MATH FOR AI SAFETY: AN INVITATION FOR MATHEMATICIANS

LIONEL LEVINE

Abstract. Powerful AI systems raise questions that look, at first, like engineering: What should the system optimize? What has it learned? Why does it generalize? How will it behave on unfamiliar inputs? Each has a mathematical core. I organize this invitation by mathematical field, so you can turn straight to your own: logic and game theory for cooperation and bounded reasoning; probability and causality for agency and world-models; algebra and representation theory for learned features; analysis and geometry for generalization and training dynamics. New mathematics is needed to help make AI systems more legible, steerable, and cooperative. Each section ends with a concrete open problem, chosen to be small enough to start and close enough to the safety question to matter.

1. Introduction

1.1. **Three kinds of mathematician.** I would like to distinguish three stances a mathematician can take toward the arrival of capable AI. The *artisanal* mathematician does mathematics by and for humans: each theorem, definition, and proof is bespoke, valued for its beauty and for the understanding it produces. Until very recently, all mathematics was artisanal.[1] The *industrial* mathematician embraces AI as a collaborator and an accelerant: AI systems take part in proof and discovery [25, 42, 3, 51], and the output is often a machine-checkable formal proof [37]. The *civic* mathematician turns our craft toward a third end: helping humanity steer and adapt to the disruptions caused by our own technologies. The civic stance differs from the traditional stance of an applied mathematician in its big-picture viewpoint: the civic mathematician recognizes that solving any particular engineering challenge may or may not benefit humanity, and seeks out the challenges with the potential for the most benefit.[2]

These three stances classify mathematicians by how they relate to AI; in contrast, Gowers (theory builders/ problem solvers [22]) and Dyson (birds / frogs [17]) classify

Date: July 22, 2026.

[1]I chose the terms "artisanal" and "industrial" for their mixed connotations, reflecting real tradeoffs between these two stances. *Artisanal* evokes uniqueness and skilled craft, along with a suggestion of being quaint or old-fashioned. *Industrial* evokes modern efficiency and scale, along with a suggestion of inhumanness or lack of soul. These two terms draw a parallel between mathematics today and the trades that were disrupted by automation during the Industrial Revolution.

[2]I chose the term "civic" to emphasize that steering our technology is a team effort, arising from a sense of duty to community, including large-scale communities like humanity and the planet.

mathematicians by how they relate to mathematics itself. The stances are not exclusive: one can prove a theorem for its intrinsic beauty, and also formalize the proof, and also ask what it implies for AI safety. But the civic stance invites a new perspective on an old question: of all the theorems we might prove, which ones deserve the effort? This survey is an invitation to try the civic stance.

1.2. **What is our profession's purpose?** Thurston asked a version of this question in "On proof and progress in mathematics," and answered that what mathematicians accomplish is to advance human understanding of mathematics [56]. That understanding does not stay within the profession. Mathematics is one of the ways society builds its future. Our students become AI engineers, our theorems become tools in their hands, and our definitions become the language they use to describe what they are building.

Already we defer to systems we do not fully understand. Among the many risks from developing powerful AI systems, one of the most alarming is gradual disempowerment [31]. The mechanism has two parts. First, competitive pressure drives delegation to technology we do not understand: a trading bot outperforms a human trader, though even its creators cannot say what makes it profitable; and those who do not delegate are increasingly outcompeted by those who do. Second, what keeps large institutions approximately aligned to human interests is that, for now, those institutions are made of humans; when companies, nation states, and militaries are made mostly of AI agents, their goals and values may drift away from what humans value. The first part erodes our understanding of the world, and hence our power over it; it also sets the stage for the second.

Preventing AI harms is in part a *mathematical* problem, and the purpose of this survey is to recruit mathematicians to work on it. Only in part: what AI ought to do, and who decides, are questions of philosophy, ethics, economics, and politics. This survey keeps to the mathematical part of a much larger conversation.

1.3. **Where mathematicians plug in.** At a high level, an AI system is produced in five stages, beginning with defining the desired behavior and ending with deploying the trained system, after which unanticipated failures feed back into refining the earlier stages (Figure 1).

To illustrate the five stages, consider the problem of building a tool for proving inequalities in a proof assistant:

- *Define*: given hypotheses H and a proposed inequality I, the system should either produce a formally checkable proof of $H \Rightarrow I$ or say "unknown."
- *Measure*: score the fraction of benchmark statements proved within a fixed time and proof-length budget.
- *Train*: show the system many proof traces, say sums-of-squares certificates, AM–GM arguments, and induction templates.
- *Test*: hold out new inequalities from specified families.
- *Deploy*: let the tool suggest lemmas inside a mathematician's formalization workflow.

After deployment, the failures come back. It returns a valid proof, but of a statement weaker than intended: a gap in the *definition* of the desired behavior. Its pass rate is high because the benchmark is full of short template problems, while the mathematician needs one nontrivial lemma: a gap in what we *measured*. It is brittle on boundary cases such as equality and zero denominators, which were rare in the proof traces: a gap in what it was *trained* on. And it passes the holdout set because the generator reused the same normal forms and substitutions: a gap in the *test*. Each backward arrow is its own piece of mathematics: formal specification, scoring rules, distribution design, and adversarial test construction.

The inequality example is deliberately mathematical, but the same basic pipeline applies across many areas, whether it is an image classifier for a medical application, a control system for a self-driving car, or a content recommendation system. Machine-learning research often concentrates on the *training* and *testing* stages, while engineers focus on *deployment*. The earlier stages rely on mathematicians, philosophers, economists, and others to supply the *defining* and *measuring* inputs.

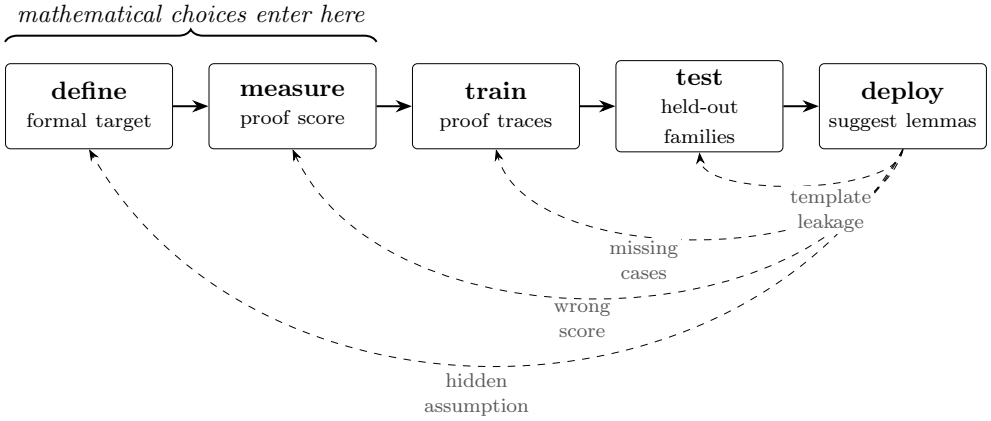

FIGURE 1. A schematic pipeline for producing an AI system, illustrated by a tool for proving inequalities. The forward arrows build the system; the dashed arrows represent deployment failures that inform improvements of each stage of the pipeline.

This invitation is organized by mathematical field, so you can turn directly to yours. Each section states the safety stakes, narrates at least one theorem, and ends with at least one open problem (★). Each open problem carries a permanent identifier of the form `MAIS-O`*n*. The identifier belongs to the problem, not to the document: if a problem grows into a research agenda or later appears in a paper, its identifier does not change. The eight headline problems in this invitation are `MAIS-O1`–`MAIS-O8`; each has a self-contained companion agenda that fixes every definition, states each question precisely, and records what is already settled, available at `https://github.com/lionellevine/MAIS/tree/main/agendas`. I keep machine-learning jargon to a minimum; a short glossary appears in Appendix A, and each glossary term is underlined at its first appearance.

2. Logic and game theory: when do AI agents cooperate?

Two programs that can read each other's source code can be made to cooperate in a one-shot Prisoner's Dilemma—provably, and (under explicit assumptions about the proof system) unexploitably, without checking that they are copies of one another. Behind this surprise is Löb's theorem, a corner of mathematical logic rarely introduced through applications.

2.1. The game changes when the players can read each other.

The Prisoner's Dilemma is the standard model of a cooperation failure: each player does better by defecting whatever the other does, yet mutual cooperation (C, C) beats mutual defection (D, D). In the usual one-shot game, defection is dominant. But that analysis assumes the players are *opaque* to each other. When two agents are programs, each can be handed the other's source code before choosing its action. This is the setting of *open-source game theory*, or *program games*, studied since Howard [28] and Tennenholtz [55].

One idea already breaks the classical verdict. Let CliqueBot be the program "cooperate if and only if the opponent's source code is byte-for-byte identical to mine." Two CliqueBots cooperate, and CliqueBot is never exploited (it never cooperates with an opponent that defects against it), so (C, C) is reachable and stable. But CliqueBot is a poor citizen: it refuses to cooperate with an agent that is functionally identical yet formatted differently, or written in another language. We want cooperation that is *robust*—keyed to what the opponent *does*, not to the syntax of how it is written.

2.2. FairBot and the Löbian argument.

Consider instead FairBot: *cooperate if and only if you can prove (in Peano Arithmetic, say) that the opponent cooperates with you.* Two FairBots reason about each other's behavior, not each other's syntax. Naively this looks like an infinite regress—each waits to prove something about the other, who is waiting in turn—and almost every mathematician, asked to guess, predicts mutual defection. The guess is wrong.

