AN EXPERIMENTAL COURSE

FIRST EDITION / 2026

# Neural Programming

*OBSERVE* *INTERVENE* *EVALUATE* *REVISE*



*The experimental engineering of learned computation.*

Four weeks. Twenty questions. One experimental loop.

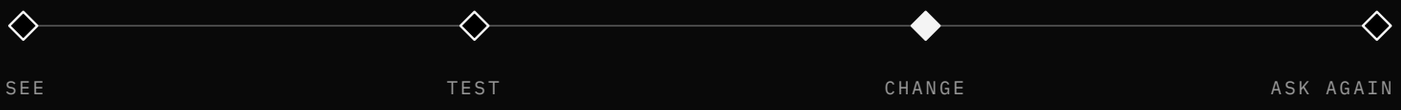
DANIEL RODRÍGUEZ

NOVEMBER 2026

01 / WONDER, GIVEN INSTRUMENTS

# What if learned computation became *a material we could work?*

A model can be observed from the outside. It can also be opened to experiment: a state recorded, a contrast constructed, a computation changed, a consequence measured.




Neural Programming is the working name of this course: an experimental practice for investigating and deliberately modifying learned computation. It brings together internal measurements, causal tests, activation interventions, direct weight edits and behavioral evaluation.

The ambition is a portable ability: to enter an unfamiliar model with a precise question, build the instruments to investigate it, and decide which changes the evidence justifies.

---

Every lesson inherits a question.
*Every experiment leaves a sharper one.*

02 / THE COURSE MAP

# A month of *better questions.*

Four weeks / Twenty sessions / One cumulative experimental practice

01 Can Neural Networks Be Programmed?

02 From Symbols to Learned Computation

03 What Is a Behavioral Phenomenon?

04 What Makes an Experiment Reproducible?

05 What Happens Inside a Forward Pass?

06 How Do Internal States Differ?

07 Does Correlation Imply Computation?

08 Where Does a Behavior Come From?

09 Can Representations Be Identified?

10 How Do We Know an Explanation Is Correct?

11 Can We Steer an Activation?

12 When Should an Intervention Occur?

13 How Do We Control Without Breaking?

14 Can We Program Weights Directly?

15 Can a Model Retain an Intervention?

16 Can Discoveries Become Programs?

17 What Closes the Control Loop?

18 Can Experiments Design Experiments?

19 What Does It Mean to Understand a Neural Network?

20 What Are the Limits of Neural Programming?



10% Initiation 40% Science 40% Control 10% Synthesis

03 / THE OBJECT AND THE METHOD

# Observe a computation. Make a deliberate change. *Earn the next claim.*

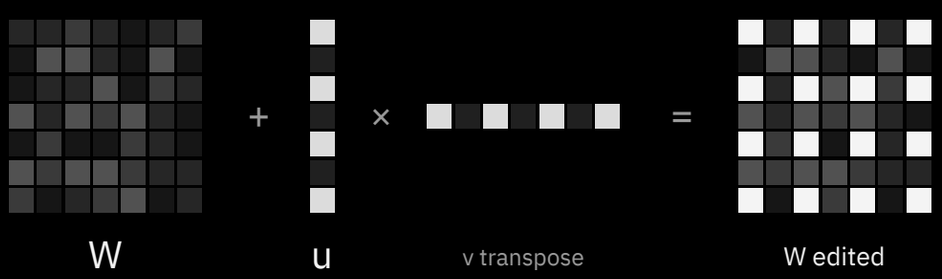
ACTIVATIONS

**Change a state during a run.**

WEIGHTS

**Change a structure across runs.**

A COMPACT OPERATION ON A PERSISTENT STRUCTURE




An algebraic update is precise; its behavioral scope is empirical.

An intervention is an operation; its value is an experimental question.

| | |
|---|---|
| MEASURE | Specify behavior, record conditions and preserve a baseline. |
| INTERVENE | Change a defined part of the computation and compare consequences. |
| JUDGE | Test transfer, collateral effects and alternatives before keeping the edit. |

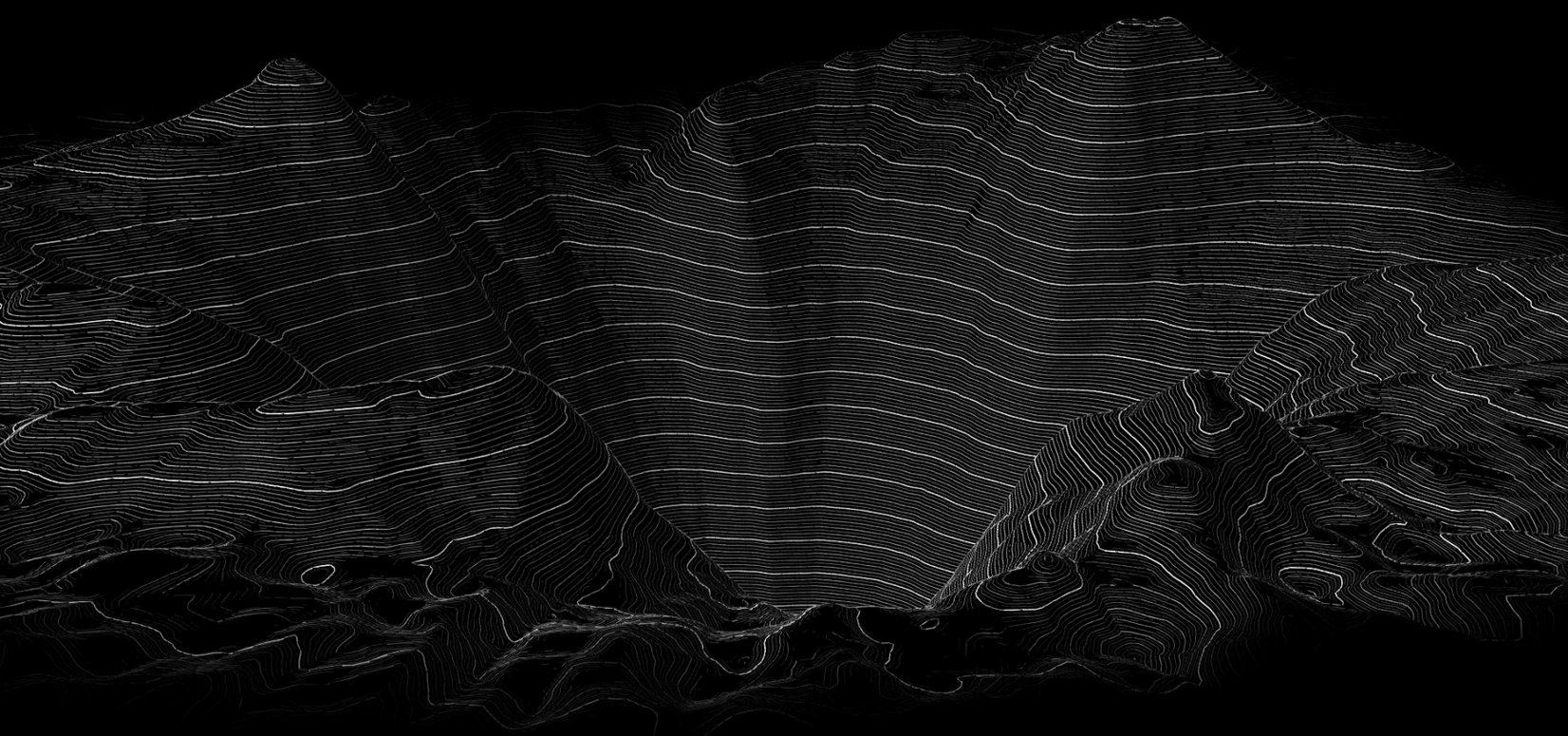
Scope matters. The core pursuit is steering without retraining the base model. Some readings train an auxiliary controller or representation edit; some weight editors optimize a target before changing a matrix. Their learning requirements stay explicit.

SESSIONS 01-05

01

# Encounter

*First, learn to see.*



Begin with images and possibilities. Then give the phenomenon a definition, its observation a record, and its computation an instrument.

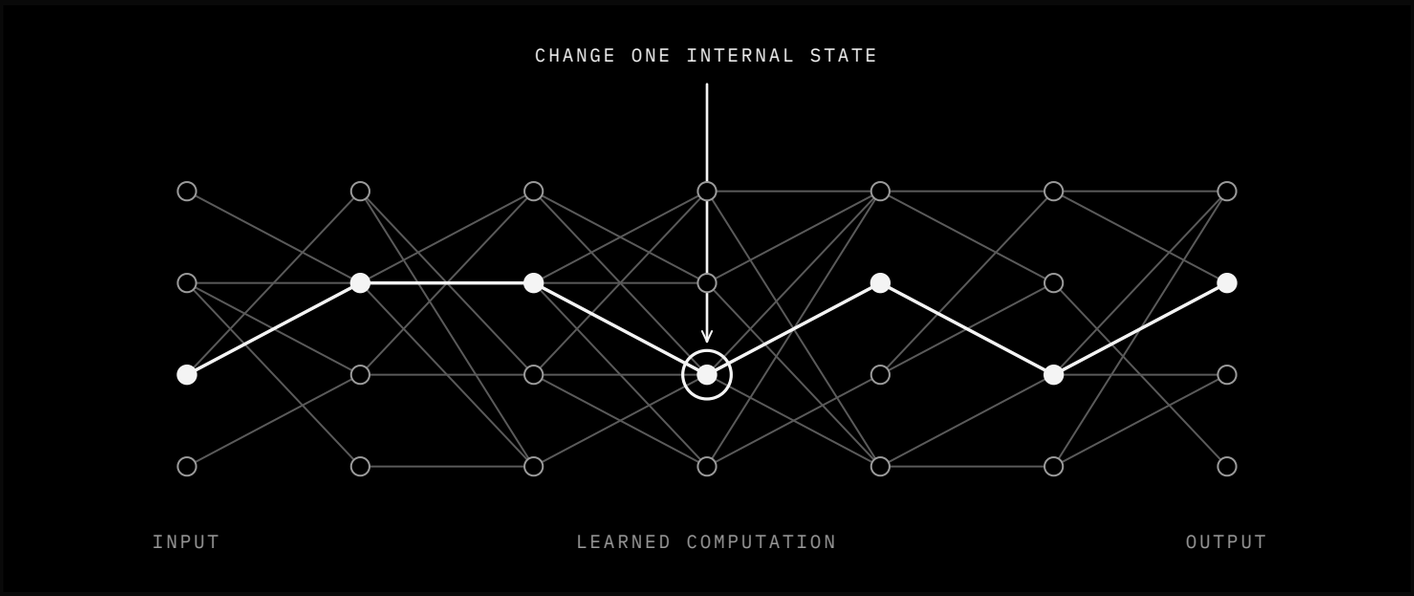
BY THE END OF THIS WEEK

A reproducible behavioral baseline and a first view inside a forward pass.

