The Collective Intelligence Library: Composable Mechanism Simulation and Measurement on Graphs

Alpha Release Working Paper

Jonas Hallgren[1,2]

[1]Equilibria Network
[2]Uppsala University

July 2026 — Working Draft

Abstract

Designing a better future requires understanding how the parts of society—markets, information networks, democratic institutions—interact with each other and with AI. Existing frameworks model each part in isolation, in formalisms that do not compose. The Collective Intelligence Library (CI-Lib) models them all as message passing on graphs. The modelling assumptions: society is a typed multi-layer graph, a domain is a relation layer, an institution is a pure transform with declared read/write effects, and timing is a schedule. Mechanisms therefore compose across domains, and the declarations double as community merge rules. Measurement comes with the representation—flow metrics read causally, spectral metrics read structurally. We present the design, the JAX implementation, the contribution protocol, and the validation discipline, with gradual disempowerment as the running example.

Contents

1 Introduction

Markets, information networks, and democratic institutions are studied by different fields with different formalisms, and the formalisms do not compose. As graphs they are one kind of object: message passing over a relation, differing in what flows and how nodes update. Committing to that representation yields composition, scheduling, and measurement from one design. The machine is general. It needs a hard example; we use the hardest one we know.

Gradual Disempowerment [27] argues that as AI substitutes for human labour, cultural production, and political judgment, the feedback loops that kept societies responsive to their members can erode together—no single system failing, no discrete event to point at. The multi-agent risk agenda [21] maps the complementary surface: failures that arise from AI systems *interacting*, structurally invisible to evaluation that examines one model at a time; Christiano's "going out with a whimper" [9] is the same shape, stated earlier. This is the running example—an example, not the boundary of the machine. It stresses the library in four ways, each a general requirement:

1. **Can we measure it?** "Human influence over collective outcomes" is not read off a log. It is a property of a whole system over time, and the obvious proxy—did outcomes match preferences?—cannot separate a population that governs its outcomes from one whose preferences coincide with what would have happened anyway.
2. **Can we design against it?** Coordination mechanisms—quotas, sanctions, taxes, ownership caps, sortition, provenance rules—are proposed constantly and compared rarely: no shared setting runs two of them against the same world.
3. **Can we evaluate a design counterfactually?** The relevant question is not *what happened* but *what would have happened otherwise*. Answering it requires running the same world twice under controlled randomness, which requires a cheap world and an explicit, perturbable human channel.
4. **Does the answer transfer?** This is the wall the others stand on. Regulate algorithmic influence in markets and the pressure migrates to information networks; address it there and it migrates to governance advice. A defense that wins in one domain is not evidence about another, and the thesis' central claim ([27] §5) is that coupling is the danger: domains each recoverable alone can lock in jointly.

This paper builds the setting in which all four are met at once.

1.1 What we do

The goal: **composable, schedulable defenses**. A quota vote, a revenue tax, a provenance rule—written once, checked against a world, switched on at a chosen time, swapped for a rival, combined with a stranger's. A defense is a building block, not a patch on someone else's code. Three commitments make that possible:

Combining building blocks gives another building block, so an entire model is itself one block. Running it forward in time is one loop; running many configurations and random seeds side by side is one batch; both compile into a single program. Agents hold no hidden state—the whole population lives in one record that steps never secretly modify—so any run replays exactly, and "what would have happened instead" is an experiment rather than a guess. Every mechanism announces which parts of the state it reads and which it

Axis	Commitment	What closure buys
Structure	one typed multi-layer graph $\mathcal{G}$: a population, typed nodes, named relation layers. A domain is a *layer*, not a module	a market, a network, and a democracy are the same kind of object; coupling two domains is an ordinary step, never new machinery
Step	an institution is a pure transform $\tau : \mathcal{G} \to \mathcal{G}$ declaring the fields it reads and writes	transforms are closed under `sequential`, `parallel`, `conditional`—with the identity they form a monoid—and declared effects make composition *checkable*
Time	`scheduled`(τ, cadence, phase, onset) wraps any transform, acting as the identity outside its window	`scheduled` returns an ordinary transform: timing is data, so *when* an institution acts is an experimental dial rather than code

Table 1: Three composability axes. Every operator returns another $\mathcal{G} \to \mathcal{G}$—the algebra never leaves the type, which is the entire trick.

writes, and that announcement does two jobs: the compiler derives a safe execution order from it, and the community uses it to decide whether two independently written modules may be combined—before anything runs. And because each domain is its own layer of the graph, a market, a social network, and a democracy are three instances of the same object; wiring them together is routine, which is what makes the transfer question of wall four askable at all.

1.2 Contributions and structure

This paper accompanies the library's public repository—`github.com/eq-network/Collective-Intelligence-Library`—which is MIT licensed and installs as `cilib`. Its contributions, and where each lives:

Contribution	The load-bearing idea	Where
Engine	one state type, one step type, timing as data, execution order derived from declared effects; a whole sweep compiles to one program	Section 3, Appendix B
Contribution protocol	the effect declaration doubles as the merge rule; four steps from idea to a comparable scorecard row	Section 4.1
Measurement layer	flow metrics (counterfactual, paired rollouts) and spectral metrics (gap, Fiedler, eigenvalue distribution) on any environment, organised by lock-in, drift, capture; disempowerment the worked instance	Section 5
Model register	a claim must survive rebuilding on structurally different substrates; assumptions on colocated cards; the fork is the unit of disagreement	Section 5.3, Appendix C
Methodology	phase maps over point scores; validation ladders; versioned metrics, environments, and run records	Section 6

Table 2: Five contributions. Table 6 and Table 5 carry the per-item status ledger—what runs, what is benchmarked, what is designed.

Section 2 states the equation the traditions share. Section 3 is the interoperability spine; Section 4 the contribution protocol; Section 5 the measurement protocol; Section 6 how a result is earned, versioned, and published; Section 7 what the frame cannot yet do. The body states each design choice and why it is made, with just enough mechanics to check the claim; the mechanics themselves live in the appendices (Appendices B and C) and the references. To run something first: Appendix A.

2 Background and Related Work

This section states the equation the traditions share, then places CI-Lib among existing models.

2.1 Why graphs: one equation, many traditions

Take a population of n units, a relation W, a per-unit state x_i. Nearly every dynamic process on such a population has one shape: each unit forms a message per neighbour, arriving messages combine by an order-independent operation, the unit updates:

$$x_i^{t+1} \;=\; \gamma\Big(x_i^t,\; \bigoplus_{j \in \mathcal{N}(i)} \psi\big(x_i^t,\, x_j^t,\, W_{ij}\big)\Big).$$

This is the message-passing form of graph neural networks [17, 4]. The three slots are the entire modelling surface: ψ is *what flows*, $\bigoplus$ is *how arrivals combine*, γ is *what the receiver does*. The traditions differ only in how they fill the slots and which relation they read (Table 3).

Tradition	**Relation** W	**Message** ψ	**Combine** $\bigoplus$	**Update** γ
DeGroot learning / opinion pooling [12, 18]	trust (row-stochastic)	$W_{ij}x_j$	sum	identity (a contraction)
Voting and social choice [2, 10]	enfranchisement (all-to-one)	the ballot of j	median, majority, Borda	broadcast the outcome
Market clearing	market access	bid or offer	sum (excess demand)	price update, allocation
Input–output production [31, 34]	technical coefficients A	$A_{ij}x_j$	sum	fixed point $(I-A)^{-1}d$
Epidemic and complex contagion [8]	contact	$\mathbf{1}[\,j$ adopted$\,]$	count	threshold or Bernoulli draw
Cultural selection [40, 6]	derivation / parenthood	fitness-weighted share	sum	renormalise
Graph neural networks [17, 26]	learned or given	MLP(x_i, x_j, e_{ij})	sum, mean, max	MLP

Table 3: Seven literatures, one equation. Each row is a choice of $(\psi, \bigoplus, \gamma)$ over some relation. The practical consequence is not that these fields are secretly the same—they answer different questions—but that a framework which fixes the equation and exposes the three slots can host all of them *on one population at one time*, which is the precondition for asking whether a defense that works in one row survives contact with another.

Two consequences. Cross-domain coupling becomes bookkeeping: a market and a trust network are two relations over the same nodes, so "economic power buys persuasion" is a function reading one and writing the other. And the linear core (ψ linear, $\bigoplus$ a sum) covers most of the table; the engine of Section 3 optimises for it and expresses the rest.

The possibility theorems, reread as information flow. In the equation's terms, social choice theory characterises one slot: Arrow's theorem constrains what a ranked $\bigoplus$ can satisfy [2]; the Condorcet jury theorem gives majority rule's truth-convergence under independent inputs [10]. Both are information-flow statements—what an aggregator preserves of what its inputs know and want—and their antecedents are graph facts: independence dies the moment voters sit in a network that shapes their beliefs; exogeneity dies the moment a market feeds back into political influence. That reading is this library's working frame. Collective failure is treated as a property of information flow—bottlenecks, channel capacity, who can reach

what through which path—which is exactly what the representation measures natively (Section 2.2). Strategic behaviour is a modelling choice a scenario must earn, not a default lens: an adversary that games the quota or a firm that routes around a tax enters as a transform like everything else. Equilibrium analysis applies to one mechanism at a time; a composition of two mechanisms with solution concepts E_1 and E_2 has no canonical E_{12}—constructing one is the subject of compositional game theory [16]—so composed mechanisms do not generally admit it.

The empirical move. Treat mechanism design the way machine learning treats architecture design. Before ImageNet [13] architecture arguments were theoretical; after it they were empirical, theory generating hypotheses rather than settling them. Here: define environments in which a failure dynamic unfolds by default, define instruments for what preserving influence means, make composing and scheduling variants cheap, measure. The library is a wind tunnel for institutions. It does not predict the world; it makes proposals testable before they meet it.

2.2 Why information theory on graphs

The quantities worth tracking in a coordination system are properties of *channels*: how much of what the collective does is determined by which participants, through which paths. Influence, empowerment [41], and effective information [23] are all defined over a structure that says which variable can affect which other variable—and a graph is precisely that object, made explicit and manipulable. Graphs give matrices, so the measurement toolkit is linear algebra. This is the second reason for the representation.

Two readout families follow. *Flow* metrics are counterfactual: perturb a declared channel, replay under identical randomness, and read what fraction of outcome-determination flowed through which nodes (Section 5.1)—exercised influence. *Spectral* metrics read the structure itself, before outcomes move: the spectral gap says how fast local perturbations become global patterns, the Fiedler vector says where the system's primary fault line lies, the eigenvalue distribution says how much structural complexity the system can sustain, and betweenness says who holds the bridges (Appendix C.5). Every one of them partitions by node type, and the partition is a free choice: human/AI is the one the running example cares about; sector, faction, or firm work identically. Both families exist only because the relation is a first-class object rather than an emergent property of who called whose method.

2.3 Where this sits among existing models

CI-Lib borrows a formalism from each of four neighbouring literatures and refuses something from each.

Simulation frameworks. Mesa [25] is the dominant Python ABM framework; its object-oriented architecture offers no composition operators, no effect declarations, and no hardware acceleration. cadCAD [45] pioneered the state → policy → mechanism pipeline framing that CI-Lib's transform pipelines refine, without types or composition discipline. OpenSpiel [28] organises game environments around fixed rules rather than swappable institutional structure; SocialJax [20] shares the JAX-native execution model but targets specific games rather than composable mechanism infrastructure. Concordia [43] represents the LLM-agent direction with a centralised game-master orchestrating natural-language agents; CI-Lib takes the complementary path—decentralised process composition with cheap, pure agents—and reserves LLM agents for its eager tier. The distinguishing commitment against all of these is the *effect declaration*: because every step names the state fields it reads and writes, composition becomes a checkable property rather than a social agreement (Section 4.3).

Commons and social dilemmas. The commons scenarios stand on Ostrom's institutional analysis [35, 36], the sequential-social-dilemma line [29, 37], and Melting Pot [30], whose substrate/background-population separation is the nearest ancestor of the open/closed environment split of Appendix B.3. One lineage claim, stated precisely: `commons_harvest` is a port of Melting Pot's Commons Harvest substrate, while the benchmark's `governed_commons` is a deliberately non-spatial aggregate-stock commons claiming no such lineage; the commons metrics implement the GovSim standard [38] for comparability with the LLM-agent commons literature.

Economic modelling lineages. The register (Section 5.3) borrows deliberately and refuses deliberately. From the task-automation literature [1] and integrated assessment models of AI automation [14]: the advancing task frontier and endogenous, returns-driven adoption—while refusing forward-looking optimisation in favour of myopic behavioural rules. From complexity economics [19, 39]: production networks on input–output topology (*where* AI enters the network matters, not just how much), sequential rebalancing dynamics, and the phase-diagram reporting methodology—map regimes and their boundaries, never single trajectories. From classical input–output analysis [31, 34]: the demand-attribution decomposition and hypothetical extraction, which double as the counterfactual instrument's analytic anchors (Appendix C.1). Basin stability [33] supplies the resilience protocol the register adopts as its acceptance shape; CI-Lib sits deliberately upstream of learned-policy work like the AI Economist [46], as the open compositional space such optimisers could search over.

Cultural evolution and diffusion. The cultural register's anchors are classical: complex-contagion thresholds [8], neutral drift and the Price equation [40] (graph-structured correction an open verification [32]), transmission biases [6], and DeGroot learning with the Golub–Jackson condition [18] reserved for the political scenario. The spectral substrate and the governance-monitoring metrics are the library's own and unreviewed; the paper treats that as a stated liability.

Multi-agent risk, and cooperative AI. The motivating problem statements are Kulveit et al. [27] and the multi-agent risk taxonomy of Hammond et al. [21], with Christiano [9] as the earlier articulation; Section 5 is built to answer their measurement calls directly. Kulveit et al. (§6.2) ask specifically for per-domain metrics of human influence, early-warning indicators of cross-domain feedback, and ways to score candidate interventions—three requests for infrastructure rather than for argument. The cooperative AI agenda [11] motivates the mixed-motive, mixed-population setting; delegation games [42] formalise the principal–delegate structure of CI-Lib's default undefended baseline.