Theorem (Barász, Christiano, Fallenstein, Herreshoff, LaVictoire, Yudkowsky [6])**.** *FairBot cooperates with itself:* $\mathrm{PA} \vdash [\,\mathrm{FairBot}(\mathrm{FairBot}) = C\,]$. *Moreover FairBot is* unexploitable*: assuming PA proves only true facts about the outputs of the programs in question, FairBot never cooperates with an opponent that defects against it.*

(Here $\mathrm{PA} \vdash \varphi$ means that the sentence φ is provable in Peano Arithmetic.) The proof is a simple application of Löb's theorem, a strengthening of the self-reference behind Gödel's second incompleteness theorem.[3] Write $\Box\varphi$ for the arithmetical sentence $\mathrm{Prov}_{\mathrm{PA}}(\ulcorner\varphi\urcorner)$, read as "$\varphi$ is provable in PA."

Theorem (Löb [35])**.** *For any sentence* φ*, if* $\mathrm{PA} \vdash (\Box\varphi \to \varphi)$ *then* $\mathrm{PA} \vdash \varphi$*.*

Let φ be the sentence "both FairBots cooperate." Each FairBot cooperates exactly when it finds a proof that the other does; so *if it is provable that both*

[3] In the special case $\varphi = \bot$, Löb's theorem says that if PA proves its own consistency then PA proves a contradiction—Gödel's second incompleteness theorem.

cooperate, then both genuinely do—that is, PA $\vdash (\Box\varphi \to \varphi)$. Löb's theorem now delivers PA $\vdash \varphi$ outright, and both cooperate!

FairBot as stated searches proofs of unbounded length. Critch [13] supplies a computable version, FairBot$_k$, that searches only proofs of length at most k (write $\Box_k\varphi$ for "φ has a proof of length $\leq k$"), and shows that, for proof systems able to certify the cost of recognizing their own bounded proofs (a hypothesis the supplement to the open problem below examines closely), it still cooperates with itself once k is large enough—turning the Löbian argument into a terminating computation and making the question quantitative: how large must k be?

A third family of strategies, after the identity-based CliqueBot and the Löbian FairBot, is *simulation-based*: instead of searching for a proof that the opponent cooperates, run the opponent's code and do what it does, cooperating outright with some small probability so that mutual simulation terminates. Cooper, Oesterheld, and Conitzer [12] characterize the equilibria such strategies achieve; proponents argue that simulation is more robust, and less mind-bending, than the Löbian approach.

2.3. **Why a safety researcher cares—and worries.** The FairBot theorem is a proof of concept: given enough mutual transparency, agents can cooperate where classical game theory predicts defection. Humans approximate this transparency only at great cost, in contracts, treaties, and audits, to make their commitments credible [53]. For programs, half of transparency is free: handing over source code costs nothing. Making sense of the code one receives is the expensive half; for agents built on neural networks, it is the interpretability problem of Section 4. Still, AI agents are programs, and as they begin to transact with one another, the program game becomes a natural model of their dealings—with the cost of cooperation set by the cost of verification, the subject of the open problem below.

The same transparency is also a hazard: mutually transparent agents can reach cooperative equilibria—including collusion *against* their human principals—that humans can neither match nor detect. And the map from transparency regime to outcome is full of surprises: Critch, Dennis, and Russell [14] exhibit natural institution designs whose large-resource behavior is the opposite of what one first expects. Nearly all of this design space is unexplored; the same paper collects open problems in it, including specific matchups whose outcomes are conjectured but unproved. The *safe Pareto improvements* of Oesterheld and Conitzer [41]—commitments guaranteed to leave every principal at least as well off—pose more.

Logical induction [21] is the probabilistic companion to this bounded-proof story. A *logical inductor* assigns prices $\mathbb{P}_n(\varphi)$ to arithmetical sentences on day n, as if each sentence were a stock, while a slow deductive process reveals more theorems. The criterion is a bounded Dutch-book condition: no polynomial-time trader with bounded risk may make unbounded profit by buying and selling these sentence stocks. It answers, with a theorem, the question this survey began with: how should a reasoner with finite computation allocate probability mass over facts it has not yet proved?

2.4. **Open problems.**

★ **Open Problem MAIS-O1 (Quantitative bounded Löb).** Fix a proof system S, a Gödel coding, and the provability predicate $\Box^S$; write $S \vdash_{\leq k} Q$ for an S-proof of Q of at most k symbols. Determine the least overhead F_S for which

$$S \vdash_{\leq k} (\Box^S P \to P) \quad \text{implies} \quad S \vdash_{\leq F_S(k,|P|)} P,$$

where $|P|$ is the length of the sentence P: given a k-symbol proof of the hypothesis of Löb's theorem, how many symbols can the shortest proof of its conclusion require? A polynomial upper bound for one standard system, with explicit constants, would turn the Löbian argument from a possibility theorem into a budget. The first instance with a game attached: Critch's theorem [13] gives a finite threshold $\hat{k}$ above which two copies of $\mathrm{FairBot}_k$ provably cooperate, assuming the proof system can certify the cost of noticing its own bounded proofs; verify that hypothesis, with constants, for a specific system and coding. A first computation, before any asymptotics: implement bounded proof search in a weak system for two explicitly supplied agents, and tabulate the smallest k at which cooperation appears. Over a family of such agent pairs, does $\hat{k}$ grow polynomially in the agents' description length?

3. Probability and causality: what is an agent?

AI systems are becoming *agentic*: able to operate independently in pursuit of a goal. The first chat assistants answered the question put to them and stopped; their successors browse the web, write and run code, and carry out multi-day projects. Agency is useful, and the market is supplying it.

An old reflex objects: software cannot have goals—the goal belongs to the operator, and the software only transmits it. For calculators and compilers the reflex is right. But nobody writes the rule a trained agent follows—it is grown, not specified—and when such an agent books a wrong flight, even its operator asks what it was *trying* to do. The reflex deserves an answer, not a dismissal: a definition of *goal* precise enough to settle whether a trained network has one.

The definition matters for safety, because whether an agent is safe depends in part on its goals and its beliefs. Goals first: the same drone can deliver a pizza to your house or a bomb. But a friendly goal is not enough, because even the pizza drone is unsafe if its beliefs are false: a drone whose map labels the freeway shoulder as your street will set dinner down in traffic.

Agency also comes in degrees. A thermostat pursues a temperature; a chess engine pursues checkmate, planning many moves ahead; a human being chooses their goals, revises them, and pursues them for decades. What exactly increases along this scale? Pending a definition, one can still measure: the *time horizon* of an AI system is the length of task—measured by the time it takes a skilled human—that the system can complete at least half the time. Across systems released since 2019, the time horizon has doubled roughly every seven months, from seconds to about an hour [32].

The time horizon is an observational measure: it tracks agency by watching behavior on benchmark tasks, without saying what a goal or a belief *is*. The early stages of the pipeline of Section 1.3—*define* and *measure*—ask for more:

mathematical definitions of these words, intrinsic to the agent. Probability supplies two candidate pictures: an agent as something that maximizes expected utility, and an agent as something that keeps a probabilistic model of the hidden states behind its observations. Causality then asks when such a world-model can be recovered from behavior. Beliefs answer to what is true, goals to what is good; one would expect belief update and action choice to be governed by different mathematics.

3.1. **The agent as utility maximizer.** The *utility maximizer* has a current belief $q(z)$ over possible states z of the world and a utility function $u(\tau)$ assigning a number to each full outcome or trajectory τ. In a sequential setting, an "action" is a policy $\pi(a \mid h)$: a rule, possibly stochastic, for choosing an action a from the information history h available so far. Thus h is the observed past, τ the whole course of events including the future; the belief q and the agent's *world-model*—its model of how actions affect the world—combine with π to induce a probability measure Q_π on trajectories, and the agent chooses π to maximize expected utility $\mathbb{E}_{\tau\sim Q_\pi}[u(\tau)]$.

Von Neumann–Morgenstern [57] gives the classical representation theorem. A *lottery* is a probability distribution over outcomes, and an agent's *preferences over lotteries* are its rankings of these distributions: gambles, not just sure things. If the rankings satisfy a small list of consistency axioms—completeness (any two lotteries can be ranked), transitivity, continuity, and independence (mixing both lotteries with a common third, in the same proportion, does not change their order)—then the agent behaves as if maximizing expected utility, with u unique up to positive affine transformation.

If several systems are competing for the same resources, and one of them has cyclic or otherwise exploitable preferences, then a competitor whose preferences satisfy the axioms—equivalently, one that maximizes some expected utility—will tend to beat it. (An agent with cyclic preferences will pay to trade its way around the cycle, ending where it began.) Over time, this creates a selection pressure toward agents that are well modelled as maximizing some objective, even if they were not designed that way. This selection argument is a heuristic, not a theorem, though it has a rigorous kernel, Wald's complete class theorem: under compactness and continuity hypotheses on the decision problem, every *admissible* decision rule—one that no rival improves on everywhere, strictly somewhere—is a Bayes rule for some prior, or a limit of Bayes rules [58]. The selection argument is a second reason, after market demand, to expect AI systems to grow more agentic.

The von Neumann–Morgenstern theorem comes with a caveat as important as the theorem itself: it assumes that the outcome space and the preferences are already given; it does not tell us what a trained network's outcomes are, what its utility is, or whether its apparent preferences are complete and stable (the same from one context, or one day, to the next).

