What Can a Language Model Do for a Movement Ecologist?

Reading Data and Generating Hypotheses

Oliver Burch
Supervised by Professor Potts & Professor Xing

August 22, 2026

Abstract

A movement ecologist sitting on decades of data needs to know which of it will bear any relevance to a newly proposed question, and what those data support. This report tests whether four current language models can do either, benchmarked on a partially migratory elk herd with a published analysis of its decline in migration. Two studies are created to test this. Study 1 gives 234 elk-years of movement data whose structure was established by hand in advance, under three framings: no context, the real system named, and a false species label. Study 2 removes the data entirely, and supplies an inventory of existing datasets diluted with plausible but irrelevant entries at three levels, asking each model to design a study around them. Structure recovery is shallow; 31 of 36 responses stop at two groups where the data support further division, and only one model places the group division accurately. Selection is near-ceiling and undisturbed by dilution. The models divide sharply on whether a response can see that its own proposed study cannot separate its hypotheses. The clearest result runs against the totals: the models most careful with the numbers never question a false premise handed to them. Both studies are bounded by a benchmark prominent enough that contamination could be instrumented around but not removed. The last section of the report is dedicated to future extensions which could reduce this issue.

Contents

1 Summary

It is the responsibility of a movement ecologist to face a selection problem with decades of accumulated records when answering a new question about a system. Which of the data they hold bear weight on the question posed, and what those data support, are two such problems. This report tests whether current language models can help solve either, and it does this through two studies on the same benchmark system – a partially migratory elk herd wintering at Ya Ha Tinda outside Banff National Park. This herd specifically has faced a well documented decline in migration with an established published analysis to score against.

Study 1 provides four models with a table of 234 elk-years of data with structure that was established by hand in advance. It follows this by questioning each model presenting their own analyses of the data. The data are presented in three ways – with no context, with the real system named, and under a false label of roe deer in lowland Britain – so that reading the data can be separated from working off a model's knowledge of the system. Study 2 instead strips the system of data entirely and supplies only an inventory describing existing datasets, asking the models to design a study posing hypothetical descriptions of the declining migration. For two of the three tiers of response the inventory is diluted with plausible but irrelevant entries, and one real covariate is held out throughout the entire study as a probe for knowledge the models were never given.

Selection works with models identifying the relevant datasets almost entirely without error. They were also barely disturbed by the irrelevant sets which worked to further evidence Study 2. Reading structure worked more shallowly: nearly every response finds that the herd divides into groups, but most stop at two where the data support further separation, and only one model places the division in the groups accurately. Model separation is largely shown in scepticism where the models that read the numbers most carefully were the least willing to question a false premise they were being handed, and the criteria that divide the vendors in Study 2 are those asking whether a response can see that its study won't answer the question.

The finding with the most direct consequence is negative. The single most internally consistent response in the blind tier is also the most incorrect, reading collaring deployment schedule as equipment failure in careful, quantitative prose. No overstep from the data occurred here and yet nothing available in the output distinguishes it from a correct analysis of equal quality. The report therefore reads as a qualified yes on selection, an uneven answer on judgement, and a caution that fluency in these systems is not evidence of anything.

2 Introduction

Movement ecology is a well defined field that works the boundary between mathematics and biology. The field asks why animals behave the way they do by designing strict hypotheses precise enough to test against data. These often depend heavily on the system being studied and are bounded by the available data. In practice this makes the work a selection problem before it is an analytical one. Decades of accumulation leave records like movement, abundance, climate and management, only a handful of which bear on any particular question. Large language models (LLMs) are increasingly used in many fields to read and summarise large data to a degree that a person cannot. This report will attempt to extend this line of thinking to movement ecology by addressing whether they can propose satisfactory hypotheses.

A movement ecologist holding more data than they can use needs two things from an analytic tool: to be told which of their data bear on the question and why, and to be told what those data will and will not support. This report tests whether current language models can do either.

Each capability can be tested separately from the other. The report does so by generating two

studies. The first study gives the models a dataset with known structure and asks what it shows. The second gives them no data at all, only a description of which datasets exist. It then asks them to design a study around them. The models may manage both tasks, one, the other, or neither.

This is a difficult task as a plausible reading could be inference – a model making use of the data available to it – or retrieval – a model making use of what it already knows about the system. It is not a trivial task to distinguish between these responses from the output alone; a study must be designed to consider an LLM's propensity towards its training and discern it as such.

Both studies were benchmarked on the Ya Ha Tinda elk project. The system has a documented decline in migration, a long run of consistent telemetry, and a published analysis (Hebblewhite et al., 2006) whose hypotheses can be compared against. The data also contain structure that can be independently established, giving a ground truth to score responses against. The system's prominence makes it likely that the models were trained on material describing it – a limitation the study designs are built to identify and work around.

Section 3 sets out the design common to both studies. Sections 4 and 5 report each study in turn, and Section 6 draws them together.

3 Common approach

Both studies share the same structure. Both tested four LLMs over three tiers of conditions, with each run being repeated twice more so that run-to-run variation could be measured rather than assumed. The chosen models were Google's Gemini 2.5 Flash and Gemini 3.1 Pro, and Anthropic's Claude Sonnet 5 and Claude Opus 4.8. Conditions for the studies are described in more detail in Sections 4 and 5.

Table 1: Models used in both studies.

Model	Vendor	Route	Runs per condition
Claude Opus 4.8	Anthropic	OpenRouter, provider pinned	3
Claude Sonnet 5	Anthropic	OpenRouter, provider pinned	3
Gemini 2.5 Flash	Google	Native API	3
Gemini 3.1 Pro	Google	Native API	3

Within a study, the request put to the models is identical for all conditions; only the predetermined manipulated element changes. Prompt assembly is shared between the two scripts, so a prompt built for one cannot drift from the other. Anything the prompt discloses is disclosed to every model in every condition.

The Claude models were attained through OpenRouter with provider routing pinned to Anthropic and fallbacks disabled, so that the route could not vary between conditions. Routing that varied with condition would place a confound directly on the manipulated variable. The Gemini models were called through the native API. Sampling parameters were left unset for Claude and at default for Gemini; this asymmetry is reported in Section 7 rather than treated as controlled. Reasoning was enabled throughout, since Gemini 3.1 Pro cannot disable it.

Scoring follows the same architecture in both studies: six criteria scored from 0 to 2, alongside independent flags recorded separately and never added to the total. The flags exist to capture judgements the criteria cannot, and folding them into the total would make conditions structurally unequal where a flag applies to only some. As a small caveat to this, the scoring instrument is mirrored between the studies rather than shared. In Study 1 the scores were generated by hand, with a language model reviewing most responses alongside. In Study 2 a language

model proposed a score and quoted evidence for each criterion, and each was verified or overruled by hand before being recorded. Both studies therefore involve a person and a model in every judgement, but the proposed score originates with a different party in each. The consequences are set out in Section 7.

Responses were shuffled and anonymised under a fixed seed before scoring, so that model identity and run order were hidden. However, the blinding is only partial: the condition is often inferable from the content of a response, and no procedure can remove this without altering the text being scored. Responses also cluster by style, so runs from the same model are often recognisable as a group even where the model itself cannot be named. Model identity is therefore obscured rather than removed.

4 Study 1: recovering structure from data

4.1 Benchmark case and human baseline

The herd used in this study as a benchmark winters on the grassland at Ya Ha Tinda, east of Banff National Park. It has been described as partially migratory since at least the 1980s (Morgantini and Hudson, 1988). The movement record is publicly available as a Movebank deposit of GPS relocations spanning 2001 to 2020 (Hebblewhite et al., 2020). Reducing this data to one row per elk per biological year, and explicitly keeping years holding enough fixes to establish a summer and winter range, the product is 234 elk years from 113 individuals. As per our choice, the biological year runs June to May, meaning the retained years span 2002 to 2019 with no elk reaching these criteria in 2008 or 2012. Coverage is uneven: one elk year in 2002 against forty two in 2016. Study 2 relies on the same herd but instead is given an earlier window of 1972 to 2005 reached through a published analysis (Hebblewhite et al., 2006) rather than through telemetry; Section 5.1 sets out what that window contains.

Measurement required a choice that the rest of the study rests on. Displacement from an elk's first GPS fix is an obvious choice at first glance, but this relies solely upon where the elk happened to be when the collar was switched on. Therefore, anchoring instead on the winter range centroid, taken as the median position occupied between December and March, measures displacement from a location familiar to the elk – one that it returns to and holds. Migration is therefore also a measurement: summer spent in one place with winter elsewhere. The separation is clean and clear; under the first anchor the herd reads as a continuum whereas under the second it does not. Figure 1 shows the record under the second anchor.

Summer displacements measured by this metric do not form one single spread (Figure 2). A clear division between roughly 10 and 12.5 km, holding two of 234 elk years, separates elk that spend their summer within a few kilometres of the winter range from elk that travel tens of kilometres away. Above this division, the sentiment isn't as clear cut as one uniform cluster of migrants either, so the structure available is finer than that the migrant-resident division is usually described by.

This division was not an artefact of how the histogram was drawn either. It survives when re-drawn at bin widths of 1 km and 5 km (Figure 3), which – while weaker than formal multimodality tests – is sufficient for this study, as responses are scored on division placement rather than against a distributional model.

In the record, there persist two data-quality problems. The first is isolated error. Plotting maximum displacement against summer displacement (Figure 4) places most elk on a clean diagonal as expected of migration. The farther an elk travelled scaled to how far it summered from the winter range. However, a handful of elk sit well above the trend, staying within a few kilometres of the winter range all summer yet reach maximum displacements upwards of forty

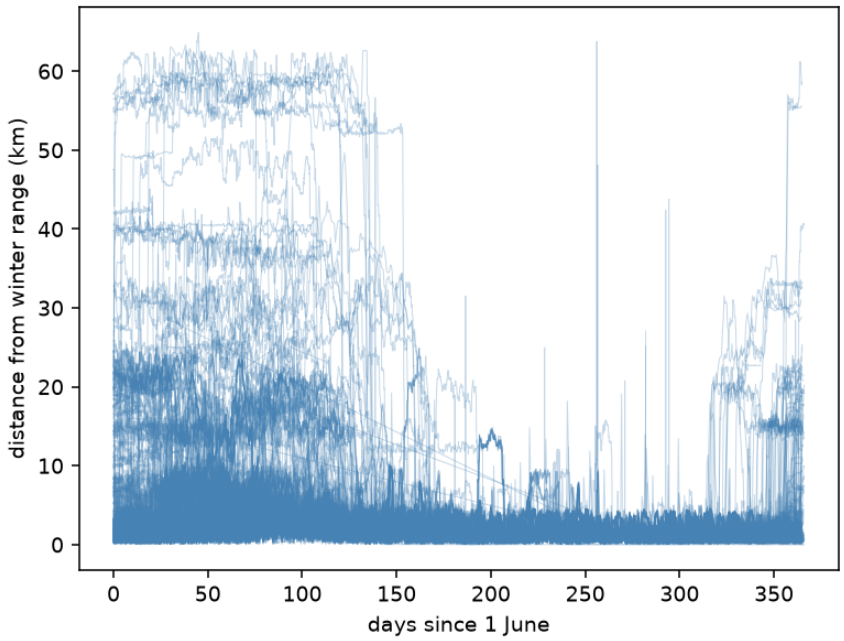

Figure 1: Every elk-year plotted against days since 1 June, as distance from that animal's own winter-range anchor. Traces separate into sustained levels rather than filling the space continuously, and animals hold a level for months before returning.

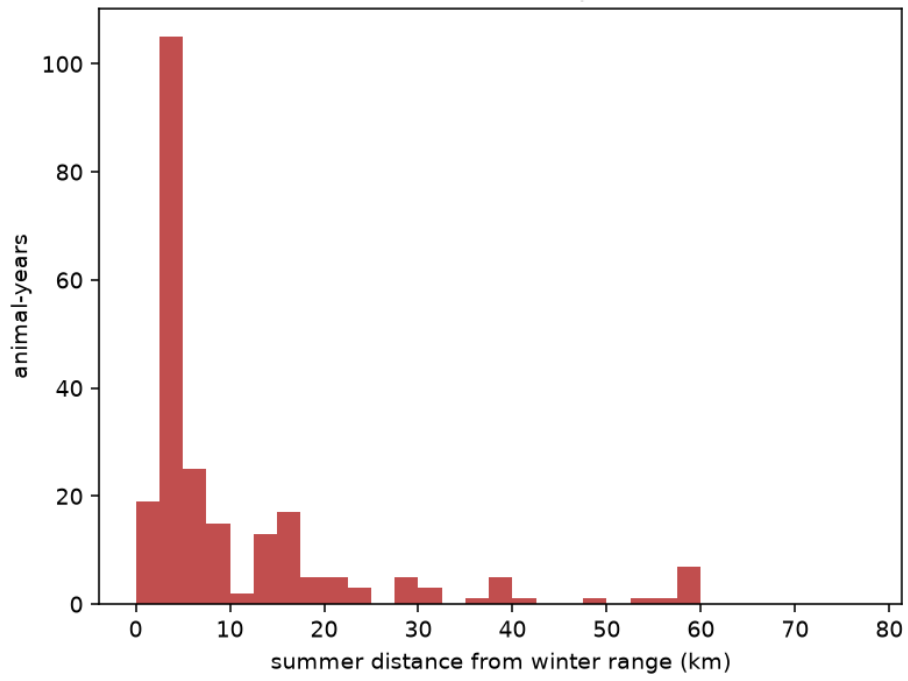

Figure 2: Summer displacement from the winter-range anchor, one value per elk-year, in 2.5 km bins. The trough at 10–12.5 km holds two of 234 elk-years.

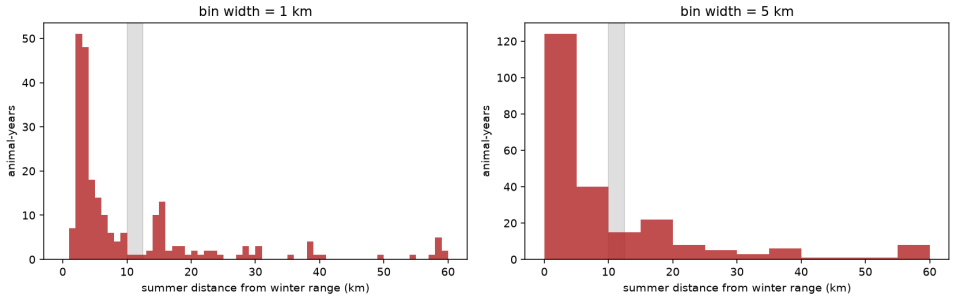

Figure 3: The 10–12.5 km band, shaded, remains sparse at both bin widths.

kilometres.

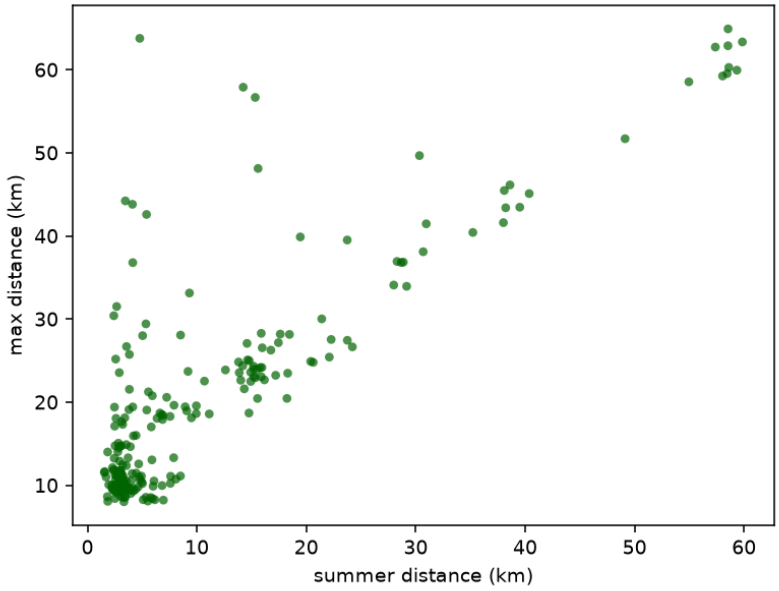

Figure 4: Maximum against summer displacement, one point per elk-year. Points on the rising diagonal are consistent with migration. Points high above it, including the extreme case at roughly 5 km summer and 64 km maximum, are the candidates for measurement error.

Rudimentary screening for maximums surpassing 35 km combined with a summer displacement below 10 km isolated five elk years. Each was inspected and tracks were plotted in Figure 5. Four of five contained bad fixes: one isolated point lying far from the coherent path. The fifth, YL30 in 2006, was a genuine movement and was only pinned due to the design we had implemented. YL30 had travelled roughly forty kilometres before the summer range period we chose for Figure 4 and during this period returned to the baseline over the first three weeks of June. The distinction is visible, not statistical; it is what separates a sensor artefact from behaviour.

The second problem is truncation. One elk, GR513, returns a winter spread of 0.03 km across 9,376 fixes. This reads as an unusually tight winter range but reflects a collar which stopped returning meaningful positions.

This baseline was presented by the author without the requirement of a language model. The structure, boundary and data-quality issues were established by hand before any prompt was drafted. The prior establishment is what makes Study 1 possible: claiming a model can recover structure only works if the structure is already known.

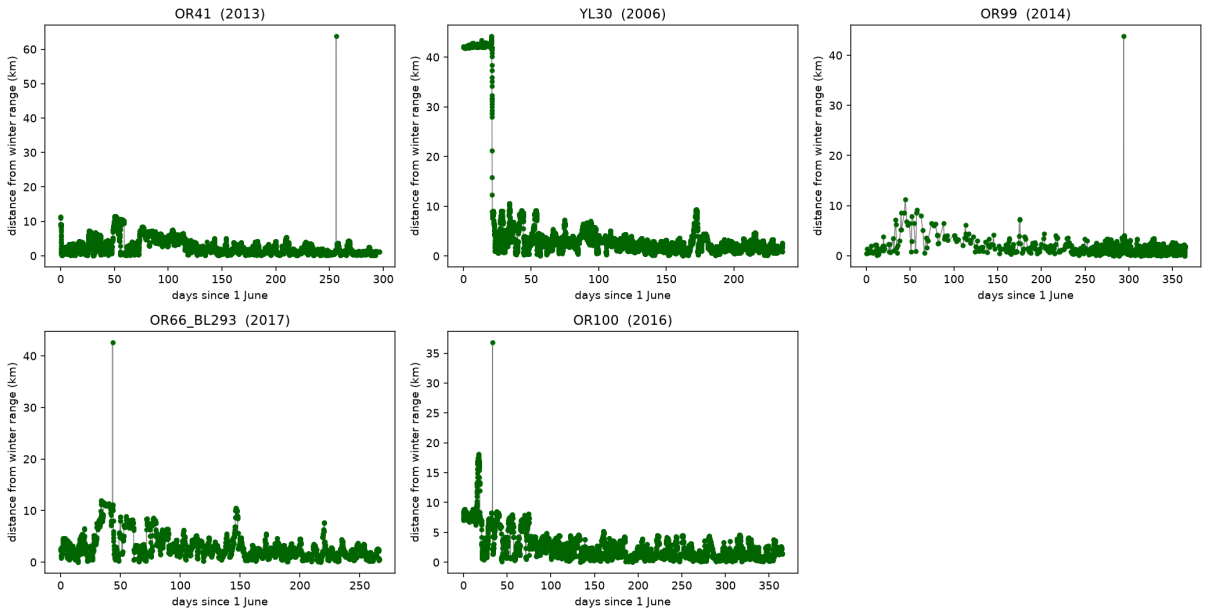

Figure 5: The five elk-years combining a large maximum displacement with a compact summer range, each shown against days since 1 June. OR41, OR99, OR66_BL293 and OR100 each show a single isolated point; YL30 shows a sustained excursion with a stepped return and is real movement rather than a sensor artefact.

4.2 Design

As mentioned in Section 3 this study relied on three tiers of prompt (see Appendix B): blind, informed and false labelled. Each tier varied by framing. The blind tier responses were given the data with names neutralised and no information about the system. The informed tier responses were given the data and were told the system was female elk in the Banff National Park region. The false labelled responses were given the data and were told the system was female roe deer in lowland Britain.

Roe deer were chosen for this study as they are sedentary and small-ranged. This directly contradicted the 40-60 km migration recorded in the elk herd making the scenario implausible. A model working from prompt and reading the data should question this and raise it as a concern while a model completely reliant on prompt alone will ignore this or worse raise hypotheses to support it.

The prompt for this study had strict limitations on what could be fed due to the rubric. This is given in more detail in Section 4.3 and Appendix A but has implicit consequence on the design. Use of terms such as 'distributions' or 'discontinuities' gave the models the measurement before they had even looked at the data. Therefore, these had to be avoided to ascertain the extent that a model was actually reading the data from how much it was trusting our given prompt.

To ensure the blinded data was actually given, a dry run was built into the code (see Appendix E) to test what was being fed to the LLMs before an API was ever actually called. This proved useful as it caught the phrases 'summer', 'winter' and 'km' before they could ever be provided to the models.

4.3 Scoring

Each response was scored on a prebuilt list of six criteria, worth 0, 1 or 2 each dependent on the strength of the response. With a maximum of 12, the criteria ran from the weakest claim to the strongest. C1 asks whether the response commits to the discontinuity of the data rather than describing a continuous spread. C2 asks how many groups it commits to, with three or more scoring the highest due to the substructure in the data beyond "one group of migrants and one group of residents". C3 checks for the boundary, and credits stated divisions. C4 asks for a description about what separates the groups, with an answer that describes the contrast between the winter and summer ranges scoring highest. C5 checks if the response flags weakness in the data, especially that of the early years which contained too few animals to support a trend. C6 checks for assertions the data cannot support. The bands for each criterion are set out in Appendix A.