SESSION 01

# Can Neural Networks Be Programmed?

An initiation into the possibility of working inside learned computation.




READINGS / IN ORDER

01 **The Building Blocks of Interpretability**
Chris Olah et al. / 2018 / Distill

A visual entrance into the subject. Feature visualization, attribution and dimensionality reduction become complementary instruments for exploring internal computation. Begin with the interfaces and the possibilities they make visible.

02 **Feature Visualization**
Chris Olah et al. / 2017 / Distill

Shows how to generate inputs that activate a neuron, channel or layer. It gives students something to see before formal theory, while separating an evocative visualization from an explanation of a decision.

03 **Towards Neural Programming Interfaces**
Zachary Brown et al. / 2020 / NeurIPS

An early precedent for controlling a pretrained model through hidden activations. Its auxiliary controller is trained while the base model stays fixed: a conceptual ancestor, with a different learning requirement from direct activation editing.

THE EXERCISE

Watch one controlled intervention change a model output. Identify what was observed, what was changed, and what remains unexplained.

THE QUESTION THAT FOLLOWS

*If a learned network can be treated as a computational object, how does working on it differ from conventional programming?*

SESSION 02

# From Symbols to Learned Computation

The program can be written as rules, learned as weights, or revised through intervention.

AUTHORED RULES

```
if condition:
    choose action A
else:
    choose action B
```

Structure is specified directly.

LEARNED PARAMETERS




Structure emerges through fitting.

READINGS / IN ORDER

01 **Software 2.0**

Andrej Karpathy / 2017 / Author’s essay

The philosophical provocation: learned weights can be treated as software produced through optimization. What would it mean to inspect and revise a program that no one authored line by line?

02 **A Mathematical Framework for Transformer Circuits**

Nelson Elhage et al. / 2021 / Anthropic

Introduces computational paths that read from and write to a shared residual stream. Selected diagrams connect the idea of learned software to concrete objects; the mathematical derivations become a reference to revisit.

03 **Thinking Like Transformers**

Gail Weiss, Yoav Goldberg, Eran Yahav / 2021 / ICML / PMLR

Proposes RASP, a small programming language whose primitives correspond to attention and feed-forward operations. A RASP program shows one way to solve the task and how many layers and heads are enough. Students gain an authored program in the network’s own terms to set beside the learned one.

THE EXERCISE

Compare a small explicit program with a learned network for the same task. Mark the objects each makes available to inspection and change.

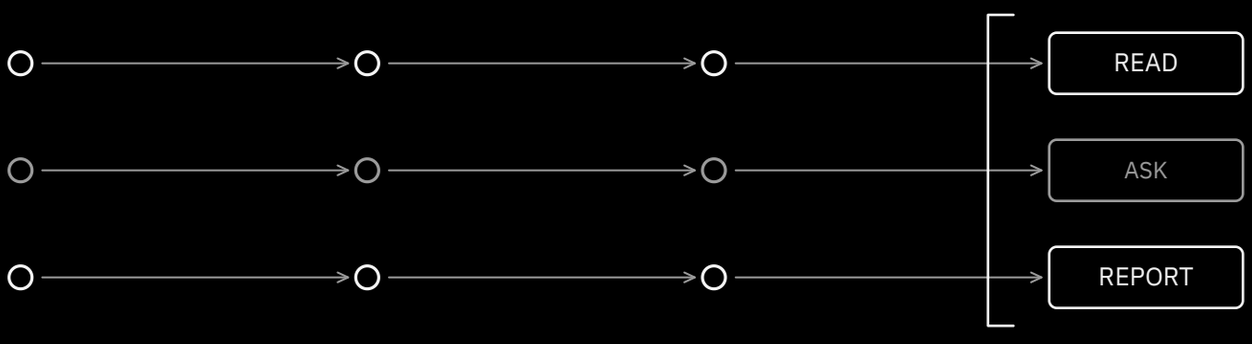
THE QUESTION THAT FOLLOWS

*If learned computation is still computation, how can we observe what a network actually does?*

SESSION 03

# What Is a Behavioral Phenomenon?

Turn a surprising output into a question that an experiment can answer.




The same task may admit more than one acceptable outcome.

READINGS / IN ORDER

01 **AgentBench: Evaluating LLMs as Agents**

Xiao Liu et al. / 2024 / ICLR

Moves evaluation into interactive environments where sequences of actions and observations determine success. The experimental object includes tools and environment; a plausible sentence alone cannot establish that an agent completed its task.

02 **SWE-bench: Can Language Models Resolve Real-World GitHub Issues?**

Carlos E. Jimenez et al. / 2024 / ICLR

Grounds success in executable consequences: the model must change a repository to resolve an issue. Use it to distinguish a convincing explanation from an externally assessed result when defining behavioral experiments.

THE EXERCISE

Define an observable behavior and an acceptance table. Include cases where the same action is useful and cases where it is costly.

THE QUESTION THAT FOLLOWS

*Once a behavior is defined, how do we know our observations are reproducible?*

SESSION 04

# What Makes an Experiment Reproducible?

An experimental result includes the conditions under which it was produced.

EXPERIMENTAL LEDGER

| | | |
|---|---|---|
| MODEL | weights + tokenizer | FIXED |
| CONTEXT | prompt + tool state | RECORDED |
| EXECUTION | build + cache + seed | RECORDED |
| INTERVENTION | site + strength | VARIED |

A reproducible run includes the context you did not write.

READINGS / IN ORDER

01 **Improving Reproducibility in Machine Learning Research**

Joelle Pineau et al. / 2021 / JMLR

A foundation for clear reporting, shared code and explicit experimental conditions. Translate its principles into a notebook that records requests, settings, versions, data and evaluation rules closely enough for another person to examine the result.

02 **Holistic Evaluation of Language Models (HELM)**

Percy Liang et al. / 2022 / Stanford CRFM

Organizes evaluation across scenarios and desirable properties. Use its methodology to build an acceptance table, make tradeoffs visible, and document what the experiment covers, omits and measures.

THE EXERCISE

Build a record of requests, model versions, environment state and evaluation rules. Repeat a baseline while varying one declared condition.

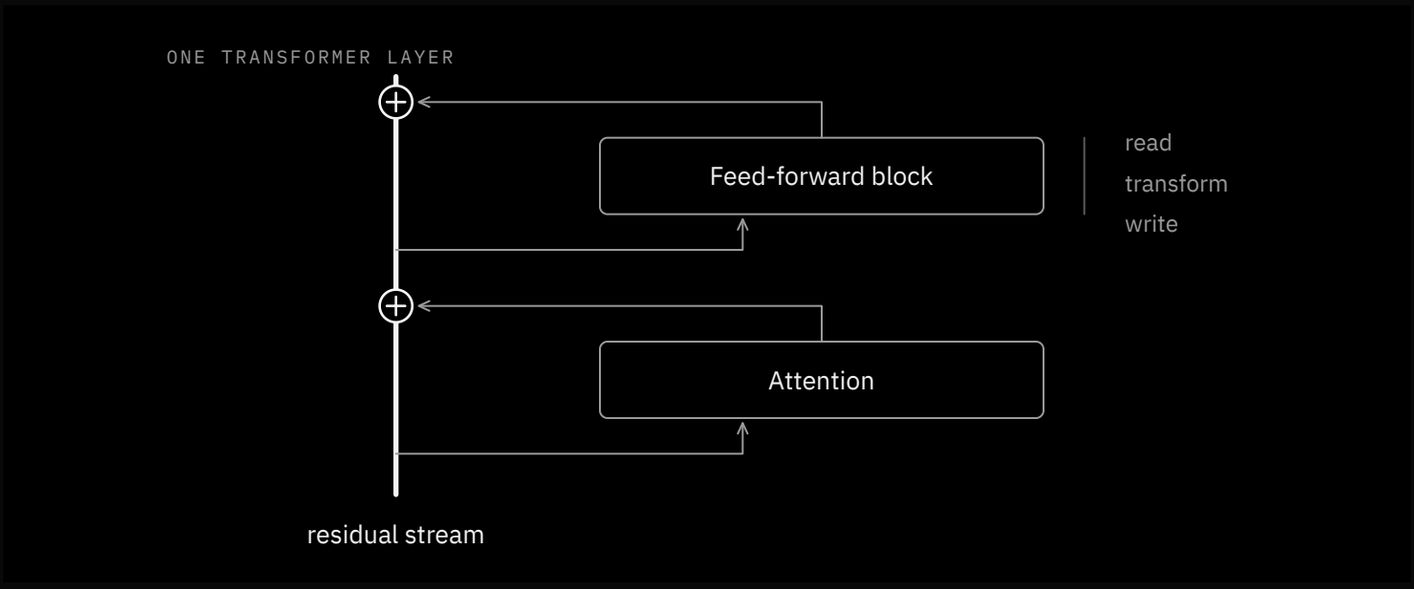
THE QUESTION THAT FOLLOWS

*We can measure the input and output. What happens between them?*

SESSION 05

# What Happens Inside a Forward Pass?

Make an internal computation visible, one measurement at a time.




READINGS / IN ORDER

01 **How Might LLMs Store Facts?**
Grant Sanderson; text adaptation by Justin Sun / 2024 / 3Blue1Brown

A visual introduction to the matrix multiplications and nonlinearities inside transformer MLPs. Pair the illustrative construction with an instrumented forward pass, keeping the example distinct from a full account of factual storage.

02 **ARENA: Introduction to Mechanistic Interpretability**
ARENA educational project / Living resource / Curriculum

Hands-on exercises for caching activations and inspecting attention. Reuse the teaching infrastructure while pinning a compatible environment: the software changes faster than the conceptual lessons about observation and intervention.