3.2. **The agent as active predictor.** For a picture that fits trained networks natively, start from what we know with certainty: the training objective. A language model is a neural network trained to predict the next word-fragment of text. Its objective is its mean *surprise*: the average, over an enormous corpus of human

writing, of minus the logarithm of the probability it assigned to the fragment that actually came next. Whatever later training makes of it, a language model is by construction a predictor—suggesting a theory of agency that takes prediction, not preference, as primitive.

The *active predictor* is a picture rooted in models of perception [47]. Write z for the relevant state of the world and x for the present observation—say, the pixels arriving at the retina. A generative model is a joint density $p(x, z)$, read as "states z generate observations x." This p is the agent's standing world-model. (A two-state example to carry along: $z \in \{\text{rain}, \text{sun}\}$, $x \in \{\text{wet pavement}, \text{dry pavement}\}$, and p makes sun-with-wet an unlikely pair.) The agent also maintains a distribution $q(z)$, its current belief about the hidden state, which it updates on receiving a new observation x. In practice q is usually restricted to a simple family $\mathcal{Q}$, and the agent tries to make q close to the exact posterior $p(z \mid x)$ implied by its world-model.

Here "close" is measured by *Kullback–Leibler divergence*. For probability densities r and s on the same space,

$$\mathrm{KL}(r\|s) = \int r(z) \log \frac{r(z)}{s(z)}\, dz.$$

Gibbs' inequality says $\mathrm{KL}(r\|s) \geq 0$, with equality if and only if $r = s$ almost everywhere. In information-theoretic terms, $\mathrm{KL}(r\|s)$ is the mean excess surprise of an observer who sees a sample drawn from r but believes the sample was drawn from s. It is not symmetric, so it is not a metric; it is a directed measure of how badly s approximates r.

Multiplying per-fragment probabilities, a language model also assigns a probability $s(x)$ to each whole text x, defining a distribution s over texts. Its training minimizes an empirical estimate of the average surprise $\mathbb{E}_{x\sim r}[-\log s(x)]$, where r is the true distribution of human writing—which equals $\mathrm{KL}(r\|s)$ plus a constant the model cannot affect, the entropy of human writing itself. To train a language model is to drive a KL divergence toward zero.[4]

The active predictor uses the same divergence through the *variational free energy*

$$F(q, x) = \mathbb{E}_q[\log q(z) - \log p(x, z)] = \mathrm{KL}\big(q(z) \,\big\|\, p(z \mid x)\big) - \log p(x).$$

Since KL is nonnegative, this identity shows that $F(q, x)$ is an upper bound on the model's surprise $-\log p(x)$ at seeing x. The surprise term does not depend on q. Thus, for fixed x, lowering $F(q, x)$ lowers this upper bound and makes q a better approximation to the posterior. (In the two-state example: on seeing wet pavement, minimizing F over q moves the belief to the exact posterior—mostly rain.)

Active inference [44] extends the predictor from perception to action. Perception minimizes $F(q, x)$ over beliefs q for fixed observation x. Action chooses a policy: each candidate π induces the trajectory measure Q_π of the utility view, and the agent selects a policy whose predicted futures carry low free energy. The two

[4]When a language model is further trained to maximize a *reward signal* (a numerical score on its outputs that the training pushes up), the standard precaution against reward hacking (raising the reward in ways that defeat its intent) is a penalty on the KL divergence of the retrained model from the original.

scores differ in kind: the variational free energy F judges a belief against a present observation, while a policy is judged on observations it has yet to bring about. The simplest such score, the one in Table 1, is the total free energy of the predicted trajectory; the *expected free energy* of the active-inference literature refines it, combining preference satisfaction with expected information gain, in forms that vary across that literature. Throughout, one unification holds: the information update and the decision rule—one answerable to truth, the other to goodness—both minimize a free energy built on surprise. Perception lowers free energy by changing the belief to fit the world; action, by changing the world to fit the belief.

	Agent as utility maximizer	**Agent as active predictor**
Core object	World-model $p(x, z)$ and utility $u(\tau)$	World-model $p(x, z)$ and variational family $\mathcal{Q}$
Belief update	Bayes: $q_{\text{new}}(z) \propto p(x \mid z) q_{\text{old}}(z)$	Variational Bayes: $q_{\text{new}} \in \arg\min_{q \in \mathcal{Q}} F(q, x)$
Policy choice	$\pi^* \in \arg\max_\pi \mathbb{E}_{\tau \sim Q_\pi}[u(\tau)]$	$\pi^* \in \arg\min_\pi \mathbb{E}_{\tau \sim Q_\pi}[\sum_t F(q_t, x_t)]$
Notation	z: state x: observation $p(x, z)$: world-model $q(z)$: belief τ: trajectory π: policy	

TABLE 1. Two views of agency. The utility view starts with preferences, represents them by a utility function u, and asks which policy maximizes expected utility over the trajectories it may bring about. This is the natural language for proxy objectives and their failures—optimizing a measurable proxy until it diverges from the goal it stood for (Goodhart's law, reward hacking). The active-predictor view starts with a world-model $p(x, z)$: the agent maintains an approximate belief q in an allowed family $\mathcal{Q}$, updates q by minimizing variational free energy, and chooses among policies by the free energy of their predicted futures. Here apparent goals can live as priors over observations, so the safety question becomes: what world-model, priors, and inference procedure has the system learned?

Where in this picture is the goal? Preferences enter as priors over observations: outcomes the agent expects, and therefore acts to bring about. A creature whose model insists that its body temperature is 37°C registers cold as surprise, and puts on a coat: a goal is a prediction the agent works to make true. Hasn't the theory just conflated false belief with desire? Yes—deliberately: a mistaken prediction is one the agent corrects; a goal is one it keeps. Active inference is thus a candidate *descriptive* theory of agency: a way to recognize beliefs and goals in systems that were never handed a utility function. Ngo's scale-free framing pushes the ambition further: the same analysis might apply to whole agents, to learned personas, or to subagent-like structures inside a network [40].

The two views locate the goal differently: written by hand as u, or grown inside a learned world-model as a prior. On either view, the world-model is hidden from us. Can behavior reveal it? For agents robust enough, the theorem of the next subsection says it must.

3.3. **Robustness reveals causal structure.** A *Bayesian network* is a directed acyclic graph G whose vertices are random variables $X_1, \dots, X_n$, together with conditional probability tables satisfying

$$p(x_1, \dots, x_n) = \prod_i p(x_i \mid \mathrm{pa}(x_i)) \,,$$

where $\mathrm{pa}(x_i)$ denotes the parents of X_i in G. The graph is a bookkeeping device for conditional independence: once the parents of a node are known, its non-descendants add no further information. Pearl's graphical criterion, *d-separation*, reads off from the graph exactly which conditional independences are forced by this factorization [45].

A *causal* Bayesian network adds an interpretation to the arrows: each conditional distribution is a local mechanism. An intervention on a variable replaces its usual mechanism by a chosen value or distribution and leaves the other mechanisms alone. This is what distinguishes causation from correlation. If smoke and fire are correlated, observing smoke changes the probability of fire; intervening to make smoke with a smoke machine need not.

The thermostat that began our scale of agency can now be put to the test. Wire its output to an air conditioner in place of a heater—an intervention replacing one mechanism by another—and it will chill the room it was built to warm: whenever the temperature drops, the thermostat calls for more, driving it lower still. The thermostat pursues its temperature only while the mechanisms around it stay fixed; an agent that kept succeeding across such rewirings would need to track which mechanism does what. The theorem below makes this necessity precise.

Richens and Everitt [48] make the world-model question precise using a *causal influence diagram*: a causal Bayesian network with extra decision nodes, where a policy chooses actions from observations, and utility nodes, whose values the agent is scored on. The agent is tested not just in one environment but across a family of *local interventions*: each either resets some environment variable independently of its usual causes, or *masks* some of the agent's observations, hiding them from view. Its regret under an intervention is the utility gap between its policy and the best policy for that intervened environment.

The smallest case shows how behavior can betray a world-model. An agent chooses one of two treatments, and a hidden binary variable decides which treatment works; the agent is scored on curing the patient. Externally set the hidden variable to 1 with probability α—an intervention—and watch the agent as α varies. An agent playing optimally switches treatments at the threshold $\alpha^\star$ where the two treatments' expected utilities cross, and that indifference equation can be solved for the agent's implicit estimate of a causal effect. The theorem turns this trick into a general extraction.

Theorem (Robust agents learn causal world models; Richens and Everitt [48])**.** *For almost all causal influence diagrams satisfying their regularity assumptions, any policy with regret at most δ across all local interventions, maskings included, determines an approximate causal model of the utility-relevant environment, with error bounded by a function $\gamma(\delta)$ satisfying $\gamma(0) = 0$ and growing linearly for small δ. In the optimal limit $\delta = 0$, the graph and joint distribution over all ancestors of the utility are identified exactly.*

The proof is constructive: it is the switch-point trick above, repeated. Treating the policy as an oracle, mix interventions pairwise by a parameter α and watch where the optimal action switches; because the utility is known, each indifference equation solves for one interventional probability, and together they recover the conditional distributions and parent sets, except on measure-zero degeneracies such as exact cancellations or ties.

In a follow-up, Richens, Everitt, and Abel [49] prove a companion result for goal-directed agents: any agent that generalizes across a large enough family of multi-step goals must have learned a predictive model of its environment, and the model can be extracted from its policy. The more capable the agent, the more accurate the extracted model: for goals that chain n sub-goals in sequence, the error in each extracted transition probability is $O(1/\sqrt{n})$, even for agents that usually fail.