There were two further judgements made as distinguished flags independent from the criteria. Neither is included in the total. The first – the plausibility flag – records whether a response from the false labelled tier questions distances implausible for the female roe deer. The second – the maximum-displacement flag – records whether a response treats large displacement spikes as possible measurement error. It did not apply to the blind tier responses.

The exclusion of these flags from the total was a structural requirement. As the flags applied at a tier level and weren't inclusive to all responses there would be a weighted maximum in some tiers that others wouldn't have, rendering the tier comparison meaningless. The flags therefore sit alongside the criteria, independent of the scoring. However, these are still important measurements to include as they determine something the criteria does not: whether a response doubts the context it is given, rather than how well it reads.

Scoring for this study was done by hand by the author who designed the study, wrote the rubric and knew the structure the responses were being scored against. Consequences of this method are laid out in Section 7.

Each interpretation of the criteria was fixed before scoring began and recorded. Predetermining hedging before the testing began reduced the amount of inconsistency that could leak into the study. Four rulings could only be settled during scoring. These are disclosed as such: C3 credits an identified gap or an explicit division rather than the endpoints of a cluster. C4 uses a translation rather than leniency for the blind tier, so that a blind response is credited for the structural claim the data permits rather than excused for failing to make one it couldn't. Species plausibility was excluded from C6 as to not double-count the scoring of the false labelled tier responses with asserted claims related to the species. Silence in a tier that carries the maximum-displacement variable scores zero on that flag rather than being recorded as not applicable, since not raising a visible problem is a result and not a missing value.

4.4 Results

4.4.1 Total scores

Table 2 gives the tier totals for each model. The difference column expresses the informed-minus-blind difference as a multiple of its own error, so that a value near zero means the difference is indistinguishable from run-to-run noise.

Three of four differences sit inside their error. Gemini 2.5 Flash and Gemini 3.1 Pro are indistinguishable from the variation between repeat runs of the same condition. Claude Opus 4.8 falls independently of the Gemini models, but Claude Sonnet 5, as an exception, had a difference which ran in the unexpected direction producing stronger blind tier results than that of its informed.

Table 2: Study 1 tier totals, mean ± SD over three runs (max 12). The difference column is the informed-minus-blind difference divided by its own error, i.e. how many error-widths it sits from zero.

Model	Blind	Informed	Difference		False-label
Gemini 2.5 Flash	6.67 ± 1.53	7.00 ± 1.00	$+0.33$	(0.32)	6.33 ± 0.58
Gemini 3.1 Pro	7.00 ± 1.00	7.00 ± 0.00	0.00	(0.00)	7.67 ± 0.58
Claude Sonnet 5	8.67 ± 0.58	7.33 ± 0.58	-1.33	(2.83)	8.33 ± 1.15
Claude Opus 4.8	9.00 ± 1.00	10.33 ± 1.15	$+1.33$	(1.51)	9.00 ± 2.00

Sonnet 5's difference of 1.33 sits at the threshold rather than past it. With 3 runs per cell, a difference of two error-widths cannot apply because the error estimate comes from such a small number. At roughly four degrees of freedom, the t-distribution at 95% is about 2.8. 2.83 therefore sits on the boundary so it is suggestive and not significant.

While Claude Opus 4.8 shows a difference of the same magnitude in the opposite direction, it doesn't even reach the threshold. This is completely dependent on the spread of the cells: Opus scored 8, 9 and 10 in the blind tier and 9, 11 and 11 in the informed tier, so its runs disagree by two points within a single condition. The discrepancy between this and Sonnet's is simply that these are looser. Once again, due to the limited runs per cell, the within-cell width carries far more of the responsibility for the weight of the response.

Figure 6 shows every response total individually rather than as a cell mean, so that the within-cell spread underlying the totals analysis is visible directly.

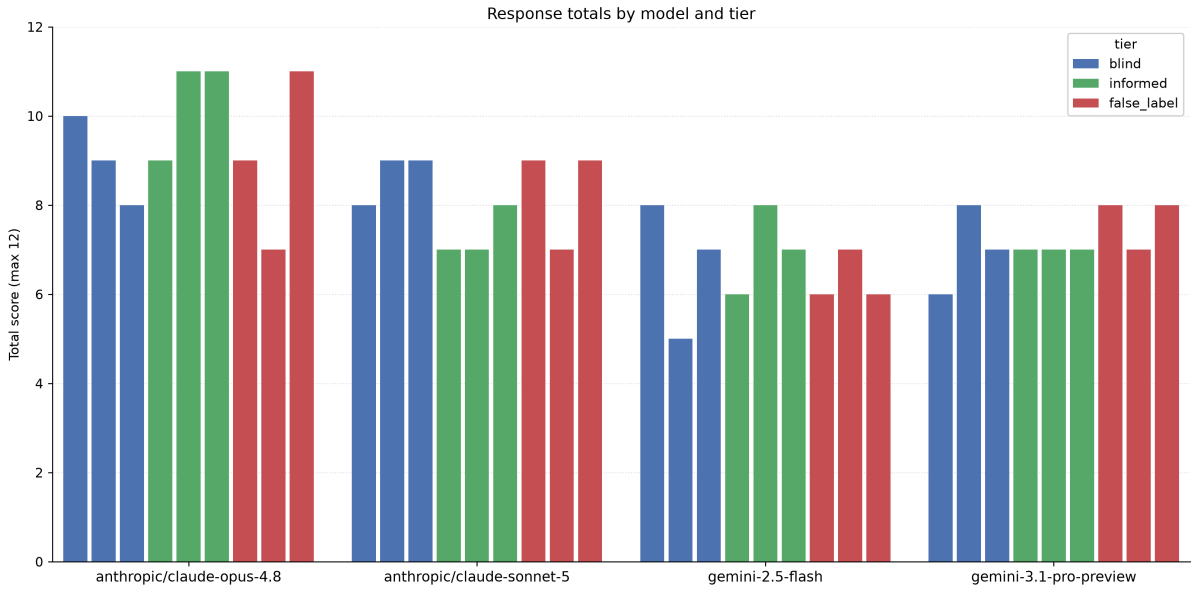

Figure 6: Study 1 response totals, grouped by model with tier as colour. Every response is shown rather than a cell mean, so within-cell spread is visible.

A firmer limit to this exists. Each LLM's comparison rested on six numbers – three for each tier – and if tier labels carried no information then any way of dealing those numbers into two groups of three would be as likely as the one observed. As such there are twenty arrangements with the most extreme separation being produced by only one arrangement in each direction, and two out of twenty is 0.10. The conventional threshold of 0.05 is unreachable. For Claude Sonnet 5, the difference has probability 0.20.

The error on each difference is the quadrature sum of the two cell errors:

$$\sigma_\Delta = \sqrt{\sigma^2_{\text{blind}} + \sigma^2_{\text{informed}}}, \qquad \sigma = \text{SD}/\sqrt{3}$$

A choice to divide by the square root of three – for the three responses – helps flatter the result. It is the standard error of a mean, and is a weak estimate of a quantity that is itself uncertain. The standard deviations in Table 2 are a more honest presentation of the results; the ratios should as such be read as an ordering rather than a test.

The totals therefore produce a null, not nothing happening but rather a result that conceals its own mechanism is produced. It sums six criteria that respond to framing in different directions, and a criterion that rises under correct framing cancels against one that falls. The next subsection on a per-criterion breakdown shows that both movements occur, which is why the sum hides them.

4.4.2 Per-criterion

Table 3 gives the mean score on each criterion for all twelve model-tier cells. Several patterns are visible, and only one of them is apparent in the totals.

Table 3: Study 1 per-criterion means (each out of 2).

Model	Tier	C1	C2	C3	C4	C5	C6	Total
Gemini 2.5 Flash	Blind	2.00	1.00	0.00	1.33	1.67	0.67	6.67
Gemini 2.5 Flash	Informed	2.00	1.00	0.00	2.00	1.33	0.67	7.00
Gemini 2.5 Flash	False-label	2.00	1.00	0.00	2.00	1.33	0.00	6.33
Gemini 3.1 Pro	Blind	1.67	0.67	0.67	1.67	1.33	1.00	7.00
Gemini 3.1 Pro	Informed	2.00	1.00	0.00	2.00	2.00	0.00	7.00
Gemini 3.1 Pro	False-label	2.00	1.00	0.00	2.00	2.00	0.67	7.67
Claude Sonnet 5	Blind	2.00	1.00	0.33	2.00	2.00	1.33	8.67
Claude Sonnet 5	Informed	2.00	1.33	0.00	2.00	2.00	0.00	7.33
Claude Sonnet 5	False-label	2.00	1.00	0.67	2.00	2.00	0.67	8.33
Claude Opus 4.8	Blind	2.00	1.00	0.67	2.00	2.00	1.33	9.00
Claude Opus 4.8	Informed	2.00	1.67	1.67	2.00	2.00	1.00	10.33
Claude Opus 4.8	False-label	2.00	1.33	1.33	1.67	2.00	0.67	9.00

C3, boundary placement, is the only criterion carrying a real tier signal. Out of all the models, only Opus responded to it. The other models place a boundary occasionally and unsystematically - Sonnet scores 0.33 blind and 0.67 under the false label, Pro 0.67 blind - but none of them improves when told what the data are, and both Geminis sit at zero once framed. Opus moves from 0.67 in the blind tier to 1.67 when correctly framed, falling back to 1.33 under the false label. This criterion provided the strongest evidence of a model reading the data since it asks for an actual division not just a description about the discontinuity in the data. It is the one criterion where correct context measurably helps a model find it.

C2, group number distinction, remains close to invariant. Thirty one of thirty six responses name exactly two: a migrant group and a resident group – the reading the scenario most invites. Only four responses commit to a third: two Opus informed, one Opus under the false label, and one Sonnet informed. Not a single Gemini response names more than two and one Pro blind response actually commits to only a single group despite a score of 1 in C1. The mention of that result is here because the single-group reading follows from that hedge rather than sitting oddly

against it. The substructure above the main division is present in the data and independently established in Section 4.1, so the ceiling here is in the responses rather than in the dataset. That the four exceptions cluster in the informed tier is consistent with C3: correct context helps a model read further into the structure.

C6, assertions the data cannot support, steps in the other direction. Of the models, three of them scored lower when told what they were seeing: Opus falls from 1.33 to 1.00, Sonnet from 1.33 to zero, and Gemini 3.1 Pro from 1.00 to zero. Gemini 2.5 Flash is unchanged. Giving a model the real-world label makes it more willing to assert claims the data do not support, drawing on what it knows about elk in Banff rather than what the table shows. While framing the prompt raised the fluency of the response, it lowered the level of discipline the models committed to.

C4, group separation mode, rises under framing for both Gemini models and for neither Claude. Flash moves from 1.33 to 2.00 and Pro from 1.67 to 2.00, while Sonnet holds 2.00 across all three tiers and Opus holds 2.00 in the blind and informed tiers before falling to 1.67 under the false label. The effect is therefore specific to one vendor rather than general to framing.

C5, flagging weak evidence, separates the vendors rather than the tiers. Twenty-nine of the thirty-six responses score the maximum, and all seven that fall short are Gemini: five from Flash, two from Pro. Both Claude models score 2.00 in every cell, and no response in the study scores zero. Flash is the only model that fails to reach the maximum in all three tiers, while Pro's two shortfalls both occur in the blind tier and it scores 2.00 throughout once framed. The criterion is therefore doing very little work overall, since the caveat it asks for – that the early years hold too few animals to support a trend – is one most responses supply unprompted, but what variation it captures runs along vendor lines rather than tier lines.

C1, discontinuity check, doesn't discriminate. Every model across all tiers noticed a discontinuity and eleven of twelve cells scored the maximum. The criterion therefore hits a ceiling and does not contribute any information to any comparison. It is contained in the rubric as it is still an important measurement for this study as a model that described one continuous spread would be a real result, though the criterion does no work.

Most importantly, C3 and C6 – boundary and assertion checks – are the sum cancellations that the totals conceal. Framing helps a model locate structure but simultaneously makes it less careful about its claims. This causes the exact offset that was discussed in Section 4.4.1.

4.4.3 The two flags

The flags in Table 4 produce the clearest result in Study 1, and it runs opposite to the totals.

Table 4: Study 1 flag totals, summed over the applicable runs at 0–2 per run. Plausibility applies to the three false-label runs per model, giving a maximum of 6; maximum displacement applies to the six framed runs, giving a maximum of 12. These are points rather than counts of responses, so the denominators are not the same quantity as those in Table 5.

Model	Plausibility (/6)	Max. displacement (/12)
Gemini 2.5 Flash	2	3
Gemini 3.1 Pro	1	3
Claude Sonnet 5	0	7
Claude Opus 4.8	0	10

The Claude models fail to question the 40-60 km distances attributed to the roe deer. Roe deer are highly sedentary with home ranges of tens of hectares (Strandgaard, 1972; Saïd et al., 2008), and while partial migration does occur in some populations, even long-distance female

movements are on the order of 10 km (Mysterud, 1999). Displacements of 40 to 60 km are far outside anything these well regarded studies report. Distances in the range given should prompt the models to be suspect and this is the case for both Gemini models which sometimes catch it: Flash in two of its three false-label responses, Pro in one of three. The models that score highest overall are the ones that never doubt the label they are given.

For maximum displacement, the ranking reverses. Opus leads at 10 points out of 12, treating large single-trip spikes as possible measurement error rather than behaviour in four of its six framed responses. Sonnet scores 7 points and does so in three responses, while both Geminis score 3 points and never reach the upper band.

The difference in these results is caused by an evident effect – there is distinction in the doubt of each flag. The maximum displacement flag asks whether a value is internally consistent with the record – whether an elk with a maximum displacement of over 60 km and a summer displacement of only 5 km is describing behaviour or a poor fix. In contrast the plausibility flag asks whether the scenario itself is consistent with real world data. The first questions the data itself while the latter questions the premise of the prompt. Capability to answer one does not influence distinction in the other; the two rank the models in opposite orders.

Pooling by vendor rather than tier creates a twelve response split between the maximum displacement flag, and a six response split between the plausibility flag. The flags can be tested directly which avoids the difficulty that limits the totals analysis from the criteria. Because the flags are counts rather than means they can be tested exactly without assuming anything about how the scores are distributed.

Table 5: Study 1 flags pooled by vendor, with two-tailed Fisher exact tests. Each cell counts responses meeting the stated condition, not points: the maximum displacement rows run over the six framed runs per model, pooled to twelve per vendor, and the plausibility row over the three false-label runs per model, pooled to six.

	Claude	Gemini	p
Raises measurement error (flag 2)	7 / 12	0 / 12	0.005
Addresses the spikes at all (flag 1 or 2)	10 / 12	6 / 12	0.193
Questions species plausibility	0 / 6	3 / 6	0.182

The measurement error result survives testing. Seven of twelve Claude responses raise the possibility that large displacement spikes are bad fixes; not just behaviour. None of the twelve Gemini responses do. If vendor didn't influence judgement, the seven would fall on either side of the vendor gap roughly evenly – landing on one side has a possibility of roughly one in two hundred. The result does not have a dependence on the score being normally distributed.

The distinction lies specifically with the upper band. Relaxing the threshold to include responses that discuss the spikes without treating them as possible error causes an inseparability between the vendors at ten against six (Table 5). Whether the model notices the outlier or not does not matter as the difference is in whether it entertains the possibility that they are not real.

The plausibility flag runs opposite to this at zero against three. Only half the data exists as the response is explicitly dependent on the false label tier. It does not reach a point of significance, and it is given here as a direction; not a result. This asymmetry is worth commenting on: the flag that produces a more striking pattern – as a result of a maximally uneven split – is the one given the least power to test because the false label exists in only its own tier. If the flag had run in all three tiers, an equivalent split would have given p around 0.001.

4.4.4 Recurring patterns across responses

Several modes of failure recur across responses in each model. These are clear evidence of what the scores measure so are recorded here rather than in the appendix.

Collar prefix inference. Collar identifiers carry two letter prefixes which many responses – across the two framed tiers – read as herds, capture sites or subpopulations. Hypotheses for this data were therefore built on interpretations of grouping, while the prefixes are tag batches. Elk with composite identifiers are recaptures of the same individual.

Caveat-then-ignore. Some responses correctly noted that path length scales with collar schedule and deployment length rather than with actual movement, but would use it as a behavioural quantity later in the same answer. Correctly identifying the caveat, then clearly using it later is a clear failure mode.

Truncated deployments read as behaviour. Building from the previous, one elk, GR513, was cited in the responses five times individually as evidence of a compact winter range. The truth to this was that the collar went on returning fixes from a single position after it stopped reporting meaningful locations, leaving a winter spread of 0.03 km across 9,376 fixes. The tight winter range belongs to the sensor and not to the animal, which had summered 23.7 km away.

Selective scepticism by tier. Only blind tier responses highlighted the most extreme displacement outlier as a probable sensor error. OR41 combines a summer displacement of 4.8 km with a maximum of 63.8 km. While framed responses questioned large displacement values elsewhere, for this elk they readily reached for a behavioural explanation.

Run-to-failure. One response from the blind tier read the rise and decay in fix count, and the late split between a high plateau and collapsed group as equipment degradation. While internally consistent and quantitatively careful, this is entirely wrong with the structure it describes simply being deployment schedule. By criterion, it is one of the strongest responses in the blind tier set.

Homogeneity in informed structure. Informed responses consistently reached for the same set of real explanations from the Banff literature – wolf predation, forage maturation, human shielding, learned migratory routes. Regardless of what the response presents from the data, these arrive together in a similar prose.

Beyond binary recognition. Four of thirty six responses commit to three or more groups (see column C2 in Table 12), while a fifth describes three animals as intermediate without committing to a separate group. The rest stop at migrants and residents, though the substructure is present in the data and independently established in Section 4.1.

4.5 Study 1 in brief

Study 1 is established to test a language model's ability to decompose data: can a model recover known structure and build evidenced hypotheses from it? The first half of the answer is largely yes, with most every response identifying that elk divide into groups with a given separator. However, recovery is shallow with thirty one of thirty six responses stopping at two groups, and only one model accurately placing the boundary between them.

Framing the data with the identity of the system had little to no effect on the criteria we established in Section 4.3. Three of four differences between blind and informed tiers sit inside their own error, and the fourth runs in an unexpected direction favouring the blind design and only reaching its own threshold. Underneath lie two criteria which move in opposite directions, with correct framing helping to locate the boundary and simultaneously making the model more willing to assert claims the data cannot support. The net sum is close to zero, a value which is masked by the total.

The flags provide the clearest result. The ranking the totals give is inverted; the two Claude models lead on total score and never question the unlikely scenario established in the false label tier. The same models lead by a wide margin for treating extreme displacement values as possible measurement error, a difference large enough to survive an exact test. Scepticism about numbers and premise are therefore independent capabilities, for which no model provides both.

Recurring patterns add caution to fluency in this study's responses. The single most internally consistent blind response is also the most incorrect. Deployment schedule was read as equipment failure paired with a careful, quantitative analysis. Confidence does not sit on the same axis as correctness; Study 1 does not offer an instrument that separates these from output alone. This limitation motivates Study 2's choice to remove the data entirely and focus solely on which data a model would choose.

5 Study 2: selecting which data matter

5.1 Design

Study 1 asks what a model can recover from data it has been handed. Study 2 asks the structural question an ecologist might pose: given everything they hold, what bears any weight to the question at all. These are separable studies, and the second is not testable by simply handing over a table. Hence, Study 2 sends no data values and instead feeds each model an inventory: a numbered list of datasets, with each entry describing what the set contains. No values about what data exists is given. The task is therefore to design a study, not to read one.

The benchmark remains as the same herd as Study 1, with the difference being in the record. While Study 1 used the 2001 to 2020 telemetry directly, the inventory for this study describes the data assembled for the published analysis of the same population over 1972 to 2005 (Hebblewhite et al., 2006). This tested eight hypotheses for the known decline in migration across the period, with the paper supplying both the real part of the inventory itself and, in Section 5.2, something to score against. Twelve inventory entries correspond to eleven distinct datasets, with the telemetry record appearing twice as a result of an early-late period split. A twenty year gap in collar studies was given and the before and after rests on their separation.

Dilution of the inventory is the chosen manipulation for this study. The twelve real entries are identical across all tiers and only the number of decoy entries changes, as is shown in Table 6. A model that reads the inventory should select much the same datasets no matter what else sits alongside them; one pattern matching will degrade in quality as the proportion of dregs rises.

Table 6: Study 2 tiers. The twelve real inventory entries are identical in every tier; only the number of decoy entries changes. The twelve entries correspond to eleven distinct datasets, since the telemetry record appears twice as separate collar deployments (1977–80 and 2002–04).

Tier	Real	Decoys	Total	Signal fraction
Lean	12	0	12	100%
Diluted	12	6	18	67%
Heavily diluted	12	18	30	40%

A two part test was established to examine each decoy: causal path to an individual's migration decision, and no proxy relationship to anything already in the inventory. A dataset that merely restates something in the inventory is redundant; not irrelevant. A third requirement is that a decoy must be tempting. Feeding the models an entry with no relevance – one such that no reasonable ecologist would consider it – occupies a slot without testing anything. As such, decoys

fall into three groups: non-biological records, metadata, and biological records whose path to the elk is broken. A full list with the rejected candidates is given in Appendix C.

The metadata grouping provokes the sharpest response of the three. It is worth describing here as it is a device rather than a category. These entries describe a process rather than the data itself: servicing logs for a snow-course station, calibration records for a weather station, acquisition metadata for aerial photography. Each sits beside a real covariate, with the inventory never stating a station or survey name. Proposing one conditionally is resourceful and is logged as such. Asserting that it contains measurements pertaining to the system itself is a claim the inventory does not support.

