

AGI-Alpha-Agent-v0

Meta-agentic Alpha-Factory stack for deterministic, inspectable multi-agent runs

Offline-first demos • Orchestrator + agent swarm • Verifiable execution primitives



Verified autonomy, by design

Autonomy becomes deployable when every action is treated as an untrusted proposal until it clears proof, policy, and settlement.



Fail-closed: no actuation without gates + acceptance tests + replayable receipts.

Institutional posture: treat every model output like an untrusted package.

Design stance: prototype-first, guardrail-heavy, audit-ready.



Positioning

What the repository is (and is not).

Research prototype, demo-first

- Concept lab for meta-agentic iteration
- Aspirational language describes goals—not deployed general intelligence
- Designed for inspection: deterministic runs + replay

Offline-first demos

- Browser simulation (Pyodide) + local CLI
- Mode toggle: Offline vs cloud providers
- Clean upgrade path to hosted runtimes

Verifiable surfaces

- Ledgered envelopes + replay
- Pragmatic security hardening for packaging & CI
- Optional on-chain modules for validation/disputes



Executive overview

Run locally, replay deterministically, extend safely.

What it is

- Orchestrator routing “envelopes” across specialist agents
- Persistent step ledger for replay, audit, and debugging
- Offline-first demo gallery with clear interfaces (CLI, REST/WS)

Why it matters

- Repeatability: seeded runs + deterministic cycles
- Traceability: provenance for outputs and decisions
- Operational readiness: monitoring hooks + guardrails

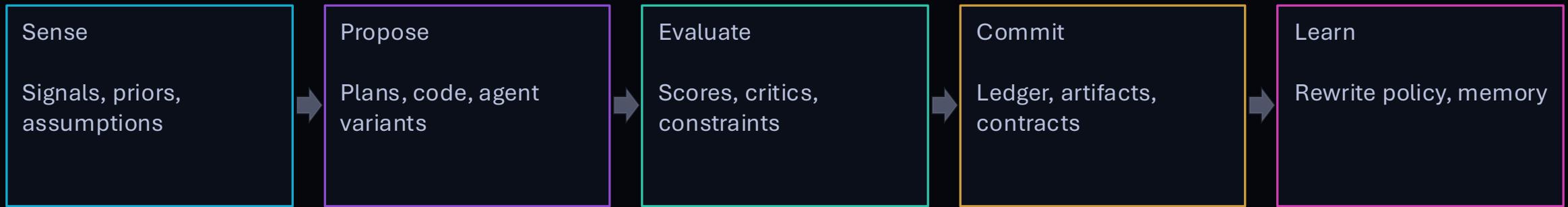
What's different

- Fail-closed control plane + sandboxed execution
- Evidence bundle + two-phase commit for actuation
- Optional settlement modules: escrow, validators, receipts



A single loop, many instantiations

Sense → Propose → Evaluate → Commit → Learn



Two complementary lenses (demo-grade, auditable)

- Game theory: staking, reputation, validator-gated decisions
- Statistical-physics metaphors: free-energy triggers, landscape search, Hamiltonian intuition
- Lightweight toy models—easy to replace with domain-grade evaluators



Architecture: Alpha-Factory stack

Interfaces → orchestrator → agent mesh → ledger → optional providers.

Interfaces

- CLI for automation
- Web UI for demos and dashboards
- REST/WS for integrations

Runtime

- Orchestrator + AgentManager
- Role-separated agent mesh
- Sandboxed tools + memory fabric

Evidence & settlement

- Ledger + replay + provenance
- Optional on-chain validation/escrow
- Receipts and certificates

Design goals: optional deps fail gracefully • deterministic lifecycles • clear extension points.



The agent mesh

Separation of duties: no single agent gets to plan, execute, validate, and settle alone.