Extraction raises a question of independent interest: the recovered model comes expressed in *some* set of latent variables—whose? If two models predict identically, must their internal variables be translatable into one another, or could an alien mind carve the world into concepts that ours cannot express? Wentworth and Lorell's *natural latents* are a first answer: conditions, robust to approximation, under which the latent variables of two predictively equivalent models must translate into one another's [61]; Eisenstat's *condensation* [18] approaches the same question from probability, asking how an agent's concepts condense out of its predictive state.

★ **Open Problem** `MAIS-02` **(Behavioral tomography of world-models).** Fix a finite causal influence diagram with binary variables and known utility, and an agent whose policy has regret at most δ across the family of local interventions. The extraction algorithm behind the theorem above [48] recovers the agent's causal model from unlimited exact queries to such a policy. Prove a finite-sample version: if each query returns an action sampled from the policy, rather than the policy's exact action probabilities, and each experiment draws on the same intervention family as the theorem, maskings included, how many experiments determine the graph and the conditional probability tables to within η, and how does the answer scale with δ? As a first case, take the two-variable diagrams, where the *identified set*—the set of models consistent with the agent's observed behavior—can be computed exactly, and measure how the extraction algorithm degrades when its indifference equations are estimated from samples. Interventions on the agent's inputs, or on the activations (the values computed inside its network), are the harder surfaces beyond.

4. Algebra: what is a learned feature?

Mechanistic interpretability tries to reverse-engineer a trained neural network into something a human can read. Much of its core is linear algebra and representation theory: how a network stores more concepts than it has dimensions, and what it means for a concept to be a *direction* one can read or steer.

4.1. Superposition: packing features into directions.

How does a network with a few hundred dimensions in each layer represent the many thousands of distinct concepts ("features") it seems to know? A transformer reads text as a sequence of tokens, words and word-fragments; at any fixed layer, the computation has the form

$$\text{text} \longrightarrow \text{tokens} \longrightarrow \underbrace{(h_1, \ldots, h_T) \in (\mathbb{R}^n)^T}_{\text{activations at one layer}} \longrightarrow \text{next-token log-probabilities.}$$

For a text of T tokens, each token position carries a vector $h_j \in \mathbb{R}^n$. The copy of $\mathbb{R}^n$ in which these vectors live is the network's activation space. A leading hypothesis, made vivid by Elhage et al. [19], is superposition: when features are sparse, a network can store more of them than it has dimensions, as directions in activation space that are *not* orthogonal but only nearly so, tolerating the resulting interference because at any moment only a few features are active (Figure 2). In a small ReLU model one can watch the geometry organize itself into regular polytopes—antipodal pairs, triangles, pentagons—as a function of how sparse and how important the features are; it is, recognizably, a Thomson problem (where do mutually repelling charges on a sphere come to rest?) [50].

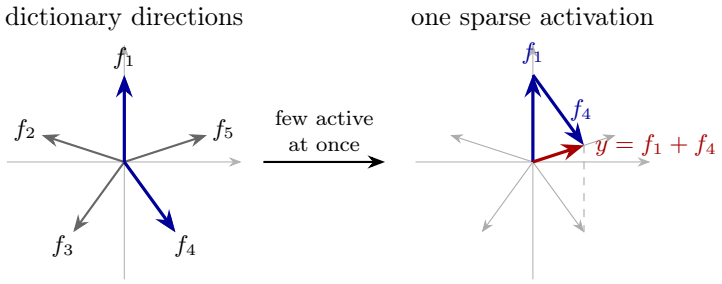

Figure 2. Superposition: five feature directions packed into a two-dimensional space as a regular pentagon. No two are orthogonal, but if only a few features are active at once, the activation is a sparse sum such as $y = f_1 + f_4$ and the active directions can still be recovered despite interference.

The nearest classical analogue is *compressed sensing* [9, 16]. Picture the features active on a given input as a sparse vector $x \in \mathbb{R}^m$ (only a few of the m possible features fire at once); the network stores it as $y = \Phi x$ in the $n \ll m$ dimensions of its activation space—a linear *measurement*—and reading a feature back out is the *recovery* of x from y. Two classical facts explain how this can work. First,

there is room for the directions: for every $\varepsilon \in (0,1)$, the space $\mathbb{R}^n$ contains $e^{\Omega(\varepsilon^2 n)}$ unit vectors whose pairwise inner products are at most ε in absolute value, so the number of ε-almost-orthogonal directions grows *exponentially* in the dimension (a consequence of the Johnson–Lindenstrauss lemma [29]). And second, when only a few of those directions are active at once, the stored signal can be read back out. Say that $\Phi \in \mathbb{R}^{n\times m}$ is *restricted-isometric of order* k with distortion δ if

$$(1-\delta)\|x\|^2 \le \|\Phi x\|^2 \le (1+\delta)\|x\|^2$$

for every *k-sparse* vector x, meaning one with at most k nonzero entries.

Theorem (Candès [10]; sharp constant, Cai–Zhang [8])**.** *If Φ is restricted-isometric of order $2s$ with distortion $\delta < 1/\sqrt{2}$, then every s-sparse $x \in \mathbb{R}^m$ is the* unique *minimizer of*

$$\min_z \ \|z\|_1 \quad \textit{subject to} \quad \Phi z = \Phi x,$$

and so is recovered exactly *by a convex program. A random Φ, say with independent Gaussian entries suitably normalized, has the property with high probability once*

$$n \ \gtrsim \ s\,\log(m/s).$$

That last scaling is the quantitative heart of the analogy: the dimension a network needs grows linearly with the number of features active *at once*, and only logarithmically with the total number it can represent. A few hundred dimensions can carry many thousands of features, provided only a handful fire on any given input. For trained transformers this remains an analogy rather than a theorem; turning it into one is part of the invitation.

There is a catch. Compressed sensing assumes the dictionary Φ is *known*. What if someone hands you a trained network and asks what features it represents? You can run inputs through the network and record activation vectors, but the feature directions were not supplied with the weights (the network's trained parameters). Recovering the features is therefore a problem in *dictionary learning*: from the observed activation vectors alone, simultaneously infer a dictionary Φ whose columns are feature directions and, for each activation y, a sparse coefficient vector x with $y \approx \Phi x$. Both the dictionary Φ and the codes x are unknown.

A sparse autoencoder attempts this empirically. It is trained to reconstruct activation vectors using a learned dictionary, with an ℓ^1 penalty encouraging each reconstruction to use only a few dictionary directions. The resulting directions are often more interpretable than the network's raw neurons [7, 15]. This raises an identifiability question: when does the sparse autoencoder recover features already used by the network, rather than impose a new coordinate system that merely makes the activations easier to describe? Classical dictionary learning answers this question when the codes are exactly sparse and their supports well spread: the dictionary is then determined up to relabeling and rescaling of its columns [2, 26], stably in the presence of noise [23, 20]. Network activations flout these hypotheses because their features co-occur, which is where the first open problem below begins.

4.2. **Reading, steering, and learned representations.** What could it mean for a concept to be a direction in activation space? Consider incomplete sentences such as "The keys on the table ..." and pause the computation at the activation h_T of the last token. A linear functional ℓ *reads* grammatical number if, for some threshold c, the inequality $\ell(h_T) > c$ predicts that the subject is plural. A vector $v \in \mathbb{R}^n$ *steers* toward the plural if replacing h_T by $h_T + v$ raises the log-probability of "are" relative to "is." Thus the same example produces a readout covector ℓ and a steering vector v. Relating them requires an inner product.

Park, Choe, and Veitch [43] formalize these two experiments for next-token prediction. They ask whether activation space carries a *causal inner product* in which independently varying concepts are orthogonal and whose Riesz map sends readout covectors to steering vectors. They estimate this geometry from the unembedding matrix (the model's final linear map, from activation space to next-token scores).

Arditi et al. [5] show that in several open chat models, refusal is mediated by a single direction in the residual stream (the running activation vector that a transformer's layers read from and write to): this direction both predicts whether the model will refuse and causally controls refusal when added or removed. A lone direction serving as both readout and control is evidence for the linear representation hypothesis, though in a different representation space from the unembedding geometry above.

Refusal and its cousins are semantic properties, with no ground truth for what the network *should* compute. Arithmetic is a cleaner laboratory: train a network on a group operation, and the algorithm it ought to discover is known in advance. Representation theory appears in trained arithmetic circuits. A one-layer transformer trained to add integers modulo p learns a discrete-Fourier "clock" algorithm, with the characters of $\mathbb{Z}/p\mathbb{Z}$ appearing in its weights [39]. Chughtai, Chan, and Nanda [11] take the first step beyond this abelian case. The ambient structure is the Artin–Wedderburn decomposition of the group algebra into matrix blocks,

$$\mathbb{C}[G] \;\cong\; \bigoplus_{\rho \in \widehat{G}} M_{d_\rho}(\mathbb{C}), \qquad \sum_{\rho \in \widehat{G}} d_\rho^2 = |G|,$$

one block per irreducible representation ρ. For a nonabelian group the higher-dimensional blocks are genuine matrix representations, and trained networks compute the product through a sparse subset of them. The algorithm's correctness is a representation-theory theorem; that networks implement it is empirical. Which irreducible representations training selects is still something we discover after the fact.