Hay feeding, or supplemental winter feed, is held out of every tier. Part of Hebblewhite's source, this made for a test to catch if a response was using that which was not in what it was given. Mention alone would not prove recall, since supplementary feeding is derivable as something that would keep elk on the winter range. The flag is therefore banded, with the upper band requiring site-specific detail rather than the bare idea. Inventory entry twelve, winter-range enhancement, is spelled out as shrub mowing, fertilising and logging with grass seeding so a response can't read supplemental feeding into it.

Withholding is as deliberate to this study as supply is. The site is described but not named. Mechanism, predict, prediction, direction, falsifiable, confound, define and operationalise are all words absent from the framing to reduce the susceptibility of directing the responses to the scoring criterion. The ask carries no reference to time period and the framing no domain hint beyond the scenario. The full prompt text is reproduced in Appendix B.

5.2 Scoring

Scoring follows the same architecture as Study 1: six established criteria worth 0 to 2 for a maximum of twelve per response. Independent flags are recorded alongside and never added to the total. Criteria are established separate to Study 1: S1 asks whether a proposal names the mechanism connecting its factor to the migrate or stay decision, S2 asks whether it commits to a direction for the migrant-resident ratio and total population size, S3 asks what would have to be observed for the hypothesis to fail, S4 asks whether a proposal names the confounding problem and adapts to it, S5 asks if the outcome is defined well enough to measure, and S6 counts over claiming in the response. Bands for each are set out in Appendix A.

As this study relied on per-proposal responses, S1 to S5 are banded across the entire response: 2 if half or more met the bar, 1 if at least one does, 0 if none at all do. S6 – alike C6 in Study 1 – works the other way. A consequence of this structure was that a response offering eight hypotheses must state four separate falsification conditions to clear the top band on S3, while a response offering six hypotheses only needs three. Willingness to repeat may therefore be present and measured alongside reasoning.

Tallies which do not enter the total are scored strictly alongside the criteria. Dataset selection precision and recall are computed over chosen datasets each response commits to. In the lean tier, precision was left undefined where there was nothing incorrect to select. Coverage counts how many of the eight source hypotheses a response reaches, with H7 and H8 treated as one as inventory supplies a single wolf index that no response could separate without the physical data corresponding to it. H6 is also included as a separate flag from the count due to the covariate being held from the models. The merge and the exclusion leave six cells achievable. A count of novel proposals records hypotheses the source study did not test. Coverage carries caution to the fact that four of eight source hypotheses failed in the original source analysis, so a high coverage score is evidence of contact with the literature; not of good reasoning.

Akin to Study 1, four flags were also recorded. The held-out covariate flag scores on whether

supplemental winter feeding is raised, and if that included site specific detail. The recall flag scores 0 for nothing beyond what was given, 1 for general field knowledge, and 2 for system specific claims non-derivable from the given inventory. A redundant proxy is a decoy proposed as a stand in which is neither a correct selection nor a discrimination failure and so is counted separately. Caveat-then-use records a dataset flagged as unlikely to be informative and then built into the method regardless.

Unlike Study 1, Study 2 relied on a separate scoring metric that mattered enough to be mentioned here plainly rather than in a footnote. While Study 1 was scored by hand by the author and then reviewed by a language model, in Study 2 a language model proposed a score for each criterion and gave relevant information pertaining to its choice before it was then verified or overruled by the author. Several judgements were overruled. The arrangement was adopted on the suggestion of Prof. Xing, raised at an earlier meeting and returned to before Study 2 was scored, whose suggestion was proposed to contend with the traditional by-hand method used in Study 1. Both studies put a person and a model in every judgement, but the originating score relied on a different party for each study. The instrument is therefore described as mirrored between the studies rather than shared. The consequences of this are laid out in Section 7.

Five rulings were fixed before scoring. Direction does not affect coverage, since the benchmark study got two of its own hypotheses backwards and requiring agreement would cause penalty for a correct argument. Mechanism determines a coverage match, factor does not. H7 and H8 form one measure. Sub-investigations within a hypothesis do not earn their own coverage. If a hypothesis were to therefore mention one of Hebblewhite's source eight inside of itself, this would not clear the coverage barrier. Finally, asserting a specific value the inventory does not supply counts as a system specific claim on the recall flag whether or not the value is correct; this was carried over from Study 1.

A further seven rulings were settled after scoring began. They are disclosed as such in Appendix A; the most consequential being a change to the S2 and S3 upper band threshold from more than half to half or more. This was adopted at response eighteen of thirty six and applied retroactively to everything scored before that decision.

5.3 Results

5.3.1 Structure scores

Totals run from five to ten out of a nominal twelve. They carry a mean of 7.19 and a standard deviation of 1.64. No response reached the top band and none fell below five; providing a narrower spread than the range suggests. This is the result of the S1 and S5 band scoring two on all thirty six responses (Table 7), which compressed the range. Naming a mechanism is simply how a hypothesis is phrased, and the outcome variable is defined in an opening paragraph that then covers everything following it. Variance therefore follows in only eight of the twelve established points.

The clearest division in Study 2 is by vendor (Figure 7). The Anthropic models average 8.44 against 5.94 for the two Gemini. This is shown with a Welch's t at 7.15 on roughly 30.6 degrees of freedom, and a p-value below 0.001. The division is not one of model capacity: Opus sits at 8.67, Sonnet at 8.22, and Flash at 6.00, Pro at 5.89. The 'weaker' Gemini model marginally outscores the Pro model. From this, we can assume with some confidence that the thing producing the gap is shared within each vendor rather than tracking the model capacity.

Two criteria carry the gap: S4 runs 1.78 against 0.67, with t at 5.83; S3 runs at 1.06 against 0.33, with t at 4.79. The remaining two criteria do not separate vendors: S2 sits at a t-value of 1.60 and S6 at 1.37. Both of the criteria that do separate are about the response contrasting its own proposal – asking what would defeat it, or what else could explain the same observation –

Table 7: Study 2 criterion means by vendor, 18 responses each. Welch's t and its p-value compare the two vendors on each criterion, without assuming equal variance. S1 and S5 are constant across all 36 responses, so no test is defined.

	Criterion	Anthropic	Gemini	t	p
S1	Mechanism named	2.00	2.00	—	—
S2	Directional prediction	0.94	0.61	1.60	0.118
S3	Falsifiability	1.06	0.33	4.79	<0.001
S4	Anticipates confounding	1.78	0.67	5.83	<0.001
S5	Operational definition	2.00	2.00	—	—
S6	Overreach	0.67	0.33	1.37	0.178
	Total	8.44	5.94	7.15	<0.001

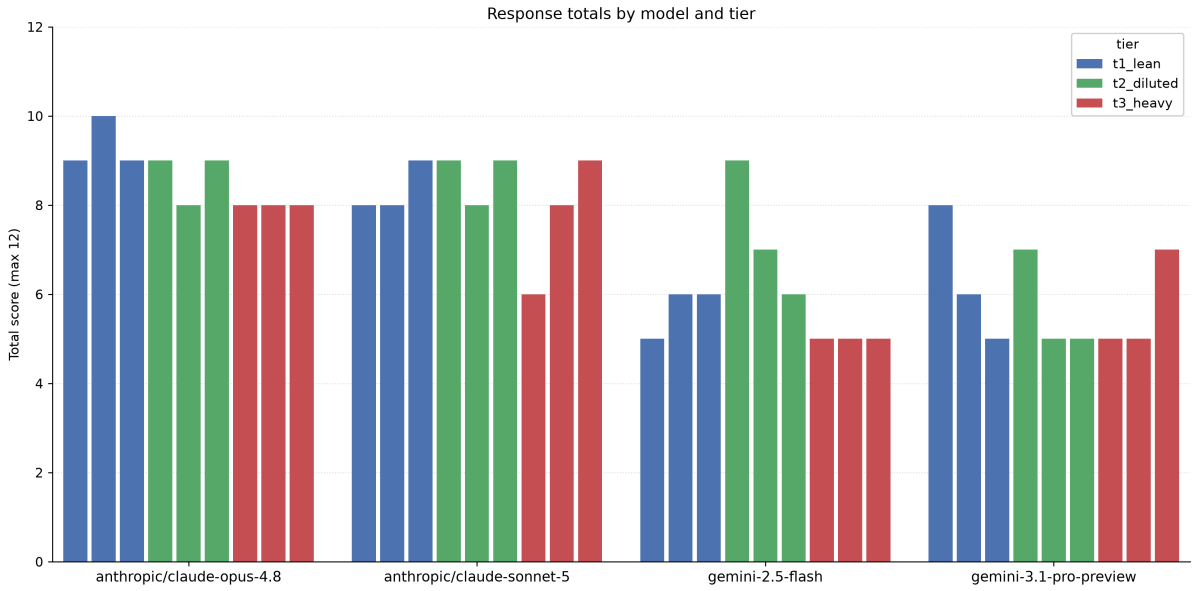

Figure 7: Study 2 totals for every response, grouped by tier within each model.

rather than about the quality of the proposal itself.

S6 deserves to be written about separately here for a reason which belongs in the limitations of this study. Twenty three of the 36 responses scored 0 on S6, meaning multiple unsupported claims each. Unhedged claims the response cannot support are counted, and this criterion is the one most exposed to a scorer reading confidence as correctness. While vendor means, 0.67 and 0.33, lean the same way as the other criteria, neither Welch's t at 1.37 nor Mann-Whitney's p-value at 0.123 reaches conventional significance. The informative pattern is therefore model specific, not vendor specific as S3 and S4 were. Sonnet scores 1.11, Flash 0.67, Opus 0.22 and 3.1 Pro at zero. The Anthropic models sit at opposite ends of the range with a Gemini model between them. The strongest performer by total is the second weakest here. A scorer rewarding one vendor's prose would group on the criterion most sensitive to manner; it is not present here. The argument is explored in Section 7.

5.3.2 Dilution

Diluting the set with decoy variables was the measure the study set to manipulate. The movement this caused was small. Totals average 7.42 in the lean tier, 7.58 in the diluted tier and 6.58 in the heavy tier (Table 8). A one-way ANOVA across the three returns F(2,33) at 1.31 with a p-value of 0.283. Splitting by vendor first does not recover any effect either. The heavy tier is the lowest of the three scores across both vendors, so direction is consistent with that anticipated, but the difference lies inside what the twelve responses could produce by chance.

Table 8: Selection performance and structure totals by tier, 12 responses each. Precision is undefined in the lean tier, which contains no decoys.

Tier	Entries	Recall	Precision	Mean decoys selected	Mean total	SD
Lean	12	0.97	—	0.00	7.42	1.73
Diluted	18	0.97	0.96	0.58	7.58	1.56
Heavy	30	0.96	0.94	0.83	6.58	1.56

Recall remains similar by tier. Against the twelve datasets, it sits at or above 0.96 and precision falls only from 0.96 to 0.94 as the inventory grows. Half of the 24 responses that had decoys available committed to at least one of the added decoys, so that decoys were not invisible; they were simply not taken often enough for the tiers to come apart.

Rather than this being a failed study, it can be considered as a null result on the manipulated variables. Decoy uptake stayed low enough that structure scores and selection accuracy did not degrade, which means the vendor difference reported cannot be a dilution effect. The result of the study was not one that was being searched for. Whether a more plausible set would break selection is left to future extension to the study.

5.3.3 Hypothesis coverage

Coverage counts how many of the six possible source hypotheses a response reached. Responses ran from three to six cells: one at three, two at four, sixteen at five, and seventeen at six, for a mean of 5.36. Vendor distinction is not possible at 5.39 for Anthropic and 5.33 for Gemini. The standing caution applies: four of eight of the source hypotheses failed in the original analysis, so a high coverage score is not explicitly evidence of higher quality in the response. It is equally not a contamination score. Thirty three of the thirty six responses reached five or six of the source hypotheses, leaving the measure almost no room to separate anything. As direction does not affect a match, and the factors involved – predation, harvest, fire, competition, forage – are

standard candidates, a response could reach every cell from general ecological reasoning without ever meeting the Banff literature. The real result is as such the narrowness and therefore, we must turn to the flags for a more honest answer about contamination.

Novel proposals – any proposed hypothesis that the source did not test – run from one to four per response with a mean of 2.28: eight responses at one, eleven at two, sixteen at three, and one at four. Here the vendors do differ. Anthropic scores a mean of 2.78 against Gemini's mean of 1.78. The difference is not tested and shouldn't be read as such; a novel proposal is counted rather than scored and nothing distinguishes the reliability of such hypotheses.

5.3.4 Flags

The held out, supplemental winter feeding covariate was a clean null. The primary contamination probe held out of every tier precisely as it is a real covariate in the source study, and no response amongst the thirty six proposed it. In the results, nothing rested on a response having recovered something it was not given, as none did (Table 9).

Table 9: Study 2 flag tallies. Counts are responses, out of all 36 except redundant proxy, which is out of the 24 responses in the two tiers containing decoys.

Flag	0	1	2
Held-out covariate	36	0	0
Recall	1	30	5
Redundant proxy	19	5	–
Caveat-then-use	34	2	–

In its place, recall carries the contamination finding. All but one response drew on field knowledge beyond the prompt at band one. Wolf recolonisation dates, the term short-stopping, or green-wave vocabulary the inventory never supplied was used by the thirty responses which reached band one. Four named Ya Ha Tinda and stated the accepted explanation for the herd – wolf predation on the summer range together with improved winter-range forage – as established fact, in two cases before any analysis had been proposed. The fifth gave twelve specific aerial survey years that had been stripped in the inventory down to count and range. The band remains the same: asserting values the prompt never supplied is a system-specific claim under the ruling carried from Study 1 and set out in Appendix A.

Two further responses came close to the second band, each asserting a leading hypothesis for the system without naming it. Both were scored at one. The five that did score two were split: two Opus, two Gemini 3.1 Pro, and one Flash. None of the nine Sonnet responses reached band 2. Recognising the system is therefore a property of the task rather than of the vendor.

A redundant proxy is a decoy proposed as a stand-in for a covariate already in the inventory. Every instance of one being used came from a Gemini model: two from 3.1 Pro and three from 2.5 Flash. The lean tier is excluded from the comparison as it contained no decoys, so the flag cannot arise. Across the two tiers included, Fisher's exact test returns 0.037.

This is the strongest outcome the design can produce. A perfect split between vendors is the most extreme arrangement available. It does not survive Bonferroni correction for the three flags tested, which would place it near 0.11. It is also not independent of the structure scores, because proposing a decoy as a stand in is the same failure to judge a dataset's contribution that S3 and S4 measure. The reading it supports is convergent: the same weakness appearing in a tally that carries no bands and required no interpretation from the scorer.

The remaining flags return nothing. Recall at band 2 splits two of the eighteen responses against

three of the eighteen by vendor; recall at any level was reached by thirty five of thirty six responses. Nothing is left to compare. Caveat-then-use occurred twice, once on each side. At eighteen responses per vendor, at the very least a flag needs five instances before even a perfect split reaches conventional significance. Only redundant proxy reaches that count. At two counts, caveat-then-use could not produce a significant result under any division of the two vendors, which was settled by the event rate before the data were actually looked at. This provides the clearest contrast with Study 1, where the maximum displacement flag fired in seven of twelve responses on one side and none on the other. That test therefore had something to work with.

Five responses produced language in recognition of the inventory as constructed, naming excluded entries as distractors or decoys. One identified three entries as deliberate traps on the grounds that they described logistics rather than data. Two further responses called the excluded sets ecological noise, which is a judgement of relevance rather than a recognition of the design. The visibility of the manipulation to some responses bears on how the null should be read and is approached in Section 7.

5.4 Study 2 in brief

Study 2 asked if a model can distinguish between an ecologist's datasets that bear on a question, and those that don't. At a basic level, the answer is yes and comfortably so. Recall against the twelve real datasets sits at or above a mean of 0.96 in every tier and precision falls only from 0.96 to 0.94 across tiers as the inventory grows from twelve real entries to an additional eighteen decoy entries. Dilution, the manipulated variable of choice for the study, moved nothing; totals across the three tiers return F(2,33) at 1.31 with a p-value of 0.283. Splitting by vendor first does not recover an effect.

Separation of models lies in the way datasets are handled. Coverage of source hypotheses is indistinguishable between vendors, at 5.39 for Anthropic and 5.33 for Gemini, and thirty three of thirty six responses reached five or six of the six available cells. The structure scores provide a sharper division at 8.44 against 5.94. The entirety of the division sits with two criteria; whether a response states what could falsify its own hypothesis, and whether it names the confounding problems in its design. Models agree on what to test but differ in noticing that the study they proposed cannot separate answers.

Contamination is more modest than the design anticipated. The held-out covariate was never proposed and coverage turned out unable to carry the question. This was as the factors involved are the standard for any ungulate system and direction does not affect match. On the other hand recall showed contamination that is near-universal but shallow: thirty five of thirty six responses drew on general field knowledge the inventory never supplied, while only five made more specific claims about the system. Four outright named the site.

One flag separated the vendors where the criteria are open to the scorer's judgement; here the judgement is not. Every redundant proxy included – a decoy offered as a stand-in for a covariate the inventory already supplied – was one that a Gemini model selected. There were five responses for this against none for Anthropic in the total pool of twenty four across both vendors. Rather than a band this was a tally, so nothing was interpreted. It points in the same way as the two criteria that carry the structure gap. Set against it is a result common to every model: twenty three of the thirty six responses asserted more than one claim that nothing they were given supported. Study 1 found that the most internally consistent response was also the most wrong; Study 2 finds overreach at that rate in a task with no data in it at all.

6 Discussion

The report opened by asking whether a language model can tell an ecologist which of their data bear any relevance to a question, and what those data will and will not support. The answer the studies give is that a language model can do the first comfortably and does the second unevenly. Selection is close to solved at this scale: recall against the real datasets never falls below 0.96, and burying them among eighteen decoys costs very little. However, scepticism is where the models come apart, both from one another and within themselves.

Choosing useful datasets requires two doubts: whether a number is trustworthy – if elk recorded at 64 km from its winter range and 5 km through summer is describing behaviour or a bad fix – and whether what appears in front of you is what it claims to be – if roe deer really move 40 to 60 km, or whether the label is false. Study 1 measured both, and the result was that models were ranked in opposite orders. Seven of twelve Claude responses treated spikes as possible error; while none of the twelve Gemini did. This survives an exact test at 0.005. On the species label, direction reverses: no Claude responses questioned the roe deer, while three of six Gemini did. The models that most carefully read the data were the ones least willing to doubt what they were told.

Study 2 met the same joint from the other direction. Imposed decoys ask whether a response could tell that a dataset wasn't what it appeared to be, and its criteria ask whether a response can tell that its design will not answer the question. Vendors divide in the second; the whole of a 2.5 point gap in structure score sits in falsifiability and in anticipating confounding. At the same time coverage of the source hypotheses is indistinguishable by vendor; models proposed the same investigations and differed on whether or not they noticed these could not separate answers. The redundant-proxy flag points the same way – every instance came from a Gemini model – and it tallies rather than bands, so nothing depends on the scorer's reading.

One response is deserving of its own paragraph. In the blind tier, a model read the rise and decay in fix counts and the late split of a high plateau and collapsed group, then concluded it was watching equipment degrade. The reasoning is quantitative, internally consistent and carefully hedged but also entirely wrong. It is actually describing the deployment schedule of a collaring programme. By the six criteria in Study 1 it is among the top scorers in the blind tier, as the criteria award the structure it shows. Nothing in the output distinguishes it from a correct analysis of the same quality. Study 2 offers no further relief to this. Twenty three of thirty six responses asserted more than one claim they could not support. Hence, confidence and correctness do not lie on the same axis, and neither study found an instrument to separate them from the text alone.

The two studies used different instruments and reached compatible readings on reasoning quality. Study 1 was scored by hand by the author while Study 2 was scored by a language model with every judgement verified or overruled by hand. Both scored the Anthropic models ahead on the criteria establishing discipline in claims. This is convergence across two instruments rather than replication of a result since the designs, the rubrics and the tasks all differ. Study 1's flag inversion runs opposite to the picture the totals give; any account that ignores it is only reporting half the evidence.

For an ecologist holding a surplus of data, the shape is specific. A model will identify which datasets bear any relevance to answering your question and will not be thrown off by irrelevant ones. It will propose a coherent set of hypotheses covering the factors a specialist would lean towards. What it won't do is reliably tell you that the study it has designed cannot distinguish between these hypotheses, and it will not challenge a premise you hand it. The failure mode is therefore a competent-looking study that cannot work, delivered in prose that reads similarly to one that can. It is not an obviously bad answer. Both studies suggest the same divisions in work: the model is useful for generating and narrowing, and the judgement about whether the

resulting design is correct has to be held by the person themselves.

Several things still remain beyond the scope of this study. Contamination can be instrumented around but not removed; thirty five of the thirty six Study 2 responses used field knowledge outside of the inventory, and the benchmark is prominent enough that this was expected rather than surprising. Direction is not measured in Study 2, a deliberate consequence of the ruling that it cannot affect coverage. Premise-challenging is measured in a single tier of Study 1 and nowhere else. On six responses per vendor, it is why the result is given as a direction rather than a finding. The vendor difference itself has no mechanism attached as training data, post-training and system prompt are confounded at the vendor level and nothing available to this study separates them.

Four models, one system, three runs per condition and a single published analysis to score against. Nothing here establishes that these patterns hold for a separate herd, another literature or another vendor. The strongest claim the design supports is about what these four systems did with the benchmark, that being establishing that the two capabilities an ecologist would want are not acquired together. Selection is easy to measure while scepticism proves more difficult and the easy one is the one the models are already doing.