03 **TransformerLens**
Neel Nanda and contributors / Living resource / Research library

An experimental workbench for recording and changing internal activations. It makes measurement and intervention possible; the meaning of those measurements remains a question for the experiment.

04 **interpreting GPT: the logit lens**
nostalgebraist / 2020 / LessWrong

Decodes the residual stream at every layer with the model's own unembedding, so a prediction can be watched as it forms. It is the cheapest instrument in the course: one matrix product per layer.

THE EXERCISE

Instrument a small model. Record its residual stream, attention outputs and MLP activations for a complete, token-aligned forward pass.

THE QUESTION THAT FOLLOWS

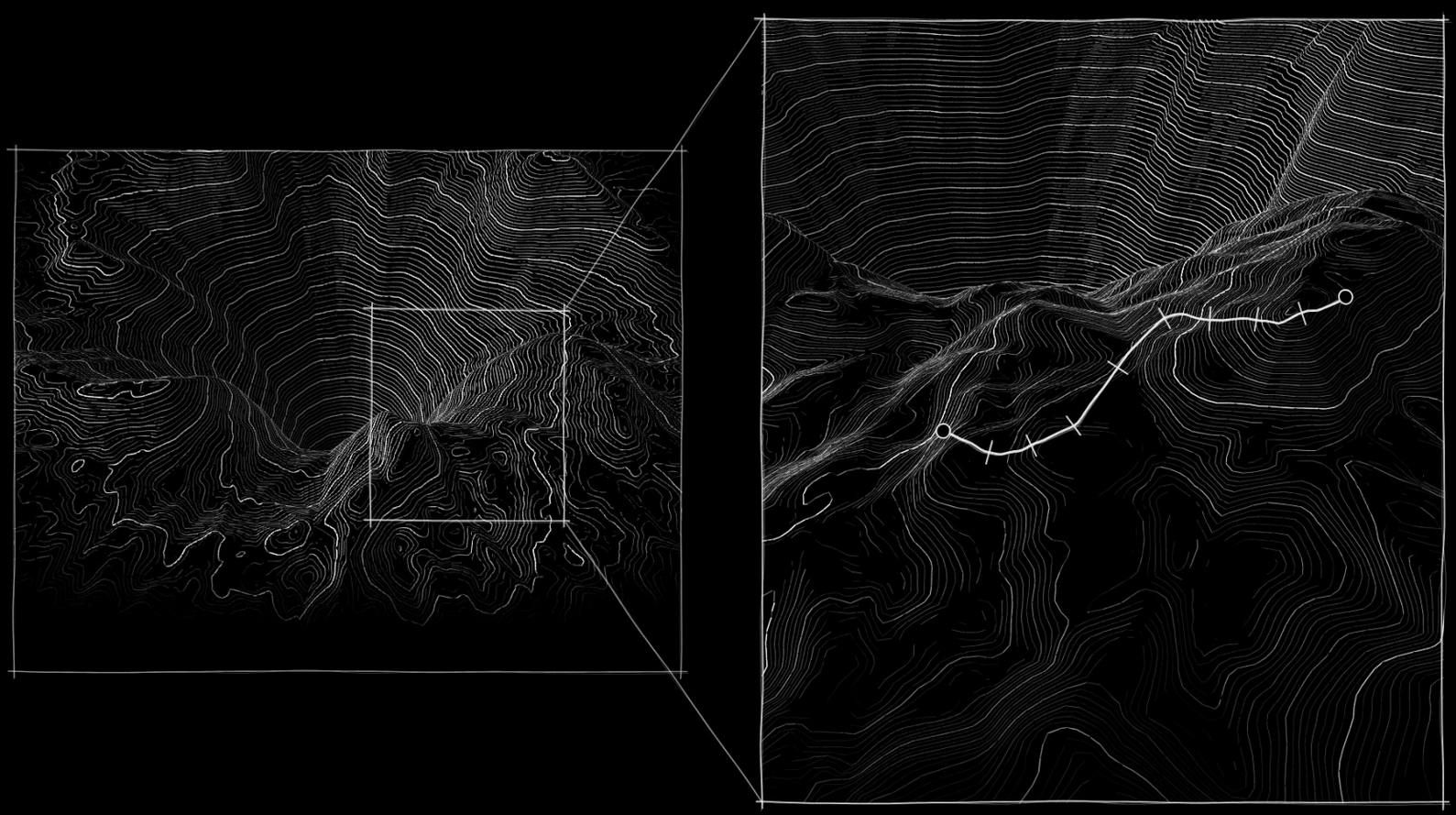
*How do we distinguish meaningful computational differences from incidental variation?*

SESSIONS 06-10

02

# Evidence

*Then, learn what a change can establish.*



Compare states, make interventions, locate effects and test explanations. Each experiment narrows the distance between a suggestive image and a defensible claim.

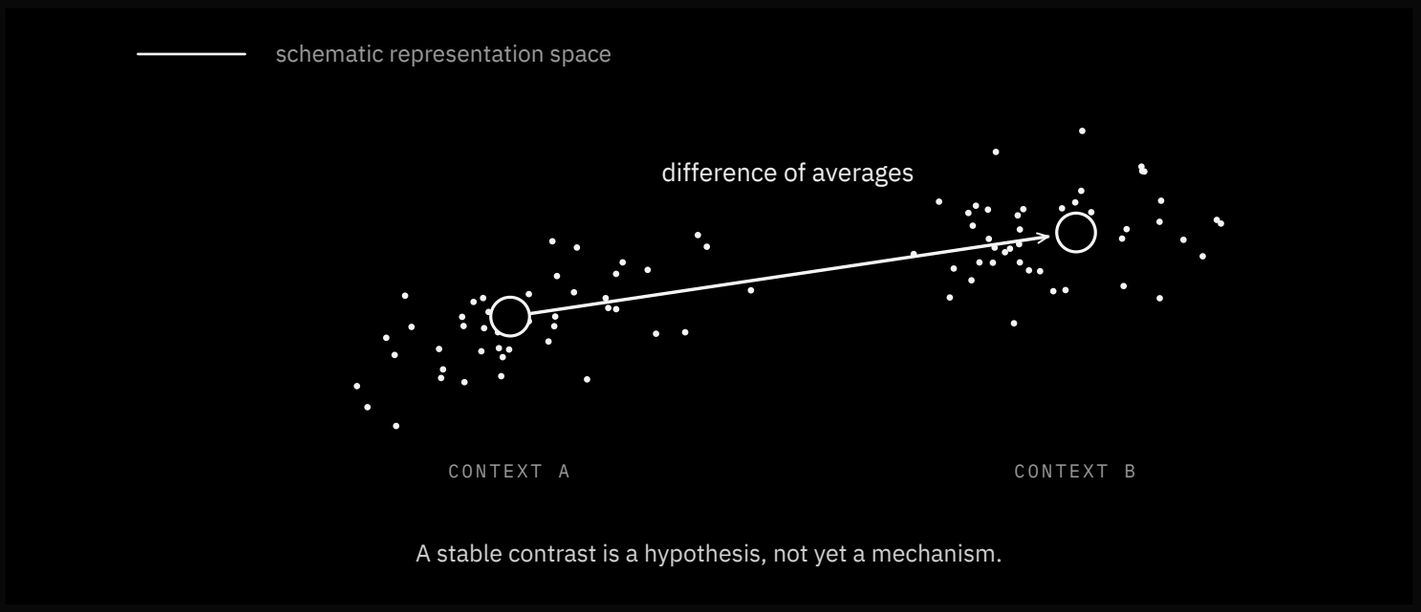
BY THE END OF THIS WEEK

A bounded causal claim, supported by an intervention and challenged by alternatives.

SESSION 06

# How Do Internal States Differ?

A difference in coordinates becomes useful only through a well-defined comparison.




A stable contrast is a hypothesis, not yet a mechanism.

READINGS / IN ORDER

01 **Toy Models of Superposition**
Nelson Elhage et al. / 2022 / Anthropic

Tractable networks show how more features than dimensions can coexist in a representation. The lesson is foundational: the largest coordinate difference need not identify a single concept or behavioral mechanism.

02 **The Linear Representation Hypothesis and the Geometry of Large Language Models**
Kiho Park, Yo Joong Choe, Victor Veitch / 2024 / ICML / PMLR

Examines what it means for concepts to correspond to directions and which geometry makes comparisons meaningful. A counterfactual formalism connects representations, probing and steering, complicating the apparent simplicity of a similarity score.

03 **Eliciting Latent Predictions from Transformers with the Tuned Lens**
Nora Belrose et al. / 2023 / arXiv

Trains a small affine translator for each layer before decoding, correcting the logit lens where earlier states use a different basis from the last. Two instruments can read the same state differently; a difference between contexts earns more trust when it holds under both.

THE EXERCISE

Construct paired inputs and compare their activations. Inspect token alignment, context differences and stability across several examples.

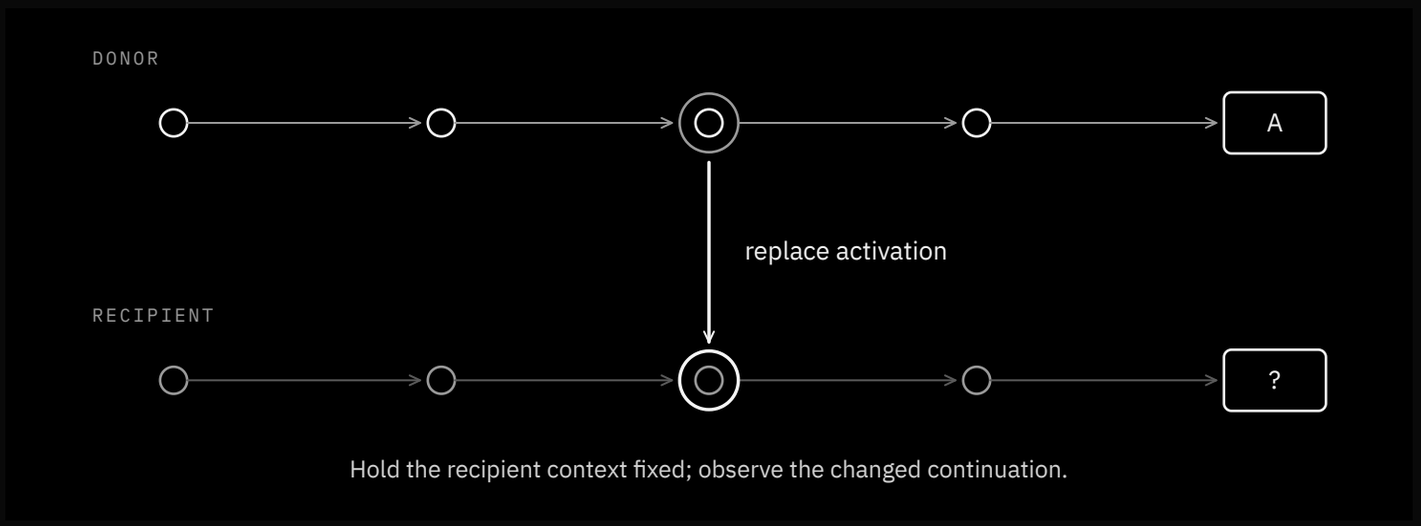
THE QUESTION THAT FOLLOWS

*If an activation differs between two behaviors, does that difference help cause the behavior?*

SESSION 07

# Does Correlation Imply Computation?

Replace a state, observe a consequence, and state the scope of the causal claim.