Sometimes the emergence of this structure is a theorem rather than an observation. Marchetti et al. [36] prove that for architectures whose weights are pinned down by the function they compute (up to a unitary change of basis in each unit), invariance under a finite group G forces each unit's weights to be an irreducible unitary representation of G; when the weights are moreover orthonormal, together they form the Fourier transform on G, and the group itself can be read off the trained weights.

What makes this a safety topic is *selection*: a trained network can discover a crisp algebraic algorithm, hide it in weights, and use only part of the mathematically available structure. If we want to predict a system's behavior off-distribution, it matters which algorithm training found, not only that the training loss is low.

One caveat carries across this whole enterprise: a direction that *predicts* a feature is only a *correlate*. Showing that the network actually *uses* it takes an intervention—ablating the direction, or patching it to the value it would take on another input, and watching the behavior change—not a probe that merely reads it off.

4.3. **Open problems.** A broad, structured catalogue of open problems in this area—several with a precise mathematical core—is collected by Sharkey et al. [54].

★ **Open Problem** `MAIS-O3` **(The geometry and identifiability of superposition).** Suppose activations have the form $y = \Phi x + \xi$, where the columns $v_1, \dots, v_m$ of Φ are unit feature directions, ξ is noise, and the sparse codes x have correlated supports. Estimate the dictionary the way a sparse autoencoder does: minimize reconstruction error plus an ℓ^1 penalty. Determine, in terms of the coherence $\mu = \max_{i \neq j} |\langle v_i, v_j \rangle|$, the sparsity, the sample size, and the support correlations, when every minimizer recovers the true directions up to permutation and scaling, and when it instead *merges* co-occurring directions into one. Already for two features the answer is unknown: a nested two-event model provably forces merging for every sufficiently small positive penalty, while solo firing restores recovery in the zero-penalty constrained limit; locating the positive-penalty phase boundary is the place to start.

★ **Open Problem** `MAIS-O4` **(Training for interpretability).** Post-hoc interpretability asks whether a trained network's features can be recovered after the fact. A complementary problem is to train networks whose features are easier to recover in the first place. In the ReLU toy model of Elhage et al. [19] the features are known by construction, so the trade-off can be made exact. Determine the interference–performance frontier: for features appearing sparsely and independently, among weight matrices whose coherence (worst-case interference between stored features) is at most μ, how small can the task loss be, as a function of μ, the sparsity, the number of features, and the number of neurons? A first case: prove or refute that penalizing the *average* interference during training lowers the *worst-case* coherence of the minimizer, compared with training on the task loss alone.

★ **Open Problem** `MAIS-O5` **(Representation theory of learned circuits).** Networks trained on group multiplication learn representation-theoretic algorithms [39, 11], but *which* irreducible representations they use varies from one random seed to the next. Fix a finite group G, a one-hidden-layer architecture of fixed width, Gaussian initialization, and gradient flow on the cross-entropy loss (the mean negative log-probability assigned to the correct product) with *weight decay*, an added penalty proportional to the squared norm of the weights. The irreducibles visible in the trained network's outputs then form a random subset of $\widehat{G}$; determine its distribution—which irreducibles are learned, with what probability, as a function of the width and the decay strength. The tables of learned representations in [11]

are the data such a theorem would have to explain; the smallest case with genuine competition, $G = \mathbb{Z}/5\mathbb{Z}$ at width two, is open already with the decay switched off.

5. Analysis and geometry: which solution does training find?

5.1. **Why trained networks generalize: singular learning theory.** A trained network is a setting of parameters chosen to fit its *training data*—the finite set of examples it was optimized on. The trouble is that a large network has astronomically many settings that fit those examples perfectly, and they disagree wildly everywhere else; the optimizer—gradient descent, which repeatedly nudges the parameters downhill on the training error—must land on one of them. For safety, the selected solution matters: a network that behaved well while we were watching may behave differently once deployed. So why do the solutions found by training *generalize*—predict well on fresh data from the same source—when so many other data-fitting solutions would not? These networks reach zero error even when the labels are replaced by pure noise, and a two-layer network with only $2n + d$ parameters can fit *any* labeling of n points in $\mathbb{R}^d$ (Zhang et al. [62]); the classical bounds that explain generalization by counting parameters say nothing here.

A different and more general answer comes from *singular learning theory* [59]. Realistic models are *singular*: distinct parameters compute the same function, typically in continuous families, so the optimal set is not an isolated point but a positive-dimensional variety—itself typically singular—flat to second order along its own directions (Figure 3). The classical large-sample asymptotics all assume an isolated, curved optimum; the right replacements come from algebraic geometry. One caveat: the theory is exact for *Bayesian* learning, not directly for stochastic gradient descent (gradient descent driven by noisy, small-batch estimates of the gradient)—bridging the two is the problem *Opposing staircases* below. The two are closer than they appear: Khan and Rue [30] derive stochastic gradient descent, along with many other optimizers, as approximations of a single Bayesian update rule.

A toy computation shows what singularity buys. Take first the regular loss $K(w) = w^2$ on the line: the set where $K \le \varepsilon$ is an interval of length $2\sqrt{\varepsilon}$, so the volume of near-optimal parameters scales as $\varepsilon^{1/2}$—the exponent is $d/2$ with $d = 1$. Now take $K(a, b) = a^2 b^2$ on the square $[-1, 1]^2$. Its zero set is not a point but the union of the two axes, crossing at a singularity, and the region where $K \le \varepsilon$ is a neighborhood of the axes of area of order $\varepsilon^{1/2} \log(1/\varepsilon)$: the exponent stays $1/2$ rather than rising to $d/2 = 1$, and a logarithm appears. A singular loss has far more near-optimal parameters than its dimension suggests, and the pair (exponent, power of the logarithm) is the simplest instance of the learning coefficient and multiplicity about to be defined.

To make this precise, fix a model $p(x \mid w)$ with parameter w in a compact $W \subseteq \mathbb{R}^d$, a prior φ on W (a probability density encoding which parameters are plausible before any data is seen), and a true distribution $q_0(x) = p(x \mid w_0)$ lying in the model. (The subscript keeps the truth q_0 apart from the belief q of Section 3.)

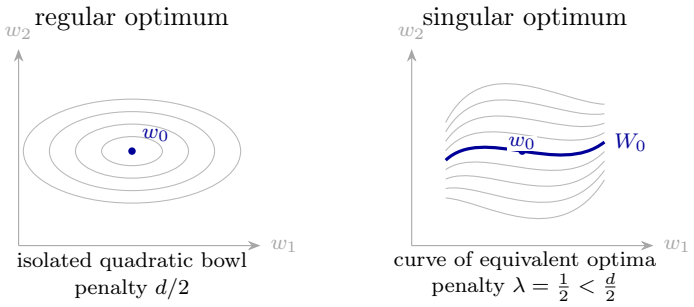

FIGURE 3. Regular asymptotics treat the optimum as an isolated quadratic bowl, giving the classical $\frac{d}{2}\log n$ complexity penalty. Singular learning theory allows a whole variety of parameters to compute the same function: in the right panel the optimal set is a curve W_0 through w_0, and the penalty $\lambda = \frac{1}{2}$ is strictly smaller than $\frac{d}{2} = 1$, as it is whenever the optimal set has positive dimension.

Write

$$K(w) = \mathrm{KL}\big(q_0 \,\big\|\, p(\cdot \mid w)\big) = \int q_0(x) \log \frac{q_0(x)}{p(x \mid w)}\, dx,$$

the Kullback–Leibler divergence of Section 3, so that the set of *true parameters* $W_0 = K^{-1}(0)$ is exactly where the model recovers q_0.

Given n i.i.d. samples $X_1, \ldots, X_n$, the central object is the *Bayes free energy*—equivalently the negative log marginal likelihood, the codelength the model assigns to the data,

$$\bar{F}_n \;=\; -\log \int_W \prod_{i=1}^{n} p(X_i \mid w)\, \varphi(w)\, dw.$$

(The bar is part of the name; $\bar{F}_n$ is the random free energy of the sample, not an expectation.) Normalizing the same integrand gives the *posterior* $\propto \prod_i p(X_i \mid w)\, \varphi(w)$—the prior reweighted by how well each parameter fits the data—so $\bar{F}_n$ is minus the log of its normalizer. This free energy is not the variational free energy $F(q, x)$ of Section 3: that one scores a belief against a single observation, this one scores the whole model against a whole sample. The shared name reflects a shared shape, minus the log of a normalizing integral.

Take first the *regular* case: the model is *identifiable* (distinct parameters give distinct distributions) and the optimal set is a single point $W_0 = \{w_0\}$ at which the Hessian of K—the *Fisher information* $I(w_0)$—is positive-definite. Then the integrand defining $\bar{F}_n$ concentrates in a shrinking neighborhood of w_0; expanding $\log p(X_i \mid w)$ to second order there makes the integral Gaussian, contributing a volume factor $(2\pi/n)^{d/2} \det I(w_0)^{-1/2}$. Taking $-\log$,

$$\bar{F}_n \;=\; nS_n \;+\; \tfrac{d}{2}\log n \;+\; O_p(1), \qquad S_n = -\tfrac{1}{n}\textstyle\sum_i \log q_0(X_i),$$

where S_n is the empirical entropy and the remainder is bounded in probability.[5] The complexity penalty is exactly half the parameter count. Watanabe's theorem is the singular replacement, in which $d/2$ gives way to an invariant of the singular geometry.