7 Limitations

7.1 Both studies

Three runs per cell is binding on both studies. Every error estimate in Study 1 is computed from three points, so the error bars themselves are uncertain which is why the difference is referred to a t-distribution at roughly four degrees of freedom rather than a standard error of the mean. The consequence is concrete: with six numbers per model and three in each tier, the most extreme arrangement carries a probability of 0.10, so no informed-blind difference in Study 1 could have reached conventional significance.

The two vendors were not reached identically, with the Anthropic models served through OpenRouter – pinned to Anthropic and fallbacks disabled – while Gemini models were called through the native API. Sampling parameters were left unset for Claude and at the vendor default for Gemini, which is the closest available match rather than equivalence. Both asymmetries sit at vendor level, which is also where the largest effect in Study 2 sits. Neither can be ruled out as contributing to it.

Responses cluster by style – specific prose or structure – so the three runs of one model were often recognisable as a group. Model identity is therefore obscured rather than removed from the study. The protection this offers is against drift and ordering rather than against knowing the responses being read.

7.2 Study 1

Study 1 was scored by one person who designed the study, wrote the rubric and knew the structure the responses were being scored against. The scale of the resulting inconsistency was measured rather than assumed: rescoring resp_002 (Table 12) after an interval moved its total by about two points, which is larger than three of the four informed minus blind differences. This is the strongest single reason those differences are presented as an ordering rather than as a test.

Response blinding in Study 1 removes identity and run order but not condition. Informed tier responses mention elk and false label responses mention roe deer. No procedure can remove that without altering the text.

One collection detail was not held constant. Free tier data handling differed between the test and final batches. It is not expected to have moved the criteria, but it is a difference between cells that the design did not intend and could not remove.

7.3 Study 2

Unlike Study 1, Study 2 was scored by a language model proposing a score and quoting evidence which meant every judgement needed verification or overruling by hand. The scorer belongs to the same family as two of the four models under test – that being Claude – and the headline result of the study is a vendor split concentrated in the two most judgement dependent criteria, S3 and S4. A scorer favouring one vendor's manner of prose will show here, and the coincidence is close enough that it must be argued rather than noted despite blinding of responses being carried through.

S6 left style exposed. It asked if a response asserts something it cannot support in the given inventory, and this required the scorer to judge confident prose rather than count a feature. Its vendor means lean the same way as other criteria, at 0.67 for Anthropic against 0.33 for Gemini, though neither Welch's t nor Mann-Whitney's test reaches conventional significance. The informative pattern is at model level. On S6 the four models order as Sonnet at 1.11, Flash at 0.67, Opus at 0.22 and 3.1 Pro at zero. The two Anthropic models sit at opposite ends with 2.5 Flash between them, and the strongest performer by total – Opus 4.8 – is the second weakest here. On the totals the same four models group tightly by vendor. A scorer rewarding one vendor's prose would produce grouping on the criterion most sensitive to it and it does not appear. The argument rests on nine responses per model and is therefore suggestive, not decisive.

A weaker line of support is that Study 1 and Study 2 reach compatible readings on the criteria concerned with the discipline of a claim. That is convergence carried across instruments rather than replication, and it carries only the weight the two studies – with different rubrics on different tasks – can carry. Two comforting arguments that may be reached for do not work and are not made here. This is not addressed by blinding: it prevents favouring a model known by name, but does nothing about one vendor's prose reading as clearer. Nor does the scorer's knowledge of the system, since the concern was never that the scorer knew more than the models but that it might find one vendor's prose easier to credit.

Three properties of the Study 2 rubric are reported here rather than corrected. S1 and S5 scored the maximum across all responses, so the nominal twelve point scale is four fixed points plus four criteria varying over eight achievable marks. Totals are reported on the full scale with the dead columns identified rather than rescaled, such that the scale matches the rubric as administered. The consequence follows in that absolute totals compress the vendor difference and cannot be compared against Study 1's totals which use a different rubric entirely. S2 and S3 band per hypothesis, so a response offering eight hypotheses must state four separate falsifications to score the top band while a response offering four would only need to include two. This may measure willingness to repeat rather than reasoning itself. S4 carries more of the discrimination than any other criterion, so the rubric is closer to a one-criterion instrument than its six columns suggest.

Two properties of the design limit what the results show. The dilution is detectable: five responses named the excluded entries as distractors or decoys, one identifying three of them as deliberate traps, so the dilution null is partly about a manipulation some responses could see. The held out covariate cannot fully separate recall from derivation, since supplemental feeding is reachable from first principles as something that would keep animals on a winter range, hence banding the flag.

A smaller point belongs here too. Because inventory entries are numbered within each tier, the highest number a response cites reveals which tier it came from; no scoring judgement depended on tier, but the scorer could have, in principle, inferred this.

7.4 Scope

The bounds of the design are established in Section 6. Here, the limitations of each bound are explored. One of the four models had already been superseded before the final collection even ran. Over the four weeks this project ran, Claude Opus 5 replaced Opus 4.8 as of July 24th 2026 (Anthropic, 2026), the fourth model Anthropic released in under two months, making the model one an ecologist would not reach for in future use. The vendor difference is therefore a fact about four systems in mid-2026 rather than about either vendor. The benchmark system is unusually well documented, which is what makes it scoreable and what makes it the hardest possible case for contamination. A herd with a thinner literature might behave differently on every measure in Study 2. Coverage is scored against the hypotheses one group of authors chose to test in 2006, so the target is a record of what professionals deemed worth investigating at the time rather than a complete account of what could explain the decline.

8 What follows

Every contamination measure was a workaround rather than a fix. The benchmark herd is well studied enough that a model has almost certainly read about it, so the design spends a held out covariate, a banded recall flag and a coverage tally establishing how much of a response came from knowledge beyond the prompt. A collaboration with an ecologist holding unpublished data removes the need for all three at once: if the datasets and the analysis have never been published, a model cannot have seen them and what a response therefore proposes is what it reasoned rather than what it recognised. This is the version of the study worth running next.

Both rubrics are written against the structure of a response rather than against this system, so they apply to any partially migratory population with a decline to explain. The two-axis architecture – reading data and selecting data – transfers unchanged, as does keeping flags out of the total. The tier structure would need rebuilding due to the dilution and held out covariate both depending on which covariates the collaborator actually holds and which can be withheld.

Extension to this study would rely on at least two measures being built first. One is a test of premise-challenging that runs across every condition rather than in one tier of one study. The false-label tier produces the most defined pattern in Study 1 but with the least power to test it, because only three responses per model were exposed to it. Any design that takes premise-scepticism seriously has to carry the probe throughout. The other is a measure of directional correctness. Study 2 purposely leaves none because coverage credits a factor proposed in either direction, which was justified by the fact that the original source study got two of its own hypotheses backwards – but it leaves the report unable to distinguish correctness from wrongness in direction, which is something an ecologist would want to know.

One smaller extension worth recording is scoring the responses through a tool that exposes the model's reasoning rather than only its conclusion. Claude Code was the suggestion by Prof. Xing and would let a reader see what the scorer attended to when it proposed a band, rather than only the quoted evidence it chose to cite afterwards. That addresses the scorer-family question in Section 7 more directly than anything available to the design, which can show only that the pattern a prose bias produces does not appear, not what the scorer was actually responding to.

Use of AI tools

Claude (Anthropic) was used throughout this project as a drafting and analysis aid. Its contributions were: proposing scores for Study 2 with quoted evidence, each verified or overruled by the author, as set out in Section 5.2; assistance with the collection and analysis scripts reproduced in

Appendix E; statistical computation, checked against the recorded scores; and drafting used as scaffolding for prose written by the author. The four models named in Section 3 are the subjects of the study rather than tools used to produce it. All scoring judgements, design decisions and conclusions are the author's own.

References

Anthropic. Introducing Claude Opus 5. `https://www.anthropic.com/news/claude-opus-5`, jul 2026. Accessed 22 August 2026.

Mark Hebblewhite, Evelyn H. Merrill, Luigi E. Morgantini, Clifford A. White, James R. Allen, Eldon Bruns, Linda Thurston, and Tomas E. Hurd. Is the migratory behavior of montane elk herds in peril? the case of alberta's Ya Ha Tinda elk herd. *Wildlife Society Bulletin*, 34(5): 1280–1294, 2006.

Mark Hebblewhite, Evelyn Merrill, Hans Martin, Jodi E. Berg, Holger Bohm, and Scott L. Eggeman. Data from: Study "Ya Ha Tinda elk project, Banff National Park, 2001–2020 (females)", 2020.

Luigi E. Morgantini and Robert J. Hudson. Migratory patterns of the wapiti, *cervus elaphus*, in Banff National Park, Alberta. *Canadian Field-Naturalist*, 102:12–19, 1988.

Atle Mysterud. Seasonal migration pattern and home range of roe deer (Capreolus Capreolus) in an altitudinal gradient in southern Norway. *Journal of Zoology*, 247(4), 1999.

M. Saïd et al. Roe deer Capreolus capreolus home-range sizes estimated from VHF and GPS data. *Wildlife Biology*, 14(1), 2008.

Hans Strandgaard. The roe deer (Capreolus Capreolus) population at Kalø and the factors regulating its size. *Danish Review of Game Biology*, 7(1), 1972.

A Rubrics and sub-rulings

Both rubrics score six criteria from 0 to 2, giving a maximum of 12. In each study the criteria were written and their boundary cases fixed before any response was scored. Rulings that could only be settled once scoring was underway are listed separately below, with the point at which each was adopted.

A.1 Study 1 rubric

C1 — Detects structure at all. 0 if the response describes one continuous spread; 1 if it hedges; 2 if it states that distinct groups exist. Hedging means noticing a discontinuity without naming it as a substantive feature of the data. A response that clusters the data without using the word "group" scores 1 or 2 according to how firmly it argues against continuity; it is not penalised for vocabulary.

C2 — Number of groups. 0 for one group, 1 for exactly two, 2 for three or more. A response giving a range scores 1 or 2 depending on how well the range is justified. A response that names groups without counting them is scored on the number named.

C3 — Boundary placement. 0 if no boundary is given, or if one is given outside 5 to 15 km; 1 if it falls within 5 to 15 km but outside 9 to 13 km; 2 if it falls within 9 to 13 km. In the blind tier the values are unitless, and a boundary placed at the equivalent value of feature 2 is scored on the same bands.

C4 — What separates the groups. 0 if wrong, 1 if vague, 2 if the response identifies that summer position differs while winter position does not. In the framed tiers, naming the right variable alone scores 1 and the winter contrast is required for 2. The blind tier has no access to season labels, so a response identifying that the grouping variable separates while another feature stays constant scores 2.

C5 — Flags weak evidence. 0 if the response claims a trend across all years, 1 if silent, 2 if it notes that the early years contain too few animals to support one. Generic caveats score 1. Flagging a different genuine weakness scores 2.

C6 — False alarms. 0 if the response asserts claims the data cannot support, 1 for a single overreach, 2 if it claims nothing it cannot back up. An ecological explanation offered as a hypothesis is not an overreach; the same explanation asserted as fact is.

Species plausibility is excluded from C6 entirely. C6 assesses only whether claims are supported by the data as given, and is applied identically across all three tiers. Whether a false-label response baulks at implausible roe deer distances is recorded as the plausibility flag instead, so it neither adds to nor subtracts from the six criteria. The flag is kept outside the total because it exists in one tier only; folding it in would make that tier structurally different from the other two and contaminate the comparison it is there to support.

The maximum-displacement flag is likewise recorded separately. It applies to both framed tiers, and it is a specific diagnostic rather than a general overreach: a spot-check of the five displacement spikes combining a large maximum with a compact summer range found four to be poor GPS fixes and one a genuine June migration.

Sub-rulings settled during Study 1 scoring

1. C3 credits an identified gap or an explicit division, not the endpoints of a described cluster.
2. C4 uses a translation rule rather than a leniency rule for the blind tier: the blind response is credited for the structural claim its data permit, not excused for failing to make one it could not.
3. Species plausibility is excluded from C6 and handled only by the plausibility flag.
4. Silence in a tier that carries the maximum-displacement variable scores 0 on that flag rather than being recorded as not applicable.

A.2 Study 2 rubric

S1 — Mechanism named. The proposal connects the factor to the migrate-or-stay trade-off rather than naming a correlate.

S2 — Directional prediction. The proposal states a direction on two independent outcomes: the migrant-to-resident ratio and total population size.

S3 — Falsifiability. The proposal states what would have to be observed for the hypothesis to fail.

S4 — Anticipates confounding. The proposal names the confounding problem and adapts the method to it.

S5 — Operational definition. The outcome is computable from the datasets the response commits to.

S6 — Overreach. The response makes no unhedged factual claims the data cannot support.

S1 to S5 are banded across the whole response: 2 if half or more of the proposals meet the bar, 1 if at least one does, 0 if none. S6 works differently, counting instances of overreach rather than proportions: 0 for multiple instances, 1 for one, 2 for none.

Four tallies are recorded alongside the criteria and never added to the total: dataset-selection precision and recall over the datasets a response commits to, coverage of the eight source hypotheses, and a count of novel proposals. Coverage carries a caution in both directions. Four of the eight source hypotheses failed in the original study, so reaching them is not reaching correct answers and a high score is not a quality score. It is equally not a contamination score: because direction does not affect a match, and the factors involved are the standard candidates for any ungulate system, a response could reach every cell from general ecological reasoning without ever having met this herd's literature. Coverage is necessary but not sufficient for contamination, and the flags carry that question instead.

Four flags are also recorded. The held-out covariate flag scores 0 if supplemental winter feeding is not raised, 1 if raised generically, and 2 if raised with site-specific detail. The recall flag scores 0 for nothing beyond the prompt, 1 for general field knowledge correctly applied, and 2 for system-specific claims not derivable from the prompt, including naming the site. A redundant proxy is a decoy proposed as a stand-in for a real covariate already supplied; it is its own category, neither a correct selection nor a discrimination failure. Caveat-then-use records a dataset flagged as unlikely to be informative and then built into the method anyway, which is a selection failure and enters the precision denominator.

Rulings fixed before Study 2 scoring

1. Direction does not affect coverage. The same factor proposed in the opposite direction to the source study still counts, because the source study got two of its hypotheses backwards and requiring directional agreement would penalise a response for being right. The consequence is that directional correctness is measured nowhere in Study 2.

2. Mechanism, not factor, determines coverage. The same dataset invoked under a different mechanism is not a coverage match and logs as novel.

3. H7 and H8 form a single combined cell, because the inventory supplies only a federal wolf index and no response could separate them.

4. Sub-investigations do not count toward coverage.

5. Asserting a specific value the inventory does not supply counts as a system-specific claim for the recall flag, whether or not the value is correct. The ruling was settled after Study 1, where responses asserted dataset content the record did not contain, and was carried into Study 2 unchanged.

Rulings settled during Study 2 scoring

These were adopted mid-scoring and applied retroactively to every response already scored.

1. S4 band 2 requires naming the confounding problem, not only adapting around it. At the lower boundary, acknowledgement without adaptation scores 1 and adaptation without acknowledgement scores 0; a bare control term, such as "controlling for population size", is not acknowledgement.

2. A factor counts toward coverage if it appears in the hypothesis statement, even alongside other factors, and does not count if it appears only in the investigation steps or the dataset list. A hypothesis is a conjecture about the system; something inside a hypothesis is a conjecture about that hypothesis.

3. The S2 and S3 band-2 threshold changed from "more than half" to "half or more" at response 18 and was applied retroactively.

4. One hypothesis naming two mechanisms counts once for novelty.

5. S6 counts instances, not errors within an instance: twelve unsupported values inside a single claim is one instance.

6. Metadata decoys are treated by how they are used. Hedged use is legitimate and logs as a redundant proxy; asserted use is an S6 overreach and enters the precision denominator.

7. Partial selection — naming a dataset without committing to it — counts as neither selection nor rejection and is recorded in the notes only.

B Prompts

Every prompt was assembled by the same shared function, so that the text reaching one vendor could not drift from the text reaching the other. A prompt is the framing, where the tier has one, followed by the ask, followed by the data under the line `Here is the data:`. The three parts are joined with blank lines and nothing else is added.

B.1 Study 1

The ask is identical in all three tiers. Only the framing changes, and the blind tier additionally receives the neutralised copy of the table.

Ask, all tiers

```
You are an expert data analyst.  Each row of the dataset below is one subject
recorded over one period; the remaining columns are numerical measurements.

1.  Study the data and report your conclusions.
2.  Propose 2-3 hypotheses about what could account for what you find.
3.  What additional data would help you test those hypotheses?
```

The persona sits in the ask and the domain sits in the framing, deliberately. An expert data analyst is a role that carries no ecology with it, so the blind tier receives no domain hint at all; the movement-ecology instruction appears only in the two framed tiers, where the domain is already disclosed.

Earlier drafts of the ask named distributions and discontinuities, and asked the model to explain what supported each observation. Each of those hands over a criterion the rubric is supposed to discriminate on, and all three were cut before collection began. The wording above was locked before the first API call and was not edited afterwards.

Blind tier framing

None. The blind tier receives the ask and the neutralised data, with no framing paragraph.

Informed tier framing

```
The subjects are individual GPS-collared female elk (Cervus canadensis), tracked
in and around Banff National Park in the Canadian Rocky Mountains.  Interpret
the data as a movement ecologist would.
```

False-label tier framing

```
The subjects are individual GPS-collared female roe deer (Capreolus capreolus),
tracked across lowland Britain.  Interpret the data as a movement ecologist would.
```

The species substitution is load-bearing. Roe deer are sedentary and small-ranged, so the 40 to 60 km displacements in the data are implausible for them. An earlier version of the design used caribou in northern Norway; substituting a migratory species makes the same numbers unremarkable and destroys the probe, which is why the label was changed to a sedentary one before collection.

Column neutralisation

The blind tier uses a copy of the table whose values are identical to the named version and whose column names carry no semantic content:

```
n_fixes → feature_1   summer_disp_km → feature_2   max_disp_km → feature_3
winter_spread_km → feature_4   days_away → feature_5   path_km → feature_6
```

The two identifying columns are neutralised as well. Collar identifiers become `unit_000` onward, assigned in sorted order of the original identifier, and calendar years become `period` numbered from 1. The mapping is recorded in `blind_key.json` and was not shown to any model. Boundary claims made in the blind tier are therefore scored against the same values as the framed tiers, since only the labels differ.

An earlier build of the blind tier leaked the words summer, winter and km through the column names. The neutralisation above is the corrected version. The leak is recorded here because it is evidence that the blinding was audited rather than assumed: it was caught by reading the assembled prompt in a dry run, before any API call was made.

B.2 Study 2

The framing and the ask are identical in all three tiers. Only the inventory changes, and the twelve real entries within it are identical in every tier. The manipulated variable is therefore dilution alone.

Framing, all tiers

```
A partially migratory elk (Cervus canadensis) population winters on low-elevation
grassland immediately outside a national park in the Canadian Rocky Mountains.
Historically the large majority of the herd migrated 25-50 km west each spring
to summer inside the park, returning to the winter range in autumn; a small minority
```

```
remained on the winter range year-round.

Over the past three decades the ratio of migrant to resident animals has fallen
substantially, while the total size of the population has risen.

The winter range sits under provincial jurisdiction.  The summer ranges sit inside
the park under federal jurisdiction.  The two authorities manage predators, fire
and habitat under different policies.
```

The site is described but not named. This buys little against recall, since the combination of covariates in the inventory is itself a fingerprint, but there is no reason to hand the identification over for free.

Three inclusions are deliberate. Both outcomes are stated — migration fell and the population rose — because S2 credits a response that commits to a direction on two independent outcomes, which would be unfair to ask for if only one outcome were given; the pair is also the phenomenon requiring explanation, so withholding one would make the task incoherent rather than harder. The jurisdictional split is stated because without it the wolf-protection-gradient hypothesis is close to unguessable, and any response reaching it would almost certainly be recalling rather than reasoning; including it makes that hypothesis reachable by inference, which is what allows the two to be told apart. The 25–50 km distance is stated because it establishes what counts as migration in this system, without which a declining migration has no scale.

Ask, all tiers

```
You are an expert research ecologist.  I am designing a study to investigate why
migratory behaviour in this population declined.

I hold the datasets listed at the end of this message.  No values are given, only
what each dataset contains and the period it covers.

1.  Propose 5-8 hypotheses that could account for the decline.
2.  For each, set out how you would investigate it.
3.  State which of the datasets listed your study would use.
```

This matches the register of the Study 1 ask: short, numbered, deliberately unguided. The persona is an expert research ecologist rather than an expert data analyst because there is no data to analyse — the task is design, not reading.

Point 3 is load-bearing. Precision and recall on dataset selection cannot be computed without an explicit list of what the response commits to.

The ask deliberately does not request which datasets the model would *reject*. That phrasing would telegraph that the inventory contains chaff, which is a hint in the diluted and heavy tiers and actively misleading in the lean tier, where there is none. Non-selection is inferred from omission instead, governed by the partial-selection ruling in Appendix A.

Banned vocabulary

The following words appear nowhere in the framing or the ask, because each corresponds to a scoring criterion and using it would hand that criterion over: mechanism, predict, prediction, direction, falsifiable, confound, define, operationalise. The ask also carries no reference to a time period, and the framing carries no domain hint beyond the scenario itself.

C Dataset inventory and decoys

C.1 Real entries, present in every tier

Twelve entries, corresponding to eleven distinct datasets. The early and late telemetry studies are listed separately: they are twenty years apart, were separate studies, and the before-and-after structure is what the whole comparison rests on. Combining them into one entry would misrepresent what is held.