Hold the recipient context fixed; observe the changed continuation.

READINGS / IN ORDER

01 **How to Use and Interpret Activation Patching**

Stefan Heimersheim, Neel Nanda / 2024 / Research tutorial / arXiv

A methodological guide to replacing internal activations and interpreting the effect. It makes the choice of patch, reference and outcome metric explicit, and helps students state the limited claim an experiment supports.

02 **Towards Best Practices of Activation Patching in Language Models: Metrics and Methods**

Fred Zhang, Neel Nanda / 2024 / ICLR

Shows that metrics, corruption methods and other patching choices can change the conclusions. Use it to design a second experiment that checks whether the first result depends on the measurement procedure.

THE EXERCISE

Patch a recorded activation into a matched run. Compare with unpatched and control interventions using a predeclared outcome metric.

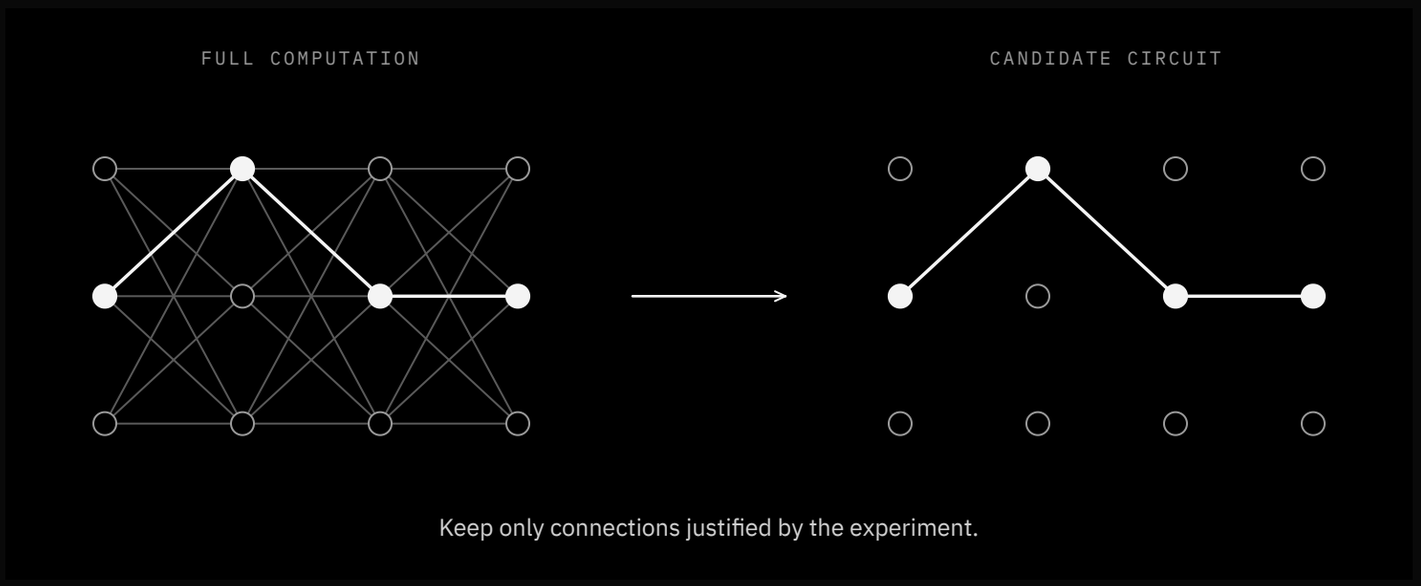
THE QUESTION THAT FOLLOWS

*If an intervention affects an output, can we localize the components responsible?*

SESSION 08

# Where Does a Behavior Come From?

Move from an influential location toward a tested account of interacting components.




READINGS / IN ORDER

01 **Interpretability in the Wild: A Circuit for Indirect Object Identification in GPT-2 Small**

Kevin Wang et al. / 2022 / Research paper; ICLR 2023

A worked investigation of a language behavior and its candidate circuit. The analysis tests faithfulness, completeness and minimality, giving students a concrete model of how to challenge a proposed mechanism.

02 **Towards Automated Circuit Discovery for Mechanistic Interpretability**

Arthur Conmy et al. / 2023 / NeurIPS

Introduces ACDC, which automates parts of circuit discovery. Its inputs make a central dependency visible: an automated search still begins with a defined behavior, dataset and metric.

03 **Transformer Feed-Forward Layers Are Key-Value Memories**

Mor Geva et al. / 2021 / EMNLP

Reads each feed-forward layer as a key-value memory: the first matrix detects patterns in the input, and the second writes what each pattern promotes in the output vocabulary. This suggests a place to try editing a fact. Later in the course, Meng and colleagues put that idea to the test.

THE EXERCISE

Localize an effect across layers and positions. Remove or restore candidate paths, then check which claims survive the new interventions.

THE QUESTION THAT FOLLOWS

*Can we describe these computations using units more meaningful than individual neurons?*

SESSION 09

# Can Representations Be Identified?

A feature is a proposed way of describing a computation, with tests still to pass.




Reconstruction error is part of the measurement.

READINGS / IN ORDER

01 **Towards Monosemanticity: Decomposing Language Models With Dictionary Learning**
Trenton Bricken et al. / 2023 / Anthropic

Uses sparse dictionary learning to find features more interpretable than individual neurons. Read it as a proposal for the units of explanation, then ask how reconstruction and interventions test those units.

02 **Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet**
Adly Templeton et al. / 2024 / Anthropic

Extends feature discovery to Claude 3 Sonnet and explores changes to behavior through feature manipulation. The scale is inspiring; the causal and interpretive limits remain part of the lesson.

03 **Gemma Scope: Open Sparse Autoencoders Everywhere All At Once on Gemma 2**
Tom Lieberum et al. / 2024 / Google DeepMind / BlackboxNLP

Released sparse autoencoders allow practical feature experiments without first training a dictionary. Coverage spans all layers and sublayers of Gemma 2 2B and 9B, with selected layers of 27B.

THE EXERCISE

Compare raw coordinates with sparse features. Inspect examples, reconstruction error and the outcome of a carefully chosen feature edit.

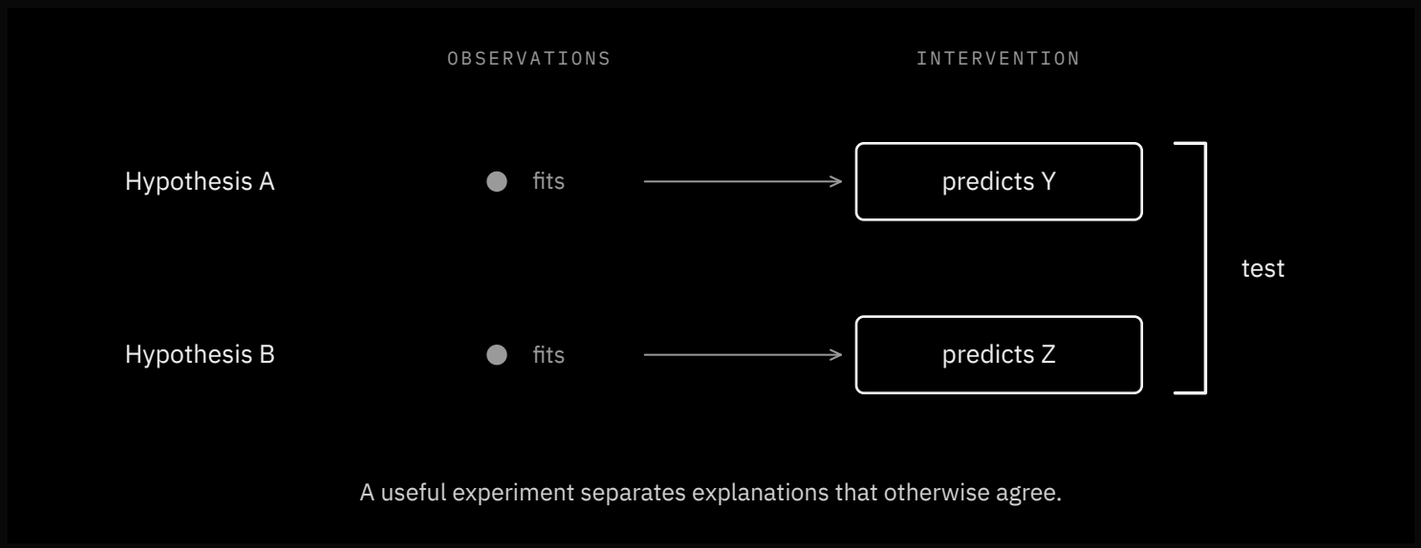
THE QUESTION THAT FOLLOWS

*How do we know that a feature description is faithful to what the network computes?*

SESSION 10

# How Do We Know an Explanation Is Correct?

Make competing explanations disagree on an experiment before choosing between them.




A useful experiment separates explanations that otherwise agree.

READINGS / IN ORDER

01 **Causal Abstraction: A Theoretical Foundation for Mechanistic Interpretability**

Atticus Geiger et al. / 2025 / JMLR

A theoretical account of when a higher-level description faithfully abstracts lower-level computation. It gives students language for distinguishing an understandable story from an explanation that survives causal tests.