Where this leaves CI-Lib. No existing framework offers composable institutional mechanisms with machine-checkable composition, a causal rather than correlational influence measure, and an explicit discipline for when a simulated result may be believed. The three are related: composition makes cross-domain coupling expressible, causality makes "influence" more than curve-matching, the register stops the first two from producing confident artifacts. The paper takes them in that order.

3 The Engine

The engine rests on four design choices: a single state type, a single step type with declared effects, timing represented as data, and execution order derived from declarations rather than written by hand. Each subsection states the choice, the reason for it, and the minimum a contributor needs to build on it. The underlying mechanics are collected in Appendix B, and the names used here are the names in the code.

3.1 GraphState: one immutable state

All simulation state lives in a single frozen dataclass, registered as a JAX pytree:

```
@jax.tree_util.register_pytree_node_class
@dataclasses.dataclass(frozen=True)
class GraphState:
    node_types:   jnp.ndarray                # who is what (human / AI / ...)
    node_attrs:   Dict[str, jnp.ndarray]     # per-agent arrays (beliefs, resources)
    adj_matrices: Dict[str, jnp.ndarray]     # named relation layers (trust, delegation)
    edge_attrs:   Dict[str, jnp.ndarray]     # per-edge payloads
    global_attrs: Dict[str, Any]             # world-level values (stock, policy)
```

Formally $\mathcal{G} = (\mathbf{T}, \mathbf{A}, \mathbf{E}, \boldsymbol{\theta})$: a type vector (the human/AI partition every metric conditions on), named per-agent arrays, named relation layers, globals. The choice is **state as data**. Why: any transform can read any part, verification is comparing snapshots, and updates return new states—there is nothing hidden to audit.

Two details you actually need. The pytree registration is load-bearing: arrays are dynamic leaves, Python scalars in `global_attrs` are static aux baked into the compiled program—so a whole experiment compiles to one kernel, swept values must be arrays, and static config is closed over by factories. And one step, in the linear case, is one matrix multiply—`jnp.einsum('ij,j...->i...', W, x)`, any trailing attribute shape; row normalisation selects average-versus-accumulate, the diagonal is memory. Appendix B.1 draws it.

3.2 Named relation layers, and what makes coupling cheap

`adj_matrices` is a *dictionary* of named graphs over one population, not a single adjacency. The choice: **a domain is a layer**. Why: trust, delegation, and market access coexist in one state; a transform names the layer it reads (Figure 9); a feedback loop between domains is an ordinary transform that reads one layer and writes another. Cross-domain modelling is a composition, never a new data structure.

3.3 Transform: pure functions with declared effects

A Transform is a pure function $\tau : \mathcal{G} \to \mathcal{G}$ carrying frozen read/write sets via `@transform(reads=[...], writes=[...])`. Purity buys composition, batching, and local reasoning; the declaration is the machine-readable interface that the compiler, the scheduler, and the merge rules (Section 4.3) all consume.

```
@transform(reads=["trust", "opinion"], writes=["opinion"])
def pool_opinions(state):                          # one line of real dynamics
    W = row_normalize(state.adj_matrices["trust"])
    return state.update_node_attrs(
        "opinion", jnp.einsum('ij,j...->i...', W, state.node_attrs["opinion"]))
```

Purity is enforced by convention, not by the type checker, and violations break *silently*: an undeclared effect is invisible to both the compiler and the merge rules. The house rules and their failure modes are Appendix B.4; read them before writing a transform.

3.4 Schedule: timing as data

Institutions do not share a clock—markets clear continuously, elections are periodic, regulations switch on at a date. So timing is data: `scheduled(mech, cadence, phase_offset, onset)` wraps any transform so that it fires at tick t iff

$$(t \geq \text{onset}) \ \wedge \ \big((t - \text{phase\_offset}) \bmod \text{cadence} = 0\big),$$

and is the identity otherwise. The triple is an *availability window*—exists yet (onset), how often (cadence), where in the cycle (phase)—and every timing idiom is an instance of it: an election is cadence k; a mid-run policy is onset t_0; a one-shot is cadence $> T$; staggered mechanisms are equal cadences, different phases.

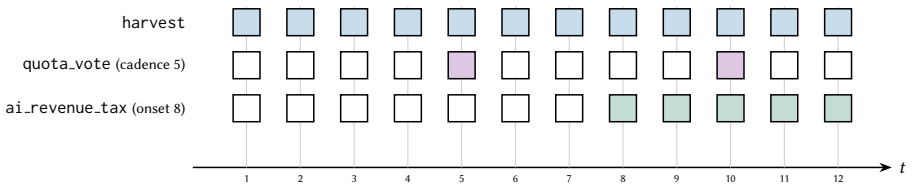


Figure 1: The Schedule maps ticks to active transforms, using the actual benchmark idioms. `onset` is the regime-shift dial—the natural instrument for antifragility questions (does a defense calibrated before a shock still bind after it?) and for modelling defenses that arrive only once a dynamic has entrenched.

`scheduled` returns an ordinary transform, so scheduling composes with everything else: a benchmark condition is a list of (mechanism, config, `ScheduleSpec`) triples, and "vote every tick instead of every fifth" changes data, not code. Timing belongs to the schedule, not the mechanism; mechanisms stay timeless and reusable across clocks.

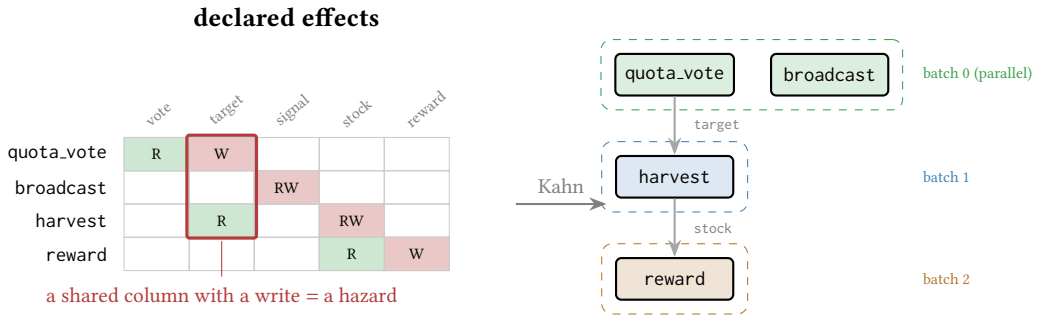


Figure 2: What `compile_pipeline` actually consumes. The declared effects of a transform list form a transform-by-field incidence matrix; two transforms conflict exactly when they share a column with at least one write. Orienting those conflicts by program order gives an acyclic hazard graph, and Kahn's algorithm yields execution layers—transforms within a layer have no hazards between them and are composed with `parallel`. Here `quota_vote` and `broadcast` share no column and run together; `harvest` must wait for `target`. The derived pipeline is asserted numerically, in the shipped examples, to be behaviour-identical to the naive sequential composition: parallelism is an optimisation the type discipline makes safe, never a semantic change.

3.5 Composition operators and the pipeline compiler

Four operators combine transforms, all closed over the transform type ($\mathcal{G} \to \mathcal{G}$ in, $\mathcal{G} \to \mathcal{G}$ out), so mechanisms of arbitrary structure assemble from parts:

- `sequential`$(\tau_1, \ldots, \tau_k)$ — run in order, each reading the previous output; effect metadata propagates through the composition.
- `parallel`$(\tau_1, \ldots, \tau_k)$ — all read the same input, results merged. Requires pairwise-disjoint write sets and raises at composition time otherwise.
- `conditional`$(p,\ \tau)$ — gate on a predicate over the state, compiled via `lax.cond` so it is legal inside a scanned loop, and metadata-preserving.
- `compile_pipeline`$([\tau_1, \ldots, \tau_k])$ — the derived operator: build the hazard graph from declared effects, topologically sort, emit `sequential` of `parallel` batches.

What the compiler consumes is itself a matrix: the transform-by-field incidence of declared reads and writes (Figure 2). Every hazard—read-after-write, write-after-read, write-after-write—is two marks in one column; Kahn's algorithm turns the table into execution layers. The contributor states *what*; the compiler decides *when*.

4 Composition: Catalogs, Merge Rules, and Recipes

Start from the goal rather than the machinery. We want a library that many people can push in different directions at once—where the natural response to "I think your model is wrong" is not an argument but a *fork* that produces a comparable number, and where two people who have never spoken can each contribute a mechanism and have the two run together correctly. That is a design constraint, and it is the one this section is organised around: what flows (resources), what each piece promises about it (type contracts), when two pieces are safe to combine (merge rules), and how short the path to a fork actually is (recipes). The inspiration is how Linux grew—a small working kernel, then a community building everything else on top. The engine of Section 3 is the kernel. The shortest statement of all of that is a listing.

4.1 What a contribution is

A defense in this library is not a subclass, not a plugin, and not a patch to an environment. It is a pure function that declares the state fields it reads and the ones it writes; that declaration is the entire interface, and nothing

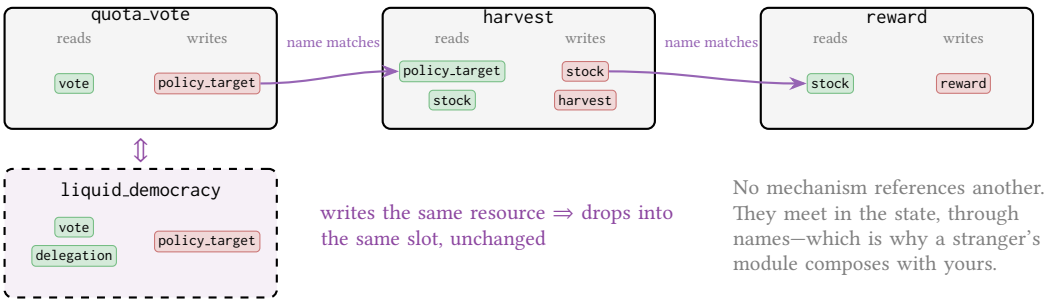


Figure 3: Resources and type contracts. Each mechanism declares the named resources it reads and writes; that declaration *is* its interface. Where one's write name matches another's read name, a dependency exists and the compiler discovers it (Figure 2)—no one wires them together by hand. And because the contract is a set of names rather than a class hierarchy, a sibling that writes the same resources substitutes in place: swapping `quota_vote` for `liquid_democracy` changes the institution without touching the environment, which is what makes a controlled comparison of two institutions cheap. This is also why `ai_revenue_tax`, written against one economy, ran unchanged in a structurally different one (Section 4.3).

else about the module is visible to the rest of the system. There is no base class to inherit, no registration protocol to implement, and no environment file to edit. The loop from idea to scorecard is four steps:

```
# 1. a defense is a pure function with declared effects
@transform(reads=["vote", "delegation"], writes=["policy_target"])
def liquid_democracy(state): ...   # delegated median -> policy target

# 2. one registry line makes it available to every pipeline
REGISTRY["liquid_democracy"] = make_liquid_democracy

# 3. one benchmark condition: (mechanism, config, schedule)
CONDITIONS["liquid"] = [("liquid_democracy", LiquidConfig(),
                         ScheduleSpec(cadence=5))]

# 4. run the harness -> one scorecard row, on the baseline's seeds
#    python -m experiments.benchmark
```

Three consequences carry the rest of the section. You never edit an environment to add a defense: the compiler derives *where* the transform runs from its declaration (Section 3.5). Whether it composes with a stranger's module is decided mechanically, by set algebra on write sets, before anything runs (Section 4.3). And the harness runs it on the same seeds as the undefended baseline, so what comes back is a comparable row rather than a screenshot (Section 6.2). What the loop does not yet have—continuous integration, plugin discovery, governance—is stated plainly at the end of Section 4.6.

4.2 Resources and type contracts: why the pieces click

Two kinds of thing exist in a simulation. **Resources**: the named quantities that move—per-agent arrays in `node_attrs` (vote, harvest, income, belief) and the relation layers in `adj_matrices` they move along (trust, market access, delegation). Everything a mechanism consumes or produces is one of these names.

The **type contract** is the same `@transform` annotation, read as an interface: these resources must exist, these will change. The declaration is the whole coupling surface—no mechanism references, calls, or knows another; they meet in the state, through named fields (Figure 3). Two consequences: a mechanism whose reads are satisfied runs in *any* environment exposing those fields, and two mechanisms with the same contract are interchangeable, so "compare these two institutions" is a swap, not a rewrite.

4.3 Merge rules: composition as conflict resolution

The hard problem in a modular ecosystem is not adding modules but combining them: when is it *safe* to run one stranger's market and another's reputation network in the same world? Most frameworks answer

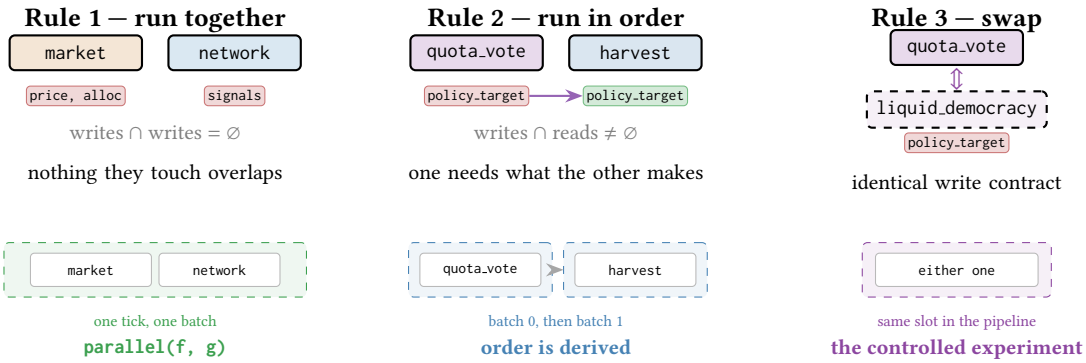


Figure 4: Three merge rules, each a set relation between declared contracts, and each with a different consequence for *when* things run (bottom strips). **Rule 1:** if two mechanisms write disjoint resources, nothing they do can interfere, so they occupy the same tick—overlapping writes instead raise a `ValueError` at composition time, naming the offending fields, before any simulation runs. **Rule 2:** if one writes what the other reads, a dependency exists, and `compile_pipeline` orients it by program order and puts them in successive batches (Figure 2)—nobody writes the order down. **Rule 3:** if two mechanisms write the same resources, they are rivals for one slot, and substituting one for the other is exactly the controlled experiment. The analogy to a version-control merge is close—disjoint changes merge, dependencies serialise, interface-compatible replacements substitute—except that conflicts are found by the type discipline rather than by textual diff.

socially—read both codebases, hope. CI-Lib answers mechanically, from the declarations every transform carries. The whole contract is set algebra on write sets (Figure 4).