Role-separated mesh (typical)

- PlanningAgent — goal decomposition & constraints
- ResearchAgent — evidence & uncertainty tracking
- StrategyAgent — portfolios, scenarios, trade-offs
- MarketAgent — selection & incentives

Institutional constraint

- CodeGenAgent — tool/code synthesis under sandbox
- SafetyGuardian — policy enforcement & filters
- MemoryAgent — structured persistence + provenance
- Validators — replay + attest + dispute handling (optional)



a-AGI Insight v1

Zero-data meta-agentic disruption forecasting (offline-first).



Highlights

- Runs with or without an API key (offline fallback)
- Best-first search over rewrite chains
- MATS: multi-objective evolutionary loop
- Thermodynamic trigger used as a phase-transition rule (toy model)
- Deterministic seeds for replay

CLI: `alpha-agı-insight-v1 --episodes 5`



Determinism & replay

Ledger-first execution: reproducible runs, inspectable artifacts, explicit seeds.

What gets recorded

- Every cycle as a discrete envelope mutation
- Full interaction stream persisted to a ledger
- Replayable traces for analysis and audit
- Provenance for outputs and decisions

Replay surfaces

- CLI: replay / show-results / agents-status
- REST: GET /results/{sim_id}
- WebSocket: /ws/progress
- Deterministic seeds make behavior testable



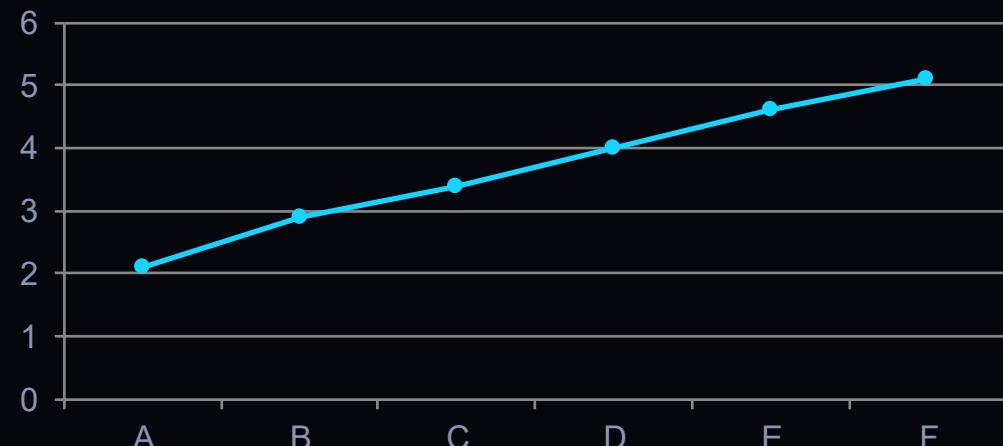
MATS: multi-objective evolutionary search

NSGA-II style loop for exploring trade-offs (Pareto fronts).

What MATS provides

- Non-dominated sorting + crowding distance
- Configurable mutation/crossover
- Island populations + elite exchange
- Optional novelty pressure + critics

Illustrative Pareto improvement



Baseline algorithm: NSGA-II (Deb et al., 2002).



Thermodynamic trigger (toy model)

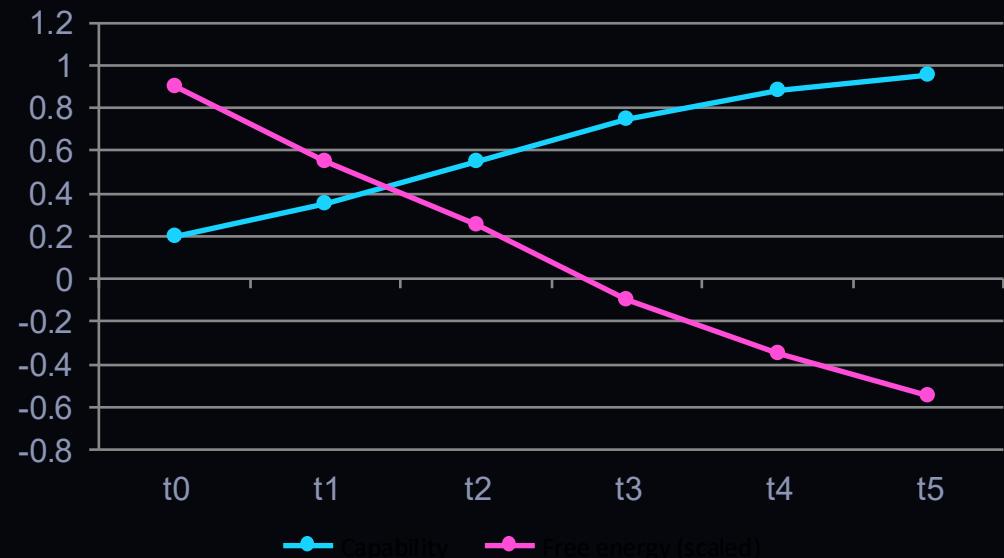
A simple, auditable phase-transition rule for disruption events.