02 **Challenges with Unsupervised LLM Knowledge Discovery**

Sebastian Farquhar et al. / 2023 / Google DeepMind / arXiv

A methodological counterweight: a procedure that appears to discover knowledge can instead recover another prominent feature. Use it to develop alternative explanations before accepting the interpretation of a probe.

03 **Towards Reliable Evaluation of Behavior Steering Interventions in LLMs**

Itamar Pres et al. / 2024 / NeurIPS Workshop on Foundation Model Interventions

Argues for more reliable comparisons of behavioral steering through realistic contexts and evaluation choices. Read alongside the causal studies to separate a mechanism claim from a claim of useful control.

THE EXERCISE

Write a causal claim with explicit scope. Challenge it with alternative explanations, held-out inputs and matched control interventions.

THE QUESTION THAT FOLLOWS

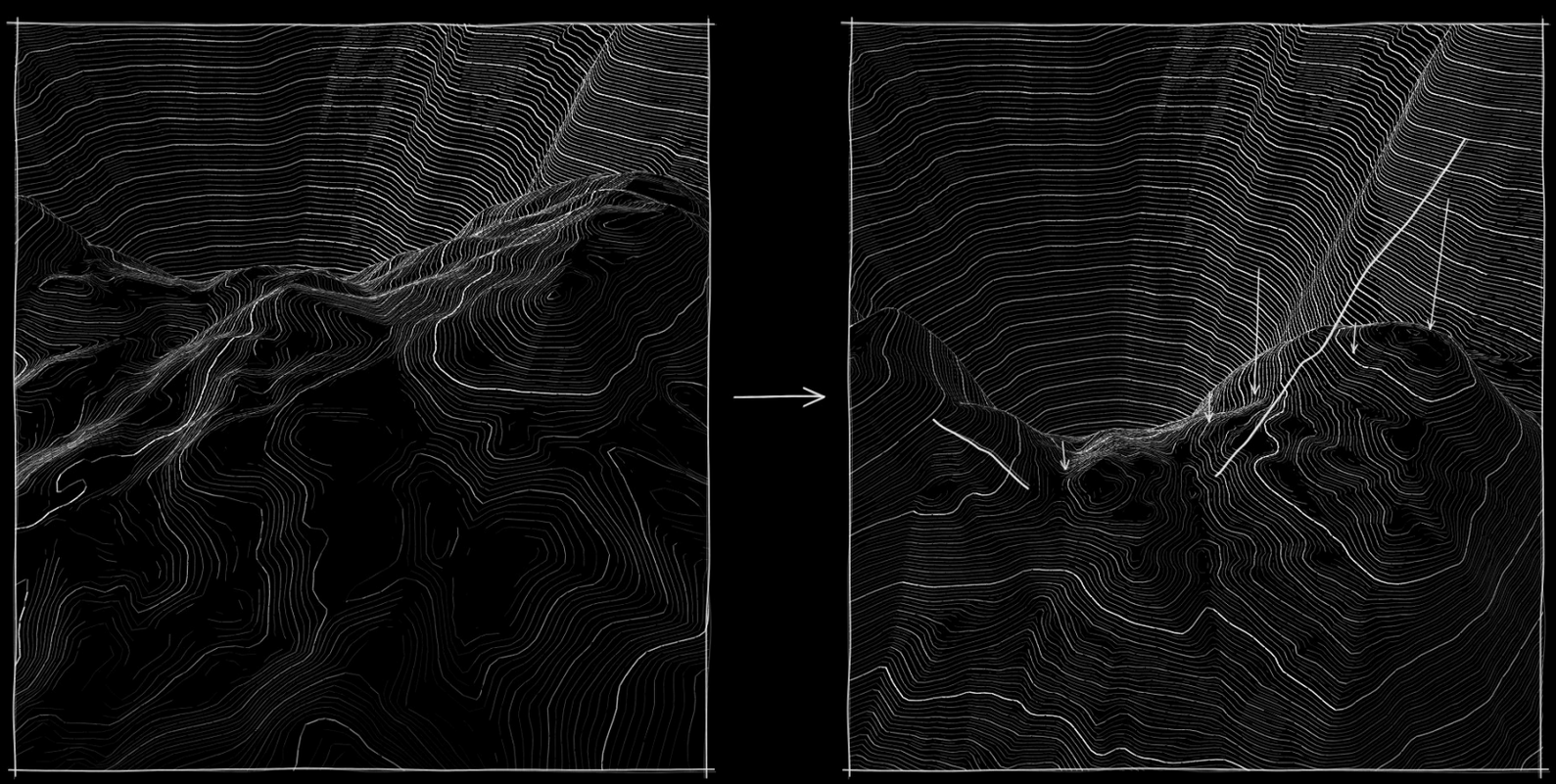
*If we have causal evidence about a computation, can we use it to deliberately produce a new behavior?*

SESSIONS 11-15

03

# Intervention

*Learn to change the computation.*



Build activation edits, decide when they should fire, and measure what they disturb. Then examine how direct parameter edits alter the computation across future runs.

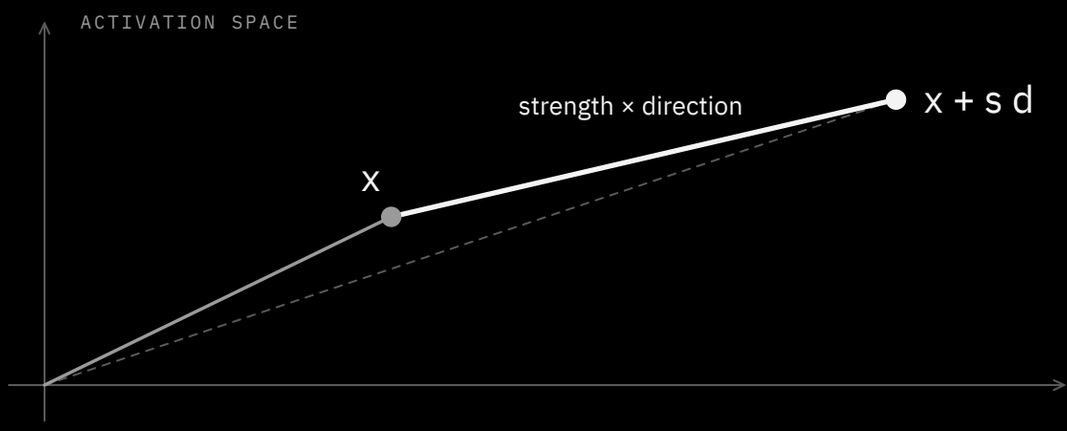
BY THE END OF THIS WEEK

Activation and weight interventions assessed for effect, specificity and persistence.

SESSION 11

# Can We Steer an Activation?

Construct a direction, choose a strength, and measure the behavior that follows.




Direction, strength, and insertion site are independent choices.

READINGS / IN ORDER

01 **Steering Language Models With Activation Engineering**
Alexander Matt Turner et al. / 2023 / arXiv

Introduces Activation Addition: construct a direction from a contrast between prompt activations, then add it during inference. The paper gives students a concrete first operation for changing output properties without optimizing model weights, and a reason to measure effects beyond a single attractive example.

02 **Steering Llama 2 via Contrastive Activation Addition**
Nina Rimsky et al. / 2024 / ACL

Builds steering vectors by averaging differences between paired examples of a behavior. Students can reproduce the contrast, vary its coefficient, and inspect how a direction changes generation. The central lesson is that an intervention is constructed from a dataset, whose design shapes what the vector captures.

03 **Representation Engineering: A Top-Down Approach to AI Transparency**
Andy Zou et al. / 2023 / arXiv

Proposes representation engineering as a top-down approach: read a concept's direction from the activations of contrasting stimuli, then control behavior by adding to or projecting along it. Students place the two earlier methods within one vocabulary of reading and control, and compare ways of extracting a direction.

THE EXERCISE

Build a contrastive steering direction. Sweep its strength and compare target effects with a prompt-only baseline and unrelated behaviors.

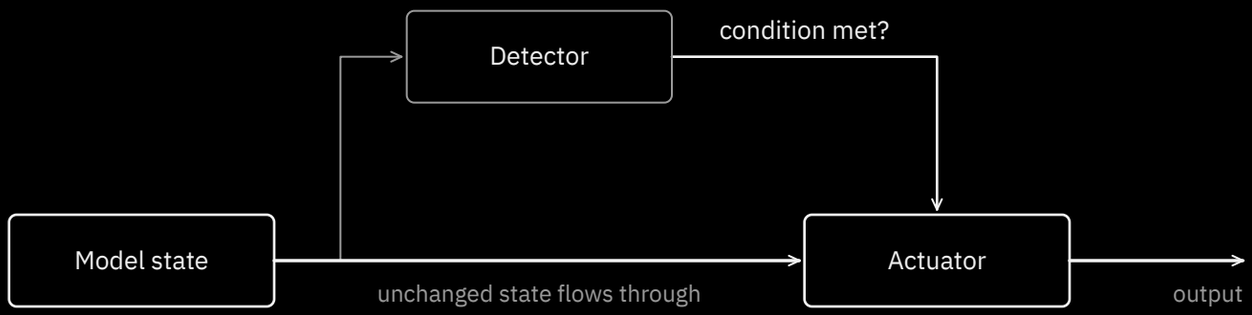
THE QUESTION THAT FOLLOWS

*A steering direction changes outputs. Should it be applied to every input?*

SESSION 12

# When Should an Intervention Occur?

A detector decides when; an intervention determines what is changed.




Detection quality and intervention quality need separate tests.

READINGS / IN ORDER

01 **Programming Refusal with Conditional Activation Steering**
Bruce W. Lee et al. / 2025 / ICLR / Spotlight

Introduces Conditional Activation Steering, which uses internal activation patterns to determine whether a steering operation should fire. Its selective-refusal experiments motivate a general design exercise: separate the question of what an intervention does from the classifier that decides when to apply it.

02 **Steering Without Side Effects: Improving Post-Deployment Control of Language Models**
Asa Cooper Stickland et al. / 2024 / arXiv

Studies what happens when selective steering affects benign inputs, including cases misclassified by the gate. KL-then-steer first trains the model to reduce unwanted changes. Read it as a comparison that sharpens the course's training-free objective: conditional application alone does not guarantee low collateral damage.