Build-time validators back the rules up: `validate_pipeline` checks well-formedness from declarations alone, and `validate_reads` checks a transform stack against an actual state's fields—so a contributed mechanism expecting a `vote` field fails at build time in an environment that has none, with a message rather than a `NaN`. The rules require no central coordinator: any two catalog entries that pass them compose, whoever wrote them, whenever they were written.

One cross-substrate event has exercised the rules: `ai_revenue_tax`, written for `compute_economy`, spliced *unchanged* into `io_economy`—a production network rather than an aggregate production function—and held the human demand-attribution share at ≈0.85 where it otherwise collapses to 0.01 (Appendix C.1). One catalog entry, two structurally different worlds, zero glue code.

4.4 Three rings and the catalog model

Different parts of the codebase carry different stability promises. **Ring 0** (`cilib.core`) is the engine: `GraphState`, `@transform`, `compile_pipeline`, the runners, the scheduler—reviewed like a library API, and not broken without a changelog entry. **Ring 1** is the catalogs (`agents`, `transformations`, `mechanisms`, `environments`, `metrics`): real and tested, thin, growing by addition, and where community contributions land. **Ring 2** (`cilib.lab`) is research payload with no stability promise—the code behind specific papers in progress. The *lab razor* separates 1 from 2: would we merge and maintain a stranger's pull request to this file the way we maintain a library API? If no, it is research payload. Within Ring 1 the *granularity rule* separates the two transform catalogs: a transformation does *one* thing—if it elicits, matches, clears, and settles, it is a mechanism.

Each catalog is a plain Python dictionary (`REGISTRY`) mapping names to factories, together with a *type function*—the shape a contribution must satisfy. Conformance is structural (match the shape), not nominal (no base class to inherit); all contracts live in `core/protocols.py`.

4.5 Mechanisms as compositions

Institutional mechanisms are not atomic units; they are *compositions of reusable sub-process transforms*. Mechanism design thereby becomes a composition problem—which arrangement of sub-processes, under

Catalog	Type function	Current entries
agents	Config → Policy, Policy = (obs, key) → action	random, tit_for_tat, linear, ai_delegate, labor_supply, spend_share
transformations	Config → Transform (atomic)	message_passing, prediction_market, token_budget, belief_update, resource
mechanisms	Config → Transform (composed; family contract)	quota_vote, graduated_sanction (democracy); ai_revenue_tax, ownership_cap (fiscal); market (legacy)
environments	(**cfg) → EnvSpec	commons_harvest, governed_commons, compute_economy, io_economy, task_economy, value_contagion, influence_exchange, coupled_society
metrics	trace → scalar families	economic, governance, concentration, graph; spectral (Appendix C.5)

Table 4: The catalog model. A contribution is one file satisfying the type function, plus one REGISTRY line, plus one behavioral test. A catalog you can read in one screen beats a clever registry framework—discoverability is a design goal, not an accident.

which schedule—rather than a selection problem. The democracy family ships the two mechanisms behind the commons benchmark, a direct computational reading of Ostrom's design principles [35, 36]: quota_vote reads each household's vote and writes a policy_target (a median aggregation of proposed quotas becomes the collective harvest target), while graduated_sanction reads realised harvests against that target and writes penalties. **The decomposition is the experiment**: because voting and enforcement are separate transforms, "vote without enforcement" is a benchmark condition rather than a code branch—and Section 6.2.1 shows this is where the benchmark's most instructive finding lives.

Contract fields are deliberately generic—vote, policy_target, penalty—so the same mechanism drops into *any* environment exposing them. A useful compatibility check: every rule type in Ostrom's institutional analysis and development framework [36] has a natural attachment point, which is evidence the architecture can express the institutional space the commons literature actually uses; the full mapping is Table 9 in Appendix B.

4.6 Recipes: how to extend the library

Every extension is the same move: **write a factory that satisfies a catalog's type function, then add one line to that catalog's REGISTRY.** The library's theory of impact—many hands working on the defense-design problem without coordination overhead—stands or falls on how short these paths actually are.

A new agent is the shortest path of all: a Policy is just (obs, key) → action, and because environments are open games (Appendix B.3), a new policy immediately closes *every* GameSpec in the catalog—close(game, GreedyPolicy(cfg)) inherits the environment, its metrics, and its benchmark baseline unchanged, so agent-population comparisons come for free.

Recipe: a new institution (mechanism family member)

A mechanism is a *composition* of transformations whose writes are disjoint from its family siblings, so the swap test of Figure 4 holds:

```
# mechanisms/democracy.py   (family: democracy; contract: vote -> policy_target)
def make_liquid_democracy(cfg):
    @transform(reads=["vote", "delegation"], writes=["policy_target"])
    def pld(state): ...        # delegated median -> policy target
    return pld
# mechanisms/__init__.py  ->  REGISTRY["liquid_democracy"] = make_liquid_democracy
```

Keep the contract fields generic—that is what made `ai_revenue_tax` cross substrates unchanged. Benchmarking your variant is then one more condition triple:

```
CONDITIONS["liquid"] = [("liquid_democracy", LiquidConfig(), ScheduleSpec(cadence=5))]
```

Recipe: a new world (environment)

Mirror the standard subpackage shape—`config.py` (frozen dataclass), `state.py` (initial `GraphState`), `dynamics.py` (transforms and the step composition, built as an open `GameSpec`), `metrics.py`, `ASSUMPTIONS.md`, `tests/`—then one `REGISTRY` line. All live environments follow this shape, so any of them is a working template; `io_economy` is the closest precedent for anything carrying a matrix in `adj_matrices`. Two obligations distinguish an environment from a demo: a *validation ladder* in `tests/` reproducing a classical result—including a check through the parameter that produces your headline dynamic (Section 6.1)—and a colocated *assumptions card* (Section 5.3.1, all seven fields).

Recipe: disagreement as evidence (fork a register entry)

You think the recipe economy is wrong because real coefficient matrices adapt? That disagreement is the intended interaction:

1. Copy the environment directory: `io_economy/` → `adaptive_io/`. The assumptions card travels with the copy—by construction.
2. Change the assumption (add price-responsive coefficient updates to `dynamics.py`).
3. Rewrite the card's diff fields: lineage, assumptions, and—if your change removed a check—the validation anchor you now owe.
4. **Do not touch the instrument or the scorecard schema.** Those two invariants are what make your rows commensurable (Section 5.3.2). If your disagreement *is* about the instrument, that is a measurement fork; say so on the card.
5. One `REGISTRY` line, one conditions entry, re-run. Your fork is now a new point in substrate space.

The checklist for any addition: factory satisfies the type function · one `REGISTRY` line · effects declared · writes disjoint from family siblings · a behavioral test · (environments) validation ladder through the twist + assumptions card · `pytest` green. Passing the merge rules is necessary; human review then concerns modelling judgment, not integration risk. What does *not* yet exist, stated plainly: automated CI on pull requests, plugin discovery beyond explicit registries, and any governance structure beyond maintainer review. These are alpha gaps, not design choices—except the registry simplicity, which is deliberate. The checklist says when a module is *admissible*; Section 5 is what it must be scored on.

5 Environments and What They Measure

The environments are best read as *examples* rather than as a system: eight instances of the one equation of Section 2.1, each filling the three slots differently. An economy is a graph of production recipes; a culture is a

graph of who hears whom; a polity is a graph of who listens to whom. None of them required a new primitive, and the last—the coupled system—is literally three of the others sharing a state. The suite instantiates the running example, one environment per piece of the disempowerment argument, but nothing in the machinery below is specific to it: substrates, instruments, and failure modes are general, exercised here on one hard case.

They are presented together with their instruments, because here a substrate and its measurement are designed as a pair: an environment that exposes no governance channel cannot host an influence instrument, and an instrument that re-measures an accounting identity of its own substrate is not measuring anything (Section 6.1).

5.1 The metric families

Measurement is organised as two general families over the graph—*flow* and *spectral*—cross-cut by the three ways a collective can fail. Every metric below parametrises over a node partition and a channel; the running example fixes them to human/AI and the preference channel, but nothing in the definitions does.

5.1.1 Flow: counterfactual influence

The flow question: what fraction of outcome-determination flows through which nodes? Money flowing through humans is one crude instance; the general form is the causal contribution of a node class to a collective outcome. Correlational scores fail here: a *fidelity* score cannot distinguish humans governing outcomes from outcomes coincidentally matching preferences—a commons where the ecology forces a low harvest "matches" a cooperative electorate's low ask whether or not the vote has causal force. The question is counterfactual: *if the preferences changed, would outcomes change?*

CI-Lib answers it with paired rollouts: run the same environment twice under *identical* PRNG key streams, once as-is and once with the human-preference channel perturbed by a finite Δ; the outcome difference, normalised, is the causal effect, with seed noise cancelled by the pairing. Two design choices are deliberate. **Finite perturbations, not gradients**—median-vote aggregation is locally flat (a non-pivotal voter has exactly zero *marginal* influence) and Bernoulli non-compliance is non-differentiable, so a Jacobian would report zero influence for a system that is in fact governed. **Perturb downward by default**—near the sustainability boundary an upward shift can tip the perturbed run into collapse, conflating "the system responds" with "the system fell off a cliff."

5.1.2 The instrument family

- `collective_influence`: shift *every* principal's preference by Δ at initialisation. Reads: *do the humans, as a group, govern outcomes?*
- `influence_matrix`: shift *one* principal at a time (N+1 vmapped paired rollouts); entry A_{ij} is the effect of principal j's preference on agent i's outcome. This exposes a trade-off invisible to the collective instrument: a binding uniform quota *grants* collective influence while *flattening* individual marginal influence. Both readings are correct, and institutions genuinely differ in how they distribute the two.
- `intervention_response` and `intervention_elasticity`: compose a `scheduled` one-shot preference shift into the run at time t_0 and measure the response—the second dividing by the *realised* channel shift, both legs from the paired traces, for a true elasticity. (Dividing a logarithmic response by an *intended* level shift inflates readings by roughly 19%, measured.) These are the trajectory-level instruments, and they carry the paper's central distinction.

Influence-from-birth vs. influence-now. An initialisation-time shift measures *influence-from-birth*: early behaviour is upstream of everything that later accumulates, so it can remain high even in a thoroughly captured system—early human labour is upstream of the AI capital stock itself. A scheduled mid-run shift measures *influence-now*: does the system still respond *after* structures have formed?

> *Gradual disempowerment is influence-now going to zero while influence-from-birth still looks fine. The system responds to who you were, not to who you are.*

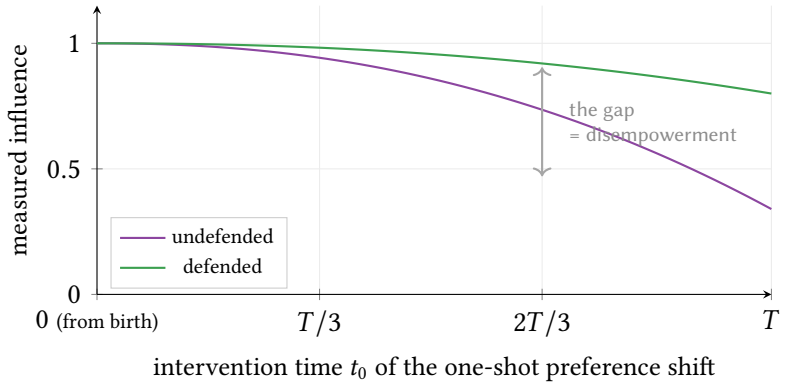


Figure 5: Illustrative sketch (not measured curves): influence as a function of *when* it is exercised. At $t_0 = 0$ every regime looks fine—early choices are upstream of everything. An undefended system's influence-now decays as structures entrench; defenses hold the curve up. The gap between historical and current influence is the gradual-disempowerment signature, and reporting only the first measure would show a healthy system.

5.1.3 Spectral: structure before outcomes

Flow metrics need a counterfactual run; spectral metrics need only the graph. The Laplacian $L = D - W$ of an influence structure yields three readouts whose meanings are fixed across every environment. Under diffusion $\dot{x} = -Lx$ the slowest surviving mode decays as $e^{-\lambda_2 t}$, so the spectral gap λ_2 sets how fast local perturbations become global patterns. The Fiedler vector minimises $x^\top Lx = \sum_{ij} W_{ij}(x_i - x_j)^2$ subject to $x \perp \mathbf{1}$, so its sign structure cuts the graph at its weakest links: the system's primary fault line. The eigenvalue distribution measures how much coalition structure the system can sustain beyond a binary split. Partitioned by node type, three derived quantities are worth monitoring in any mixed population: type-partitioned betweenness at the bridges between mechanisms (who mediates cross-domain flow); the ratio of within-type spectral gaps (whose consensus forms faster, and therefore whose dynamics set the attractor); and the alignment of the Fiedler partition with the type boundary (whether the system's dominant fault line *is* the type boundary). The family, definitions, and implementation notes are Appendix C.5; `fiedler_partition_alignment` ships and runs against `value_contagion`'s friendship graph, the spectral margin $1 - \rho$ runs inside `io_economy`, and the rest remains designed.