$$F = E - \text{capability} \cdot S$$

Interpretation

- E: sector energy (momentum)
- S: sector entropy (resistance)
- capability: configured growth curve
- Trigger when $F < 0$; disruption may add an innovation gain via MATS

Illustrative crossing



Threshold: $F = 0$



Interfaces: REST + WebSocket + CLI

Instrumented services and deterministic replay paths.

API server (FastAPI)

- POST /simulate
- GET /results/{sim_id}
- WS /ws/progress
- GET /metrics (Prometheus)
- Auth: Bearer token; rate limiting

CLI

- simulate • show-results • agents-status • replay
- Ledger persisted under ./ledger/ for inspection
- Script-friendly for CI + automation

Web demo surface

- GitHub Pages static simulation
- Offline mode first; switch to providers when ready
- Designed for restricted environments



Offline-first by design

Demos remain runnable when the network is unavailable.

Practices baked into the repo

- Wheelhouse workflows for offline installs
- Browser simulation via Pyodide
- Asset fetching with checksum verification
- Network-disabled tests (PYTEST_NET_OFF=true)
- Deterministic seeds for repeatability

Outcome

- Reproducible evaluation becomes the baseline
- Cloud capability becomes an optional acceleration layer
- Review cycles tighten: replay → diff → verify



Security & integrity (pragmatic hardening)

Controls that keep demos inspectable and deployments sane.

Supply chain & packaging

- Signed artifacts (where configured) + reproducible builds
- Safe archive extraction in tooling paths
- Dependency checks and lint/type gates in CI

Runtime & operational guardrails

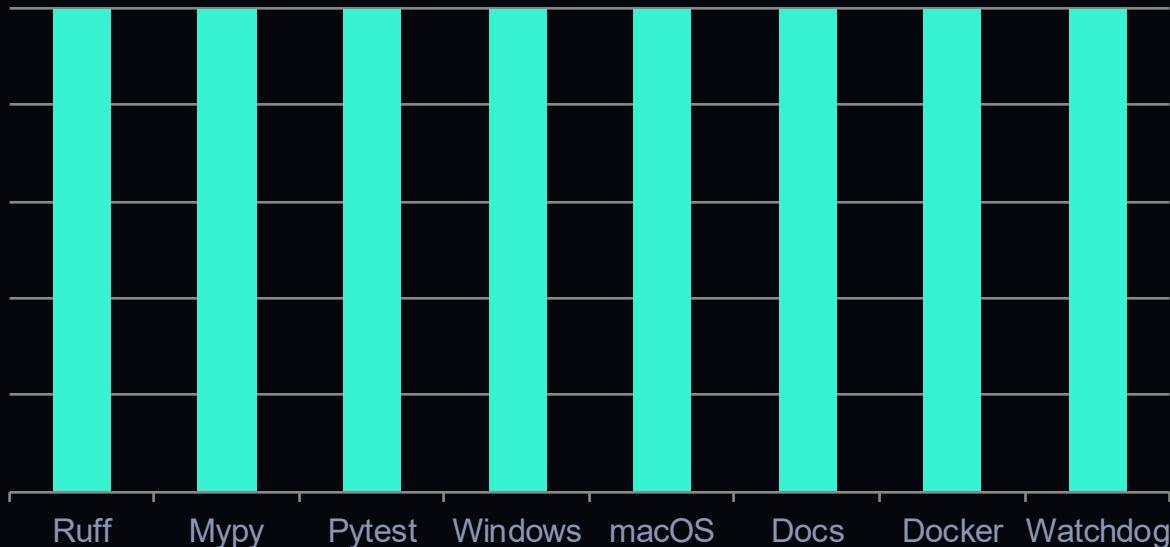
- Bearer-token auth + rate limiting for services
- Sandbox CPU/memory/time caps; environment-bound secrets
- Branch protection verification + CI watchdog