03 **Refusal in Language Models Is Mediated by a Single Direction**
Andy Arditi et al. / 2024 / NeurIPS

Read for one method, weight orthogonalization: each matrix that writes to the residual stream is projected so that it can no longer write along a chosen direction. The intervention then occurs in every forward pass, with no detector and no hook. Students set this permanent form beside the conditional one, apart from the behavior the paper studies.

THE EXERCISE

Evaluate a gate separately from its edit. Measure missed opportunities and false triggers before testing the combined intervention.

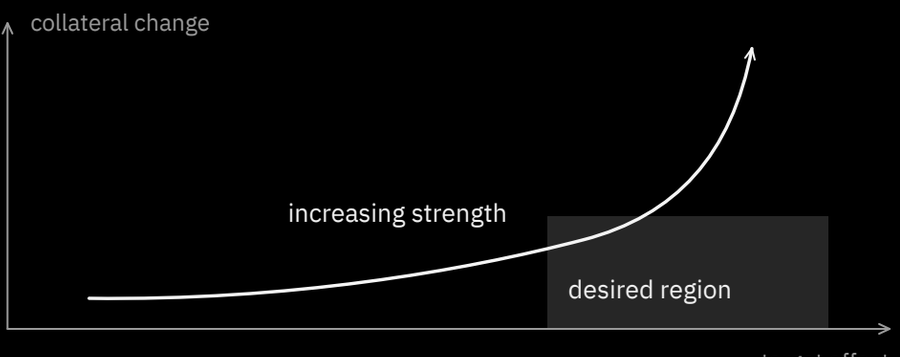
THE QUESTION THAT FOLLOWS

*Even a selective intervention can cause damage. What would make its effect proportionate and reliable?*

SESSION 13

# How Do We Control Without Breaking?

An intervention earns its value across both intended effects and collateral consequences.




Conceptual tradeoff; the curve is not measured data.

READINGS / IN ORDER

01 **Analyzing the Generalization and Reliability of Steering Vectors**
Daniel Tan et al. / 2024 / arXiv

Tests steering both within and outside its original distribution. Effects vary across inputs, spurious biases influence measured success, and reasonable prompt changes can break some directions. This reading turns a successful demonstration into a more demanding question: under which conditions does the intervention remain useful?

02 **AxBench: Steering LLMs? Even Simple Baselines Outperform Sparse Autoencoders**
Zhengxuan Wu et al. / 2025 / ICML / PMLR

Compares steering and concept detection across methods on Gemma 2. Prompting outperforms the tested existing steering methods, while other techniques fare better at detection. Students learn to evaluate the intervention against inexpensive alternatives and to distinguish finding a concept from controlling its expression.

03 **Towards Reliable Evaluation of Behavior Steering Interventions in LLMs**
Itamar Pres et al. / 2024 / NeurIPS workshop

Returns as a laboratory protocol: choose realistic contexts, account for model likelihoods, compare behaviors consistently, and include baselines. Students use these criteria to turn visually impressive steering into a quantitative evaluation, reporting both the intended behavioral effect and where the evidence is weaker.

THE EXERCISE

Compare several strengths on target and preservation tasks. Include simple prompting, norm-matched controls and independent evaluation examples.

THE QUESTION THAT FOLLOWS

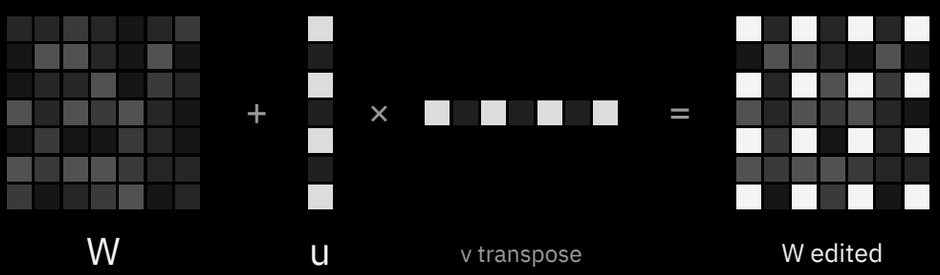
*If a useful edit must be reapplied during inference, can the underlying weights be changed instead?*

SESSION 14

# Can We Program Weights Directly?

Treat parameters as an editable computational structure, with consequences to verify.

A COMPACT OPERATION ON A PERSISTENT STRUCTURE




An algebraic update is precise; its behavioral scope is empirical.

READINGS / IN ORDER

01 **Locating and Editing Factual Associations in GPT**

Kevin Meng et al. / 2022 / NeurIPS

Connects causal localization to Rank-One Model Editing, changing feed-forward weights to update a factual association. It supplies a historical bridge from observing internal states to persistent edits. Its value-selection step uses optimization, so direct weight editing should not be confused with an entirely optimization-free procedure.

02 **Sparks of Weight-Space Programming: LLMs can natively read & write neural network weights**

Rikiya Takehi et al. / 2026 / MIT CSAIL / research blog

Reports early experiments in which a frontier language model reads, writes, and edits another network’s parameters without training loops. The examples include small control policies and a language-model edit. Treat this as a concrete research invitation to reason about weights as executable structure, with claims bounded by the experiments.

03 **Weight-Space Programming Doesn’t Need Transformers**

Hyunjong Ok, Jaeho Lee / 2026 / Preprint / Work in progress

Asks which architectures make sense when an agent writes weights directly. In the reported experiments, agents favor explicit associative tables over a fixed transformer design. The provocation is architectural: a system designed for gradient training need not be the best substrate for deliberate weight construction.

THE EXERCISE

Make a small, reversible weight edit. Inspect the changed computation and compare its effects with an activation-level intervention.

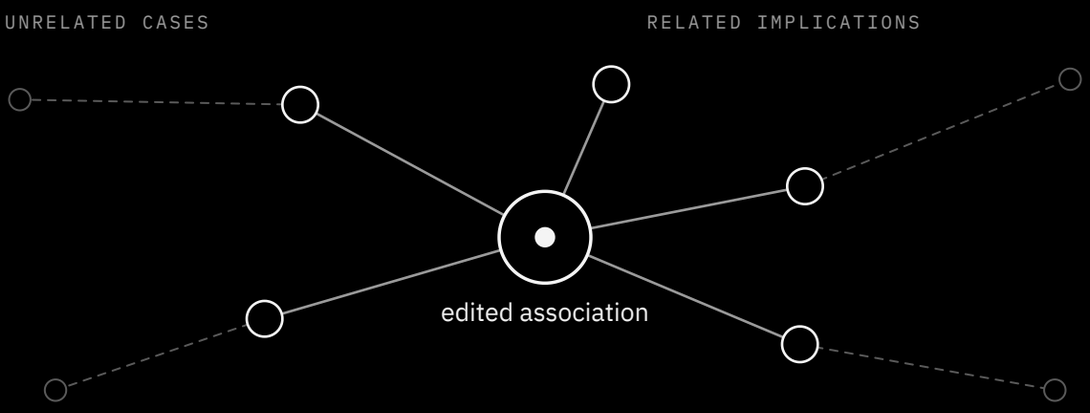
THE QUESTION THAT FOLLOWS

*One persistent edit is possible. What happens when edits accumulate or interact with related knowledge?*

SESSION 15

# Can a Model Retain an Intervention?

Persistence is only the beginning: related behavior must remain coherent.




Persistence does not guarantee coherent consequences.

READINGS / IN ORDER

01 **Mass-Editing Memory in a Transformer**
Kevin Meng et al. / 2023 / ICLR

Introduces MEMIT, extending direct factual editing to many associations across multiple layers. Students study how persistent changes accumulate and how efficacy, generalization, and specificity must be assessed together. Like ROME, the method uses optimized target representations, so it is an adjacent technique rather than a wholly gradient-free recipe.

02 **Evaluating the Ripple Effects of Knowledge Editing in Language Models**
Roi Cohen et al. / 2024 / TACL

Introduces RippleEdits to test implications beyond the one fact explicitly changed. An edit can succeed locally while leaving related answers inconsistent. Students expand the acceptance test from direct recall to the network of consequences an update should have, including a comparison with in-context editing.

03 **Model Editing Harms General Abilities of Large Language Models: Regularization to the Rescue**
Jia-Chen Gu et al. / 2024 / EMNLP

Measures general capabilities after factual model edits and investigates regularizing the updates. It makes persistence an engineering concern beyond successful recall: a model must retain its broader usefulness. Students define tests for unrelated tasks and consider how edit magnitude can trade targeted gains against collateral losses.

THE EXERCISE

Test one edit across paraphrases, related implications and unrelated capabilities. Repeat after saving and reloading the edited model.