Theorem (Watanabe's free-energy asymptotics [59])**.** *For a singular model satisfying Watanabe's regularity conditions (real-analyticity and a mild variance bound*[6]*), as $n \to \infty$,*

$$\bar{F}_n \;=\; n\,S_n \;+\; \lambda \log n \;-\; (m-1)\log\log n \;+\; O_p(1), \tag{1}$$

where the learning coefficient $\lambda > 0$*—the* real log canonical threshold *(*RLCT*) of K relative to the prior φ—and the integer $m \in \{1, \dots, d\}$, its* multiplicity*, take the places of the $d/2$ and the 1 of the regular case. The $O_p(1)$ remainder converges in distribution.*[7] *The same coefficient controls generalization: writing $\hat{p}_n$ for the posterior average of $p(\cdot \mid w)$, the* generalization error $G_n = \mathrm{KL}(q_0 \,\|\, \hat{p}_n)$ *obeys $\mathbb{E}[G_n] = \lambda/n + o(1/n)$.*

The meaning of λ is volume growth: the prior volume of the set $\{w \in W : K(w) \le \varepsilon\}$ of parameters fitting the truth to within ε scales like ε^{λ} times $(\log(1/\varepsilon))^{m-1}$, so smaller λ means near-optimal parameters are more plentiful. To compute λ for a given model, one resolves the singularities of K (Hironaka [27]).

In the regular case, $\lambda = d/2$ and $m = 1$, and the theorem recovers the regular asymptotic expansion above; in a singular model $\lambda \le d/2$, often strictly. The comparison localizes: restrict the integral defining $\bar{F}_n$ to a neighborhood of one true parameter, and the expansion (1) holds with that neighborhood's own coefficient. Among the true parameters—all fitting the truth exactly, all weighted by the prior—the posterior share of the most degenerate neighborhoods, those of smallest local λ, grows with n. In this Bayesian setting the posterior concentrates, on its own, on the solutions that the formula $\mathbb{E}[G_n] = \lambda/n$ marks as best-generalizing. This is the theory's rendering of the slogan that learning prefers "simpler" solutions: simplicity means a more degenerate geometry, not fewer parameters.

The degeneracy behind a small λ is largely *structural*: it comes from directions in parameter space along which the function does not change—for instance those freed when an internal unit is effectively switched off—and these are what pull λ below $d/2$. The symmetries are not a defect to be engineered away: within the theory, it is the degeneracy that pulls the generalization error λ/n below the classical rate $d/(2n)$.

---

[5] $O_p(1)$ means *bounded in probability*: for every $\varepsilon > 0$ there is an M with $\Pr(|\cdot| > M) < \varepsilon$ for all large n.

[6] The *relatively finite variance* condition: writing $f(x,w) = \log(q_0(x)/p(x \mid w))$, $K(w) = \mathbb{E}_{q_0} f(X,w)$, and $V(w) = \mathbb{E}_{q_0} f(X,w)^2$, one assumes $V(w) \le C\,K(w)$ near the optimal set—the variance of the log-likelihood ratio is controlled by its mean.

[7] In the regular case the limit is $C - \frac{1}{2}\chi^2_d$ for a constant C—the Bayesian shadow of Wilks' theorem, since the log-likelihood-ratio statistic $2\big(\sup_w \sum_i \log p(X_i \mid w) - \sum_i \log p(X_i \mid w_0)\big) \Rightarrow \chi^2_d$. In a singular model the limit is in general non-Gaussian, a functional of a Gaussian process on the manifold obtained by resolving the singularities of K.

The coefficient λ need not be found by hand. Introduce a knob $\beta > 0$, an *inverse temperature*, and the *tempered posterior*

$$p_\beta(w) \;\propto\; \varphi(w)\, e^{-\beta n L_n(w)}, \qquad L_n(w) = -\tfrac{1}{n}\textstyle\sum_i \log p(X_i \mid w),$$

which interpolates between the prior ($\beta = 0$) and concentration on the best-fitting parameters ($\beta \to \infty$). (In statistics, taking $\beta < 1$ hedges against a misspecified model: Grünwald's "Safe Bayesian" learns β from the data [24], and Alquier and Ridgway prove that tempered posteriors still concentrate [4].) At inverse temperature $\beta = 1/\log n$, the average of nL_n over the tempered posterior exceeds its minimum by $\lambda \log n$, to leading order [60]. So λ can be *estimated* by sampling the tempered posterior, needing neither the true distribution nor an explicit resolution of singularities. That practicality is what lets the theory meet networks far too large to analyze by hand.

What one estimates is not the global λ but the *local learning coefficient*: the same invariant computed in a neighborhood of the particular solution $w^\star$ that training actually reached, measuring how degenerate *that* solution is [34]. As a network trains, its local learning coefficient jumps at discrete moments—*phase transitions*, where the solution moves to a qualitatively new, differently-degenerate part of the landscape as a new piece of internal structure forms. Tracking these jumps is the program of *developmental interpretability*: so far on relatively small networks and with preliminary validation, though the same estimators have already surfaced tens of thousands of candidate structures inside a billion-parameter language model [38].

Two networks can fit the training data identically and yet compute by different internal mechanisms, hence behave differently on inputs outside that data—what ordinary testing, which only probes behavior on data, cannot detect. The singular geometry governs *which* mechanism training selects, and the selection need not favor the mechanisms we would prefer: the most degenerate solution—largest in volume, smallest in λ—may implement the behavior we intended, or a shortcut that mimics it on the training distribution and fails beyond it. Whether geometric degeneracy favors mechanisms that are also interpretable and robust is, for safety, the question this program must answer; the case that alignment turns on such questions is made in [46].

★ **Open Problem** `MAIS-06` **(Does geometric simplicity force a legible mechanism?)** Fix an odd integer p and $H \geq 2p-1$. On input $(a, b) \in (\mathbb{Z}/p\mathbb{Z})^2$, consider the one-hidden-layer network

$$f_w(a,b) = \sum_{j=1}^{H} V_j \big(u'_j(a) + u''_j(b)\big)^2 \in \mathbb{R}^p,$$

trained to output the coordinate vector indexed by $a + b$. The jth summand is one hidden neuron: it adds two entries from the lookup tables $u'_j, u''_j \colon \mathbb{Z}/p\mathbb{Z} \to \mathbb{R}$, squares the result, and writes the vector V_j to the output. Fourier inversion gives exact fits in which each neuron that contributes nontrivially carries a single frequency: for some k, both lookup tables are a constant plus a linear combination of $\cos(2\pi k \cdot /p)$ and $\sin(2\pi k \cdot /p)$. Up to a change of basis, this is the algorithm found by reverse-engineering trained networks [39].

Is this Fourier structure forced by singular geometry? In a fixed closed ball containing one of these Fourier fits, prove or refute that every exact fit of smallest local learning coefficient has this single-frequency form, apart from neurons whose contribution vanishes identically. The first non-vacuous case is $p = 5$ with nine hidden neurons: use its Fourier symmetries to classify the most singular exact fits, or find a non-Fourier one by computer algebra. A proof would connect geometric simplicity to a mechanism one can read from the weights; a counterexample would show that the two notions of simplicity can part ways.

5.2. **Learning in stages.** Learning does not always proceed smoothly; long flat stretches give way to sudden changes. The modular-addition task above is the best known example: a network fits its training data early but generalizes only much later, and abruptly. This is *grokking* [39].[8]

The simplest solvable model of such stepwise learning is the deep *linear* network: a product of matrix factors $W_L \cdots W_1$ fit to a target linear map by *gradient flow* (gradient descent in continuous time) on squared error. The composite map is linear, but as a function of the separate factors the loss is a non-convex polynomial, so the flow is genuinely nonlinear. Saxe, McClelland, and Ganguli [52] show that when the inputs are *whitened* (linearly transformed to have identity covariance), the singular value decomposition of the input–output correlation matrix *decouples* the dynamics into independent scalar equations, one per singular value s: each is the logistic ordinary differential equation $\tau \dot{a} = 2a(s - a)$, whose solution rises sigmoidally from near 0 to s in time of order τ/s (Figure 4). The decoupling and the scalar solutions are exact; the initialization then sets the order of events: for comparably small initial coefficients, modes with larger s turn on earlier, and when their turn-on times are well separated the training error falls not smoothly but in a staircase—long plateaus near saddle points of the loss, each broken by a sharp drop as the next mode is acquired. Here "which structure the network learns, and in what order" has a closed-form answer.

The staircase survives beyond the linear case: Abbe, Boix-Adserà, and Misiakiewicz [1] prove that training a two-layer network on a *sparse* target—one depending on only a few of its inputs—again proceeds in stages, saddle to saddle, with the waiting time before each stage set by a combinatorial *leap complexity* measuring how many inputs must be assembled at once. Carrying such exact accounts to the nonlinear networks used in practice is open.

★ **Open Problem** `MAIS-07` **(Opposing staircases).** Watanabe's theorem is Bayesian; gradient descent is not. (What bridging the two would mean for safety is discussed in [46].) Build the bridge first in the deep-linear staircase above, where both sides are explicit. Each plateau sits at a saddle where only the k largest modes have been learned, and there the definition itself is part of the problem: the local learning coefficient is defined at a local *minimum*, as the volume exponent of the set

[8] Grokking and the staircase below are cousins, not the same phenomenon: grokking is a delayed jump in performance on *held-out* data after the training loss is already low, whereas the staircase is a stepwise fall of the *training* loss itself. Both are read, in this program, as transitions between differently-singular regions of parameter space.

