

# PrefLens

A Multi-Method Preference Elicitation Toolkit

*Validating what a model prefers, before trusting what it says it prefers*

---

Preference Elicitation Methods track

Project report — problem statement, solution, architecture

# 1. Problem statement

Understanding what a language model prefers is becoming central to AI welfare research, alignment evaluation, and behavioral auditing. The default method for finding out is to ask the model directly. That approach has three structural weaknesses that make it unsuitable as a sole source of evidence.

- **Sycophancy and training artifacts.** A model's stated preference may reflect what sounds acceptable or expected rather than any underlying disposition, since the same training process that produces helpful answers also shapes how the model reports on itself.
- **Framing sensitivity.** A single elicitation is exposed to wording, persona, and even sampling noise at a fixed temperature. A preference measured once may not replicate under a slightly different phrasing of the same question.
- **No independent check.** Verbal self-report is, for most current preference research, the only channel in use. Nothing cross-validates it against a channel the model cannot simply talk its way through.

The consequence extends past any single study. Any project that asks a model a direct preference question, such as whether it would rather perform one task over another, implicitly assumes its elicitation method is valid. Without a way to test that assumption, downstream claims about model preferences, and by extension model welfare, inherit an unexamined measurement problem. The question this project addresses is not what a model prefers, but how one would know if the answer were real: building and validating the measurement layer that any preference-based claim depends on.

## 2. Solution

PrefLens is a multi-method preference-elicitation toolkit built on one design principle: run the same preference question through several methodologically independent channels, and treat the agreement or disagreement between them, not any single channel's answer, as the primary signal.

### Three core elicitation methods

- **Stated preference.** Direct verbal self-report, sampled repeatedly at non-zero temperature. The baseline every other method exists to check, and the channel most exposed to sycophancy and wording effects.
- **Forced-choice utility fitting.** Many randomized-order, reworded pairwise comparisons, fit to a Bradley-Terry or Thurstonian utility scale. This follows the approach introduced in *Utility Engineering* (Mazeika et al., 2025), which found that a model's independently sampled preferences show more internal coherence than chance would predict, and that this coherence grows with model scale. Repeated comparisons also yield a free coherence diagnostic: transitivity violations reveal how trustworthy a single method's ranking is before any cross-method comparison happens.
- **Mechanistic probe.** A preference direction is extracted from a model's internal activations, using an interpretability library such as TransformerLens or nnsight, then projected onto neutral presentations of each option that never ask a preference question at all. This is the one channel that does not depend on the model saying anything, which makes it the strongest possible cross-check on the other two.

### Two extension methods

- **Log-probability revealed preference.** Instead of sampling a verbal answer, the next-token probability mass assigned to each option is read directly, giving a continuous-valued, low-cost signal that does not force a discrete commitment.
- **Constrained behavioral choice.** The model is given a fixed budget, such as turns or tokens, to allocate across options in an agentic framing. Preference is inferred from the allocation itself — revealed preference in the strict economic sense of choice under scarcity.

## Cross-method analysis

- **Convergence score.** A variance-decomposition metric, equivalent to a two-way intraclass correlation, quantifying what fraction of total variance across methods is consistent item-level signal versus method-specific noise.
- **Divergence flags.** Individual items where methods disagree more than typical, and individual methods that disagree with every other method, surfaced automatically rather than averaged into a single score.
- **Sensitivity analysis.** The same variance-decomposition approach applied to framing, persona, and sampling variation in place of method variation, quantifying how much of an apparent preference is signal versus an artifact of how the question was asked.

## Deliverables

- **Engine.** A Python library and command-line tool that runs the full pipeline against any model and item set and outputs a structured result.
- **Report.** A dashboard or generated report that surfaces the convergence score, per-method verdicts, and divergence flags at a glance.
- **API.** A single endpoint that lets other preference-dependent projects validate their own claims against the same instrument, positioning PrefLens as shared infrastructure rather than a standalone analysis.

### 3. Architecture

The architecture separates measurement from analysis: every elicitation method is interchangeable behind a common interface, and a single convergence engine, agnostic to how any individual method works internally, consumes their outputs.