THE QUESTION THAT FOLLOWS

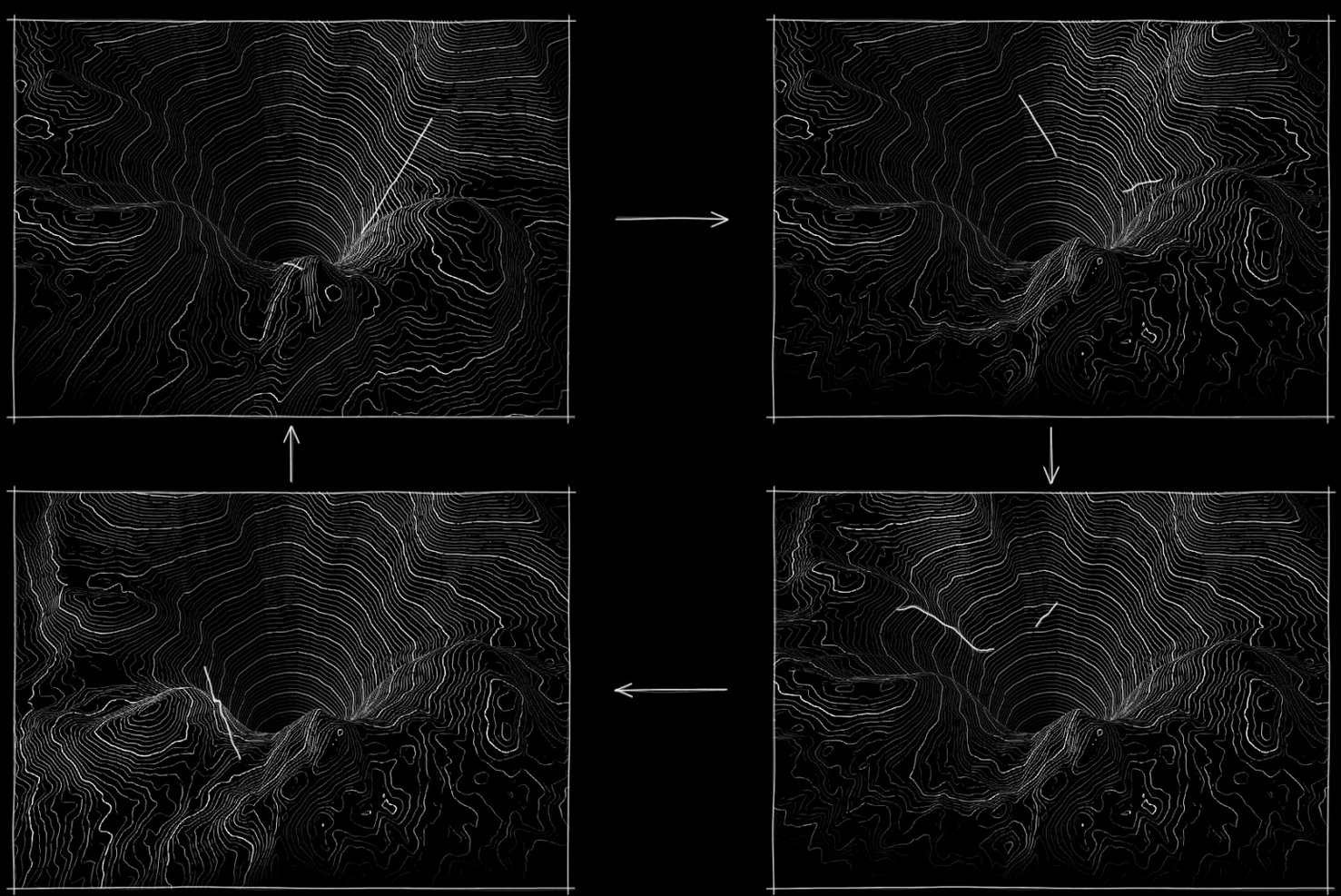
*Can transient and persistent interventions become parts of one repeatable programming practice?*

SESSIONS 16-20

04

# Discovery

*Make the method capable of continuing.*



Package discoveries, connect feedback, and investigate automated experiments. Return to the philosophical questions with a clearer account of what is known and what remains open.

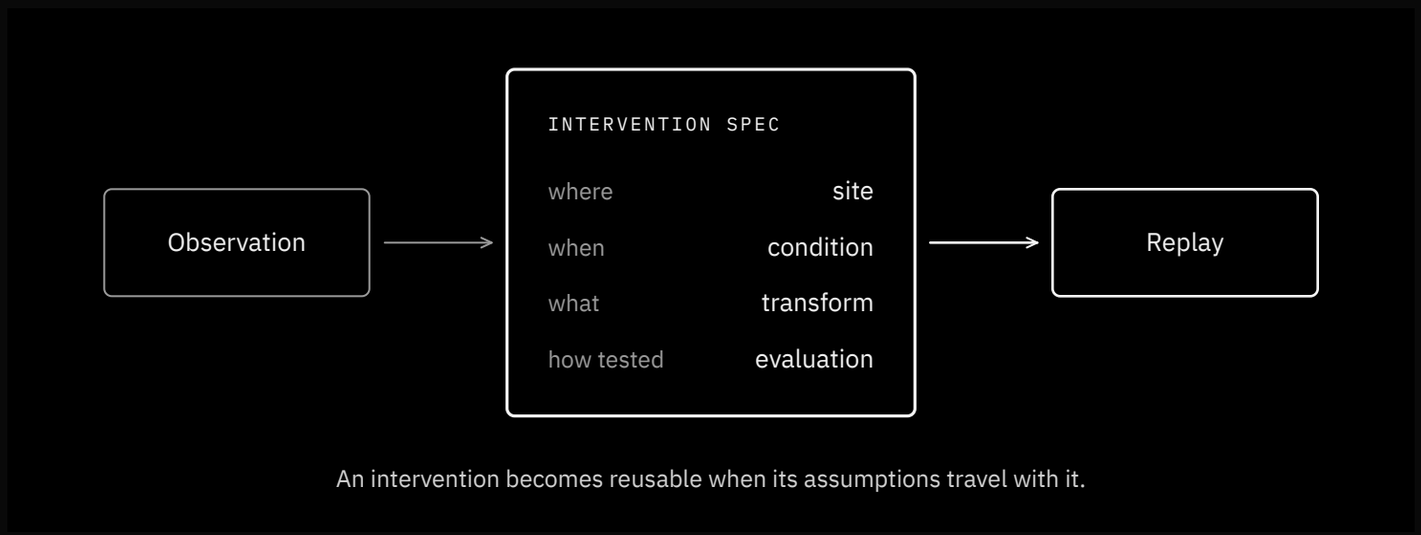
BY THE END OF THIS WEEK

A reusable experimental system and an honest account of its boundaries.

SESSION 16

# Can Discoveries Become Programs?

An experimental change becomes reusable when its assumptions travel with it.




An intervention becomes reusable when its assumptions travel with it.

READINGS / IN ORDER

01 **pyvene: A Library for Understanding and Improving PyTorch Models via Interventions**
Zhengxuan Wu et al. / 2024 / NAACL demos

Provides a common interface for configurable interventions across PyTorch models. Static edits and trainable interventions can be specified, composed, and shared. Students turn an experimental notebook into a reusable operation with an explicit location, source, transformation, and evaluation contract that others can inspect and execute.

02 **ReFT: Representation Finetuning for Language Models**
Zhengxuan Wu et al. / 2024 / NeurIPS

Learns compact interventions on hidden representations while keeping the base model frozen. This is a useful neighboring approach, because unchanged base weights do not imply that no training occurred. Students compare a hand-constructed edit with a learned intervention and identify exactly which parameters each procedure changes.

03 **Tracr: Compiled Transformers as a Laboratory for Interpretability**
David Lindner et al. / 2023 / NeurIPS

Compiles RASP programs into the weights of a working transformer, so the mechanism is known before any experiment begins. Students return to the authored program of Session 02 as a network they can run, and test an interpretability method where its findings can be checked against a known mechanism.

THE EXERCISE

Package an intervention with its model, site, condition, parameters and evaluation contract. Replay it independently and record its limits.

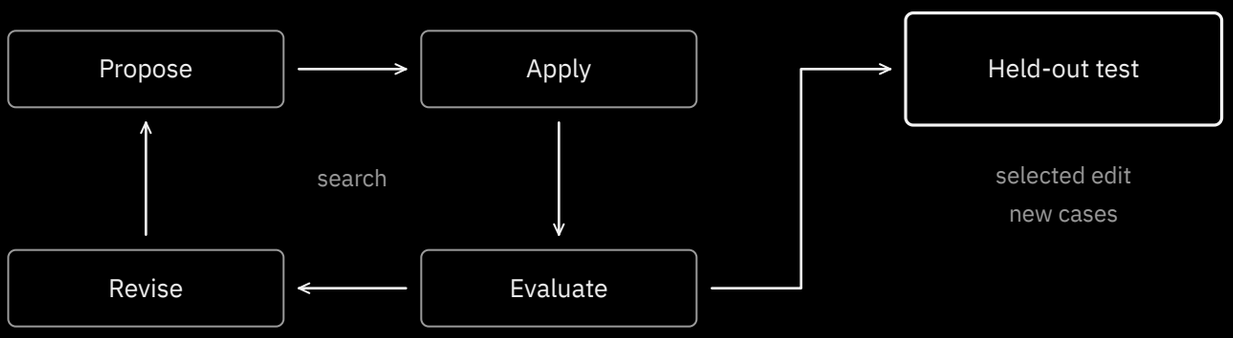
THE QUESTION THAT FOLLOWS

*How should an experimental system decide which candidate modifications are worth keeping?*

SESSION 17

# What Closes the Control Loop?

Feedback turns a sequence of trials into a method for selecting the next experiment.




Search feedback improves candidates; independent evidence tests the claim.

READINGS / IN ORDER

01 **Sparks of Weight-Space Programming: LLMs can natively read & write neural network weights**

Rikiya Takehi et al. / 2026 / MIT CSAIL / research blog

Return to the experiments as a workflow: inspect a network, form a hypothesis, construct a modification, and test the result. The focus shifts from one surprising edit to the feedback available to the editing agent, and to which parts of its reasoning can be checked independently.

02 **Exploring Weight-Space Programming: Capabilities, Generalization, and Limitations**

Junru Hu et al. / 2026 / Preprint

Use the reported trajectories to study how feedback can guide search without guaranteeing a transferable result. Students separate the data that chooses an intervention from the data that judges it, and specify budgets, rollback rules, and a stopping condition before their own edit search begins.

03 **WeightBench: Evaluating Language Models as Weight-Space Programmers**

Aayush Marasini, Barsat Khadka, Nick Rahimi / 2026 / Preprint

Organizes direct weight programming into reading, fixing, writing, recognizing impossible edits, and data-free compression. The benchmark makes the objective executable and exposes distinct capabilities and limitations. Students use its framing to ask whether their controller measures success or simply rewards a plausible-looking modification.

THE EXERCISE

Connect proposal, application, evaluation and revision. Reserve an independent final test, and define budgets and stopping rules before search.

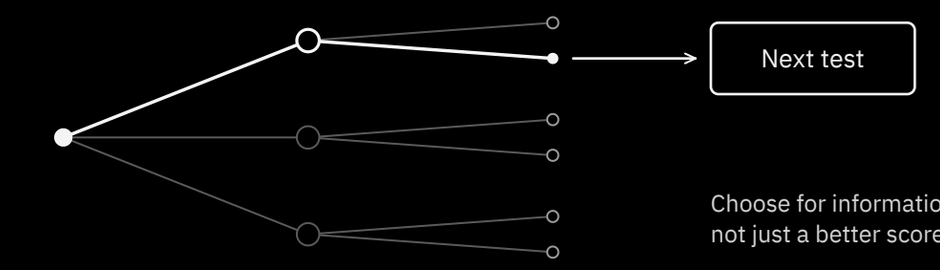
THE QUESTION THAT FOLLOWS

*If the loop is procedural, can we automate discovery rather than only execute experiments?*

SESSION 18

# Can Experiments Design Experiments?

Give an automated researcher a question, instruments, a budget and an accountable record.