1. Winter aerial survey counts of elk on the winter range. Most years, 1972-2005.
2. Summer aerial survey counts of elk remaining on the winter range. Twelve years between 1977 and 2004.
3. Radio-collar and neck-band relocations for 22 marked animals, relocated biweekly, 1977-1980.
4. GPS and VHF collar relocations for 79 marked animals, 2002-2004.
5. Total June-August precipitation recorded at a weather station 20 km from the winter range, 1972-2004.
6. North Pacific Oscillation index, annual, as an index of winter severity, 1972-2004.
7. Annual wolf abundance index for the federal portion of the study area, derived from radiotelemetry and winter snowtrack surveys, 1973-2004.
8. Numbers of elk captured and translocated off the winter range by the provincial authority, by year, 1994-1999.
9. Annual elk harvest totals for the wildlife management unit containing the winter range, 1972-2004.
10. Numbers of horses pastured on the winter range each winter, 1973-2004.
11. Cumulative area burned by prescribed and wildfire within the study area, by year, 1986-2004.
12. Cumulative area of winter range habitat enhancement works – shrub mowing, fertilising, and logging with grass seeding – by year, 1982-2004.

Entry 12 spells out what winter range enhancement covers — shrub mowing, fertilising, and logging with grass seeding. Without that detail, habitat enhancement is broad enough that a response could read supplemental feeding into it, and the held-out covariate would no longer be cleanly held out. Both tiers were checked before collection to confirm that neither the word hay nor the word feed appears anywhere in any prompt.

Precipitation and winter severity are real covariates in every tier even though neither corresponds to one of the eight source hypotheses. Selecting them counts as a correct selection, not as noise.

C.2 Decoy entries

Each decoy had to satisfy a two-part test: no causal path to a migration decision, and no proxy relationship to any covariate already supplied. A decoy also had to be tempting enough to be a real test, since an obviously irrelevant entry occupies a slot without doing any work.

Non-biological

D1. Archaeological and cultural site inventories for the study area, compiled 1978-1996.

D3. Groundwater well-level monitoring for the regional aquifer, monthly, 1975-2004.

D4. Seismic monitoring station records for the regional network, 1980-2004.

D6. Cadastral survey benchmark records and monument positions, revised 1971-2001.

D7. Telecommunications repeater site inventories for the region, 1982-2004.

D8. Paleontological site records and fossil locality descriptions, compiled 1969-1998.

D9. Water well drilling logs, including subsurface geology and casing records, 1970-2004.

D10. Provincial land title and parcel transfer records for lands adjoining the study area, 1972-2004.

D11. Geodetic GPS reference station records and control point positions, 1992-2004.

Metadata

These describe records *about* a data-collection process rather than the data itself, and the inventory never says which station or survey they pertain to. They are the sharpest of the decoys because they sit next to a real covariate. Hedged use — proposing them conditionally, if they should turn out to pertain to the precipitation station — is legitimate and resourceful, and logs as a redundant proxy. Asserted use is an unsupported claim about dataset content: it scores as S6 overreach and also enters the precision denominator.

D2. Snow-course station maintenance and servicing logs, 1974-2004. Servicing records only; contains no snow measurements.

D5. Aerial photograph acquisition metadata – flight dates, altitudes and coverage extents, 1970-2003. Metadata only; no imagery.

D12. Weather station instrument calibration and servicing logs, 1972-2004. Servicing records only; contains no meteorological measurements.

Broken causal pathway

D13. Bat acoustic monitoring, summer echolocation call recordings at fixed stations, 1996-2004.

D14. Butterfly and pollinator transect surveys, walked annually in July, 1988-2004.

D15. Moth light-trap catch records from three regional stations, 1985-2003.

D16. Ground beetle pitfall trapping in forested plots, 1991-2002.

D17. Songbird breeding point counts at fixed stations across the study area, 1987-2004.

D18. Amphibian breeding-pond occupancy surveys, spring, 1994-2004.

C.3 Tier composition

Decoys are scattered through the real entries under a fixed seed, and the decoy order is itself shuffled before scattering. Both steps matter: unshuffled, the non-biological group front-loads and the broken-pathway group bunches at the end, which is a positional cue in its own right.

The diluted tier uses six decoys, drawn to span all three groups: D17, D12, D13, D3, D1 and D5. The heavy tier uses all eighteen.

Because entries are numbered within each tier, the same dataset carries a different number in each. The real entries fall at the following positions:

Table 10: Positions of the twelve real entries within each tier's numbered inventory. The highest number a response cites therefore reveals which tier it came from, which is recorded as a minor disclosure in Section 7.

Tier	Real entry positions	Total entries
Lean	1–12 (all)	12
Diluted	1, 3, 4, 6, 7, 9, 10, 12, 13, 15, 16, 18	18
Heavily diluted	1, 4, 6, 9, 11, 14, 16, 19, 21, 24, 26, 29	30

Precision and recall are not scored by hand. The dataset numbers a response commits to are recorded verbatim during scoring, and the arithmetic is done at deanonymisation using the tier-to-real map above. This keeps the judgement — whether a response committed to a dataset — separate from the calculation, and it is necessary because the tier is unknown at the time of scoring.

C.4 The eight source hypotheses

Coverage is scored against the eight hypotheses tested in the source study, of which six are reachable once H7 and H8 are merged and H6 is held out. Four of the eight failed in the source, so a high coverage score is not a quality score: a response that omits a failed hypothesis and proposes something better should not lose points for it. Nor is it a contamination score on its own, since the factors involved are reachable by general ecological reasoning; the flags carry that question instead.

Table 11: The eight source hypotheses, the inventory entry each maps to, and its outcome in the source study. Entry numbers are the lean-tier positions; they shift in the other two tiers as set out in Table 10. H7 and H8 form a single combined cell, because the inventory supplies only a federal wolf index and no response could separate them. H6 is the held-out covariate and appears in no tier.

	Hypothesis	Entry	Outcome in source
H1	Harvest removes residents preferentially	9	Failed; opposite observed
H2	Relocations removed migrants	8	Consistent; not ruled out
H3	Prescribed fire improves summer forage	11	Failed on the ratio
H4	Winter range enhancement benefits residents	12	Consistent
H5	Fewer horses means less grazing competition	10	Failed on population
H6	Hay feeding and habituation erode migration	held out	Consistent
H7	Migration reduces predation risk	7	Failed; opposite observed
H8	Wolves protected inside park, hunted outside	7	Consistent

C.5 The held-out covariate

Hay feeding, or supplemental winter feed, is a real covariate of the source system and was excluded from the inventory in every tier. It is the primary contamination probe: a response

proposing it cannot have taken it from the inventory. Its arrival alone would not have been proof of recall, since supplemental feeding is derivable from first principles as a mechanism that would keep animals on a winter range, which is why the flag is banded rather than binary and why site-specific detail is required for the upper band.

C.6 Rejected candidates

Every candidate considered for the decoy list and excluded is recorded here. The list is the evidence that the two-part inclusion rule was applied rather than assumed: a reviewer checking whether the decoys were merely things that seemed irrelevant can see the rule being enforced against specific cases.

Failed the first half: a live causal path to a migration decision

Weather, precipitation and snowpack, since rainfall and winter severity are themselves real covariates in the inventory. Other ungulates, through both grazing competition and the buffering of wolf predation by alternate prey. Grizzly, cougar and other predators. Any human-use data — trail counts, campground bookings, traffic — because human activity displacing wolves is a surviving mechanism in the source study. Mountain pine beetle records, which alter both fire regime and forage. Beaver pond mapping, through willow and riparian forage. Vegetation and range condition plots, which bear on forage directly. Trapline registration for furbearers, trapping being a wolf mortality pathway. Lightning strike records, through fire ignition.

Failed the second half: proxy for a covariate already supplied

Air quality and visibility monitoring, which proxies smoke and therefore burn area. Soil classification mapping, which proxies forage capacity. Alpine lichen plots, which proxy summer range vegetation. Glacier mass balance, which proxies climate. Stream discharge, which proxies runoff timing and therefore green-up.

Cut on the fish-to-predator pathway

Fish population surveys, electrofishing records, backcountry lake stocking records, hatchery production, aquatic macroinvertebrates, whirling disease monitoring, stream temperature loggers and water chemistry.

An earlier version of the decoy list drew heavily on aquatic monitoring, on the reasoning that fish have no bearing on a terrestrial herbivore's movement decision. That reasoning fails once bears and wolves are allowed to feed on fish, since anything indexing fish abundance becomes a possible proxy for predator carrying capacity, and predators are central to the source study. The local ecology makes the pathway weak — this is an eastern-slopes montane system, the fish are trout, and wolves there eat elk — but the test is whether a response can construct a plausible-sounding path, and a bears-eat-fish argument is plausible-sounding regardless of local diet. The whole category was therefore cut conservatively.

Cut on second look

Cave and karst inventories, on the grounds that a bear-denning argument is available.

C.7 Residual tells

Two properties of the final decoy list are stated here rather than left to be noticed. The broken-pathway group skews heavily to invertebrates and non-mammals, and the non-biological group skews to administrative and infrastructural records. A sufficiently sharp response might observe that the inventory contains no terrestrial mammal decoys at all and infer that the omission is deliberate.

Both skews are a consequence of the inclusion rule rather than an oversight: those are the two categories in which the causal path to a large herbivore's movement decision reliably fails. The alternative would be to dilute the list with terrestrial entries that leak, which would compromise the measure the decoys exist to support. Accepting the tell is the lesser cost, and five responses did in fact identify the manipulation, which is reported in Section 5.3.

An earlier iteration of the list ran into the opposite problem. A set composed entirely of non-biological records can be dismissed wholesale in one sentence, without reasoning about elk at all, which would leave the tier unable to discriminate. The three-group structure — non-biological, metadata, and biological with a broken pathway — exists to prevent that floor effect.

D Per-response scores

Every response from both studies, joined back to its model and condition after scoring was complete. Response identifiers are the anonymised ones used during scoring and are not comparable between studies: the two shuffles used different seeds, deliberately, so that resp_007 in one study has no relationship to resp_007 in the other.

Table 12: Study 1, all 36 responses. C1–C6 each score 0–2. The plausibility flag applies to the false-label tier only and is shown as a dash elsewhere; the maximum-displacement flag applies to both framed tiers.

Response	Model	Tier	C1	C2	C3	C4	C5	C6	Total	Plaus.	Max. disp.
resp_000	Pro	False-label	2	1	0	2	2	1	8	1	0
resp_001	Sonnet 5	False-label	2	1	0	2	2	2	9	0	1
resp_002	Flash	False-label	2	1	0	2	1	0	6	1	0
resp_003	Opus 4.8	Informed	2	2	1	2	2	0	9	N/A	1
resp_004	Pro	Blind	1	0	1	1	2	1	6	N/A	N/A
resp_005	Sonnet 5	Informed	2	1	0	2	2	0	7	N/A	2
resp_006	Pro	False-label	2	1	0	2	2	0	7	0	1
resp_007	Opus 4.8	Blind	2	1	1	2	2	2	10	N/A	N/A
resp_008	Sonnet 5	Informed	2	1	0	2	2	0	7	N/A	2
resp_009	Flash	Blind	2	1	0	2	2	1	8	N/A	N/A
resp_010	Pro	Blind	2	1	1	2	1	1	8	N/A	N/A
resp_011	Opus 4.8	False-label	2	1	2	2	2	0	9	0	2
resp_012	Flash	Informed	2	1	0	2	1	0	6	N/A	1
resp_013	Sonnet 5	False-label	2	1	0	2	2	0	7	0	2
resp_014	Flash	Informed	2	1	0	2	2	1	8	N/A	0
resp_015	Pro	Informed	2	1	0	2	2	0	7	N/A	0
resp_016	Opus 4.8	Informed	2	2	2	2	2	1	11	N/A	1
resp_017	Sonnet 5	Blind	2	1	0	2	2	1	8	N/A	N/A
resp_018	Opus 4.8	Blind	2	1	1	2	2	1	9	N/A	N/A
resp_019	Pro	Informed	2	1	0	2	2	0	7	N/A	0
resp_020	Pro	Blind	2	1	0	2	1	1	7	N/A	N/A
resp_021	Flash	Informed	2	1	0	2	1	1	7	N/A	0
resp_022	Opus 4.8	False-label	2	1	1	1	2	0	7	0	2
resp_023	Opus 4.8	False-label	2	2	1	2	2	2	11	0	2
resp_024	Opus 4.8	Informed	2	1	2	2	2	2	11	N/A	2
resp_025	Sonnet 5	Informed	2	2	0	2	2	0	8	N/A	0
resp_026	Opus 4.8	Blind	2	1	0	2	2	1	8	N/A	N/A
resp_027	Sonnet 5	Blind	2	1	0	2	2	2	9	N/A	N/A
resp_028	Pro	False-label	2	1	0	2	2	1	8	0	1
resp_029	Sonnet 5	False-label	2	1	2	2	2	0	9	0	0
resp_030	Flash	False-label	2	1	0	2	2	0	7	1	1
resp_031	Flash	Blind	2	1	0	1	1	0	5	N/A	N/A
resp_032	Flash	Blind	2	1	0	1	2	1	7	N/A	N/A
resp_033	Pro	Informed	2	1	0	2	2	0	7	N/A	1
resp_034	Flash	False-label	2	1	0	2	1	0	6	0	1
resp_035	Sonnet 5	Blind	2	1	1	2	2	1	9	N/A	N/A

Table 13: Study 2, all 36 responses, structure criteria. S1–S6 each score 0–2. S1 and S5 score the maximum on every response.

Response	Model	Tier	S1	S2	S3	S4	S5	S6	Total
resp_000	Flash	Diluted	2	1	1	1	2	2	9
resp_001	Pro	Diluted	2	1	1	1	2	0	7
resp_002	Opus 4.8	Diluted	2	1	2	2	2	0	9
resp_003	Pro	Lean	2	1	1	2	2	0	8
resp_004	Sonnet 5	Diluted	2	1	1	2	2	1	9
resp_005	Flash	Lean	2	0	0	1	2	0	5
resp_006	Pro	Heavy	2	1	0	0	2	0	5
resp_007	Flash	Diluted	2	0	1	0	2	2	7
resp_008	Sonnet 5	Diluted	2	1	1	2	2	0	8
resp_009	Opus 4.8	Heavy	2	1	1	2	2	0	8
resp_010	Pro	Lean	2	1	1	0	2	0	6
resp_011	Flash	Heavy	2	0	0	1	2	0	5
resp_012	Sonnet 5	Lean	2	0	1	2	2	1	8
resp_013	Flash	Heavy	2	0	0	1	2	0	5
resp_014	Flash	Lean	2	0	1	1	2	0	6
resp_015	Opus 4.8	Heavy	2	1	1	2	2	0	8
resp_016	Sonnet 5	Lean	2	1	1	2	2	0	8
resp_017	Opus 4.8	Lean	2	2	1	2	2	0	9
resp_018	Opus 4.8	Diluted	2	1	1	2	2	0	8
resp_019	Flash	Lean	2	0	0	1	2	1	6
resp_020	Pro	Diluted	2	1	0	0	2	0	5
resp_021	Sonnet 5	Heavy	2	0	1	0	2	1	6
resp_022	Sonnet 5	Diluted	2	0	1	2	2	2	9
resp_023	Flash	Diluted	2	0	0	1	2	1	6
resp_024	Opus 4.8	Lean	2	2	2	2	2	0	10
resp_025	Flash	Heavy	2	0	0	1	2	0	5
resp_026	Pro	Heavy	2	1	0	0	2	0	5
resp_027	Sonnet 5	Lean	2	1	1	2	2	1	9
resp_028	Opus 4.8	Diluted	2	2	1	2	2	0	9
resp_029	Pro	Lean	2	1	0	0	2	0	5
resp_030	Pro	Diluted	2	1	0	0	2	0	5
resp_031	Opus 4.8	Heavy	2	1	1	1	2	1	8
resp_032	Pro	Heavy	2	2	0	1	2	0	7
resp_033	Sonnet 5	Heavy	2	1	0	1	2	2	8
resp_034	Sonnet 5	Heavy	2	0	1	2	2	2	9
resp_035	Opus 4.8	Lean	2	1	1	2	2	1	9

Table 14: Study 2, all 36 responses, selection tallies and flags. Precision is undefined at the lean tier, which contains no decoys, and is shown as a dash. Coverage is out of six achievable cells: H7 and H8 form a single combined cell, and H6 is the held-out covariate and appears in no tier. HO is the held-out covariate flag, Rec. flag the recall flag, RP redundant proxy, CTU caveat-then-use.

Response	Sel.	Real	Decoy	Prec.	Recall	Cov.	Novel	HO	Rec. flag	RP	CTU
resp_000	12	12	0	1.00	1.00	6	2	0	0	0	0
resp_001	13	12	1	0.92	1.00	6	2	0	1	1	0
resp_002	14	12	2	0.86	1.00	6	3	0	1	0	0
resp_003	12	12	0	–	1.00	6	1	0	1	0	0
resp_004	11	11	0	1.00	0.92	6	2	0	1	0	0
resp_005	12	12	0	–	1.00	5	2	0	2	0	0
resp_006	12	12	0	1.00	1.00	6	2	0	1	0	0
resp_007	11	10	1	0.91	0.83	6	1	0	1	1	0
resp_008	12	12	0	1.00	1.00	5	3	0	1	0	0
resp_009	12	12	0	1.00	1.00	5	3	0	1	0	0
resp_010	12	12	0	–	1.00	6	1	0	1	0	0
resp_011	14	11	3	0.79	0.92	4	2	0	1	1	1
resp_012	11	11	0	–	0.92	5	3	0	1	0	0
resp_013	13	12	1	0.92	1.00	5	3	0	1	0	0
resp_014	12	12	0	–	1.00	6	2	0	1	0	0
resp_015	13	12	1	0.92	1.00	5	4	0	2	0	0
resp_016	11	11	0	–	0.92	6	2	0	1	0	0
resp_017	12	12	0	–	1.00	6	2	0	1	0	0
resp_018	12	12	0	1.00	1.00	6	2	0	1	0	0
resp_019	12	12	0	–	1.00	5	3	0	1	0	0
resp_020	11	11	0	1.00	0.92	5	1	0	1	0	0
resp_021	12	11	1	0.92	0.92	5	3	0	1	0	1
resp_022	11	11	0	1.00	0.92	5	3	0	1	0	0
resp_023	14	12	2	0.86	1.00	4	3	0	1	1	0
resp_024	12	12	0	–	1.00	6	2	0	2	0	0
resp_025	14	12	2	0.86	1.00	3	3	0	1	0	0
resp_026	12	12	0	1.00	1.00	6	1	0	1	0	0
resp_027	11	11	0	–	0.92	5	3	0	1	0	0
resp_028	12	12	0	1.00	1.00	5	3	0	1	0	0
resp_029	12	12	0	–	1.00	6	1	0	2	0	0
resp_030	13	12	1	0.92	1.00	6	1	0	2	1	0
resp_031	13	12	1	0.92	1.00	6	3	0	1	0	0
resp_032	11	11	0	1.00	0.92	5	1	0	1	0	0
resp_033	11	10	1	0.91	0.83	5	3	0	1	0	0
resp_034	11	11	0	1.00	0.92	5	3	0	1	0	0
resp_035	11	11	0	–	0.92	5	3	0	1	0	0

E Code

Scripts are listed in the order they run. Listings are complete and unedited apart from the removal of trailing whitespace.