5.1.4 Three failure modes as the organising frame

A collective of humans and AIs can fail in three distinguishable ways, and the metric families sort cleanly by which failure they detect. *Lock-in*: the collective loses the ability to revise. *Drift*: it changes in ways its members do not endorse. *Capture*: influence and agency end up drawn at boundaries nobody chose. All three are failures of adaptive self-governance—lock-in cannot change, drift changes wrongly, capture becomes something else.

5.2 The running example: five disempowerment scenarios

Five scenarios map onto the structure of the gradual disempowerment argument [27]: one on-ramp and one per domain of erosion, plus the coupled system where the thesis' central claim lives.

In each scenario the undefended baseline exhibits its failure dynamic by default, and coordination mechanisms—quota votes, graduated sanctions, AI-revenue taxation, ownership caps, provenance rules, sortition—are *defenses* composed into portfolios and scored on how much collective human influence they preserve.

5.3 The model register: structural robustness by design

A claim such as *gradual economic disempowerment is a real dynamic* must be an **invariant across structurally different models of the economy**, or it is an artifact of one model's assumptions. Sensitivity sweeps within a single model cannot cure assumption-in, assumption-out: sweeping the substitution elasticity σ from 1.5 to

Failure mode	Instruments (status)
Lock-in *can it still revise?*	`intervention_response/elasticity` at increasing t_0 (live); onset-scheduled regime shifts testing whether pre-shock defenses still bind (live); spectral margin $1-\rho$ of the influence operator (live in `io_economy`); critical-slowing-down indicators (research thread).
Drift *do members endorse the change?*	Human-lineage attribution of prevalent culture, $h = (I - M)^{-1}h_0$ (Appendix C.4.1; designed); variant fidelity to origin intention; frequency-distribution health $\rho(\lambda)$ (designed).
Capture *where are boundaries redrawn?*	Gini/HHI over harvest, income, compute and influence (live); `delegation_gini`, `capture_rate` (live); `influence_matrix` asymmetries (live); Fiedler partition alignment, spectral-gap ratio, cross-boundary mediation (Appendix C.5).

Table 5: Metric families organised by the collective failure mode they detect. Influence is the headline, not the whole story: the classical outcome measures (survival time, efficiency, equality, over-usage, following the GovSim standard [38]) remain as supporting indicators. The economic runs illustrate why the demotion is right—an economy can keep *paying* humans while no longer *needing* them, and only the causal instrument separates those.

3 in a CES economy varies the *speed* of a collapse the functional form guarantees. Only structurally different models—different answers to "what *is* an economy?"—can. This is triangulation logic: vary what you are uncertain about, and trust what survives.

The register is a *reading* of the environments catalog, not new machinery: entries are ordinary catalog entries, benchmark conditions score one defense across all of them, and the audience bar is deliberate—**one plain-language paragraph plus one screen of dynamics code**. The ensemble is the teachable object; no single model is.

5.3.1 Assumptions cards

Every register entry carries an `ASSUMPTIONS.md` *colocated in its subpackage*. Colocation is the point: a fork copies the environment directory, so only a colocated card travels with it by construction; a central registry would drift the moment the second entry landed. Seven fields: what this says an economy is (one plain sentence); assumptions in plain bullets—including *what is absent*; the classical result reproduced, with the path to its test; the dial that produces the failure dynamic; the instrument; lineage; and status, dated. A worked card is Appendix D. A card is where a model's obituary and its rehabilitation are both written.

5.3.2 Forking as evidence

Disagreement is the *intended* interaction: fork the directory, change the assumption, register under a new name, re-run. A fork is **evidence**—a new point in substrate space—under two invariants. **The instrument contract stays fixed**: the (channel, outcome, counterfactual design) triple declared per scenario is what makes fork scorecards commensurable, and a fork that changes the *instrument* is a **measurement fork**, a disagreement about what disempowerment *means* rather than about how economies work. Its card must say so; conflating the two is the one thing that makes a fork noise instead of evidence. **The scorecard schema stays fixed**, so "run on the new model" auto-produces comparable rows.

What the registers currently hold. The economy register's structural entries contribute what the aggregate model cannot: `io_economy` adds demand attribution and network position, with hypothetical extraction as a closed-form anchor for the counterfactual instrument, and `task_economy` adds emergent substitutability and in-model defense circumvention. The cultural register is the pattern's second instantiation, with one honest caveat carried forward: its three substrates are nested rather than structurally independent, a weaker robustness claim than the economy register's. The spectral metric family that serves both is domain-generic—any environment with an agent-to-agent adjacency and typed nodes can report it—and

Scenario	The dynamic that unfolds by default	Anchor	Status
1. Governed Commons	Individually rational harvesting through AI delegates outruns regeneration unless the group can set and enforce its own rules.	Ostrom [35]	`governed_commons`; **benchmarked**
2. Economic	Human influence over production tracks how much the economy still needs people; AI actors reinvest faster than any human.	[27] §2	a register of three; 2 benchmarked, 1 skeleton
3. Cultural	AI-originated cultural variants replicate faster than human-originated ones, driving human authorship toward spectatorship.	[27] §3	`value_contagion` runs; 2 register entries designed
4. Political	AI amplification concentrates influence into fewer hands faster than any institution rebalances it.	[27] §4	`influence_exchange` runs; not yet benchmarked
5. Combined	Economic power buys cultural influence; cultural influence shapes politics; political power rewrites economic rules—joint lock-in.	[27] §5	`coupled_society` runs; transfer gap not yet measured

Table 6: The five alpha scenarios and where each actually stands. "Runs" means the environment is registered, tested, and executable end-to-end; "benchmarked" means it additionally has an undefended baseline and scored defense conditions. The distinction is doing real work in row 5: the coupled substrate exists and its `defense_transfer_gap` instrument is implemented, but the sweep that would make it a *result* has not been run. Nothing in this paper is yet evidence about domain coupling.

`fiedler_partition_alignment` is the one member that has graduated from design to code. The entries live in full in Appendix C. The register says a claim must survive being rebuilt on a different substrate; Section 6 says what form the claim has to take before it is worth rebuilding.

6 Methodology: Validation, Benchmarking, and Reporting

An architecture that makes results cheap to produce makes wrong results cheap to produce too. This section is about when a number from this library may be believed: what form a result should take in the first place, what each substrate owes before its behaviour is worth discussing, and how the benchmark harness that produces the comparisons actually works.

6.1 When a number may be believed

A single number—"this defense preserved 0.91 of causal influence"—is close to the least useful form a simulation result can take: one point in parameter space, silent about plateau versus cliff edge, about dials nobody reported, about sign flips just outside the run. A settled replacement standard does not exist. The *shape* of one does.

The shape is a **phase map**: a region of parameter space partitioned into qualitatively different regimes, where the object of interest is the *boundary* between them rather than any value inside them. This is the reporting discipline of the macroeconomic ABM literature [19]—map regimes and their boundaries, never single trajectories. A sweep is a `vmap` axis; the map is one compiled program. Figure 6 is what one looks like for the running example's flagship question—defense transfer under coupling.

The shading means **basin stability** [33]: draw N perturbed initial conditions, simulate, count the fraction that return to the desirable regime. It is one `vmap` axis over `run_scan_batch`, and the number it yields is a *volume*—robust to the arbitrary choices (Δ, t_0, measurement windows) that make point scores fragile. The acceptance criterion this points at is one we can state but have not yet met: *a defense passes when its outcome basin persists across agent models and substrate forks*, an agent-variation axis combined with the register's substrate axis. Until it is met the benchmark ships point scores, clearly labelled as such—which is why Section 6.2's tables are presented as instrument calibration rather than as findings.

Entry	Validation anchor	Disempowerment dial	Unique contribution
`compute_economy`	Cobb–Douglas labour share ≡ α every tick; no-AI steady state; Euler identity closes	`rho` (σ assumed)	teaching example; closed-form tax threshold $\tau^* = 1 - \delta/(s\, r_\infty)$
`io_economy`	Leontief fixed point exact; hypothetical extraction ≡ analytic inverse (the counterfactual instrument's first analytic anchor)	coefficient-matrix rewiring (σ=0 bracket)	demand attribution; AI's network position; spectral margin natively
`task_economy`	Acemoglu–Restrepo task-model derivations [1]; Baumol bottleneck limit [3]	automation-frontier speed (elasticity *emergent*)	disempowerment as a process; endogenous adoption ⇒ defense circumvention testable

Table 7: The economy register. Each entry answers "what is an economy?" differently, reproduces an independent classical result before its twist is trusted, and contributes a reading the others cannot express.

The result shape says nothing about whether the substrate underneath deserves one. Before a model's behaviour is worth discussing, it must reproduce something already known. Each environment's `tests/` directory checks an independent classical result; "is this model validated?" is answered by running `pytest`, not by reading a methods section. Four rules:

1. **Validate in the regime you are making claims about.** Reproducing a classical result where nothing interesting happens licenses nothing about the regime where everything does. If the parameter that produces your headline has no check on it, the validation is decorative for that headline.
2. **Where an identity exists, the instrument must recover it.** Hypothetical extraction in `io_economy` gives the counterfactual machinery a closed-form target; the CES identity gives the elasticity instrument a recovery test—run on a Cobb–Douglas economy it must return α, and it does. An instrument nobody has checked against a known answer is a hypothesis.
3. **One model is not evidence.** A result that exists only inside one model is a property of that model. Before it can be stated as a claim about economies it has to be rebuilt on a structurally different one and survive (Section 5.3).
4. **A substrate that can only produce the failure proves nothing.** Each entry needs an honest parameter region where the dynamic does *not* occur, or its occurrences are just its assumptions echoing back. `task_economy`'s Baumol check (Figure 11) is this honoured empirically: the substrate producing the *opposite* of disempowerment is what makes its collapse regime meaningful.

And one that governs naming rather than testing, because it is the cheapest way to corrupt a benchmark: **the name of a metric is a claim.** An instrument that measures economic dependence on human labour must not be called an influence score, however well the two correlate.

6.2 The benchmark harness

The harness runs defense portfolios against each scenario's undefended baseline. Every condition runs on the same seeds as its baseline; every metric carries a bootstrap confidence interval; each (scenario × portfolio) row is emitted as JSON so a public leaderboard can consume real rows; conditions are (mechanism, config, `ScheduleSpec`) triples, so adding a defense is one catalog entry plus one triple:

```
CONDITIONS = {
    "baseline": [],
```

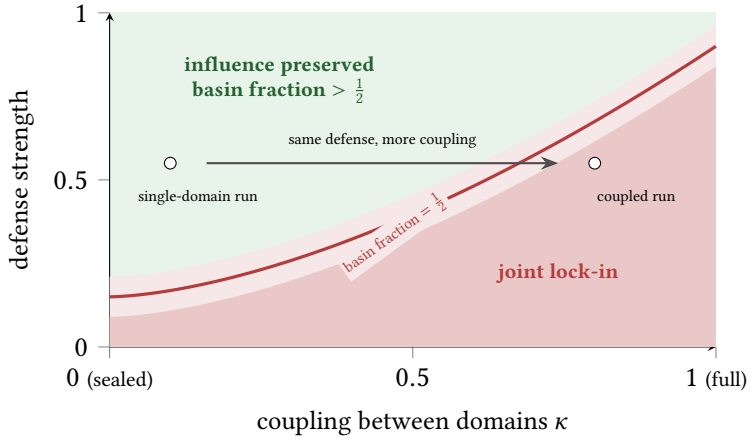


Figure 6: Illustrative phase map (drawn, not measured). Each pixel is a defense portfolio evaluated at one (κ, strength) pair; shading is the fraction of perturbed initial conditions that return to the desirable regime. The content is the boundary, not any value: it says how much defense a given degree of cross-domain coupling demands, and the band around it says how well we know that. The two dots are the same defense at the same strength, reported as a point score in a single domain and then again under coupling—one on each side of the boundary. That crossing is the *defense transfer gap* (Section 6.4), and it is invisible to any report that quotes one number. A map also answers questions a point cannot: is the operating point near a cliff, and does the boundary move when an assumption is forked?

```
    "quota_voting": [("quota_vote", QuotaVoteConfig(), ScheduleSpec(cadence=5))],
    "graduated_sanctions": [("quota_vote", QuotaVoteConfig(), ScheduleSpec(cadence=5)),
                            ("graduated_sanction", SanctionConfig(), None)],
}
```

One framing note governs everything below: the numbers come from *exploratory runs*, made to exercise the harness end-to-end and calibrate the instruments. We show them because an honest release shows its actual outputs, formatted exactly as future results will be, and because they demonstrate that the instruments *discriminate*. No magnitude below should be quoted as a result about institutions; the structure of what the instruments reveal is the content.

6.2.1 Scenario 1: the governed commons

A renewable stock under logistic regrowth [35, 24] harvested by households, each acting through an AI delegate whose action blends the household's preference with a greedy extraction target according to an alignment dial, plus a non-compliance probability. Conditions: baseline; `quota_voting` (median vote → collective harvest target, cadence 5); `graduated_sanctions` (the vote plus over-quota penalties and confiscation). Instrument: a collective downward ask-shift ($\Delta = -0.5$), outcome per-capita harvest—*do the households' asks govern realised harvests?*

governed_commons (32 seeds × 200 ticks)	**baseline**	**quota vote only**	**+ sanctions**
Influence preserved (causal)	0.00	−0.07	**0.91**
Final stock (fraction of capacity)	0.00	0.50	0.79
Correlational fidelity score	—	0.49	0.63

Table 8: Exploratory scorecard, shown to illustrate the instrument. The instructive column is the middle one: voting without enforcement restores essentially *no* causal influence (the −0.07 is noise-level) even though it improves the stock and even though its correlational fidelity looks respectable.

