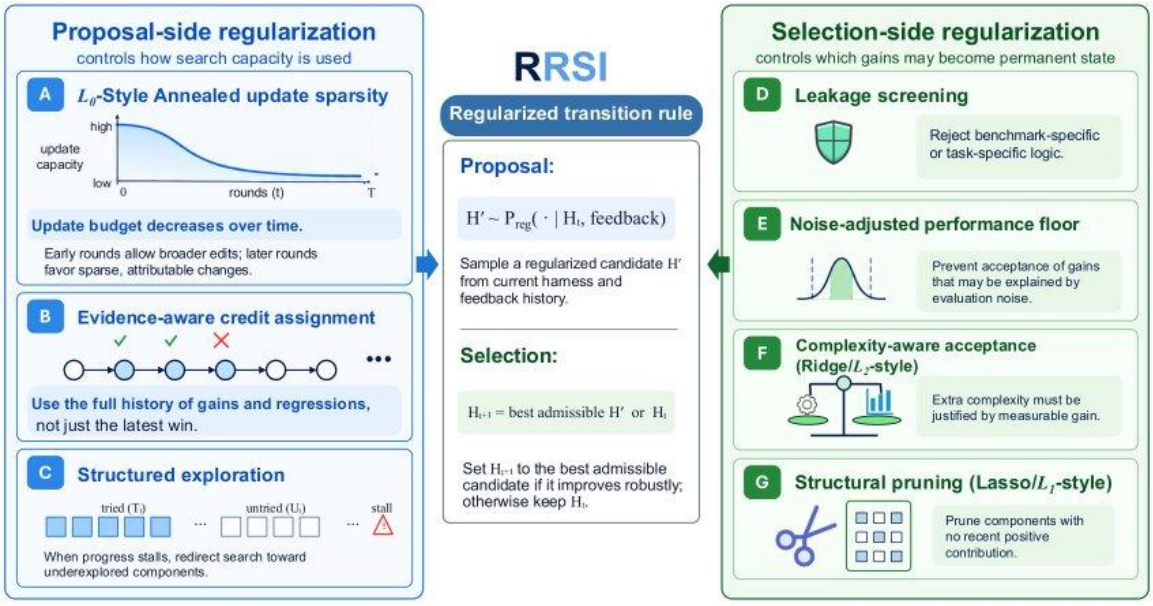
What RSI Actually Improves

Recursive Self-Improvement (RSI) is a feedback loop. Run an agent on tasks, inspect its failures, propose edits, evaluate the changes, retain the best-performing version, and repeat. In our project, the model weights remained fixed. Only the surrounding harness could change, including the system prompt, tool descriptions, how tool results are formatted, step limits, and helpers that the loop calls. Better harness design can help the same model find authoritative evidence, follow a consistent process, and submit more reliable answers.

The danger is optimizing for the examples used during development. An edit might lift a benchmark score by embedding test-specific shortcuts, while offering little help when policies, documents, or transaction patterns change.

What RRSI Adds

Regularized Recursive Self-Improvement (RRSI), described by Xia and colleagues, adds checks intended to make improvements transferable rather than merely boosting development scores. It starts with broader edits and narrows the edit budget over time. It learns from successful and unsuccessful proposals and explores neglected components when progress stalls. Before accepting changes, RRSI screens for test-specific shortcuts, accounts for evaluation noise, and weighs accuracy gains against token costs. It can also prune components that no longer help. These safeguards aim to produce improvements that carry over to unseen tasks.



Source: Xia et al. (2026), RRSI, Figure 2, p. 4.

© 2026 Coforge. All rights reserved.

Putting RRSI to Work in Banking

Our first simulated banking tasks were largely straightforward payment approval lookups. These tasks were too easy to distinguish the methods meaningfully. We redesigned the benchmark around one banking operations agent that handles two more difficult workflows: **policy interpretation** and **ledger reconciliation**.