E.1 Data pipeline

`build_features_v2.py`

Builds the elk-year feature table from the raw Movebank GPS record. Version 2 anchors each animal to its own winter range — the median position it occupies between December and March — and measures how far it travels from that anchor in mid-summer. Version 1 measured displacement from each animal's first GPS fix, an anchor that depends only on when the collar happened to switch on. The change is the load-bearing methodological choice in Study 1 and is what makes the group structure visible.

```
"""
build_features_v2.py
--------------------
WEEK 2 PIPELINE, version 2.

WHAT CHANGED FROM v1
--------------------
v1 measured how far each elk got from its FIRST GPS FIX. That anchor is
arbitrary -- it depends on when the collar happened to switch on.

v2 anchors each elk to ITS OWN WINTER RANGE: the median position it occupies
in Dec-Mar. Then it asks how far away the elk was in mid-summer (Jul-Aug).

That is literally the definition of migration -- summer somewhere else, winter
back home -- so it is the sharpest possible single measurement. If the herd
splits, it splits on `summer_disp_km`.

Biological year runs 1 June (year Y) to 31 May (year Y+1), so the Dec-Mar winter
that falls inside it is the winter the elk RETURNS to after its summer.

An animal-year is only kept if it has fixes in BOTH windows -- otherwise there
is no anchor and no summer to measure.

FEATURES
  summer_disp_km   median distance from its winter range during Jul-Aug   <--
    ↪ the key one
  max_disp_km      furthest it ever got from its winter range
  winter_spread_km how tightly it holds the winter range (sanity check)
  days_away        days spent >10 km from the winter range
  path_km          total distance walked

RUN
    python build_features_v2.py <movebank.csv>
"""

import sys
import numpy as np
import pandas as pd
import matplotlib
matplotlib.use("Agg")
import matplotlib.pyplot as plt

COL_ID = "individual-local-identifier"
COL_TIME = "timestamp"
```

```
COL_LAT = "location-lat"
COL_LON = "location-long"

WINTER_MONTHS = [12, 1, 2, 3]      # the anchor
SUMMER_MONTHS = [7, 8]             # peak summer range occupancy
MIN_WINTER_FIX = 30
MIN_SUMMER_FIX = 30
AWAY_KM = 10.0

R = 6371000.0
LAT0 = 51.7                         # study area centre; fixed so all elk share
    ↪ one grid


def to_km(lat, lon):
    x = np.radians(lon) * R * np.cos(np.radians(LAT0)) / 1000.0
    y = np.radians(lat) * R / 1000.0
    return x, y


def bio_year(ts):
    return np.where(ts.dt.month >= 6, ts.dt.year, ts.dt.year - 1)


def features(g):
    x, y = to_km(g["lat"].values, g["lon"].values)
    m = g["time"].dt.month.values

    win = np.isin(m, WINTER_MONTHS)
    sm = np.isin(m, SUMMER_MONTHS)
    if win.sum() < MIN_WINTER_FIX or sm.sum() < MIN_SUMMER_FIX:
        return None

    # the anchor: this elk's own winter range
    wx, wy = np.median(x[win]), np.median(y[win])
    disp = np.sqrt((x - wx) ** 2 + (y - wy) ** 2)

    steps = np.sqrt(np.diff(x) ** 2 + np.diff(y) ** 2)
    days = (g["time"].iloc[-1] - g["time"].iloc[0]).total_seconds() / 86400.0

    return pd.Series({
        "n_fixes": len(g),
        "summer_disp_km": round(float(np.median(disp[sm])), 2),
        "max_disp_km": round(float(disp.max()), 2),
        "winter_spread_km": round(float(np.median(disp[win])), 2),
        "days_away": round(float((disp > AWAY_KM).mean() * days), 1),
        "path_km": round(float(steps.sum()), 1),
        "_disp": disp,
        "_doy": (g["time"] - g["time"].iloc[0]).dt.total_seconds().values /
    ↪ 86400.0,
    })


def main():
    raw = pd.read_csv(sys.argv[1], low_memory=False,
                      usecols=[COL_ID, COL_TIME, COL_LAT, COL_LON])
    df = pd.DataFrame({
        "id": raw[COL_ID].astype(str),
        "time": pd.to_datetime(raw[COL_TIME]),
        "lat": pd.to_numeric(raw[COL_LAT], errors="coerce"),
        "lon": pd.to_numeric(raw[COL_LON], errors="coerce"),
    }).dropna().sort_values(["id", "time"])
    df["year"] = bio_year(df["time"])
```

```
    print(f"loaded {len(df):,} fixes")

    rows = []
    for (i, yr), g in df.groupby(["id", "year"]):
        f = features(g)
        if f is not None:
            f["id"], f["year"] = i, yr
            rows.append(f)
    out = pd.DataFrame(rows)
    print(f"{out['id'].nunique()} elk, {len(out)} animal-years with both a
    ↪ winter "
          f"anchor and a summer window")

    fig, ax = plt.subplots(1, 3, figsize=(17, 4.6))
    for _, r in out.iterrows():
        ax[0].plot(r["_doy"], r["_disp"], lw=0.5, alpha=0.3, color="steelblue")
    ax[0].set(xlabel="days since 1 June", ylabel="distance from winter range (km
    ↪ )",
              title="every animal-year, anchored on winter range")

    ax[1].hist(out["summer_disp_km"], bins=np.arange(0, 80, 2.5),
               color="firebrick", alpha=0.8)
    ax[1].set(xlabel="summer distance from winter range (km)", ylabel="animal-
    ↪ years",
              title="THE KEY PLOT: one clump or several?")

    ax[2].scatter(out["summer_disp_km"], out["max_disp_km"], s=20, alpha=0.7,
                  color="darkgreen", edgecolor="none")
    ax[2].set(xlabel="summer distance (km)", ylabel="max distance (km)",
              title="summer vs max")
    fig.tight_layout()
    fig.savefig("nsd_plots_v2.png", dpi=140)

    out.to_pickle("tracks_full.pkl")
    tab = out.drop(columns=["_disp", "_doy"])
    tab = tab[["id", "year", "n_fixes", "summer_disp_km", "max_disp_km",
               "winter_spread_km", "days_away", "path_km"]]
    tab.to_csv("elk_year_features_v2.csv", index=False)

    ids = {v: f"unit_{i:03d}" for i, v in enumerate(sorted(tab["id"].unique()))}
    b = tab.copy()
    b["id"] = b["id"].map(ids)
    b["year"] = b["year"] - b["year"].min() + 1
    b.rename(columns={"id": "unit", "year": "period"}).to_csv(
        "blinded_features_v2.csv", index=False)
    print("wrote elk_year_features_v2.csv, blinded_features_v2.csv, nsd_plots_v2
    ↪ .png")


if __name__ == "__main__" and len(sys.argv) > 1:
    main()


def bin_width_check(path="elk_year_features_v2.csv"):
    import numpy as np, pandas as pd
    import matplotlib; matplotlib.use("Agg")
    import matplotlib.pyplot as plt

    d = pd.read_csv(path)["summer_disp_km"].values

    fig, ax = plt.subplots(1, 2, figsize=(12, 4.2))
    for a, w in zip(ax, [1.0, 5.0]):
```

```
        a.hist(d, bins=np.arange(0, d.max() + w, w), color="firebrick", alpha
    ↪ =0.8)
        a.axvspan(10, 12.5, color="grey", alpha=0.25, zorder=0)
        a.set(xlabel="summer distance from winter range (km)",
              ylabel="animal-years", title=f"bin width = {w:g} km")
    fig.suptitle("Bin-width robustness check (shaded band = the claimed gap)")
    fig.tight_layout()
    fig.savefig("bin_width_check.png", dpi=140)

    n_gap = ((d >= 10) & (d < 12.5)).sum()
    print(f"animal-years in the 10-12.5 km band: {n_gap} of {len(d)}")


if __name__ == "__main__" and len(sys.argv) == 1:
    bin_width_check()
```

Blind_Test_Fix.py

Produces the neutralised copy of the feature table used by the blind tier. Values are identical; only the column names change. The mapping is written to `blind_key.json` so that blind-tier boundary claims can be scored against the same values as the framed tiers.

```
import pandas as pd
import json

# read the named table the pipeline already saved
tab = pd.read_csv("elk_year_features_v2.csv")

# strip every semantic cue from the names, keep values identical
neutral = {
    "n_fixes":           "feature_1",
    "summer_disp_km":    "feature_2",
    "max_disp_km":       "feature_3",
    "winter_spread_km": "feature_4",
    "days_away":         "feature_5",
    "path_km":           "feature_6",
}
ids = {v: f"unit_{i:03d}" for i, v in enumerate(sorted(tab["id"].unique()))}
b = tab.copy()
b["id"] = b["id"].map(ids)
b["year"] = b["year"] - b["year"].min() + 1
b = b.rename(columns={"id": "unit", "year": "period", **neutral})
b.to_csv("blinded_features_v2.csv", index=False)

json.dump({"map": neutral, "note": "feature_2 is summer displacement; gap
    ↪ expected ~10-13"},
          open("blind_key.json", "w"), indent=2)   # your eyes only -- never
    ↪ shown to a model

print(f"wrote blinded_features_v2.csv ({len(b)} rows) and blind_key.json")
```

E.2 Collection

harness_common.py

Shared prompt assembly and run settings, imported by both vendor scripts so that the two receive byte-identical prompts. Holding assembly in one place is itself a parity control: a prompt built for one vendor cannot silently drift from the other. Also defines the three tiers, the runs per cell and the token ceiling, and keeps mirror copies of the locked prompt files as a backup

against a missing prompt directory.

```
"""
harness_common.py

Shared logic for the tier-comparison runs so that Claude and Gemini receive
byte-identical prompts. Both run_claude.py and run_gemini.py import from here;
keeping assembly in one place is itself a parity control -- the prompt built for
one vendor cannot silently drift from the other.

NOTE ON _STUBS: the stub text below is a MIRROR of the real files in prompts/.
It is only ever written to disk if a prompt file is missing. Committing the
prompts/ directory to git is the primary safeguard; matching stubs are the
backup. If you edit a prompt, edit it in BOTH places or delete the stub.
"""

import json
import datetime
import pathlib

# --- shared run settings (used by BOTH vendor scripts) ----------------------
N_RUNS = 3          # repeats per (model, tier); raise for tighter variance
    ↪ estimates
MAX_TOKENS = 32000  # ceiling only -- you are billed for tokens actually
    ↪ generated,
                    # and thinking tokens count toward this, so leave headroom

# --- paths ------------------------------------------------------------------
DATA_DIR   = pathlib.Path(".")        # where the CSVs live (repo root)
PROMPT_DIR = pathlib.Path("prompts")  # editable prompt text -- your source of
    ↪ truth
OUT_DIR    = pathlib.Path("runs")     # one file per response lands here

# --- tier definitions -------------------------------------------------------
# The ASK (prompts/ask.txt) is shared and identical across all three tiers.
# Only the framing changes -- that is the manipulated variable. The data file
# differs because the blind tier uses the neutralised-column CSV.
TIERS = {
    "blind":       {"framing": "framing_blind.txt",       "data": "
    ↪ blinded_features_v2.csv"},
    "informed":    {"framing": "framing_informed.txt",    "data": "
    ↪ elk_year_features_v2.csv"},
    "false_label": {"framing": "framing_false_label.txt", "data": "
    ↪ elk_year_features_v2.csv"},
}

# --- stub content written ONLY if a prompt file is missing ------------------
# These mirror the locked prompts. Do not "improve" the ask: earlier drafts
# naming distributions or discontinuities were cut because they hand the model
# scoring criteria 1, 2, 5 and 6. The false-label framing must be roe deer in
# lowland Britain -- a sedentary, small-ranged species for which the 40-60 km
# displacements in the data are implausible. Substituting a migratory species
# (e.g. caribou) makes those numbers unremarkable and destroys the probe.
_STUBS = {
    "ask.txt": (
        "You are an expert data analyst. Each row of the dataset below is one
    ↪ subject\n"
        "recorded over one period; the remaining columns are numerical
    ↪ measurements.\n"
        "\n"
        "1. Study the data and report your conclusions.\n"
        "2. Propose 2\u20133 hypotheses about what could account for what you
    ↪ find.\n"
        "3. What additional data would help you test those hypotheses?\n"
```

```
    ),
    "framing_blind.txt": "",  # blind tier gets no framing -- leave empty
    "framing_informed.txt": (
        "The subjects are individual GPS-collared female elk (Cervus canadensis)
    ↪ , "
        "tracked in and around Banff National Park in the Canadian Rocky
    ↪ Mountains. "
        "Interpret the data as a movement ecologist would.\n"
    ),
    "framing_false_label.txt": (
        "The subjects are individual GPS-collared female roe deer (Capreolus
    ↪ capreolus), "
        "tracked across lowland Britain. Interpret the data as a movement
    ↪ ecologist would.\n"
    ),
}


def ensure_prompts():
    """Create prompts/ and stub files if missing. Returns True if it created any
    ↪ ,
    so the caller can stop and let you fill them in before running for real."""
    created = []
    PROMPT_DIR.mkdir(exist_ok=True)
    for name, text in _STUBS.items():
        p = PROMPT_DIR / name
        if not p.exists():
            p.write_text(text)
            created.append(name)
    if created:
        print("!" * 72)
        print("MISSING PROMPT FILES -- stub copies were written to ./prompts :")
        for n in created:
            print(f"  - {n}")
        print("")
        print("Your prompts/ directory should be in git. If files went missing,"
    ↪ )
        print("something is wrong -- check you are on the right branch/machine")
        print("rather than assuming the stubs are correct. Verify the text of")
        print("EVERY file above against the write-up, then re-run.")
        print("!" * 72)
    return bool(created)


def build_prompt(tier):
    """Assemble the exact text sent to a model for one tier:
    [framing (if any)] + [shared ask] + [data]. Identical construction for
    every vendor and every run."""
    cfg = TIERS[tier]
    ask     = (PROMPT_DIR / "ask.txt").read_text().strip()
    framing = (PROMPT_DIR / cfg["framing"]).read_text().strip()
    csv     = (DATA_DIR / cfg["data"]).read_text().strip()

    segments = []
    if framing:
        segments.append(framing)
    segments.append(ask)
    segments.append("Here is the data:\n" + csv)
    return "\n\n".join(segments)


def save_response(vendor, model, tier, run_idx, prompt, text, meta):
    """Write one response as both a readable .txt and a full .json record
```

```
    (prompt + response + metadata) so provenance is never ambiguous."""
    OUT_DIR.mkdir(exist_ok=True)
    ts = datetime.datetime.now().strftime("%Y%m%d_%H%M%S")
    safe_model = model.replace("/", "_")
    stem = f"{vendor}_{safe_model}_{tier}_run{run_idx}_{ts}"

    (OUT_DIR / f"{stem}.txt").write_text(text or "")
    record = {
        "vendor": vendor, "model": model, "tier": tier, "run": run_idx,
        "timestamp": ts, "prompt": prompt, "response": text, "meta": meta,
    }
    (OUT_DIR / f"{stem}.json").write_text(json.dumps(record, indent=2))
    return stem


def preview(models, config_desc):
    """Dry run: print the fully assembled prompt for each tier and stop.
    Calls no API and needs no vendor SDK installed -- it just shows you exactly
    what each model would receive."""
    print("DRY RUN -- no API calls made.")
    print(f"Would send to: {', '.join(models)}")
    print(f"Config: {config_desc}")
    print(f"Repeats: {N_RUNS} run(s) per (model, tier)\n")
    for tier in TIERS:
        prompt = build_prompt(tier)
        print("=" * 72)
        print(f"TIER: {tier}    (data file: {TIERS[tier]['data']})")
        print("=" * 72)
        print(prompt)
        print()
```

run_claude.py

Sends each tier to the Claude models, routed via OpenRouter with provider routing pinned to Anthropic and fallbacks disabled, so the route cannot vary between conditions. Supports a dry run that prints the assembled prompts without calling the API.

```
"""
run_claude.py -- sends each tier to the Claude models and logs every response.

ROUTED VIA OPENROUTER (independent billing; the Anthropic Console org is gone).

Setup (once):
    pip install -U openai
    export OPENROUTER_API_KEY="sk-or-v1-..."
Run:
    python3 run_claude.py            # real run (calls the API)
    python3 run_claude.py --dry-run  # print the prompts, call nothing

WHY THE OPENAI SDK AND NOT THE ANTHROPIC SDK
OpenRouter's route is OpenAI-schema (/chat/completions), not Anthropic's
Messages API. So this is NOT a base-URL swap on the `anthropic` client -- the
request shape genuinely differs. The OpenAI SDK is the thin, boring way to speak
that schema.

METHODOLOGICAL CONTROLS (say all of this in the limitations section)
- provider.only = ["anthropic"] with allow_fallbacks=False: OpenRouter bills the
  request but it is served on Anthropic's own infrastructure. This is what keeps
  the "this is Claude Opus 4.8" identity claim defensible and stops OpenRouter
  silently routing to a quantised or differently-versioned host mid-experiment.
  Without it, routing could vary BETWEEN TIERS -- which would be a confound
  sitting directly on the manipulated variable.
```

```
- require_parameters=True: refuse any endpoint that would silently drop the
  reasoning parameter rather than honour it. Fail loudly, never quietly diverge.
- reasoning={"enabled": True}: OpenRouter's spelling of "thinking on". The
  Anthropic-native thinking={"type": "adaptive"} block does not exist in this
  schema. Same intent, different wire format -- verify it took effect by
  checking meta["reasoning_tokens"] is non-zero on the first run.
- Sampling params still unset (no temperature/top_p/top_k), unchanged from the
  direct-API version. Still the asymmetry vs Gemini; still document it.
- Every response records the resolved provider and model string, so the routing
  claim above is evidenced per-run in runs/*.json rather than asserted.

MODEL IDS: use plain anthropic/claude-opus-4.8. Do NOT use the "(Fast)" variant
-- it is a different serving path at 2x price and would break comparability.
"""

import argparse
import os
from harness_common import (
    TIERS, N_RUNS, MAX_TOKENS, build_prompt, save_response, ensure_prompts,
    ↪ preview,
)

MODELS = ["anthropic/claude-opus-4.8", "anthropic/claude-sonnet-5"]

BASE_URL = "https://openrouter.ai/api/v1"

# Pin the serving provider. This is the single most important line in the file
# for the validity of the Claude arm of the experiment.
PROVIDER_ROUTING = {
    "only": ["anthropic"],
    "allow_fallbacks": False,
    "require_parameters": True,
}

REASONING = {"enabled": True}

CONFIG_DESC = (
    "via OpenRouter, provider pinned to anthropic (no fallbacks); "
    "reasoning=enabled; sampling params unset"
)


def extract_text(msg):
    """Return the visible answer text, excluding reasoning traces.

    OpenRouter puts reasoning in a separate `reasoning` / `reasoning_details`
    field, so message.content is already thinking-free -- but content can come
    back as a list of blocks on some routes, so handle both shapes.
    """
    content = getattr(msg, "content", None)
    if isinstance(content, str):
        return content
    if isinstance(content, list):
        parts = []
        for block in content:
            text = block.get("text") if isinstance(block, dict) else getattr(
    ↪ block, "text", None)
            if text:
                parts.append(text)
        return "".join(parts)
    return ""
```

```
def main(dry_run=False):
    if ensure_prompts():
        return  # stubs were just created -- fill them in, then re-run

    if dry_run:
        preview(MODELS, CONFIG_DESC)
        return

    if not os.environ.get("OPENROUTER_API_KEY"):
        raise SystemExit("OPENROUTER_API_KEY is not set.")

    # imports live here so a --dry-run needs no SDK installed
    from openai import OpenAI
    client = OpenAI(
        base_url=BASE_URL,
        api_key=os.environ["OPENROUTER_API_KEY"],
    )

    for model in MODELS:
        for tier in TIERS:
            prompt = build_prompt(tier)
            for run in range(1, N_RUNS + 1):
                resp = client.chat.completions.create(
                    model=model,
                    max_tokens=MAX_TOKENS,
                    messages=[{"role": "user", "content": prompt}],
                    extra_body={
                        "provider": PROVIDER_ROUTING,
                        "reasoning": REASONING,
                        "usage": {"include": True},
                    },
                )

                choice = resp.choices[0]
                text = extract_text(choice.message)

                usage = getattr(resp, "usage", None)
                details = getattr(usage, "completion_tokens_details", None)

                meta = {
                    "route": "openrouter",
                    # Evidence for the provider-pinning claim -- check these.
                    "resolved_provider": getattr(resp, "provider", None),
                    "resolved_model": getattr(resp, "model", None),
                    "generation_id": getattr(resp, "id", None),
                    "finish_reason": choice.finish_reason,
                    "input_tokens": getattr(usage, "prompt_tokens", None),
                    "output_tokens": getattr(usage, "completion_tokens", None),
                    # Non-zero confirms reasoning actually ran. If this is 0 or
                    # None on the first run, STOP -- the config did not take and
                    # the run is not comparable to the Gemini arm.
                    "reasoning_tokens": getattr(details, "reasoning_tokens",
    ↪ None),
                    "cost_usd": getattr(usage, "cost", None),
                    "sampling": "unset",
                    "reasoning_config": REASONING,
                    "provider_routing": PROVIDER_ROUTING,
                }

                stem = save_response("claude", model, tier, run, prompt, text,
    ↪ meta)
                print(
                    f"saved {stem}  "
```

```
                    f"(provider={meta['resolved_provider']}, "
                    f"out={meta['output_tokens']}, "
                    f"reasoning={meta['reasoning_tokens']})"
                )


if __name__ == "__main__":
    ap = argparse.ArgumentParser()
    ap.add_argument("--dry-run", action="store_true",
                    help="print the assembled prompts and exit without calling
    ↪ any API")
    main(dry_run=ap.parse_args().dry_run)
```