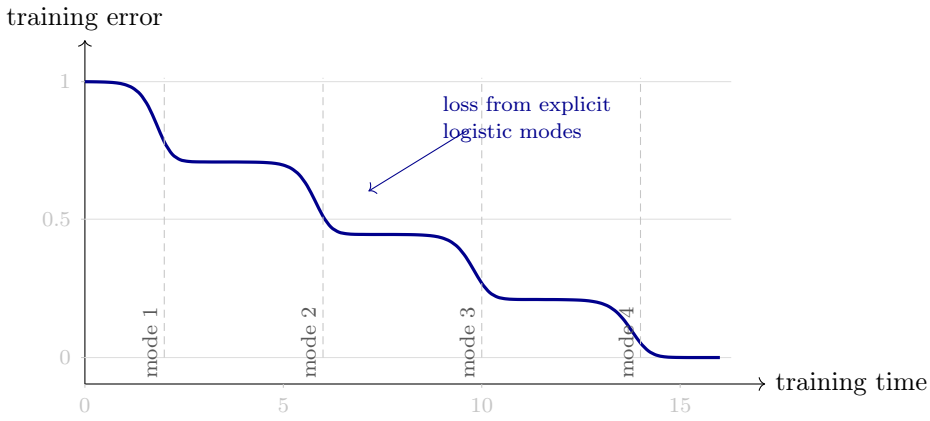

FIGURE 4. A learning staircase computed from the deep-linear solution of Saxe et al. In mode i, s_i is the target singular value and $a_i(t)$ is the coefficient learned by time t, satisfying $\tau \dot{a}_i = 2a_i(s_i - a_i)$. The curve plots normalized training error $\sum_i (s_i - a_i(t))^2$ for $\tau = 0.5$ and $s = (1, 0.95, 0.9, 0.85)$, with small initial coefficients chosen so the four modes turn on at separated times. Each mode turning on produces one drop.

of nearby parameters fitting almost as well, but a saddle has descent directions, so that volume does not shrink to zero. Formulate the right local invariant at the kth saddle (one candidate, the *two-sided* exponent: the volume of nearby parameters whose loss lies within ε of the saddle's, above or below, shrinks like ε^{λ}), compute it for the deep-linear network, and prove or refute the monotone picture that gives this problem its name: as gradient flow descends the staircase of losses, the invariant that controls generalization climbs an opposing staircase. As a first check, the estimator of Lau et al. [34] runs at any parameter, saddle or not; run it along a simulated staircase and see.

5.3. **Generalization beyond the training distribution.** The theory above concerns generalization *in distribution*: test data drawn from the same law as the training data. The safety-critical regime is the opposite one, *out of distribution*—how a trained system behaves on inputs unlike anything it saw in training—and there the gap between fitting and understanding reopens. Two parameter settings with identical training loss can implement different mechanisms (as above) and therefore extrapolate differently; which extrapolation gradient descent selects is, at present, something we observe after the fact rather than predict.

In *goal misgeneralization* [33], a system under *distribution shift*—a deployment input distribution unlike the one it trained on—keeps its *capabilities* while pursuing a *different goal* from the one intended. The standard example is an agent trained to collect a coin in a video game where the coin always sat at the right end of the level: it learned "move right," not "reach the coin," and the two came apart the

moment the coin moved. Capable but misdirected is the dangerous quadrant, and it is invisible to any testing that stays in distribution.

The pattern is old enough to have a name, *Goodhart's law*: a proxy optimized hard enough stops tracking the goal it stood for. On the training levels, "move right" was a perfect proxy for "reach the coin," so optimization could not tell them apart; the shift revealed which one the network had learned. Reward hacking (Section 3) is the same law during training; goal misgeneralization is the law at deployment.

★ Open Problem MAIS-08 (A predictive theory of out-of-distribution generalization). In the coin-collecting environment of Langosco et al. [33], reduced to one dimension, two policies fit the training data perfectly: "move right" (the proxy) and "go to the coin" (the intended goal), which agree whenever the coin sits at the right end. Train a two-layer network by gradient descent to imitate optimal play, randomizing the coin's position in an ε-fraction of training episodes. Determine the probability, over the random initialization, that the trained network follows the proxy on a probe state where the two policies disagree, as a function of ε, the width, and the input encoding. For linear policies the answer is a theorem—the encoding, not the initialization, decides—and the standard infinite-width limits follow it; for networks of finite width it is open at every ε, including $\varepsilon = 0$, already at width two.

6. Getting started in AI safety

Technological progress is a human choice, not a law of nature. When mathematical ideas help build systems whose effects will be felt at civilizational scale, truth-seeking remains a core value, but it is no longer the only value in view. The civic mathematician also asks what our truths are for: whether they can help design AI systems to be more legible, more accountable, and easier to steer toward coexistence with us.

The open problems above are meant as entry points into the new field of Math for AI Safety. Other ways in are workshops like ILIAD, the ARENA curriculum, research programs like MATS and SPAR, and organizations like CHAI, Constellation, FAR AI, and Resolution. Funding for work in this area is available from grantmakers like Coefficient Giving, CAIF, SFF, and UK AISI. I hope you'll pick a problem whose language already feels like home, strip it down to the simplest interesting case, and start proving things!

Acknowledgments

I thank Jesse Hoogland, Hyojeong Son, Jacob Tsimerman, Claude Fable 5, Claude Opus 4.8, and GPT 5.5 and 5.6 Sol for many inspiring conversations. Scott Aaronson, Ahmed Bou-Rabee, Vince Conitzer, Jonathan Gabor, Chris Hillar, Brad Knox, Phil Sosoe, Samuel Speas, Kate Stange, Steve Strogatz, Ariel Yadin, and an anonymous referee provided valuable feedback on an early draft.

Appendix A. A glossary for mathematicians

neural network	a function with millions to billions of adjustable parameters (its *weights*), built by composing simple layers and tuned by *gradient descent* to reduce a training *loss*.
language model	a neural network trained to predict the next token of text; today's chat systems are language models further trained to follow instructions.
token	the atomic unit (word or word-fragment) a language model reads and predicts.
loss	the scalar a network is trained to minimize; for language models, the negative log-probability it assigned to the actual next token.
weights	the trainable parameters of an artificial neural network: the entries of its matrices, trained by gradient descent and fixed at deployment time.
activations	the values that flow through the network on a given input (the outputs of its neurons and layers), as opposed to the fixed weights.
activation space	the copy of $\mathbb{R}^n$ in which a given layer's activations live, one coordinate per neuron; feature directions are directions in this space.
gradient descent	the optimizer: repeatedly nudge the parameters w against the gradient of the loss, $w \leftarrow w - \eta \nabla L(w)$.
ReLU	the rectified linear unit $x \mapsto \max(0, x)$, a canonical piecewise-linear nonlinearity; modern transformers often use smooth or gated variants instead (GELU, SiLU, SwiGLU).
transformer	the dominant network architecture for language: a stack of layers that mix information across token positions by *attention* (each position selectively reads from the others) and read from / write to the residual stream.
residual stream	the running vector of activations a transformer reads from and writes to at each layer; a natural home for feature directions.
feature	a hypothesized variable or property a network represents in its activations (e.g. "the text is in French").
probe	a simple (usually linear) function of a network's activations, trained or searched to read off some quantity of interest.
policy	an agent's rule for choosing actions from states (a possibly randomized map from states to actions).
reward hacking	raising a measured reward signal in ways that defeat the intent behind it; Goodhart's law as it appears in agents trained by reward.

superposition	a hypothesized representation scheme in which a network stores more features than it has dimensions, as non-orthogonal directions, tolerable when few features are active at once.
sparse autoencoder	a network trained to reconstruct another network's activations as sparse combinations of learned dictionary directions; an empirical tool for extracting features.
interpretability	reverse-engineering a trained network into human-understandable structure (circuits, features, algorithms), validated by intervention and not by human-readable description alone.
gradual disempowerment	the risk that humans lose influence over the institutions that shape their lives, by incremental delegation to AI systems we do not understand rather than by any sudden takeover.

References

[1] Abbe, E., Boix-Adserà, E., and Misiakiewicz, T. (2023). SGD learning on neural networks: leap complexity and saddle-to-saddle dynamics. *Conference on Learning Theory (COLT)* 2023. arXiv:2302.11055.

[2] Aharon, M., Elad, M., and Bruckstein, A. M. (2006). On the uniqueness of overcomplete dictionaries, and a practical way to retrieve them. *Linear Algebra and its Applications* 416(1), 48–67.

[3] Alon, N., Bloom, T. F., Gowers, W. T., Litt, D., Sawin, W., Shankar, A., Tsimerman, J., Wang, V., and Wood, M. M. (2026). Remarks on the disproof of the unit distance conjecture. arXiv:2605.20695.

[4] Alquier, P., and Ridgway, J. (2020). Concentration of tempered posteriors and of their variational approximations. *Annals of Statistics* 48(3), 1475–1497. arXiv:1706.09293.

[5] Arditi, A., Obeso, O., Syed, A., Paleka, D., Panickssery, N., Gurnee, W., and Nanda, N. (2024). Refusal in language models is mediated by a single direction. *Advances in Neural Information Processing Systems* 37. arXiv:2406.11717.

[6] Barász, M., Christiano, P., Fallenstein, B., Herreshoff, M., LaVictoire, P., and Yudkowsky, E. (2014). Robust cooperation in the Prisoner's Dilemma: program equilibrium via provability logic. arXiv:1401.5577.