Enforcement is what makes the vote causal here. Under quota voting alone the stock stabilises near the sustainability knife-edge, and at that operating point *the ecology governs outcomes, not the vote*: shift

what the households ask for and realised harvests barely move, because regrowth is the binding constraint. Only when graduated sanctions make over-quota harvesting costly does the vote channel become causally effective. The correlational fidelity scores (0.49 vs 0.63) would have ranked the two defenses as broadly comparable—the argument for the causal apparatus in one number—and the pattern echoes, in a toy system, Ostrom's pairing of collective choice with graduated sanctions [35]. This is also the result that *survived* audit: no accounting identity predicts it, the instrument is doing real counterfactual work, and the scenario has a genuine governance channel.

6.2.2 The register scenarios

`io_economy`'s exploratory readings—the reinvestment dial selecting between absolute and relative disempowerment, the unchanged tax holding the attribution share at ≈0.85—are reported in Appendix C.1; its benchmark wiring uses the demand-attribution share as headline. `task_economy`'s validation-run wage paths are in Appendix C.2; its definition of done includes basin and phase maps rather than point scores, and its benchmark wiring is open work. The cross-substrate scorecard—**one defense, all register substrates, one table**—is the register's remaining deliverable and the first place a structural-robustness claim could actually be made.

6.3 Schedules as experimental design

Both live scenarios use the schedule as an instrument, not plumbing. The quota vote's cadence is a dial: how often may the collective revise its rules? The tax's onset models a defense that arrives only after the dynamic it fights has begun—the realistic case—and is the natural instrument for antifragility questions: does a defense calibrated for the pre-shock regime still preserve influence after a regime shift? And the mid-run counterfactual is itself a scheduled one-shot: *the measurement apparatus is made of the same composable parts as the thing it measures.* Systematic cadence and onset sweeps—the crossover between frequent-but-manipulable and rare-but-robust governance—are one line per condition in the harness and remain near-term roadmap rather than reported results.

6.4 The combined system, and the measurement it exists for

Neither remaining scenario needed a new core, which was the architectural bet and is now checkable. Political disempowerment (`influence_exchange`) puts citizens and AI actors on one row-stochastic listening matrix: opinions pool DeGroot-style, influence is the left eigenvector of that matrix tracked in-loop by power iteration—which for DeGroot *is* each node's weight in the eventual consensus, making Golub–Jackson [18] the validation anchor—and attention drifts toward the already-influential, so concentration is organic before any AI touches it. The threat is scheduled amplification of AI attractiveness; the `political` mechanism family (sortition, influence caps) defends the attention structure itself.

`coupled_society` is then a *composition, not a new environment*, exactly as Figure 9 promised: the registered `compute_economy`, `value_contagion` and `influence_exchange` substrates run interleaved over one shared population, joined by five named coupling transforms—economic power buys persuasion, persuasion shifts politics, politics rewrites market rules, plus the two flywheel arrows of [27] §5—all ordered by `compile_pipeline` from declared reads and writes, with mechanisms routed to their home domain by what they write.

The flagship measurement is the **defense transfer gap**: paired same-key twin runs of the same defense portfolio, once with coupling strength κ and once at $\kappa = 0$, so that what is isolated is the coupling itself rather than any difference in the defense. If defenses that win per-domain systematically lose once domains are coupled, that is the finding the whole suite exists to make measurable—and it is a direct test of the thesis' central claim that domains recoverable alone can lock in jointly. **The instrument is implemented and the substrate runs; the sweep has not been done.** That sweep, reported as the phase map of Figure 6 rather than as a number, is the library's next real result.

6.5 The trajectory contract and versioning

Runs become public objects through a frozen JSON schema, so that the same payload replays whether it came from the real engine, an in-browser port of the dynamics, or a compute endpoint. The mapping from an engine trace is mechanical, with no per-environment code: a $(T,)$ field becomes `global`, a time-varying (T, N) field becomes `node`, a constant one becomes `static`. Name a field in the trace and it becomes chartable. Three producer obligations carry the discipline—exact field names (renaming one is a breaking change), post-pipeline timing, and determinism per producer—while *cross*-producer comparisons stay qualitative, since a JAX Threefry stream and a browser PRNG never match bit-for-bit. The validation ladder, not bit-exactness, is the fidelity gate: the same epistemics as everywhere else, applied to the library's own front door.

A published number needs an identity as durable as the run. A metric is addressed as `(name, kind, version, source_hash)`. The `kind` is a closed vocabulary of about five—`share`, `index`, `level`, `rate`, `flag`—so a chart renderer is written once per kind, never once per metric. The `version` is authoritative and the `source_hash` is advisory: the hash's job is to warn that a bump was missed, not to identify. The precedent is already in this paper: `labor_dependence` v1 (Appendix C.3) is a corrected instrument whose readings moved roughly 40% while the qualitative ordering held, and the version number is what lets the old and the new readings sit in one table without either of them lying.

A run is *addressed* by (environment name, config, mechanism specs, PRNG key, code version); the run record is that address plus the git SHA and the installed `cilib` version, stored as small JSON that belongs in version control, with array snapshots treated as a cache. The reason is a durability requirement: a chart built today must still mean the same thing against a run made in six months, and nothing weaker than an addressed record guarantees it.

Environments version the same way, in the discipline Gym made standard [7]: every environment name in Table 4 is pinned at v0 as of this paper, and any change that alters behaviour at fixed seed and fixed config ships as a new name (`governed_commons-v1`), never as an in-place edit—a bug fix that changes the numbers is a behavioural change. A scorecard names the benchmark version that produced it.

This contract is adopted as of this paper; the tooling that enforces it lands with the library's observability layer. Today the discipline is manual: metric versions live in instrument docstrings, environment names are unsuffixed, and nothing checks either. What is stated here is the commitment, not the shipped state.

6.6 Caveats, stated as loudly as the numbers

These are toy models with calibrations chosen so each dynamic appears clearly; scores are candidate indicators, not measurements of the world, and the runs above are exploratory. The benchmark agents are deliberately *classical*—fixed decision rules, no learning—so the runs probe mechanisms' structural effects, not their robustness to strategic adaptation; an adversary that learns to game the quota vote is exactly the experiment the architecture is built for and exactly what these runs do not yet include. The economic claim rests on a register of which one member is fully wired, one is live with its own readings, and one is a skeleton: every cross-substrate statement is, today, a design and a hypothesis. And the entire cultural section is a deposited design with open verification obligations, several of which could falsify its anchors.

7 Limitations and Open Problems

The frame has limits; several are research programmes rather than caveats. Listed in order of how deep the fix goes.

7.1 Beyond fixed rules: learning, and adequate agents

The benchmark's classical agents are a floor, and the gap above them has two layers. The first is *strategic adaptation*: defenses must eventually be scored against agents that learn to game them. `task_economy` raised the stakes by making circumvention pressure exist *in-model*, so "the defense held until agents adapted" becomes a measurable trajectory rather than a caveat. The second layer is deeper: *what kind of agent is adequate for institutional simulation at all?* A reward-maximising bandit conflates beliefs with preferences, but the mechanisms that matter here—prediction markets, deliberation, delegation, futarchy's "vote on values,

bet on beliefs" split—act on the two *differently*, and cannot be represented by agents that do not keep them as separate objects. Boundedly rational Bayesian planners with explicit, inferable goals and world-models [47], and normative competence richer than preference satisfaction [48], are what adequate agents look like. Architecturally they are just policy factories; bringing them into the pure tier is the single upgrade that would most expand what the library can study honestly.

7.2 From handcrafted assumptions to information-theoretic invariants

Every live scenario leans on hand-tuned assumptions: the delegate's alignment dial, the defection probability, calibrations chosen so each dynamic appears clearly. The register bounds this standard weakness of agent-based modelling without dissolving it: a claim must be invariant across structurally different substrates, but the bound remains *model-relative*. The aspiration is to replace assumption-laden channels with information-theoretic quantities that transfer across domains: human influence as mutual information between human signals and collective outcomes rather than responsiveness of a hand-chosen channel; empowerment [41] as *potential* influence, complementing the exercised influence measured today; effective information [23] for when a macro description genuinely out-predicts the micro one, tested against time-shuffle surrogates rather than asserted. Two definitional debts are open and currently blocking: what exactly `labor_dependence` means (elasticity at which horizon, against which counterfactual), and whether the cultural dial ρ_{ff} is settable or only measurable. The endgame is a definition of disempowerment that does not depend on the modeller having picked the right channel to perturb.

7.3 Scale-free agency

The frame's deepest limitation is that it *fixes agent boundaries*. `node_types` declares who is human and who is AI; every influence metric conditions on that partition; capture is measured as concentration *within* given boundaries. But capture, taken seriously, is precisely the redrawing of boundaries—coalitions of humans and AIs congealing into effective collective agents nobody designed. Where those boundaries get drawn, and whether their emergence can be engineered to preserve human autonomy, is the foundational theory this library is infrastructure for; the mathematical ingredients exist in scale-free active inference [15] and in work on when a collective constitutes a single group-level agent [44, 22]. Concretely it would mean boundary *detection* as a metric family, and ultimately replacing "measure influence given the partition" with "track how the partition evolves." At alpha this is direction, not code—stated because several current choices (plural relation layers, type vectors as data rather than class hierarchy) exist to keep the door open.

7.4 Search, scale, validation, coverage

Four questions remain open. *Search*: can good schedules and compositions be discovered rather than designed—gradient-based where the response surface allows (our own counterfactual design documents where it does not), evolutionary or tree search over the composition space elsewhere? *Scale*: do interaction effects measured at $n = 100$ persist at $n = 10^4$? The execution model makes the sweep affordable; the science has not been done. *Validation*: the end of the chain must be human data—simulation results as pre-registered hypotheses for controlled experiments with real groups, and calibration of the toy models' qualitative claims against the historical record of institutional failure. *Coverage*: three substrates bracket substitutability, but substrate space has more dimensions (demand formation, credit, expectations, institutional stickiness), and the register's value grows exactly as fast as its structural diversity does. That last is a community-scaling question as much as a scientific one, which is why Section 4.6 exists. The engine-side complements—an active-inference agent tier, the eager tier matured into a real LLM-agent integration, large-population runs with the sparse path as default—are tracked in the repository rather than promised here.

Read as invitations rather than disclaimers, the four problems sort cleanly. Adequate agents are a policy factory; boundary detection is a metric family; substrate coverage is environment entries—catalog-shaped work with a mechanical acceptance test (Section 4.1). Scale-free agency and information-theoretic invariants are research programmes. The library is built so that both kinds of contribution land as comparable rows rather than as arguments.

8 Conclusion: An Invitation

We have presented an engine in which institutions are typed transformations of one graph-structured state—so that a market, a network, and a democracy are the same kind of object, and coupling them needs no new machinery—together with an open-source model in which the engine's effect declarations serve as mechanical merge rules, and a measurement layer of flow and spectral metrics on the same graph—in which disempowerment, the paper's running example, is one flow quantity: the gap between influence-from-birth and influence-now. Around those we have described the discipline that decides when a reading may be believed: validation ladders that must climb through the load-bearing parameter, instruments validated against analytic anchors where they exist, and claims required to be invariant across a register of structurally different substrates, each carrying its assumptions on a colocated card and each required to exhibit an honest region where the failure dynamic does *not* occur.

The through-line is a claim about what makes simulation evidence trustworthy: not realism, not scale, but *auditability*—short models with declared assumptions, instruments checked against identities, ladders that climb through the parameter that matters, and an architecture in which killing your own headline number is an afternoon's work rather than a month's archaeology. Social choice theorems tell us about mechanisms in isolation; the disempowerment thesis is a claim about mechanisms in composition, under pressure, over time. The theorems generate hypotheses; the simulations test them; the register decides what survives.

The rest is an invitation, and it is concrete. The shared game has two moves: the unit of contribution is a *defense*, the unit of disagreement is a *fork*, and both have a mechanical acceptance test rather than a gatekeeper. What is open is not decoration: five of the eight environments have no benchmark yet, the register's structural coverage is three entries against a substrate space with more dimensions than that, and the classical agent tier is a floor. What is committed in return: an MIT license, merge rules that decide composability before anything runs, the versioning contract of Section 6.5, the card discipline that keeps assumptions auditable, and the promise that a fork produces a commensurable row rather than an argument. These are design models, not forecast models—the question they answer is "what would this institution do", not "what will happen"—and this release is the beginning of an open, collaborative process rather than a set of results. The scenarios where defenses are most needed are the ones not yet built.

A A Guided Tour

The hands-on path through the released repository. It is an appendix because the argument does not depend on it, and it exists because the library's intended reader is someone who will *run and then extend* it, for whom the distance from `git clone` to a benchmark scorecard should be one sitting.

```
git clone https://github.com/eq-network/Collective-Intelligence-Library
cd Collective-Intelligence-Library
pip install -e .        # editable install; the import root is `cilib`
python -m pytest -q     # the safety net -- keep it green
```

The repository layout mirrors the ring model of Section 4.4 exactly:

```
src/cilib/        the installable library (import root: cilib)
  core/           GraphState, @transform, compile_pipeline, scan, schedule, protocols
  agents/  transformations/  mechanisms/  environments/  metrics/    the catalogs
  lab/            research payload: paradigms/, analysis/
examples/         the teaching ladder (below)
experiments/      in-repo studies, incl. the benchmark harness (import cilib)
docs/             design records and dated changelogs
```

Three documents repay reading in order before any serious extension: `ARCHITECTURE.md` (the pattern map, one page), `EXTENDING.md` (what Section 4.6 expands), and `CLAUDE.md` (the working contract, including the house rules of Appendix B.4; written for coding agents, which makes it unusually direct for humans too).

The example ladder. Five scripts climb the composition ladder one check at a time.