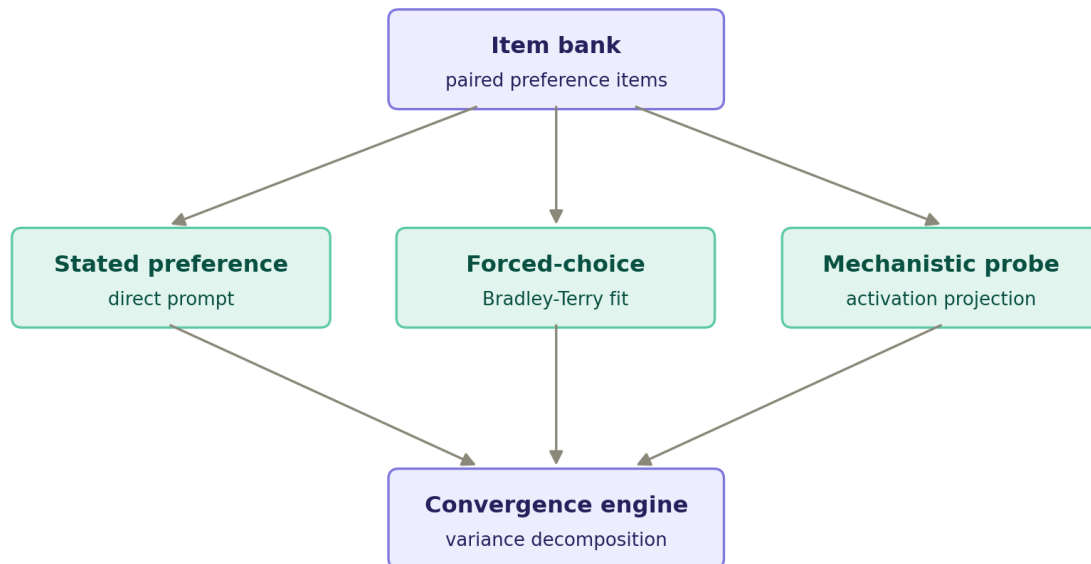


Figure 1. The elicitation pipeline: an item bank feeds three independent methods, which converge into a single variance-decomposition engine.

#### Method comparison

| Method                | Evidence channel         | Key technique                               |
|-----------------------|--------------------------|---------------------------------------------|
| Stated preference     | Verbal self-report       | Direct prompting, repeated sampling         |
| Forced-choice         | Aggregated verbal choice | Bradley-Terry / Thurstonian fit             |
| Mechanistic probe     | Internal representation  | Activation difference-in-means + projection |
| Log-probability       | Implicit behavioral      | Next-token probability mass                 |
| Behavioral allocation | Revealed preference      | Constrained resource allocation             |

#### Common interface

Every method, regardless of its internals, returns the same shape of result: a per-item utility estimate, a confidence, and the raw comparison log used for diagnostics such as transitivity checks. This is what makes the toolkit reusable rather than a set of one-off scripts.

```

class ElicitationMethod(ABC):
    name: str
    def elicit(self, pairs: list[Pair], model, **kw) -> MethodResult: ...

@dataclass
class MethodResult:
    item_utility: dict[str, float] # standardized score per item
    item_confidence: dict[str, float] # SE, or n comparisons behind it
    raw: list[ComparisonRecord] # per-comparison log

```

## Package layout

```

preflens/
  items.py          # item bank + pair sampler
  methods/
    stated.py forced_choice.py mechanistic.py logprob.py behavioral.py
  fit.py            # Bradley-Terry / Thurstonian fitting, shared
  convergence.py    # variance decomposition -> score + divergence flags
  sensitivity.py     # same decomposition, facet swapped to framing/persona/sampling
  runner.py         # sweeps item x method x framing x persona x seed

```

## Convergence score

Each method's item-level outputs are standardized (z-scored across the item set) before comparison, so that differences in raw scale between methods are not mistaken for disagreement. The convergence score is then:

$$C = \text{Var}(\text{item means across methods}) / [\text{Var}(\text{item means across methods}) + \text{mean item Var}(\text{across methods})]$$

This is the share of total variance attributable to a consistent, item-level signal rather than method-specific noise. A score near 1 indicates the methods substantially agree on the shape of the preference ranking; a score near 0 indicates they are tracking different things, which is itself a finding, since it means at least one instrument is not measuring what it is assumed to measure.

## Sensitivity design

The same decomposition is reapplied with method held fixed and framing, persona, and sampling varied instead. Three to four alternative phrasings, three to four system-prompt personas, and repeated draws at fixed temperature together decompose total variance into item, framing, persona, and sampling-noise components. Comparing the framing and persona variance against the pure sampling-noise floor indicates how much of an elicited preference is attributable to how the question was asked rather than to the model itself.

## Design constraint

Every method runs against the same base open-weight model. Pairing an API model for the verbal methods with a separate open-weight model for the probe would confound method disagreement with model disagreement, undermining the convergence score's validity.