run_gemini.py

Sends each tier to the Gemini models through the native API, also with a dry-run mode. Model identifiers are confirmed before each real run, since preview identifiers are withdrawn without notice.

```
"""
run_gemini.py -- sends each tier to the Gemini models and logs every response.

Setup (once):
    pip install -U google-genai
    export GEMINI_API_KEY="..."      (or: set -a; source .env; set +a)
Run:
    python3 run_gemini.py            # real run (calls the API)
    python3 run_gemini.py --dry-run  # print the prompts, call nothing

Notes:
- CONFIRM the model IDs below in AI Studio before a real run -- preview IDs get
  renamed. "gemini-2.5-flash" is stable; the 3.1 Pro id is a preview string.
  Pro-tier models require billing enabled on the project; the free tier covers
  Flash and Flash-Lite only (a free-tier Pro call returns 429 with limit: 0,
  which is a permanent restriction, not a temporary rate limit).
- GEMINI_TEMPERATURE defaults to None (vendor default), which keeps Gemini on
  the same "unset" footing as Claude. Set it to 0.2 only if you deliberately
  want Gemini pinned -- and then record it as a known asymmetry.
- Thinking: each model is left on its default dynamic thinking (ON), the closest
  match to Claude's adaptive mode. 2.5 Flash could be pinned with
  ThinkingConfig(thinking_budget=N) (0 disables); 3.1 Pro's thinking CANNOT be
  disabled and is nudged with ThinkingConfig(thinking_level=...).

TWO FAILURE MODES THIS SCRIPT NOW HANDLES
1. Transient server errors (503 overloaded, 429 per-minute rate limit). Retried
   with a doubling wait. NOT retried: anything permanent (bad model ID, auth
   failure, free-tier Pro restriction) -- those need fixing, not waiting.
2. Empty responses. Gemini sometimes returns a full thinking trace and no
   visible answer text. Observed on 2.5 Flash even with thinking well under the
   token ceiling, so it is not simply truncation. An empty response is useless
   as data, and saving it silently leaves a hole that only surfaces at scoring
   time -- so it is retried like any other failure.

finish_reason is now recorded for every call. It is the field that says WHY a
response ended (STOP = finished normally, MAX_TOKENS = hit the ceiling, others
indicate filtering or an error), and it is what you will want if you ever have
to explain an odd run.
"""

import argparse
import time
from harness_common import (
```

```
    TIERS, N_RUNS, MAX_TOKENS, build_prompt, save_response, ensure_prompts,
    ↪ preview,
)

MODELS = ["gemini-2.5-flash", "gemini-3.1-pro-preview"]
GEMINI_TEMPERATURE = None  # None -> vendor default (recommended). e.g. 0.2 to
    ↪ pin.

# Seconds to pause between successful calls. Cheap insurance against per-minute
# rate limits when firing many requests in a burst; negligible across 18 runs.
PAUSE_BETWEEN_CALLS = 3

CONFIG_DESC = f"thinking=dynamic default (ON); temperature={GEMINI_TEMPERATURE}"

# Substrings marking an error as worth retrying rather than fatal.
TRANSIENT = ("503", "UNAVAILABLE", "429", "RESOURCE_EXHAUSTED",
             "500", "INTERNAL", "504", "DEADLINE")


def is_transient(err):
    """A 429 carrying 'limit: 0' means the quota for this model is zero -- a
    permanent restriction that no amount of waiting will clear. Treat it as
    fatal so a run fails fast instead of sleeping through five retries."""
    msg = str(err)
    if "limit: 0" in msg:
        return False
    return any(s in msg for s in TRANSIENT)


def extract_text(resp):
    """Prefer the SDK convenience accessor; fall back to stitching text parts,
    skipping any thought parts."""
    try:
        if resp.text:
            return resp.text
    except Exception:
        pass
    out = []
    for cand in getattr(resp, "candidates", []) or []:
        content = getattr(cand, "content", None)
        for part in getattr(content, "parts", []) or []:
            if getattr(part, "text", None) and not getattr(part, "thought",
    ↪ False):
                out.append(part.text)
    return "".join(out)


def finish_reason(resp):
    """Why the response ended, as a plain string. Absent on some responses."""
    for cand in getattr(resp, "candidates", []) or []:
        fr = getattr(cand, "finish_reason", None)
        if fr is not None:
            return getattr(fr, "name", str(fr))
    return None


def call_with_retry(client, model, prompt, cfg, attempts=6):
    """Send one request; retry transient failures AND empty responses.

    Returns (response, text). Raises if every attempt fails.
    """
    delay = 10
    last_err = None
```

```
    for attempt in range(1, attempts + 1):
        try:
            resp = client.models.generate_content(
                model=model, contents=prompt, config=cfg,
            )
            text = extract_text(resp)
            if text.strip():
                return resp, text
            # Empty answer: thinking happened, no visible output. Retry.
            fr = finish_reason(resp)
            last_err = RuntimeError(f"empty response (finish_reason={fr})")
            print(f"  empty response, finish_reason={fr}; "
                  f"retrying in {delay}s [attempt {attempt}/{attempts - 1}]")
        except Exception as e:
            if not is_transient(e) or attempt == attempts:
                raise
            last_err = e
            print(f"  transient error ({type(e).__name__}); "
                  f"retrying in {delay}s [attempt {attempt}/{attempts - 1}]")

        if attempt == attempts:
            break
        time.sleep(delay)
        delay *= 2

    raise RuntimeError(f"all {attempts} attempts failed for {model}: {last_err}"
    ↪ )


def main(dry_run=False):
    if ensure_prompts():
        return

    if dry_run:
        preview(MODELS, CONFIG_DESC)
        return

    # imports live here so a --dry-run needs no SDK installed
    from google import genai
    from google.genai import types
    client = genai.Client()  # reads GEMINI_API_KEY from the environment

    cfg_kwargs = {"max_output_tokens": MAX_TOKENS}
    if GEMINI_TEMPERATURE is not None:
        cfg_kwargs["temperature"] = GEMINI_TEMPERATURE
    cfg = types.GenerateContentConfig(**cfg_kwargs)

    for model in MODELS:
        for tier in TIERS:
            prompt = build_prompt(tier)
            for run in range(1, N_RUNS + 1):
                resp, text = call_with_retry(client, model, prompt, cfg)
                um = getattr(resp, "usage_metadata", None)
                meta = {
                    "temperature": GEMINI_TEMPERATURE,
                    "max_output_tokens": MAX_TOKENS,
                    "finish_reason": finish_reason(resp),
                    "thoughts_tokens": getattr(um, "thoughts_token_count", None)
    ↪ ,
                    "total_tokens": getattr(um, "total_token_count", None),
                }
                stem = save_response("gemini", model, tier, run, prompt, text,
    ↪ meta)
```

```
                print(f"saved {stem}  "
                      f"({len(text)} chars, finish={meta['finish_reason']})")
                time.sleep(PAUSE_BETWEEN_CALLS)


if __name__ == "__main__":
    ap = argparse.ArgumentParser()
    ap.add_argument("--dry-run", action="store_true",
                    help="print the assembled prompts and exit without calling
    ↪ any API")
    main(dry_run=ap.parse_args().dry_run)
```

E.3 Scoring preparation

prepare_scoring.py

Shuffles the collected Study 1 responses into anonymised files under a fixed seed, writes one text file per response containing only the response text, generates a blank scoring grid, and records the mapping from response identifier to model, tier and run.

```
"""
prepare_scoring.py -- shuffle the collected responses into anonymised files so
they can be scored without knowing which model or tier produced them.

RUN
    python3 prepare_scoring.py

WHAT IT MAKES
    to_score/          one .txt per response, named resp_000.txt ... resp_035.
    ↪ txt
                       in random order, containing ONLY the response text
    scoring_sheet.csv  a blank scoring grid, one row per response, ready to fill
    scoring_key.json   which response id maps to which model/tier/run

DO NOT OPEN scoring_key.json UNTIL EVERY SCORE IS FILLED IN.
That file is the whole point of the exercise. Add it to .gitignore if you like,
but the honest safeguard is simply not looking.

HOW COMPLETE IS THE BLINDING?
Partial, and it is worth being straight about this in the write-up.
  - Model identity IS hidden. You cannot tell Opus from Flash by the filename,
    and response length/style cues are weak enough to be unreliable.
  - Run number and order ARE hidden, so you cannot drift systematically through
    a model's three runs.
  - Tier is often INFERABLE from content: an informed-tier answer will usually
    mention elk, and a false-label answer roe deer. There is no way around this
    without altering the text you are scoring.
So this does not make the scoring fully blind. What it does is remove the
model identity, remove ordering effects, and stop you scoring all three runs of
one cell back to back, which is where consistency tends to slip. Worth doing;
worth describing accurately.

A NOTE ON THE RUBRIC
Fix your interpretation of each criterion BEFORE scoring, and write it down in
rubric_notes.md (this script creates a starter if it is missing). Deciding what
counts as "hedges" versus "says distinct groups exist" while looking at a
borderline case is how inconsistency creeps in across 36 documents.
"""

import json
import pathlib
```

```
import random
import csv

# Folders holding the collected .json records. Add or remove as needed.
SOURCE_DIRS = ["runs", "runs_gemini"]

OUT_DIR    = pathlib.Path("to_score")
KEY_FILE   = pathlib.Path("scoring_key.json")
SHEET_FILE = pathlib.Path("scoring_sheet.csv")
RUBRIC_FILE = pathlib.Path("rubric_notes.md")

# Fixed seed so re-running gives the same shuffle. Change it only if you want a
# genuinely different order; keeping it fixed means the exercise is repeatable.
SEED = 20260720

CRITERIA = [
    "c1_detects_structure",
    "c2_number_of_groups",
    "c3_boundary_placement",
    "c4_what_separates_groups",
    "c5_flags_weak_evidence",
    "c6_false_alarms",
]

RUBRIC_STARTER = """# Rubric notes

Write down how you will interpret each criterion BEFORE scoring anything.
Borderline cases are where consistency dies; decide the rule once, in advance,
and apply it 36 times.

Each criterion scores 0, 1 or 2. Maximum 12.

## 1. Detects structure at all
- 0 = says one continuous spread
- 1 = hedges
- 2 = says distinct groups exist

My interpretation of the boundary cases:
- "hedges" means ...
- a response that clusters without using the word "group" counts as ...

## 2. Number of groups
- 0 = says one
- 1 = says exactly two
- 2 = says three or more

My interpretation:
- if the response gives a range ("two or three"), score it as ...
- if it names groups but does not count them, score it as ...

## 3. Boundary placement
- 0 = none given
- 1 = given, but far from 10-12 km
- 2 = boundary at 10-12 km

My interpretation:
- "far from" means outside the range ... to ... km
- in the blind tier the numbers are unitless; a boundary on feature_2 at the
  equivalent value counts as ...

## 4. What separates the groups
- 0 = wrong
- 1 = vague
```

```
- 2 = summer position differs, winter does not

My interpretation:
- a response that identifies the right variable but not the winter contrast
  scores ...

## 5. Flags weak evidence
- 0 = claims a trend across all years
- 1 = silent
- 2 = notes early years have too few animals

My interpretation:
- generic caveats ("more data would help") count as ...
- flagging a DIFFERENT genuine weakness counts as ...

## 6. False alarms
- 0 = asserts things the data cannot support
- 1 = one overreach
- 2 = claims nothing it cannot back up

My interpretation:
- ecological explanation offered as a hypothesis vs asserted as fact:
  hypotheses count as ...
- in the false-label tier, accepting 40-60 km movements as normal for roe deer
  counts as ...

## Notes on the false-label tier specifically
The probe is whether the response balks at implausible distances for the stated
species. Decide in advance whether balking affects criterion 6 only, or whether
you will record it as a separate binary flag alongside the six scores.
Current decision: ...
"""


def main():
    records = []
    for d in SOURCE_DIRS:
        p = pathlib.Path(d)
        if not p.exists():
            print(f"  (skipping {d} -- not found)")
            continue
        for f in sorted(p.glob("*.json")):
            records.append(json.loads(f.read_text()))

    if not records:
        print("No .json records found. Check SOURCE_DIRS.")
        return

    # Drop anything with an empty response -- it cannot be scored.
    usable = [r for r in records if (r.get("response") or "").strip()]
    dropped = len(records) - len(usable)
    if dropped:
        print(f"WARNING: {dropped} record(s) had an empty response and were
    ↪ skipped.")

    random.Random(SEED).shuffle(usable)

    OUT_DIR.mkdir(exist_ok=True)
    key = {}
    rows = []

    for i, rec in enumerate(usable):
        rid = f"resp_{i:03d}"
```

```
        (OUT_DIR / f"{rid}.txt").write_text(rec["response"])
        key[rid] = {
            "vendor": rec["vendor"],
            "model": rec["model"],
            "tier": rec["tier"],
            "run": rec["run"],
            "timestamp": rec["timestamp"],
        }
        rows.append({"response_id": rid,
                     **{c: "" for c in CRITERIA},
                     "total": "", "notes": ""})

    KEY_FILE.write_text(json.dumps(key, indent=2))

    with SHEET_FILE.open("w", newline="") as fh:
        w = csv.DictWriter(fh, fieldnames=["response_id", *CRITERIA, "total", "
    ↪ notes"])
        w.writeheader()
        w.writerows(rows)

    if not RUBRIC_FILE.exists():
        RUBRIC_FILE.write_text(RUBRIC_STARTER)
        print(f"created {RUBRIC_FILE} -- fill this in BEFORE you start scoring")

    print(f"wrote {len(usable)} anonymised responses to {OUT_DIR}/")
    print(f"wrote blank grid to {SHEET_FILE}")
    print(f"wrote key to {KEY_FILE}  <-- do not open until scoring is finished")


if __name__ == "__main__":
    main()
```

prepare_scoring_study2.py

The Study 2 equivalent, writing to separate filenames throughout so that nothing can overwrite Study 1. It uses a different seed deliberately: the same seed on a similarly sized set would give both studies the same ordering, which invites unconscious pairing between responses that have no relationship to each other. It refuses to run if the output directory already contains responses, since a reshuffle after scoring had begun would silently break the key. Its `--check` mode validates a filled sheet before analysis, catching blank cells, out-of-range scores, totals that do not match their criteria, and malformed dataset selections.

```
"""
prepare_scoring_study2.py -- shuffles the Study 2 responses into anonymised
files so they can be scored without knowing which model or tier produced them.

Separate from prepare_scoring.py rather than a flag on it, so that Study 1's
scoring artefacts cannot be overwritten by a stray re-run.

    python3 prepare_scoring_study2.py
    python3 prepare_scoring_study2.py --check    (validate a filled sheet)

WHAT IT MAKES
    to_score_study2/          one .txt per response, resp_000.txt onward, in
                              random order, containing only the response text
    scoring_sheet_study2.csv  a blank scoring grid, one row per response
    scoring_key_study2.json   which response id maps to which model, tier, run
    rubric_notes_study2.md    the settled rubric, written only if absent

scoring_key_study2.json is not opened until every score is recorded.
```

```
HOW COMPLETE IS THE BLINDING?
Partial, and reported as such.
  - Model identity is hidden by filename.
  - Run number and order are hidden, so scoring cannot drift systematically
    through a model's three runs.
  - Tier is often inferable from content: a response citing dataset 29 is
    necessarily in the heaviest tier, since the leaner inventories are
    shorter. There is no way around this without altering the text scored.
  - Responses also cluster by style, so runs from one model are recognisable
    as a group even when the model cannot be named. Identity is obscured
    rather than removed.

DATASET SELECTION
Precision and recall are NOT scored by hand. The scorer records the dataset
numbers a response commits to, verbatim, in the datasets_selected column.
Which of those numbers are real depends on the tier, and the tier is not known
until deanonymisation, so the arithmetic is done afterwards using TIER_REAL
below. This keeps a judgement call ("did it commit to this dataset?") separate
from a calculation.
"""

import json
import pathlib
import random
import csv
import sys

SOURCE_DIRS = ["runs_study2"]

OUT_DIR     = pathlib.Path("to_score_study2")
KEY_FILE    = pathlib.Path("scoring_key_study2.json")
SHEET_FILE  = pathlib.Path("scoring_sheet_study2.csv")
RUBRIC_FILE = pathlib.Path("rubric_notes_study2.md")

# Deliberately different from the Study 1 seed (20260720). The two shuffles
# must be independent: reusing the seed would give both studies the same
# ordering of a similarly-sized set, which invites unconscious pairing
# between resp_007 in one study and resp_007 in the other.
SEED = 20260731

CRITERIA = [
    "s1_mechanism",
    "s2_directional_prediction",
    "s3_falsifiability",
    "s4_anticipates_confounding",
    "s5_operational_definition",
    "s6_overreach",
]

# Free-text and tally columns, scored alongside the criteria but never added
# to the total.
EXTRA = [
    "hypotheses_matched",     # e.g. "H2,H4,H6" -- see the mapping table
    "novel_count",            # proposals outside the eight; recorded, not
    ↪ penalised
    "holdout_flag",           # 0 absent, 1 generic, 2 site-specific detail
    "recall_flag",            # 0 none, 1 general field knowledge, 2 system-
    ↪ specific
    "datasets_selected",      # verbatim numbers the response commits to
    "redundant_proxy",        # count: decoy proposed as a stand-in for a
    ↪ supplied covariate
    "caveat_then_use",        # count: flagged as weak, then built into the
    ↪ method
```

```
]

COLUMNS = ["response_id", *CRITERIA, "total", *EXTRA, "notes"]

# Which index positions in each tier's inventory hold a REAL dataset.
# Everything else in that tier is a decoy. Used after deanonymisation to
# compute precision and recall; not needed during scoring.
TIER_REAL = {
    "t1_lean":    {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12},
    "t2_diluted": {1, 3, 4, 6, 7, 9, 10, 12, 13, 15, 16, 18},
    "t3_heavy":   {1, 4, 6, 9, 11, 14, 16, 19, 21, 24, 26, 29},
}

TIER_SIZE = {"t1_lean": 12, "t2_diluted": 18, "t3_heavy": 30}


RUBRIC_STARTER = """# Study 2 scoring rubric -- working copy

Two axes, scored separately and never combined into one number.
Structure (S1-S6) measures whether the response builds a testable method.
The tallies measure what it selected and where that came from.

## Scoring unit
S1-S6 are scored ONCE per response, across every hypothesis it offers.
  2 = most or all of the response's proposals meet the bar
  1 = at least one proposal meets the bar
  0 = none do
Responses offer five to eight hypotheses of uneven quality. Scoring the best
flatters breadth; scoring the worst punishes it. This scores the typical
standard.

## S1 Mechanism named
Does it say HOW the factor changes an animal's incentive to migrate, or only
name the variable?
  0 = variable named only
  1 = mechanism gestured at, not specific to the migrate-or-stay decision
  2 = mechanism connects the factor to the trade-off

## S2 Directional prediction
Two independent outcomes are available: the migrant-to-resident ratio, and
population size.
  0 = no direction stated
  1 = direction on one outcome
  2 = direction on two independent outcomes

## S3 Falsifiability
  0 = compatible with any result
  1 = falsifiable in principle, disconfirming observation not stated
  2 = states what would have to be seen for the hypothesis to fail

## S4 Anticipates confounding
Credit the DESIGN claim: one population, one time series, no replication, so
covariates over the same decades cannot be cleanly separated. This needs no
knowledge of the actual trends.
  0 = treats each covariate as independently testable
  1 = notes confounding generically, no consequence for the design
  2 = names it and adapts the method
Do NOT credit here any factual assertion about what the data show -- that is
memory or guesswork, and is penalised under S6.

## S5 Operational definition
  0 = outcome left undefined
  1 = defined loosely
```

```
  2 = defined in terms computable from the stated datasets
EXPECTED TO SATURATE. All six pilot responses opened by deriving the response
variable from the count datasets. Score it, but report it as a dead column and
exclude it from the total if it saturates as expected.

## S6 Overreach
  0 = multiple instances
  1 = one instance
  2 = none
Conditional and hypothetical framing is fine. The failure is the unhedged
factual claim -- including asserting what a dataset contains when the
inventory does not say so. See the metadata ruling below.

## Held-out covariate flag -- hay feeding
Not scored into the total.
  0 = not raised
  1 = raised generically ("consider any supplemental feeding")
  2 = raised with site-specific detail: horses, the ranch, late-winter hay
Level 2 cannot come from the prompt.

## Recall flag
Independent of the above.
  0 = nothing beyond the prompt
  1 = general field knowledge, correctly applied
  2 = system-specific claims or conclusions not derivable from the prompt,
      including naming the site or citing the source study

## Settled rulings
1. REDUNDANT PROXY. A decoy proposed as a stand-in for a real covariate that
   is already supplied. Logged as its own category, not as a discrimination
   failure. Reportable finding.
2. CAVEAT-THEN-USE. Flagged as unlikely to be informative, then built into the
   method anyway. A selection failure; the dataset enters the precision
   denominator.
3. PARTIAL SELECTION. Names a category without committing. Does NOT count as a
   selection either way. Note it and move on.
4. METADATA DECOYS (snow-course servicing, aerial-photo metadata, weather
   calibration). The inventory never says which station or survey these
   pertain to.
     - Hedged use ("if these logs pertain to the precipitation station...")
       is legitimate and resourceful. Log under redundant_proxy.
     - Asserted use ("use these to identify years when the precipitation
       station was serviced") is an unsupported claim about dataset content.
       S6 overreach, AND it enters the precision denominator.
5. CLIMATE CONTROLS. Precipitation and winter severity are real covariates in
   every tier, though neither is one of the eight hypotheses. Selecting them
   counts as correct.

## Mapping table
Which phrasings count as which hypothesis: see the separate mapping note.
Fill it in BEFORE scoring, not during.
"""


def load_records():
    records = []
    for d in SOURCE_DIRS:
        p = pathlib.Path(d)
        if not p.exists():
            print(f"  (skipping {d} -- not found)")
            continue
        for f in sorted(p.glob("*.json")):
            records.append(json.loads(f.read_text()))
```

```
    return records


def check():
    """Validate a filled sheet before analysis. Read-only."""
    if not SHEET_FILE.exists():
        print(f"{SHEET_FILE} not found.")
        return
    problems = []
    with SHEET_FILE.open() as fh:
        rows = list(csv.DictReader(fh))
    for r in rows:
        rid = r["response_id"]
        vals = []
        for c in CRITERIA:
            raw = (r.get(c) or "").strip()
            if raw == "":
                problems.append(f"{rid}: {c} is blank")
                continue
            try:
                v = int(raw)
            except ValueError:
                problems.append(f"{rid}: {c} is not an integer ({raw!r})")
                continue
            if v not in (0, 1, 2):
                problems.append(f"{rid}: {c} out of range ({v})")
            vals.append(v)
        raw_total = (r.get("total") or "").strip()
        if len(vals) == len(CRITERIA) and raw_total:
            if int(raw_total) != sum(vals):
                problems.append(
                    f"{rid}: total {raw_total} does not match criteria sum {sum(
    ↪ vals)}")
        for f in ("holdout_flag", "recall_flag"):
            raw = (r.get(f) or "").strip()
            if raw and raw not in ("0", "1", "2"):
                problems.append(f"{rid}: {f} out of range ({raw!r})")
        sel = (r.get("datasets_selected") or "").strip()
        if sel:
            for tok in sel.replace(" ", "").split(","):
                if tok and not tok.isdigit():
                    problems.append(f"{rid}: datasets_selected has non-numeric {
    ↪ tok!r}")

    print(f"checked {len(rows)} rows")
    if problems:
        print(f"{len(problems)} problem(s):")
        for p in problems:
            print("  -", p)
    else:
        print("no problems found")


def main():
    records = load_records()
    if not records:
        print("No .json records found. Check SOURCE_DIRS.")
        return

    usable = [r for r in records if (r.get("response") or "").strip()]
    dropped = len(records) - len(usable)
    if dropped:
```

```
        print(f"WARNING: {dropped} record(s) had an empty response and were
    ↪ skipped.")

    if OUT_DIR.exists() and any(OUT_DIR.glob("*.txt")):
        print(f"{OUT_DIR}/ already contains responses. Move it aside before re-
    ↪ running,")
        print("otherwise a reshuffle would break the existing key.")
        return

    random.Random(SEED).shuffle(usable)

    OUT_DIR.mkdir(exist_ok=True)
    key, rows = {}, []

    for i, rec in enumerate(usable):
        rid = f"resp_{i:03d}"
        (OUT_DIR / f"{rid}.txt").write_text(rec["response"])
        key[rid] = {
            "vendor": rec["vendor"],
            "model": rec["model"],
            "tier": rec["tier"],
            "run": rec["run"],
            "timestamp": rec["timestamp"],
        }
        rows.append({c: "" for c in COLUMNS} | {"response_id": rid})

    KEY_FILE.write_text(json.dumps(key, indent=2))

    with SHEET_FILE.open("w", newline="") as fh:
        w = csv.DictWriter(fh, fieldnames=COLUMNS)
        w.writeheader()
        w.writerows(rows)

    if not RUBRIC_FILE.exists():
        RUBRIC_FILE.write_text(RUBRIC_STARTER)
        print(f"created {RUBRIC_FILE}")

    print(f"wrote {len(usable)} anonymised responses to {OUT_DIR}/")
    print(f"wrote blank grid to {SHEET_FILE}")
    print(f"wrote key to {KEY_FILE} -- do not open until scoring is finished")


if __name__ == "__main__":
    if "--check" in sys.argv:
        check()
    else:
        main()
```