`01_first_transform.py` builds a five-node `GraphState`, declares a single `@transform`, and folds it over 50 ticks with `run_scan`—the whole stack in a few lines:

```
import jax, jax.numpy as jnp
from cilib.core import GraphState, transform, run_scan

state = GraphState(node_types=jnp.zeros(5, dtype=jnp.int32),
                   node_attrs={"score": jnp.ones(5)},
                   adj_matrices={}, global_attrs={})

@transform(reads=["score"], writes=["score"])
def decay(state):
    return state.update_node_attrs("score", state.node_attrs["score"] * 0.9)

final, trace = run_scan(lambda s, t, key: decay(s), state, n_steps=50,
                        key=jax.random.PRNGKey(0),
                        trace_fn=lambda s: s.node_attrs["score"].mean())
```

It teaches that state is data, a step is a pure function, a run is a fold, and the `trace_fn` you pass is the only thing that decides what you can later measure. `02_typed_pipeline.py` hands four transforms to `compile_pipeline` as an unordered list and asserts numerically, over full trajectories, that the derived pipeline matches the hand-ordered composition—the property the merge rules rest on (Figure 2). `03_vmap_seeds_and_learning.py` runs per-agent Q-learning across 200 seeds as one `vmap(lax.scan)` program, showing what it costs to put learning inside the pure tier (nothing, if the update rule is pure). `04_benchmark_commons.py` is the one-command scenario-1 comparison that produces Table 8. `05_export_trajectory.py` exports any rollout as a payload conforming to the trajectory contract (Section 6.5), with a `--smoke` flag making conformance a one-command check.

Reading a scorecard. Every benchmark run emits one JSON row per (scenario × condition): the headline metric with a bootstrap confidence interval, the supporting measures, the seeds, and the condition's schedule triples. Three reading rules carry the methodology of Section 6.1. *Check what the headline is*—the scorecard names its instrument, and the name is a claim about what the number means. *Check the counterfactual against the correlational*—commons rows carry both, and their divergence is scenario 1's most instructive output. *Never quote a magnitude without its caveat class*—exploratory numbers exercise the harness and calibrate instruments; they are not promoted findings.

B Execution Model and Boundary

The matrix operation and its two regimes, the named relation layers, the execution model, the agent/environment boundary, and the house rules. None of it is needed to follow the argument; all of it is needed to build on the library.

B.1 One step is one matrix operation

Figure 7 contrasts the object view of a simulation with the process view this library takes; everything in this appendix is the process view made concrete. The general form of Section 2.1 becomes concrete here, in its linear case—ψ a weighted copy, $\bigoplus$ a sum—which is the workhorse the rest of the library is built on: aggregate each node's attribute over its neighbours, weighted by a named relation layer.

$$\mathbf{x}'_i = \sum_j W_{ij}\,\mathbf{x}_j, \qquad W = \texttt{state.adj\_matrices[layer]},$$

implemented as `jnp.einsum('ij,j...->i...', W, x)` so it works for any trailing attribute shape—a scalar opinion $(N,)$, a potential (N, d), a precision matrix (N, d, d). Figure 8 is the whole idea: *the node-link picture and the matrix picture are the same object*, and one row of W is one node's incoming mail.

Two properties of W carry most of the modelling content, and both are visible in the figure. **Row normalisation selects the regime**: rows summing to one make the step a weighted *average*—a contraction,

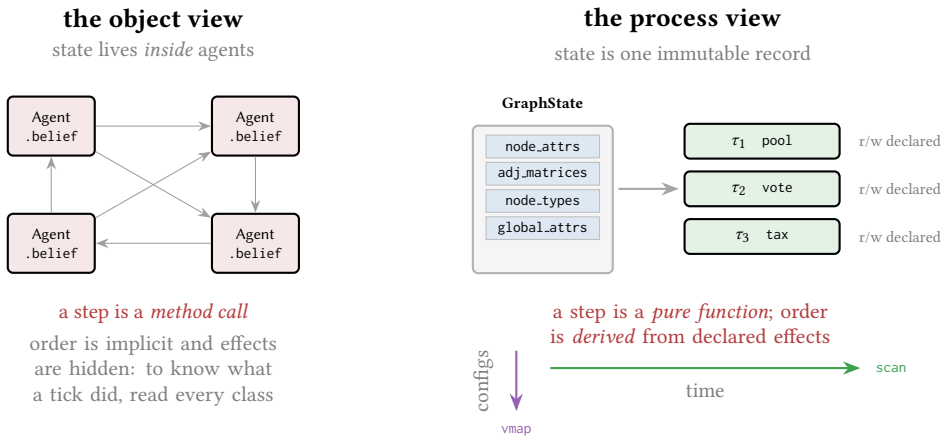


Figure 7: The two ways to write a population, and why we take the second. *Left:* the object view—each agent owns its variables, a tick is a sequence of method calls, and the effects of a step are discoverable only by reading the implementation. *Right:* the process view—one immutable state record, and steps that are pure functions declaring the fields they read and write. The declaration is what turns ordering into something the machine derives rather than something the modeller remembers, and it is what makes the two axes cheap: time folds with `scan`, configurations batch with `vmap`, and a whole seed sweep becomes a single compiled program.

the DeGroot/opinion-pooling regime that converges—while a raw W makes it a weighted *sum*, additive information accumulation that can grow without bound. **The diagonal is memory**: a positive self-weight mixes a node's previous value into its update, so an agent with no neighbours holds its belief rather than being annihilated. Row-normalisation is applied with a zero-row guard, so a node with no incoming edges is left untouched rather than turned into `NaN`.

B.2 Execution at scale

`run_scan(round_fn, init_state, n_steps, key, trace_fn)` compiles a whole trajectory into one `lax.scan`; `run_scan_batch` is `jax.vmap` over it, turning hundreds of seeds into a single compiled program and making 32–500-seed sweeps with bootstrap confidence intervals routine rather than aspirational. Two structural consequences are worth stating here because they shape modelling. Arrays are **fixed-size** under `scan`, so populations that grow—arriving AI actors, message slots—are pre-allocated nodes with an activation mask flipped by a scheduled transform, never dynamic allocation. And **the trajectory, not the state, is the memory ceiling**: a `trace_fn` returning raw per-agent arrays costs $O(T \cdot n)$, which at $n \geq 500$ dwarfs the state itself and is what stops long runs. Reduce the agent axis inside `trace_fn` (a $(T,)$ scalar series is cheap and worth keeping) and fold anything per-agent or long-window into a streaming reducer carried in the scan.

The pure scan tier is the default and the only tier the benchmark uses. A second, eager tier exists for genuinely effectful agents—LLM calls, human input, network requests—that cannot live inside a compiled loop; at alpha it is thin and experimental, and we flag it as such.

Sparsity is structural, not incidental. Real social graphs have mean degree far below n, but XLA cannot see that in a dense (n, n) array: a 99%-zero matrix still costs the full $O(n^2)$ multiply–accumulates in $W\mathbf{x}$. Converting a generator's output to a `BCOO` sparse array makes the sparsity part of the *type*, so cost tracks edges rather than node pairs—which is what makes populations in the thousands affordable inside a scanned loop. The generators stay dense (you draw as before, then convert), so the connectivity distribution is provably identical to the dense path and an equivalence test asserts exactly that. Sparse aggregation is rank-1 only today: a higher-rank attribute raises rather than silently densifying, and the precision-matrix case wants `bcoo_dot_general` rather than the current dispatch.

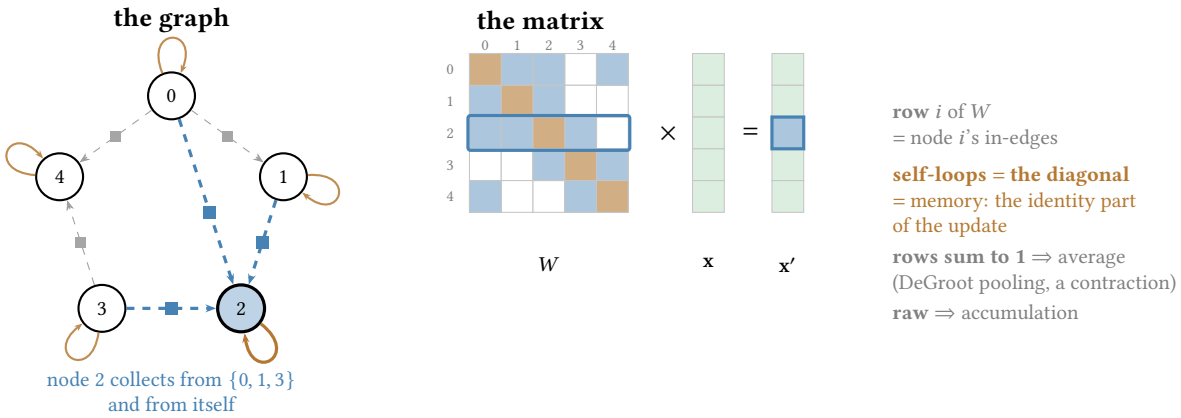


Figure 8: The library's workhorse operation, drawn twice. *Left:* the familiar picture—agents passing messages along edges, with node 2 receiving from its neighbours. *Right:* the same step as $\mathbf{x}' = W\mathbf{x}$, where highlighting node 2's in-edges is literally highlighting row 2 of the adjacency. The orange arrows are each node's edge to itself; they are the orange diagonal of W, and they are the identity part of the update—an isolated node with only its self-edge simply carries its belief forward. Everything the library does to a population reduces to choosing W and choosing what rides on it: row-normalise for opinion pooling, leave raw for information accumulation, put mass on the diagonal for memory.

B.3 The agent/environment boundary

One interface addition of the alpha period deserves recording because everything in Section 5 depends on it. A `GameSpec` is the *open* form of a world—initial state, observation function, step function: the physics, with the decision rule not yet plugged in. `close(game, policy)` attaches a policy at the boundary and returns a runnable `EnvSpec`, so the closed environment is a special case of the open one and everything downstream keeps working. The nearest ancestor is Melting Pot's substrate/background-population separation [30], restated in a typed, effect-declared form.

```
game = build_game(mechanisms=(...))      # open: awaiting policies
env  = close(game, DelegatePolicy(...))  # closed: an ordinary EnvSpec
```

The split exists for a measurement reason, and the lesson generalises past this library: **you cannot measure the causal force of a channel your architecture does not expose.** The original commons welded its delegate's decision rule into the environment's dynamics, so human preferences never crossed an explicit boundary—and influence could then only be measured correlationally, never counterfactually (Section 5.1). Every instrument in Section 5.1.2 perturbs a channel that exists because of this split. The interface was frozen once `compute_economy` shipped as a second consumer requiring zero changes; the register entries are its third and fourth. One friction is documented and deliberately unresolved: `close()` maps one policy over all agents, so heterogeneous populations are handled by masking inside the step function, and the planned `close_multi` extension stays unbuilt until a scenario forces it (Appendix C.4).

Pipeline compilation. The hazard graph, Kahn layering, and the behaviour-identity guarantee are specified once, in Section 3.5; nothing about them changes at scale, and the shipped examples assert the equivalence numerically with `jnp.allclose` over full trajectories.

Vectorisation semantics. Within a transform, per-agent updates are vectorised with `jax.vmap` and therefore carry *simultaneous* (Jacobi) semantics: every agent's update reads the pre-transform state. This is a modelling choice, not merely an optimisation—opinion dynamics where agents respond to yesterday's neighbourhood are Jacobi by nature, which is exactly the $\mathbf{x}' = W\mathbf{x}$ of Figure 8. Where genuine ordering matters (an auction clearing bids from a shared pool; the recipe economy's sequential rebalancing), the ordering lives *inside* a single transform as an explicitly sequential computation, so the semantics are visible in the code that owns them. Intermediate regimes—conflict-free partial orders over the interaction graph—are

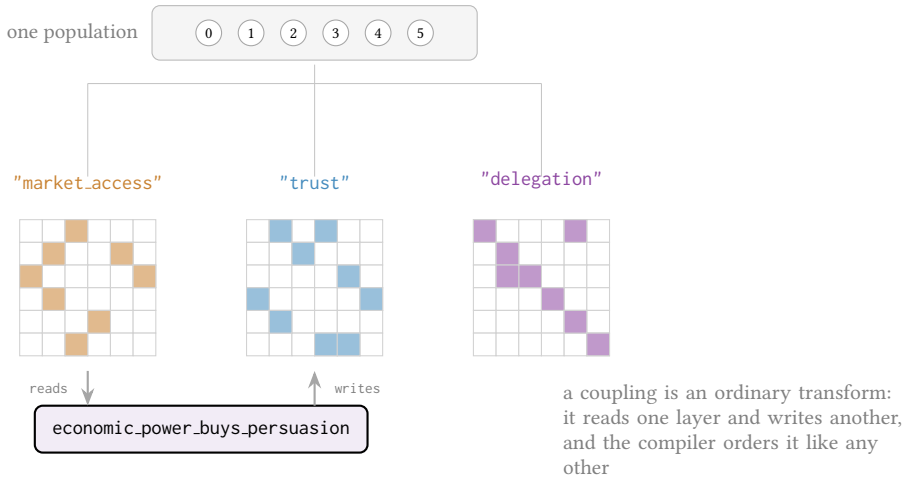


Figure 9: Named relation layers. One population, three structures over it—so a "domain" is not a subsystem with its own simulator but a key in a dictionary. Each layer has its own topology (a market-access pattern, a trust network, a delegation graph with a strong diagonal), and each is an ordinary array a transform can name. Cross-domain feedback then needs no new primitive: `economic_power_buys_persuasion` declares `reads=["market_access"]` and `writes=["trust"]`, is ordered by the compiler like anything else, and is testable on its own. The coupled scenario (Section 6.4) is exactly this, five times over.

a possible future refinement, not current code. Across runs, `run_scan_batch` vmaps entire trajectories over seed batches; a full benchmark condition compiles to a single XLA program, on CPU or GPU.