The experiment itself becomes a programmable object.

READINGS / IN ORDER

01 **Language Models Can Explain Neurons in Language Models**
OpenAI / 2023 / Research publication

An early automated-interpretability system proposes neuron explanations and tests their predictive value. Separate a readable hypothesis from an explanation that predicts activation behavior, while noticing how much remains unexplained.

02 **Automatically Interpreting Millions of Features in Large Language Models**
Goncalo Santos Paulo et al. / 2025 / ICML / PMLR

Scales automated interpretation to sparse features and evaluates explanations through several methods, including intervention effects. A fluent label must survive tests across contexts, beyond a handful of persuasive examples.

03 **Interpretable Weight-Space Programming Builds Transparent Transformers and Brain Encoding Models**
Yash Mehta et al. / 2026 / Preprint

An agent constructs models by writing weights and features, evaluates them, and revises the computation without backpropagation. Its arithmetic and neuroscience studies closely approach the experimental loop developed in this course.

04 **The AI Scientist: Towards Fully Automated Open-Ended Scientific Discovery**
Chris Lu et al. / 2024 / Sakana AI et al.

A broader research architecture for ideas, code, experiments, visualization and review. Examine what can be delegated and what independent evidence remains necessary; producing a report does not itself establish its claims.

THE EXERCISE

Let an agent propose a small set of interventions. Keep its search evidence separate from final validation, and audit one accepted claim.

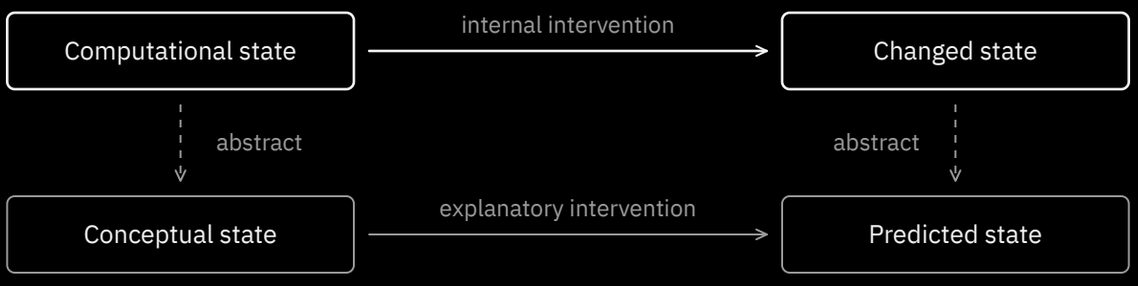
THE QUESTION THAT FOLLOWS

*If a system finds an intervention that works, what does it mean to say it understands the computation?*

SESSION 19

# What Does It Mean to Understand a Neural Network?

Return to explanation with the experience of having changed a computation.




A faithful explanation should agree along both paths.

READINGS / IN ORDER

01 **Causal Abstraction: A Theoretical Foundation for Mechanistic Interpretability**

Atticus Geiger et al. / 2025 / JMLR

Revisit the relationship between a readable explanation and a model's detailed computation. Causal abstraction offers a formal language for asking whether an intelligible description preserves the relevant effects of interventions. Students distinguish predictive description, causal influence, and an explanation that faithfully simplifies the underlying mechanism.

02 **On the Biology of a Large Language Model**

Jack Lindsey et al. / Anthropic / 2025 / Anthropic

Uses circuit tracing to investigate mechanisms in Claude 3.5 Haiku across several tasks. The detailed case studies sit alongside explicit limitations of the method. Students return to the opening visual wonder with a sharper question: which parts of a compelling computational story have actually been established?

03 **The Bitter Lesson**

Richard S. Sutton / 2019 / Author essay

Argues that scalable search and learning have repeatedly overtaken approaches built around human domain knowledge. It supplies a philosophical challenge to the course: should neural programming encode our discoveries, or build methods that discover? Students consider how automated experimentation might reconcile explicit interventions with scalable search.

THE EXERCISE

Present two accounts of the same result: a useful intervention and a proposed mechanism. State what evidence would distinguish their claims.

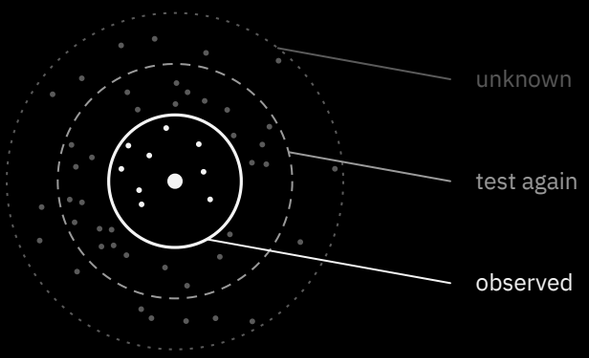
THE QUESTION THAT FOLLOWS

*What guarantees remain possible when the system is distributed, adaptive and only partially understood?*

SESSION 20

# What Are the Limits of Neural Programming?

Define the boundary of what your experiments establish, and make the next question precise.




Every claim has a domain.

READINGS / IN ORDER

01 **The Hydra Effect: Emergent Self-Repair in Language Model Computations**

Thomas McGrath et al. / 2023 / Google DeepMind / Research paper

Shows how other components can compensate when an attention layer is ablated, complicating simple attribution. This final methodological challenge asks students to consider adaptation within the forward computation: the effect of removing a component need not equal its contribution when the unmodified system is intact.

02 **AxBench: Steering LLMs? Even Simple Baselines Outperform Sparse Autoencoders**

Zhengxuan Wu et al. / 2025 / ICML / PMLR

Returns as an economic and scientific challenge: an internal intervention must earn its complexity against simpler approaches. Students assess whether the objective calls for a prompt, an external control, an activation edit, or a parameter change, using held-out behavioral evidence rather than the sophistication of the technique.

03 **WeightBench: Evaluating Language Models as Weight-Space Programmers**

Aayush Marasini, Barsat Khadka, Nick Rahimi / 2026 / Preprint

Ends the course by making limits part of the specification. Its impossible-edit and compression tasks expose capabilities that cannot be inferred from a successful small-network demonstration. Students finish with a bounded claim: what their system can change, where it fails, and what evidence would justify another experiment.

THE EXERCISE

Choose a new objective and build a defensible investigation. Report success or a well-supported negative result, including its scope and limits.

THE QUESTION THAT FOLLOWS

*Which new question can you now make tractable, and what would count as an answer?*

AFTER THE TWENTIETH QUESTION

# What the course *holds together.*

## 01 A lineage worth learning.

The readings move from feature visualization and learned software to causal intervention, representation engineering and direct weight programming. The course joins these research traditions around a common experimental practice. Their differences remain useful.

## 02 A bridge that must be built.

An intervention can work before its mechanism is understood. An explanation can be persuasive without giving reliable control. Keep the achievements distinct: an observed difference, a causal effect, a reusable intervention and a mechanistic account.

## 03 A method that can outlast its tools.

The enduring skill is to specify an objective, construct an experiment, identify candidate interventions and let evidence determine the next step. The method should remain useful as the models, libraries and preferred editing techniques change.

Interpretability supplies questions.
Interventions make them testable.
*Evaluation gives the loop its direction.*

READ FOR THE EXPERIMENT

# A reading list becomes *a practice.*

Read to answer the question in front of you. Return to a paper when a later experiment makes a different part of it necessary.

| | |
|---|---|
| VISUAL INITIATION | Explore the interfaces, diagrams and demonstrations before the formal machinery. |
| FOUNDATIONAL PAPERS | Read the central claims and methods. Reproduce the smallest useful example. |
| METHODOLOGICAL PAPERS | Study the controls, comparison choices and standards for interpreting an effect. |
| PROGRAMMING PAPERS | Reproduce an intervention. Change one design choice and explain what follows. |
| FRONTIER PAPERS | Separate what was demonstrated from what remains open. Design a stronger test. |
| PHILOSOPHICAL SYNTHESIS | Revisit the opening questions with a record of what your experiments establish. |

ONE CUMULATIVE DELIVERABLE

Maintain one experimental repository through the month: observations, hypotheses, interventions, comparisons and revised conclusions. Finish with an unfamiliar objective and an independently reproducible investigation. A well-supported negative result counts.

KEEP THESE CLOSE

# Methods to return to. Workbenches to build on.

01 **How to Use and Interpret Activation Patching**
Stefan Heimersheim and Neel Nanda / 2024
arxiv.org/abs/2404.15255

02 **Towards Reliable Evaluation of Behavior Steering Interventions in LLMs**
Pres et al. / 2024
arxiv.org/abs/2410.17245

03 **Steering Language Models With Activation Engineering**
Turner et al. / 2023
arxiv.org/abs/2308.10248

04 **Analyzing the Generalization and Reliability of Steering Vectors**
Tan et al. / 2024
arxiv.org/abs/2407.12404

05 **Exploring Weight-Space Programming: Capabilities, Generalization, and Limitations**
Hu et al. / 2026
arxiv.org/abs/2611.09382

06 **Learning by Distilling Context**
Snell, Klein and Zhong / 2022 / Context distillation, after Askell et al., 2021
arxiv.org/abs/2209.15189

HANDS-ON INFRASTRUCTURE

07 **ARENA: Transformer Interpretability**
Practical curriculum / living resource
learn.arena.education/chapter1_transformer_interp/

08 **pyvene**
Reusable intervention infrastructure
github.com/stanfordnlp/pyvene

Source note. All session readings retain the sequence of the original course map, including intentional returns to earlier papers. Publication details were checked on 26 November 2026. New 2026 work is identified as a research blog or preprint. All diagrams are original conceptual schematics, not measured results.



# The next question is now yours *to make tractable.*

---

*Wonder. Evidence. Intervention. Reflection.*

A course prospectus compiled by Daniel Rodríguez.
Four weeks / Twenty sessions / An open field of inquiry.