For policy cases, the agent had to find governing rules, resolve superseded policies, check account state and authorization, and cite authoritative records rather than informal emails. For ledger cases, the agent was required to rebuild closing balances from authorized, settled payment instructions and transactions, while identifying missing, duplicated, or invalid journal postings.

The benchmark included 80 synthetic tasks, with 40 for harness evolution, 20 held-out cases, and 20 out-of-distribution (OOD) cases with new patterns. All three variants started with the same harness and used the same frozen model and tools. Answers passed only when every structured field and required evidence citation was correct.

What the Results Show

After 20 improvement rounds, RRSI led on every evaluation split. Most importantly, its advantage extended beyond the development cases to held-out and OOD tasks that were not used to select harness edits.

Task split	Original harness	RSI	RRSI
Evolve (40)	47.5%	80.0%	**92.5%**
Held-out (20)	35.0%	65.0%	**85.0%**
Out-of-distribution (20)	30.0%	40.0%	**55.0%**

Pass rates under exact-answer, evidence-gated scoring; 80 synthetic tasks, one evaluation per task.

The improvement was especially notable for held-out ledger reconciliation. Pass rates rose from 10% for the original harness to 50% for RSI and 80% for RRSI. Over 20 rounds, plain RSI accepted two candidates and plateaued at 80% on the evolve set. RRSI accepted six candidates while its critic rejected ten proposals. Beyond prompt improvements, RRSI added control-flow checks that intercepted contradictory decisions and missing authorization evidence before submission, raised step and tool-call caps, and later accepted an episode-local memory helper.

RRSI was not the cheapest agent in this banking test. Its final harness averaged 22,514 task-model tokens per evolve case, versus 18,914 for the original harness and 16,159 for RSI, about 19% more than the original and 39% more than RSI. However, critic screening reduced the task-model tokens spent evaluating candidates during search. The original RRSI paper found lower policy-token usage than unregularized evolution on its benchmarks.

Our results show why accuracy, inference costs and search costs should be assessed separately.

© 2026 Coforge. All rights reserved.

What Remains to be Proven

Our study used one 20-round run, a small synthetic benchmark and no real customer data. It does not establish statistical significance or production readiness. RRSI passed only 55% of OOD tasks, including 40% of OOD ledger cases. This gap matters in high-stakes workflows.

Next steps include repeated evaluations, stronger exception handling, more realistic policy changes, and rigorous human review before any live use.

THE TAKEAWAY: The promise of RRSI is not autonomous perfection. It is a more disciplined way to improve an agent while checking whether those improvements transfer beyond the examples that inspired them.

About the Authors

Caleb Hollander is an Associate Forward Deployed Engineer at Momentuum blue with experience across agentic AI, Retrieval-Augmented Generation (RAG), and full-stack software development. His work focuses on designing and optimizing AI systems for performance, reliability, and scalability. He builds agentic AI solutions that move from pilot to production, integrate into enterprise workflows, and solve real-world problems.

Zachary Bonagura is an Associate Forward Deployed Engineer at Momentuum blue with experience across retrieval pipelines, system architecture, and evaluating poisoning attacks on LLM agents. He builds production agentic AI for enterprise clients that lands in live deployment, fits real workflows, and delivers reliable business outcomes.

Zhanping Zeng is an Associate Forward Deployed Engineer at Momentuum blue with experience in Retrieval-Augmented Generation (RAG), agentic workflows, multi-agent orchestration, and AI testing and evaluation. His expertise includes MCP, logging and tracing, guardrails, and Regularized Recursive Self-Improvement (RRSI). He develops production-ready agentic AI systems that integrate into real-world workflows and deliver measurable business value.

© 2026 Coforge. All rights reserved.

Website elements

SEO Title

RRSI in Banking: Improving AI Agents Without Model Retraining

Meta Description

Regularized Recursive Self-Improvement (RRSI) improved banking AI agent accuracy from 35% to 85% without retraining the underlying language model.

© 2026 Coforge. All rights reserved.