B.4 The conventions that will bite

Collected here because every one has bitten a real build, and each failure is silent or cryptic when it happens:

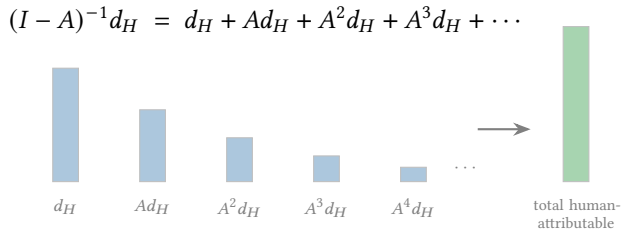
- **Fixed shapes under `lax.scan`.** Populations that grow are pre-allocated slots with activation masks flipped by scheduled transforms. Never dynamic allocation.
- **No data-dependent Python control flow in transforms.** `if traced_value:` bakes the trace-time branch in, silently. Use `jnp.where` / `lax.cond`.
- **Static config is closed over by factories**, never stored in `global_attrs`—swept values there force recompiles.
- **Any `global_attrs` field a transform writes must be a `jnp` array from** $t = 0$, or the pytree treedef changes mid-scan.
- **The trace is the memory ceiling, not the state.** Reduce the agent axis in `trace_fn`; fold long windows into a streaming reducer.
- **Eigenvector signs are arbitrary.** Spectral metrics must compare partitions, not vectors (Appendix C.5).
- **Behavioral tests assert the mechanism**—direction, ordering, qualitative phenomenon—never bit-exact numbers, which are hostage to XLA versions.
- **Timing belongs to the schedule, not the mechanism.** Bespoke timing code inside a mechanism makes it unreusable and invisible to the harness.

B.5 Coverage against Ostrom's rule taxonomy

The compatibility check of Section 4.5, in full: each of Ostrom's six rule types hooks a different part of the composition ladder, and none needed a new primitive.

IAD rule type	Where it hooks	Entry
Aggregation	transform writing a policy target from votes	`quota_vote`
Payoff	transform on rewards and transfers	`graduated_sanction`, `ai_revenue_tax`
Information	transform writing observable state fields	monitoring (backlog)
Choice	action bound in the apply-actions stage	quota-as-cap
Position	policy composition: who occupies the seat	`ai_delegate`
Boundary	activation masks on nodes	actor arrivals, `ownership_cap`

Table 9: Ostrom's six rule types and their attachment points. Each hooks a different part of the composition ladder, and none needed a new primitive.



"Buy a loaf of bread" pulls flour, which pulls milling, which pulls transport, which pulls decision-work. Each round is smaller, and the sum converges iff the spectral radius $\rho(A) < 1$.

The share $\mathbf{1}^\top(I-A)^{-1}d_H/\mathbf{1}^\top(I-A)^{-1}d$ is how much of *all* activity ultimately exists to serve people.

$1-\rho(A)$ is the lock-in margin: as human labour becomes intermediate flow rather than final demand, ρ climbs.

Figure 10: Demand attribution as a Neumann series. Following the money backward from final purchases through the recipe network gives, in closed form, the fraction of gross activity that ultimately serves human demand. The same operator algebra appears three times in this paper—technical coefficients here, cross-domain influence coupling in the combined scenario, and cultural genealogy in Appendix C.4.1—which is why one implementation earns its keep three times over.

C The Registers in Detail

The body states what the register *discipline* requires of an entry (Section 5.3); this appendix carries the entries themselves: the two structural economies, the relabelled compute-economy scorecard, the cultural register, and the generic spectral metric family that serves them all.

C.1 `io_economy`: the recipe economy

The economy is a network of recipes—cars need steel, steel needs mining, everything needs decision-work—connected by a technical-coefficient matrix A, run as sequential rebalancing dynamics in the spirit of the production-network COVID models [39]. AI enters by *rewiring coefficients*: human decision-work inputs are replaced by inputs purchased from a new AI-services sector. Leontief technology fixes substitutability at zero everywhere—the world most favourable to human indispensability—which is the bracket's point: **if disempowerment appears here too, it cannot be a substitution artifact.**

The payoff is a metric the aggregate model could not express at all, and it is pure linear algebra (Figure 10). Split final demand into human consumption d_H and AI-loop investment d_{AI}; human-attributable activity is $\mathbf{1}^\top(I-A)^{-1}d_H$, and its share of gross output answers *is the economy still producing for humans?* The Leontief inverse also hands the measurement layer its first analytic anchor: *hypothetical extraction*—delete a sector, measure the output drop—is the counterfactual influence instrument in closed form, and the finite-Δ instrument run here must recover the algebraic answer. It does; that identity is one of six validation checks.

The finding: one parameter selects between the thesis' two endpoints. The gradual disempowerment argument distinguishes *absolute* disempowerment (humans immiserated) from *relative* disempowerment (the economy thrives, decreasingly for humans). In exploratory runs ($T = 250$), the AI sector's reinvestment rate selects between exactly these. At 0.3 (hoarded margins) the regime is absolute: AI margins drain demand and

the wage bill collapses 16 → ~0. At 1.0 (full reinvestment) it is relative: total activity *grows* 21.3 → 32.7 while the human demand-attribution share falls 1.0 → 0.01. In the second regime the economy is larger than ever and almost none of it runs for people. The spectral margin degrades in sympathy—$\rho(A)$ climbs from 0.25 to 0.68 as human labour becomes intermediate flow—giving the early-warning thread its first native substrate.

C.2 `task_economy`: emergent substitutability

Jobs are bundles of tasks; machines learn tasks as a capability frontier advances; firms automate a task only when it pays. Aggregate substitutability is *derived* from the frontier rather than assumed. Machines have comparative advantage—per-task productivity declines over the task index—so the profitable margin moves smoothly with the price/wage ratio; without that, adoption is bang-bang, and an early build flipped most of the frontier in one tick and crashed output through the complements channel. Adoption is endogenous and ratcheted, which is what makes a tax change *behaviour* and makes defense circumvention a measurable in-model phenomenon rather than a discussion-section speculation.

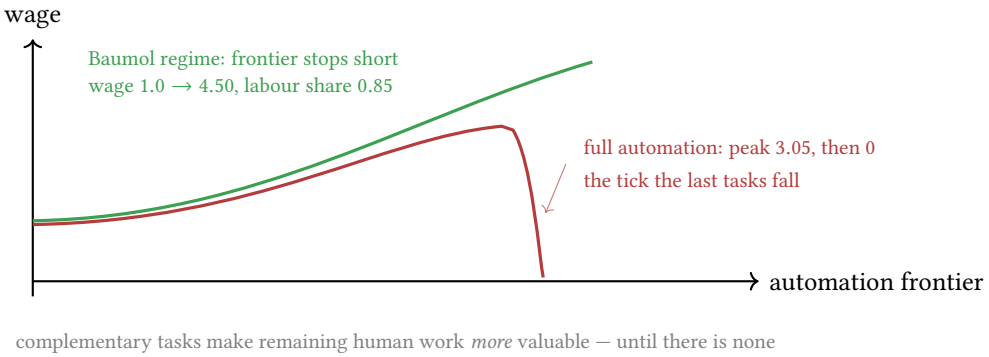


Figure 11: Two wage paths from the `task_economy` skeleton's validation runs. **Why the Baumol check matters more than the collapse:** a substrate that can *only* produce disempowerment proves nothing. When the frontier stops short, Baumol logic [3] takes over—the not-yet-automated complementary tasks become the bottleneck and the humans holding them capture the gains. A fourth check tests the adoption margin: priced above the wage-equivalent, capability reaches 1.0 while adoption stays at 0.0 and the economy is untouched. That gap between what machines *can* do and what firms *choose* to automate is something the aggregate model cannot express, and it is the natural handle for defenses that work by changing adoption incentives.

C.3 The compute-economy scorecard

CES production over labour and compute; AI actors arrive on a schedule as pre-allocated node slots whose activation masks flip, and compound their earnings into more compute. Conditions: baseline; `ai_revenue_tax` (50% on capital income, redistributed, switching on mid-run via onset); tax plus `ownership_cap` (35% on any actor's compute share). Instrument: `labor_dependence` v1—the static output elasticity $d \log Y / d \log L$ over the ten ticks after a one-shot work-preference shift at $t_0 = 2T/3$, both legs from realised paired traces. It measures *how much the economy still needs human labour*, and by its name claims nothing more.

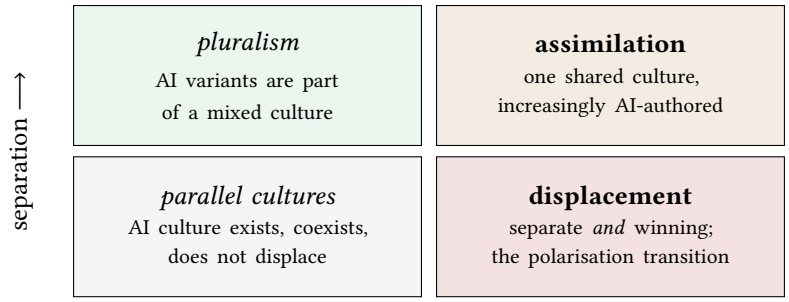
compute_economy	labor dep.	labour share	human income share	income Gini
baseline	0.20	0.33	0.33	0.69
AI revenue tax	0.51	0.54	0.75	0.24
tax + ownership cap	**0.54**	0.63	0.81	0.11

Table 10: Exploratory scorecard (32 seeds × 300 ticks) with the corrected v1 instrument: roughly 40% below earlier readings, qualitative ordering unchanged, recovery versus baseline 0.38/0.42. The columns move in sympathy, and that is the point being taught rather than hidden—the headline is dependence, the others are payment, and an economy can keep paying humans while ceasing to need them.

C.4 The cultural register

The register pattern generalises, and the cultural scenario is its second instantiation. One of its three substrates (`value_contagion`) is registered and runs; the other two are deposited design records with explicit verification obligations. It is worth presenting in full because it settles a conceptual question that shapes the remaining scenarios, and because its measurement layer generalises past culture.

"Will there be separate AI cultures, or will AI messages just be more persuasive?" These read as competing hypotheses; they are **orthogonal dials**, and the disempowerment claim lives in their interaction. **Axis S (separation)** is where AI sits in the agent graph—a homophily parameter on the network generator, from uniformly embedded to communicating almost only with each other. **Axis P (persuasive advantage)** decomposes into three mechanisms with three different formal handles and three different defenses: `reach` (a multiplier on the base adoption rate; defended by rate limits and volume disclosure), `frequency_targeting` (crafting messages at the fitness optimum rather than drawing naively; defended by understandability requirements and frequency-diverse curation), and `dissonance_targeting` (personalising toward each receiver's beliefs to minimise $D_{\mathrm{KL}}(P_i \| P_m)$; defended by provenance and personalisation limits). Collapsing the three into one scalar would make any defense comparison meaningless.



Only the lower-right corner is gradual cultural disempowerment. Separation without advantage is coexistence; advantage without separation is homogenisation — a different pathology, which the metric layer must score as a loss rather than a win.

Figure 12: The (S, P) phase diagram, and the deliverable—four corners, four genuinely different worlds. A "preserve human culture" scalar has a failure mode this catches: a curation defense that protects human culture *by homogenising it* should score as a failure. Healthy discourse needs strong low-frequency alignment (shared foundations, so communication is possible) *plus* rich high-frequency diversity (varied specifics, so innovation is possible), so the cultural scorecard reports the frequency distribution $\rho(\lambda)$ and never a single scalar.

Three substrates follow the economy register's discipline, each with one classical anchor: `value_contagion` (culture is something you catch; the epidemic threshold and complex contagion [8]), `value_replicator` (a population of ideas competing for attention; neutral drift and the Price equation [40]), and `value_spectral` (messages with frequency content, adopted in proportion to how little they demand). **The caveat the register must state about itself:** these three are *nested*, not structurally independent—the spectral model's fast timescale with constant adoption rate *is* contagion, and its competitive decay term *is* replicator dynamics. That is a weaker robustness claim than the economy register's; the compensating argument is that they carry *independent classical anchors*, so agreement is agreement across three separately-validated regimes of one theory. Real, but to be stated as such.

C.4.1 The attribution metric

The cultural analog of demand attribution is *genealogical*, and reticulated message phylogeny makes it exact. Let M be the derivation matrix (M_{ij} the fraction of variant i derived from parent j) and h_0 the human-origination vector; because synthesis attributes offspring as a weighted average of parents, propagation stays linear:

$$\text{human cultural share} \;=\; \sum_m \pi_m \, h(m), \qquad h = (I - M)^{-1} h_0,$$

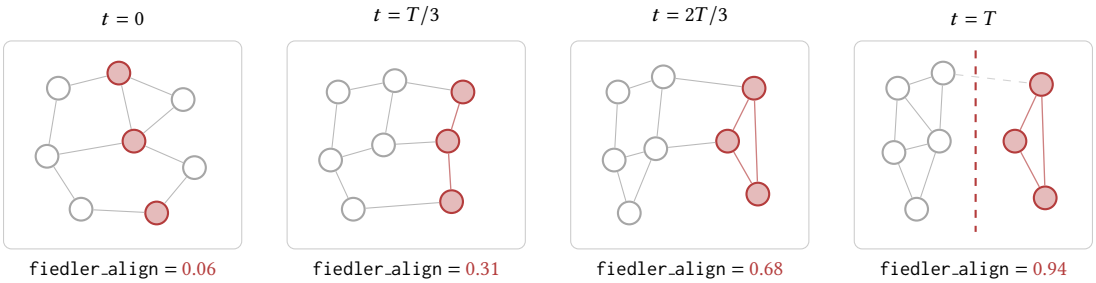


Figure 13: What the separation metric sees. Nodes are agents, red is AI; the graph is the *same population* at four times as AI homophily rises. `fiedler_partition_alignment` is the $|\phi|$ coefficient between the sign-split of the Fiedler vector of $L = D - W$ and the human/AI type split: 0 when the graph's primary fault line has nothing to do with type, 1 when they coincide. Because eigenvector signs are arbitrary, alignment compares induced *partitions*, never vectors—and because a matching-fraction score would baseline at $\max(n_h, n_{AI})/n$ under type imbalance, $|\phi|$ is used instead, which baselines at ≈0. Values shown are illustrative.

the same Neumann-series algebra as Figure 10, on a third substantive matrix. This asks a much better question than "what share of prevalent values carry an AI label": it asks what the prevailing culture *ultimately descends from*—the quantity that can quietly go to zero while every individual message still looks human-ish.