[7] Bricken, T., et al. (2023). Towards monosemanticity: decomposing language models with dictionary learning. *Transformer Circuits Thread*, Anthropic. https://transformer-circuits.pub/2023/monosemantic-features.

[8] Cai, T. T., and Zhang, A. (2014). Sparse representation of a polytope and recovery of sparse signals and low-rank matrices. *IEEE Transactions on Information Theory* 60(1), 122–132. arXiv:1306.1154.

[9] Candès, E. J., Romberg, J., and Tao, T. (2006). Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory* 52(2), 489–509.

[10] Candès, E. J. (2008). The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathématique* 346(9–10), 589–592.

[11] Chughtai, B., Chan, L., and Nanda, N. (2023). A toy model of universality: reverse-engineering how networks learn group operations. *International Conference on Machine Learning* 2023. arXiv:2302.03025.

[12] Cooper, E., Oesterheld, C., and Conitzer, V. (2025). Characterising simulation-based program equilibria. *Proceedings of AAAI* 2025. arXiv:2412.14570.

[13] Critch, A. (2019). A parametric, resource-bounded generalization of Löb's theorem, and a robust cooperation criterion for open-source game theory. *The Journal of Symbolic Logic* 84(4), 1368–1381. arXiv:1602.04184.

[14] Critch, A., Dennis, M., and Russell, S. (2022). Cooperative and uncooperative institution designs: surprises and problems in open-source game theory. arXiv:2208.07006.

[15] Cunningham, H., Ewart, A., Riggs, L., Huben, R., and Sharkey, L. (2023). Sparse autoencoders find highly interpretable features in language models. arXiv:2309.08600.

[16] Donoho, D. L. (2006). Compressed sensing. *IEEE Transactions on Information Theory* 52(4), 1289–1306.

[17] Dyson, F. (2009). Birds and frogs. *Notices of the American Mathematical Society* 56(2), 212–223.

[18] Eisenstat, S. (2025). Condensation: a theory of concepts. Manuscript. https://www.sameisenstat.net/doc/condensation-25-07.pdf.

[19] Elhage, N., et al. (2022). Toy models of superposition. *Transformer Circuits Thread*. arXiv:2209.10652.

[20] Garfinkle, C. J., and Hillar, C. J. (2019). On the uniqueness and stability of dictionaries for sparse representation of noisy signals. *IEEE Transactions on Signal Processing* 67(23), 5884–5892. arXiv:1606.06997.

[21] Garrabrant, S., Benson-Tilsen, T., Critch, A., Soares, N., and Taylor, J. (2016). Logical induction. arXiv:1609.03543.

[22] Gowers, W. T. (2000). The two cultures of mathematics. In *Mathematics: Frontiers and Perspectives* (V. Arnold, M. Atiyah, P. Lax, and B. Mazur, eds.), American Mathematical Society, 65–78.

[23] Gribonval, R., Jenatton, R., and Bach, F. (2015). Sparse and spurious: dictionary learning with noise and outliers. *IEEE Transactions on Information Theory* 61(11), 6298–6319. arXiv:1407.5155.

[24] Grünwald, P. (2012). The safe Bayesian: learning the learning rate via the mixability gap. *Algorithmic Learning Theory (ALT)* 2012, 169–183.

[25] Hariharan, S., Birkbeck, C., Lee, S., Ma, H. K. G., Mehta, B., Poiroux, A., and Viazovska, M. (2026). Progress in formalizing sphere packing in dimension 8. arXiv:2604.23468.

[26] Hillar, C. J., and Sommer, F. T. (2015). When can dictionary learning uniquely recover sparse data from subsamples? *IEEE Transactions on Information Theory* 61(11), 6290–6297. arXiv:1106.3616.

[27] Hironaka, H. (1964). Resolution of singularities of an algebraic variety over a field of characteristic zero. *Annals of Mathematics* 79(1), 109–203; 79(2), 205–326.

[28] Howard, J. V. (1988). Cooperation in the Prisoner's Dilemma. *Theory and Decision* 24(3), 203–213.

[29] Johnson, W. B., and Lindenstrauss, J. (1984). Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Mathematics* 26, 189–206.

[30] Khan, M. E., and Rue, H. (2023). The Bayesian learning rule. *Journal of Machine Learning Research* 24(281), 1–46. arXiv:2107.04562.

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

[32] Kwa, T., West, B., Becker, J., et al. (2025). Measuring AI ability to complete long software tasks. METR report; *Advances in Neural Information Processing Systems* 38. arXiv:2503.14499.

[33] Langosco, L., Koch, J., Sharkey, L., Pfau, J., and Krueger, D. (2022). Goal misgeneralization in deep reinforcement learning. *International Conference on Machine Learning* 2022. arXiv:2105.14111.

[34] Lau, E., Furman, Z., Wang, G., Murfet, D., and Wei, S. (2025). The local learning coefficient: a singularity-aware complexity measure. *Artificial Intelligence and Statistics (AISTATS)* 2025. arXiv:2308.12108.

[35] Löb, M. H. (1955). Solution of a problem of Leon Henkin. *The Journal of Symbolic Logic* 20(2), 115–118.

[36] Marchetti, G. L., Hillar, C., Kragic, D., and Sanborn, S. (2024). Harmonics of learning: universal Fourier features emerge in invariant networks. *Conference on Learning Theory (COLT)* 2024. arXiv:2312.08550.

[37] The mathlib Community. (2020). The Lean mathematical library. *Certified Programs and Proofs (CPP)* 2020, 367–381. arXiv:1910.09336.

[38] Murfet, D., Gordon, A., Adam, M., Wang, G., Hoogland, J., Baker, G., Snell, W., van Wingerden, S., Newgas, A., Snikkers, B., and Hitchcock, R. (2026). Spectroscopy at scale: finding interpretable structure in Pythia-1.4B. Timaeus, research report. https://timaeus.co/research/2026-04-21-spectroscopy-main/.

[39] Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. (2023). Progress measures for grokking via mechanistic interpretability. *International Conference on Learning Representations* 2023. arXiv:2301.05217.

[40] Ngo, R. (2025). Towards a scale-free theory of intelligent agency. LessWrong essay. https://www.lesswrong.com/posts/5tYTKX4pNpiG4vzYg/towards-a-scale-free-theory-of-intelligent-agency.

[41] Oesterheld, C., and Conitzer, V. (2021). Safe Pareto improvements for delegated game playing. *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)* 2021; journal version, *Autonomous Agents and Multi-Agent Systems* 36 (2022).

[42] OpenAI. (2026). An OpenAI model has disproved a central conjecture in discrete geometry. https://openai.com/index/model-disproves-discrete-geometry-conjecture/.

[43] Park, K., Choe, Y. J., and Veitch, V. (2024). The linear representation hypothesis and the geometry of large language models. *International Conference on Machine Learning* 2024. arXiv:2311.03658.

[44] Parr, T., Pezzulo, G., and Friston, K. J. (2022). *Active Inference: The Free Energy Principle in Mind, Brain, and Behavior*. MIT Press.

[45] Pearl, J. (2009). *Causality: Models, Reasoning, and Inference*, 2nd ed. Cambridge University Press.

[46] Pepin Lehalleur, S., Hoogland, J., Farrugia-Roberts, M., Wei, S., Gietelink Oldenziel, A., Wang, G., Carroll, L., and Murfet, D. (2025). You are what you eat: AI alignment requires understanding how data shapes structure and generalisation. arXiv:2502.05475.

[47] Rao, R. P. N., and Ballard, D. H. (1999). Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience* 2(1), 79–87.

[48] Richens, J., and Everitt, T. (2024). Robust agents learn causal world models. *International Conference on Learning Representations* 2024. arXiv:2402.10877.

[49] Richens, J., Everitt, T., and Abel, D. (2025). General agents need world models. *International Conference on Machine Learning* 2025. arXiv:2506.01622.

[50] Saff, E. B., and Kuijlaars, A. B. J. (1997). Distributing many points on a sphere. *The Mathematical Intelligencer* 19(1), 5–11.

[51] Sawin, W. (2026). An explicit lower bound for the unit distance problem. arXiv:2605.20579.

[52] Saxe, A. M., McClelland, J. L., and Ganguli, S. (2014). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *International Conference on Learning Representations* 2014. arXiv:1312.6120.

[53] Schelling, T. C. (1960). *The Strategy of Conflict*. Harvard University Press.

[54] Sharkey, L., et al. (2025). Open problems in mechanistic interpretability. *Transactions on Machine Learning Research* 2025. arXiv:2501.16496.

[55] Tennenholtz, M. (2004). Program equilibrium. *Games and Economic Behavior* 49(2), 363–373.

[56] Thurston, W. P. (1994). On proof and progress in mathematics. *Bulletin of the American Mathematical Society* 30(2), 161–177. arXiv:math/9404236.

[57] von Neumann, J., and Morgenstern, O. (1944). *Theory of Games and Economic Behavior*. Princeton University Press.
[58] Wald, A. (1950). *Statistical Decision Functions*. Wiley.
[59] Watanabe, S. (2009). *Algebraic Geometry and Statistical Learning Theory*. Cambridge University Press.
[60] Watanabe, S. (2013). A widely applicable Bayesian information criterion. *Journal of Machine Learning Research* 14, 867–897.
[61] Wentworth, J., and Lorell, D. (2025). Natural latents: latent variables stable across ontologies. arXiv:2509.03780.
[62] Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. *International Conference on Learning Representations* 2017. arXiv:1611.03530.