CI enforcement (signal surface)

Required checks keep the main branch provably green.

Required checks (illustrative)



Why it matters

- Prevents drift in determinism and token config
- Catches platform-specific regressions early
- Turns trust into continuous verification



On-chain primitives (v2 smart contracts)

Composable modules for jobs, staking, validation, disputes, reputation, credentials.

JobRegistry

Lifecycle + wiring + allowlists

StakeManager

Escrow + reward release

ValidationModule

Voting + winning set

DisputeModule

Disputes + arbitration hooks

ReputationEngine

Thresholds + blacklist

CertificateNFT

Credential minting per job



\$AGIALPHA configuration (repo-pinned)

Token configuration is treated as deterministic plumbing (not advice).

Token contract (ERC-20, 18 decimals)

```
0xa61a3b3a130a9c20768eef97e21515a6046a1fa
```

Why pin it?

- CI validates address/decimals against Solidity constants and config
- Fail-fast checks prevent drift across builds, badges, and integrations
- Treat token material as utility-only system configuration

Protocol roles (utility)

- Stake for participation (agents/validators)
- Escrow for jobs and dispute resolution
- Fee/burn controls in settlement flows
- Receipts and certificate markets as downstream primitives



Interoperability (standards wave)

Compatible with emerging agent/tool connectivity standards—never locked to a vendor.

OpenAI Agents SDK

- Agent patterns: tools, handoffs, guardrails
- Tracing and observability for agent runs

Model Context Protocol (MCP)

- Standardized tool/data connectivity
- MCP servers expose systems; clients consume them

Google ADK + A2A

- Multi-agent collaboration patterns
- Secure agent-to-agent messaging



Integration points

Extending the ecosystem boundary (optional, composable).

AGIJobsv0 (market layer)

- Job posting, matching, validation, reputation dynamics
- In this repo: contracts/v2 already define modular interfaces
- Plug into richer discovery, pricing, and governance flows

AGI-Alpha-Node-v0 (compute layer)

- Validators + runtimes + observability at the edge
- Execution template: orchestrator + lifecycle + telemetry
- Policy enforcement and operator control surfaces



Deployment paths

From one-command demos → reproducible deployments.

From local to infra

- Quickstart: `python check_env.py --auto-install; ./quickstart.sh`
- Docs + gallery: `make gallery-deploy; make gallery-open`
- Local stacks: `docker-compose.yml`
- Optional infra templates (where provided)

`./scripts/edge_human_knowledge_pages_sprint.sh`

Verified build sprint

- Mirrors the docs workflow: assets → build demos → mkdocs → publish
- Keeps the demo surface reproducible and reviewable
- Supports institutional audit cycles



Roadmap (high-leverage)

Visible in the codebase through placeholders, hooks, and tooling.

Evaluation realism

Replace placeholder scorers with transfer tests and domain evaluators.

Refinement intelligence

Targeted rewrites, tighter credit assignment, fewer wasted cycles.

Benchmarking & ablations

Expand suites; publish stable histories and sensitivity maps.

Memory fabric & tooling

Unify replay, provenance, signing, and observability.

Build

Run the demos • Inspect the ledger • Extend evaluators • Keep it auditable

BUILD

Run the demos. Replay the runs. Ship only what you can verify.



AGI Alpha

agialpha.com

Disclaimer: This repository is a conceptual research prototype. References to “AGI” are aspirational and do not indicate the presence of a real general intelligence. Nothing herein constitutes financial advice. License: Apache-2.0.