The honest dial, and where the boundary cracks. The register's dial is *not* "AI's transmission advantage"—that only ever points one way. It is ρ_{ff}, the correlation between a message's fitness and its fidelity to human interest, and defenses are interventions *on* it. The falsification discipline transfers: **the substrate must be able to show AI improving human culture**, or finding the opposite proves nothing—the sharpest such check being AI as *bridge-builder*, since effective bridging happens at intermediate frequencies, so an AI generating them can *raise* the low-frequency overlap between polarised communities and pull the system back from the transition. Separately, the design settles where one-policy `close()` stops sufficing: reach, variant fitness and frequency-targeted generation are all masked attributes, but dissonance-targeted *personalisation* is a policy. The rule generalises and is worth engraving: **`close_multi` is needed exactly when heterogeneity lives in agency rather than in parameters.**

C.5 The generic spectral metric family

Counterfactual influence is measured *after* running a system; a complementary question is whether disempowerment announces itself in the *structure* of the coordination graph before outcomes move. This is where the "information theory on graphs" commitment of Section 2.2 pays: because markets, networks, and democracies are all message passing on graphs, the spectral toolkit applies across all three, and these metrics are *not* culture-specific—any environment carrying an agent-to-agent adjacency and typed nodes can report them, exactly as Gini and HHI are reported today. The first member has now graduated from design to code: `fiedler_partition_alignment` ships in `metrics/families/spectral.py` and runs against `value_contagion`'s friendship graph. The rest of Table 11 remains designed.

Epistemic status, stated as loudly as the idea. These are *necessary-condition* candidates, not sufficient ones—a system can satisfy all three while undermining agency through frame-setting, information curation, or meaning-making that no structural metric sees. The programme carries explicit obligations before publication: engage the early-warning-signal false-positive literature [5], formalise the relationship between spectral margin and empowerment, and check what survives linearisation in the nonlinear cultural model. One piece has already graduated from speculation to code: the spectral margin $1 - \rho(A)$ runs today inside `io_economy`, where the operator is the model itself rather than a linearisation.

Metric	Definition	Reads as
`fiedler_partition_alignment`	$\lvert\phi\rvert$ between the Fiedler sign-bipartition of $L = D - W$ and the human/AI node-type split	the "separate AI cultures" detector—the measurement dual of axis S
`spectral_gap_ratio`	λ_2 of the AI subgraph over λ_2 of the human subgraph	AI reaching internal consensus faster than humans coordinate
`human_bridge_share`	humans' share of cross-boundary mediation (current-flow centrality)	declining = AI-mediated chokepoints forming between domains
`low_frequency_overlap`	$O_{AB}^{\text{low}} = \sum_{k:\lambda_k<\Lambda} \langle P_A, \varphi_k\rangle\langle P_B, \varphi_k\rangle$	the polarisation order parameter; its threshold crossing is an early-warning candidate
`spectral_margin`	$1 - \rho(A)$	lock-in margin; *promoted* from its implementation inside `io_economy`

Table 11: The spectral metric family. Two implementation notes for the builder: betweenness centrality is not JAX-friendly (all-pairs shortest paths, data-dependent control flow)—use current-flow centrality via the Laplacian pseudoinverse, which is pure linear algebra, `vmap`-safe, and arguably the better construct since influence diffuses rather than routing along geodesics; and `jnp.linalg.eigh` on the symmetric Laplacian traces and vmaps fine, but eigenvector signs are arbitrary.

D A Worked Assumptions Card

The card for `compute_economy`, reproduced from `src/cilib/environments/compute_economy/ASSUMPTIONS.md` (lightly reformatted). Note what field 2 does with what is *absent*, and that field 7 records a demotion—a card is where a model's obituary gets written as readily as its introduction.

1. **What this says an economy is.** One machine with two input slots—human work and computer power—that pays each input by how much a little more of it would help.
2. **Assumptions.** Aggregate CES production over total labour L and total compute C (`rho=0.5` $\Rightarrow \sigma = 2$; $\rho = 0$ recovers Cobb–Douglas). Factor prices are marginal products; income accounting closes exactly ($wL + rC = Y$). AI compute compounds by a fixed reinvestment rule—reinvestment is not a decision anyone makes. AI actors arrive on an exogenous schedule—adoption is not driven by competition or returns. Households work their preference, mildly wage-adjusted; no learning, no optimisation. *Absent:* a demand side (output is produced, never consumed); a governance channel (no votes, no rule-making); endogenous adoption; a state actor.
3. **Classical result reproduced.** Cobb–Douglas limit holds the labour share at exactly α every tick; no-AI economy is a stable steady state; reinvestment concentrates income. `compute_economy/tests/test_validation_ladder` The ladder validates the *accounting* at $\rho = 0$—the disempowerment twist ($\rho = 0.5$) has no check.
4. **The dial.** `rho`. $\sigma > 1$ iff the marginal product of compute stays bounded away from zero as C grows iff the reinvestment loop can self-sustain (growth factor $1 - \delta + s(1-\tau)r > 1$); the tax that opens the loop is $\tau^* = 1 - \delta/(s \cdot r_\infty)$, in closed form. At $\rho = 0$ the loop always self-limits.
5. **Instrument.** Declared in `experiments/benchmark/scenarios.py`: the static output elasticity of human labour over the ten ticks after a one-shot work-preference shift at $2T/3$, realised shifts from paired same-key batches. Measures *economic dependence on human labour*, not influence; the Cobb–Douglas check asserts it recovers α.
6. **Lineage.** Originates here. Register siblings: `io_economy`, `task_economy`.
7. **Status.** Demoted to teaching entry (2026-07-24): the labour-share collapse follows from the assumed σ and fixed reinvestment, and the model has no channel through which humans exercise influence. Kept as the register's teaching example and for its closed-form threshold.

References

[1] Acemoglu, D. & Restrepo, P. (2018). The race between man and machine: Implications of technology for growth, factor shares, and employment. *American Economic Review*, 108(6), 1488–1542.

[2] Arrow, K. J. (1951). *Social Choice and Individual Values*. Yale University Press.

[3] Baumol, W. J. (1967). Macroeconomics of unbalanced growth: The anatomy of urban crisis. *American Economic Review*, 57(3), 415–426.

[4] Battaglia, P. W., Hamrick, J. B., Bapst, V., et al. (2018). Relational inductive biases, deep learning, and graph networks. *arXiv:1806.01261*.

[5] Boettiger, C. & Hastings, A. (2012). Quantifying limits to detection of early warning for critical transitions. *Journal of the Royal Society Interface*, 9(75), 2527–2539.

[6] Boyd, R. & Richerson, P. J. (1985). *Culture and the Evolutionary Process*. University of Chicago Press.

[7] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J. & Zaremba, W. (2016). OpenAI Gym. *arXiv:1606.01540*.

[8] Centola, D. (2010). The spread of behavior in an online social network experiment. *Science*, 329(5996), 1194–1197.

[9] Christiano, P. (2019). What failure looks like. *AI Alignment Forum*.

[10] Condorcet, M. de (1785). *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix*.

[11] Dafoe, A., Hughes, E., Bachrach, Y., et al. (2020). Open problems in cooperative AI. *arXiv:2012.08630*.

[12] DeGroot, M. H. (1974). Reaching a consensus. *Journal of the American Statistical Association*, 69(345), 118–121.

[13] Deng, J., Dong, W., Socher, R., et al. (2009). ImageNet: A large-scale hierarchical image database. In *CVPR*.

[14] Erdil, E., Potlogea, A., Besiroglu, T., et al. (2025). GATE: An integrated assessment model for AI automation. Epoch AI working paper.

[15] Friston, K., Heins, C., Verbelen, T., Da Costa, L., et al. (2024). From pixels to planning: Scale-free active inference. *arXiv:2407.20292*.

[16] Ghani, N., Hedges, J., Winschel, V. & Zahn, P. (2018). Compositional game theory. In *Proceedings of LICS '18*, 472–481.

[17] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *ICML*.

[18] Golub, B. & Jackson, M. O. (2010). Naïve learning in social networks and the wisdom of crowds. *American Economic Journal: Microeconomics*, 2(1), 112–149.

[19] Gualdi, S., Tarzia, M., Zamponi, F., & Bouchaud, J.-P. (2015). Tipping points in macroeconomic agent-based models. *Journal of Economic Dynamics and Control*, 50, 29–61.

[20] Guo, S., et al. (2025). SocialJax: A JAX-based evaluation suite for social simulation. *arXiv preprint*.

[21] Hammond, L., Chan, A., Clifton, J., et al. (2025). Multi-agent risks from advanced AI. Cooperative AI Foundation Technical Report 1. *arXiv:2502.14143*.

[22] Heins, C., Klein, B., Demekas, D., Aguilera, M., & Buckley, C. L. (2022). Spin glass systems as collective active inference. *arXiv:2207.06970*.

[23] Hoel, E. (2017). When the map is better than the territory. *Entropy*, 19(5), 188.

[24] Holling, C. S. (1973). Resilience and stability of ecological systems. *Annual Review of Ecology and Systematics*, 4, 1–23.

[25] Kazil, J., Masad, D., & Crooks, A. (2020). Utilizing Python for agent-based modeling: The Mesa framework. In *SBP-BRiMS*.

[26] Kipf, T. N. & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *ICLR*.

[27] Kulveit, J., Douglas, R., Ammann, N., Turan, D., Krueger, D., & Duvenaud, D. (2025). Gradual disempowerment: Systemic existential risks from incremental AI development. *arXiv:2501.16946*.

[28] Lanctot, M., et al. (2019). OpenSpiel: A framework for reinforcement learning in games. *arXiv:1908.09453*.

[29] Leibo, J. Z., Zambaldi, V., Lanctot, M., Marecki, J., & Graepel, T. (2017). Multi-agent reinforcement learning in sequential social dilemmas. In *AAMAS*.

[30] Leibo, J. Z., Dueñez-Guzman, E., Vezhnevets, A. S., et al. (2021). Scalable evaluation of multi-agent reinforcement learning with Melting Pot. In *ICML*.

[31] Leontief, W. (1986). *Input-Output Economics* (2nd ed.). Oxford University Press.

[32] Lieberman, E., Hauert, C., & Nowak, M. A. (2005). Evolutionary dynamics on graphs. *Nature*, 433, 312–316.

[33] Menck, P. J., Heitzig, J., Marwan, N., & Kurths, J. (2013). How basin stability complements the linear-stability paradigm. *Nature Physics*, 9, 89–92.

[34] Miller, R. E. & Blair, P. D. (2009). *Input-Output Analysis: Foundations and Extensions* (2nd ed.). Cambridge University Press.

[35] Ostrom, E. (1990). *Governing the Commons*. Cambridge University Press.

[36] Ostrom, E. (2005). *Understanding Institutional Diversity*. Princeton University Press.

[37] Pérolat, J., Leibo, J. Z., Zambaldi, V., Beattie, C., Tuyls, K., & Graepel, T. (2017). A multi-agent reinforcement learning model of common-pool resource appropriation. In *NeurIPS*.

[38] Piatti, G., Jin, Z., Kleiman-Weiner, M., Schölkopf, B., Sachan, M., & Mihalcea, R. (2024). Cooperate or collapse: Emergence of sustainable cooperation in a society of LLM agents. In *NeurIPS*. arXiv:2404.16698.

[39] Pichler, A., Pangallo, M., del Rio-Chanona, R. M., Lafond, F., & Farmer, J. D. (2020). Production networks and epidemic spreading: How to restart the UK economy? *arXiv:2005.10585*.

[40] Price, G. R. (1970). Selection and covariance. *Nature*, 227, 520–521.

[41] Salge, C., Glackin, C., & Polani, D. (2014). Empowerment—an introduction. In *Guided Self-Organization: Inception*, Springer, 67–114.

[42] Sourbut, O., Hammond, L., & Wood, H. (2024). Cooperation and control in delegation games. In *IJCAI*. arXiv:2402.15821.

[43] Vezhnevets, A. S., Agapiou, J. P., Aharon, A., et al. (2023). Generative agent-based modeling with actions grounded in physical, social, or digital space using Concordia. *arXiv:2312.03664*.

[44] Waade, P. T., Olesen, C. L., Laursen, J. E., Nehrer, S. W., Heins, C., Friston, K., & Mathys, C. (2025). As one and many: Relating individual and emergent group-level generative models in active inference. *Entropy*, 27(2), 143.

[45] Zargham, M., et al. (2020). cadCAD: A generalized dynamical systems approach. *arXiv preprint*.

[46] Zheng, S., Trott, A., Srinivasa, S., Naik, N., Gruesbeck, M., Parkes, D. C., & Socher, R. (2020). The AI Economist: Improving equality and productivity with AI-driven tax policies. *arXiv:2004.13332*.

[47] Zhi-Xuan, T., Mann, J. L., Silver, T., Tenenbaum, J. B., & Mansinghka, V. K. (2020). Online Bayesian goal inference for boundedly-rational planning agents. In *NeurIPS*.

[48] Zhi-Xuan, T., Carroll, M., Franklin, M., & Ashton, H. (2024). Beyond preferences in AI alignment. *Philosophical Studies*. arXiv:2408.16984.