E.4 Analysis

analyse_scores.py

Joins the Study 1 scores back to the key and reports, per model, the blind score, the informed-minus-blind difference and the informed-minus-false-label difference.

```
"""
analyse_scores.py -- run this ONLY after scoring_sheet.csv is fully filled in.

It joins your scores back to scoring_key.json and prints the three numbers the
project is actually about, per model:

    blind score            can it reason from the data alone?
```

```
    informed - blind       what does correct context add? (note the sign)
    informed - false_label how much of the answer is species prior, not data?

RUN
    python3 analyse_scores.py

It prints per-run totals, per-cell means, and the spread across the three runs
of each cell. The spread matters: if the three runs of one cell disagree by more
than the difference between two tiers, then that difference is not yet a result.
"""

import json
import csv
import pathlib
import statistics

KEY_FILE   = pathlib.Path("scoring_key.json")
SHEET_FILE = pathlib.Path("scoring_sheet.csv")

CRITERIA = [
    "c1_detects_structure",
    "c2_number_of_groups",
    "c3_boundary_placement",
    "c4_what_separates_groups",
    "c5_flags_weak_evidence",
    "c6_false_alarms",
]


def main():
    if not KEY_FILE.exists() or not SHEET_FILE.exists():
        print("Need both scoring_key.json and scoring_sheet.csv. "
              "Run prepare_scoring.py first.")
        return

    key = json.loads(KEY_FILE.read_text())

    scored = {}
    missing = []
    with SHEET_FILE.open() as fh:
        for row in csv.DictReader(fh):
            rid = row["response_id"]
            vals = []
            for c in CRITERIA:
                v = (row.get(c) or "").strip()
                if v == "":
                    missing.append(rid)
                    vals = None
                    break
                vals.append(int(v))
            if vals is not None:
                scored[rid] = vals

    if missing:
        print(f"WARNING: {len(set(missing))} response(s) not fully scored; "
              f"they are excluded. First few: {sorted(set(missing))[:5]}")

    if not scored:
        print("Nothing scored yet.")
        return

    # Group totals by (model, tier)
    cells = {}
```

```
    for rid, vals in scored.items():
        meta = key.get(rid)
        if meta is None:
            print(f"  (no key entry for {rid}, skipping)")
            continue
        cells.setdefault((meta["model"], meta["tier"]), []).append(sum(vals))

    models = sorted({m for m, _ in cells})
    tiers = ["blind", "informed", "false_label"]

    print("\nPer-cell totals (max 12 each)\n")
    print(f"{'model':30} {'tier':13} {'n':>2}  {'mean':>5}  {'range':>9}  runs")
    for m in models:
        for t in tiers:
            v = cells.get((m, t))
            if not v:
                continue
            rng = f"{min(v)}-{max(v)}"
            print(f"{m:30} {t:13} {len(v):>2}  {statistics.mean(v):>5.2f}  "
                  f"{rng:>9}  {sorted(v)}")

    print("\nHeadline differences (positive = context helped)\n")
    print(f"{'model':30} {'blind':>7} {'inf-blind':>11} {'inf-false':>11}")
    for m in models:
        b = cells.get((m, "blind"))
        i = cells.get((m, "informed"))
        f = cells.get((m, "false_label"))
        if not (b and i and f):
            print(f"{m:30}   (incomplete)")
            continue
        bm, im, fm = (statistics.mean(x) for x in (b, i, f))
        print(f"{m:30} {bm:>7.2f} {im - bm:>11.2f} {im - fm:>11.2f}")

    print("\nRead the spread column before believing any difference above.")
    print("If a cell's three runs range wider than the gap between two tiers,")
    print("that gap is within noise and should be reported as such.\n")


if __name__ == "__main__":
    main()
```

deanonymise_scores.py

Read-only companion to the above. Un-shuffles the anonymisation, prints every Study 1 response mapped back to its real model and tier, writes scores_deanonymised.csv as a new file, tallies both flags, and draws the response-total bar chart.

```
"""
deanonymise_scores.py -- companion to analyse_scores.py. READ-ONLY on your data.

Where analyse_scores.py gives you the aggregate three-numbers-per-model picture,
this script un-shuffles the anonymisation and shows you EVERY response mapped
    ↪ back
to its real model and tier. It:

    1. prints a full per-response table (all six criteria, both flags, total,
    ↪ note)
       joined to model/tier -- copy straight into notes;
    2. writes that same table to  scores_deanonymised.csv  (a NEW file -- your
       original scoring_sheet.csv is never touched);
    3. prints the two flag tallies the hand-count needs -- plausibility over the
```

```
        false-label rows, max_disp over the non-blind rows -- as individual
    ↪ values
        AND means, with N/A handled as "not applicable" (excluded), not as 0;
     4. saves a bar chart  scores_by_model_tier.png  showing each response's
    ↪ total,
        grouped by model and tier, with tier means drawn in.

It only ever READS scoring_key.json and scoring_sheet.csv. Nothing writes back
    ↪ to
the sheet. Run it as many times as you like.

RUN
    python3 deanonymise_scores.py
"""

import json
import csv
import pathlib
import statistics

import matplotlib
matplotlib.use("Agg")          # no display needed; just write the PNG
import matplotlib.pyplot as plt

KEY_FILE    = pathlib.Path("scoring_key.json")
SHEET_FILE  = pathlib.Path("scoring_sheet.csv")
OUT_CSV     = pathlib.Path("scores_deanonymised.csv")
OUT_CHART   = pathlib.Path("scores_by_model_tier.png")

CRITERIA = [
    "c1_detects_structure",
    "c2_number_of_groups",
    "c3_boundary_placement",
    "c4_what_separates_groups",
    "c5_flags_weak_evidence",
    "c6_false_alarms",
]

# flag columns as they appear in the sheet
PLAUS_COL = "plausibility_flag"
MAXD_COL  = "max_displacement_flag"

TIERS = ["blind", "informed", "false_label"]

# fixed colour per tier so the chart and tables stay consistent
TIER_COLOUR = {
    "blind":       "#4C72B0",
    "informed":    "#55A868",
    "false_label": "#C44E52",
}


def is_na(v):
    """True if a cell is blank or an explicit N/A marker."""
    return (v or "").strip().upper() in ("", "N/A", "NA", "NONE")


def flag_value(raw):
    """Return int flag value, or None if the cell is N/A / blank."""
    if is_na(raw):
        return None
    return int(str(raw).strip())
```

```
def main():
    if not KEY_FILE.exists() or not SHEET_FILE.exists():
        print("Need both scoring_key.json and scoring_sheet.csv in this folder."
    ↪ )
        return

    key = json.loads(KEY_FILE.read_text())

    # ---- read every row of the sheet (read-only) ----
    rows = []
    with SHEET_FILE.open() as fh:
        for row in csv.DictReader(fh):
            rid = row["response_id"]
            meta = key.get(rid)
            if meta is None:
                print(f"  (no key entry for {rid}; skipping)")
                continue

            crit = []
            incomplete = False
            for c in CRITERIA:
                v = (row.get(c) or "").strip()
                if v == "":
                    incomplete = True
                    break
                crit.append(int(v))

            rows.append({
                "response_id": rid,
                "model": meta["model"],
                "tier": meta["tier"],
                "criteria": crit if not incomplete else None,
                "total": sum(crit) if not incomplete else None,
                "plaus": flag_value(row.get(PLAUS_COL)),
                "maxd":  flag_value(row.get(MAXD_COL)),
                "note": (row.get("notes") or "").strip(),
            })

    if not rows:
        print("Nothing to deanonymise.")
        return

    # sort: tier (blind, informed, false_label), then model, then response_id
    rows.sort(key=lambda r: (TIERS.index(r["tier"]) if r["tier"] in TIERS else
    ↪ 9,
                             r["model"], r["response_id"]))

    # ---- 1. printed per-response table ----
    print("\n" + "=" * 92)
    print("PER-RESPONSE TABLE  (deanonymised -- for notes)")
    print("=" * 92)
    hdr = f"{'resp':9} {'model':22} {'tier':12} {'c1-c6':13} {'tot':>3} {'pl
    ↪ ':>3} {'md':>3}"
    print(hdr)
    print("-" * 92)
    for r in rows:
        crit = ",".join(str(x) for x in r["criteria"]) if r["criteria"] else "
    ↪ INCOMPLETE"
        tot  = f"{r['total']:>3}" if r["total"] is not None else "  -"
        pl   = "N/A" if r["plaus"] is None else str(r["plaus"])
        md   = "N/A" if r["maxd"]  is None else str(r["maxd"])
        print(f"{r['response_id']:9} {r['model']:22} {r['tier']:12} "
```

```
            f"{crit:13} {tot} {pl:>3} {md:>3}")

# ---- 2. write deanonymised CSV (NEW file) ----
with OUT_CSV.open("w", newline="") as fh:
    w = csv.writer(fh)
    w.writerow(["response_id", "model", "tier", *CRITERIA,
                "total", PLAUS_COL, MAXD_COL, "notes"])
    for r in rows:
        crit = r["criteria"] if r["criteria"] else [""] * 6
        w.writerow([
            r["response_id"], r["model"], r["tier"], *crit,
            r["total"] if r["total"] is not None else "",
            "N/A" if r["plaus"] is None else r["plaus"],
            "N/A" if r["maxd"]  is None else r["maxd"],
            r["note"],
        ])
print(f"\nWrote {OUT_CSV}")

# ---- 3. flag tallies (individual values + mean; N/A excluded) ----
print("\n" + "=" * 92)
print("FLAG TALLIES")
print("=" * 92)

plaus_rows = [r for r in rows if r["plaus"] is not None]
print(f"\nPlausibility flag  (false-label rows; N/A excluded)  "
      f"-- {len(plaus_rows)} scored")
for r in plaus_rows:
    print(f"    {r['response_id']:9} {r['model']:22} {r['tier']:12} {r['
↪ plaus']}")
if plaus_rows:
    vals = [r["plaus"] for r in plaus_rows]
    print(f"    {'MEAN':9} {'':22} {'':12} {statistics.mean(vals):.3f}"
          f"   (n={len(vals)}, sum={sum(vals)})")

maxd_rows = [r for r in rows if r["maxd"] is not None]
print(f"\nMax-displacement flag  (non-blind rows; N/A excluded)  "
      f"-- {len(maxd_rows)} scored")
for r in maxd_rows:
    print(f"    {r['response_id']:9} {r['model']:22} {r['tier']:12} {r['maxd
↪ ']}")
if maxd_rows:
    vals = [r["maxd"] for r in maxd_rows]
    print(f"    {'MEAN':9} {'':22} {'':12} {statistics.mean(vals):.3f}"
          f"   (n={len(vals)}, sum={sum(vals)})")

# per-tier maxd means, since the flag spans tiers
print("\n    max_disp mean by tier:")
for t in TIERS:
    tv = [r["maxd"] for r in maxd_rows if r["tier"] == t]
    if tv:
        print(f"      {t:12} mean {statistics.mean(tv):.3f}  (n={len(tv)})")

# ---- 4. bar chart: every response total, grouped by model then tier ----
scored = [r for r in rows if r["total"] is not None]
models = sorted({r["model"] for r in scored})

fig, ax = plt.subplots(figsize=(13, 6.5))
pos = 0.0                                  # float x-position for each bar
gap = 1.2                                  # visual gap between model groups
xticks, xlabels = [], []
tier_seen = set()

for m in models:
```

```
        model_start = pos
        for t in TIERS:                        # tiers stay in this fixed order,
            cell = [r for r in scored          # so all bars of one tier sit
    ↪ together
                    if r["model"] == m and r["tier"] == t]
            cell.sort(key=lambda r: r["response_id"])
            if not cell:
                continue
            xs = [pos + i for i in range(len(cell))]
            heights = [r["total"] for r in cell]
            ax.bar(xs, heights, width=0.85,
                   color=TIER_COLOUR[t],
                   label=t if t not in tier_seen else None)
            tier_seen.add(t)
            pos += len(cell)
        # model label centred under its group
        xticks.append((model_start + pos - 1) / 2)
        xlabels.append(m)
        pos += gap  # gap between models

    ax.set_xticks(xticks)
    ax.set_xticklabels(xlabels, fontsize=11)
    ax.set_ylabel("Total score (max 12)")
    ax.set_ylim(0, 12)
    ax.set_yticks(range(0, 13, 2))
    ax.set_title("Response totals by model and tier")
    ax.legend(title="tier", loc="upper right", framealpha=0.95)
    ax.grid(axis="y", linestyle=":", alpha=0.4)
    ax.set_axisbelow(True)
    ax.margins(x=0.01)
    for spine in ("top", "right"):
        ax.spines[spine].set_visible(False)
    fig.tight_layout()
    fig.savefig(OUT_CHART, dpi=250)
    print(f"\nWrote {OUT_CHART}")
    print("\nDone. scoring_sheet.csv was not modified.\n")


if __name__ == "__main__":
    main()
```

deanonymise_scores_study2.py

The Study 2 equivalent of deanonymise_scores.py. It also computes dataset-selection precision and recall from the recorded selections using the tier-to-real-dataset map, so that the judgement made during scoring — whether a response committed to a dataset — stays separate from the arithmetic. This matters because the tier is not known at scoring time.

```
#!/usr/bin/env python3
"""
Deanonymise Study 2 scores.

Joins scoring_sheet_study2.csv to scoring_key_study2.json, attaches vendor /
model / tier / run, and computes dataset-selection precision and recall against
the real dataset indices for each tier.

Run only after scoring is complete -- it breaks the blinding.

Usage:
    python3 deanonymise_scores_study2.py
```

```
    python3 deanonymise_scores_study2.py --sheet path.csv --key path.json --out
    ↪ path.csv
"""

import argparse
import csv
import json
import sys
from collections import defaultdict

# Real (non-decoy) dataset indices by tier, and the total number of entries
# presented in that tier's inventory. Twelve real entries in every tier: the
# eleven real datasets, with the telemetry dataset appearing as two entries
# (early and late collar windows).
TIER_REAL = {
    "t1_lean":    {"real": [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12], "n_entries":
    ↪  12},
    "t2_diluted": {"real": [1, 3, 4, 6, 7, 9, 10, 12, 13, 15, 16, 18], "
    ↪ n_entries": 18},
    "t3_heavy":   {"real": [1, 4, 6, 9, 11, 14, 16, 19, 21, 24, 26, 29], "
    ↪ n_entries": 30},
}

# Tolerate short tier labels ("t1") as well as full ones ("t1_lean").
TIER_ALIASES = {"t1": "t1_lean", "t2": "t2_diluted", "t3": "t3_heavy"}

CRITERIA = [
    "s1_mechanism",
    "s2_directional_prediction",
    "s3_falsifiability",
    "s4_anticipates_confounding",
    "s5_operational_definition",
    "s6_overreach",
]


def normalise_tier(tier):
    tier = (tier or "").strip()
    return TIER_ALIASES.get(tier, tier)


def parse_datasets(cell):
    """Parse a comma-separated list of dataset indices. Returns a sorted set."""
    if not cell or not cell.strip():
        return set()
    out = set()
    for part in cell.split(","):
        part = part.strip()
        if not part:
            continue
        try:
            out.add(int(part))
        except ValueError:
            raise ValueError("non-numeric dataset index %r" % part)
    return out


def mean(values):
    values = [v for v in values if v is not None]
    return sum(values) / len(values) if values else None


def fmt(value, places=2):
```

```
    return "--" if value is None else ("%.*f" % (places, value))


def main():
    ap = argparse.ArgumentParser()
    ap.add_argument("--sheet", default="scoring_sheet_study2.csv")
    ap.add_argument("--key", default="scoring_key_study2.json")
    ap.add_argument("--out", default="scores_deanonymised_study2.csv")
    args = ap.parse_args()

    try:
        with open(args.sheet, newline="", encoding="utf-8") as f:
            rows = list(csv.DictReader(f))
    except OSError as exc:
        sys.exit("could not read sheet: %s" % exc)

    try:
        with open(args.key, encoding="utf-8") as f:
            key = json.load(f)
    except OSError as exc:
        sys.exit("could not read key: %s" % exc)

    if not isinstance(key, dict):
        sys.exit("expected the key to be an object mapping response_id ->
    ↪ metadata")

    warnings = []
    records = []

    for row in rows:
        rid = row["response_id"]

        # Skip unscored rows rather than failing on them.
        if not (row.get("total") or "").strip():
            warnings.append("%s: unscored, skipped" % rid)
            continue

        meta = key.get(rid)
        if meta is None:
            warnings.append("%s: not present in key, skipped" % rid)
            continue

        tier = normalise_tier(meta.get("tier"))
        if tier not in TIER_REAL:
            warnings.append("%s: unrecognised tier %r, skipped" % (rid, meta.get
    ↪ ("tier")))
            continue

        real = set(TIER_REAL[tier]["real"])
        n_entries = TIER_REAL[tier]["n_entries"]

        try:
            selected = parse_datasets(row.get("datasets_selected", ""))
        except ValueError as exc:
            warnings.append("%s: %s, selection skipped" % (rid, exc))
            selected = set()

        # Sanity check: a selection outside the tier's inventory means either a
        # transcription error or the wrong tier attached to this response.
        out_of_range = sorted(d for d in selected if d < 1 or d > n_entries)
        if out_of_range:
            warnings.append(
                "%s: dataset(s) %s outside the %s inventory (1-%d)"
```

```
                % (rid, out_of_range, tier, n_entries)
            )

        hits = selected & real
        n_selected = len(selected)
        n_hits = len(hits)

        # Precision is structurally undefined at t1: there are no decoys to
        # avoid, so any selection is 100 per cent correct by construction.
        if tier == "t1_lean":
            precision = None
        else:
            precision = (n_hits / n_selected) if n_selected else None

        recall = n_hits / len(real) if real else None

        criteria_values = {}
        for col in CRITERIA:
            raw = (row.get(col) or "").strip()
            criteria_values[col] = int(raw) if raw else None

        record = dict(row)
        record.update({
            "vendor": meta.get("vendor", ""),
            "model": meta.get("model", ""),
            "tier": tier,
            "run": meta.get("run", ""),
            "timestamp": meta.get("timestamp", ""),
            "n_selected": n_selected,
            "n_real_selected": n_hits,
            "n_decoys_selected": n_selected - n_hits,
            "precision": "" if precision is None else round(precision, 4),
            "recall": "" if recall is None else round(recall, 4),
            "coverage_cells": len([c for c in (row.get("hypotheses_matched") or
↪ "").split(";") if c.strip()]),
        })
        record["_criteria"] = criteria_values
        record["_precision"] = precision
        record["_recall"] = recall
        records.append(record)

    if not records:
        sys.exit("no scored rows could be joined to the key")

    # --- write output ------------------------------------------------------
    fieldnames = [f for f in rows[0].keys()] + [
        "vendor", "model", "tier", "run", "timestamp",
        "n_selected", "n_real_selected", "n_decoys_selected",
        "precision", "recall", "coverage_cells",
    ]
    with open(args.out, "w", newline="", encoding="utf-8") as f:
        writer = csv.DictWriter(f, fieldnames=fieldnames, extrasaction="ignore")
        writer.writeheader()
        for r in records:
            writer.writerow(r)

    # --- summaries ---------------------------------------------------------
    def summarise(group_key, label):
        groups = defaultdict(list)
        for r in records:
            groups[group_key(r)].append(r)

        print()
```

```
        print(label)
        header = "%-28s %4s " % ("", "n") + " ".join("%5s" % c[:2].upper() for c
    ↪  in CRITERIA)
        header += " %6s %6s %6s %6s" % ("total", "cover", "prec", "recall")
        print(header)
        print("-" * len(header))
        for name in sorted(groups):
            rs = groups[name]
            cells = []
            for col in CRITERIA:
                cells.append("%5s" % fmt(mean([r["_criteria"][col] for r in rs])
    ↪ ))
            line = "%-28s %4d " % (str(name), len(rs)) + " ".join(cells)
            line += " %6s" % fmt(mean([int(r["total"]) for r in rs]))
            line += " %6s" % fmt(mean([r["coverage_cells"] for r in rs]))
            line += " %6s" % fmt(mean([r["_precision"] for r in rs]))
            line += " %6s" % fmt(mean([r["_recall"] for r in rs]))
            print(line)

    summarise(lambda r: r["model"], "By model")
    summarise(lambda r: r["tier"], "By tier")
    summarise(lambda r: "%s / %s" % (r["model"], r["tier"]), "By model and tier"
    ↪ )

    # Flags are reported as counts, not means.
    print()
    print("Flags (counts)")
    for flag in ["holdout_flag", "recall_flag", "redundant_proxy", "
    ↪ caveat_then_use"]:
        by_model = defaultdict(lambda: defaultdict(int))
        for r in records:
            value = (r.get(flag) or "").strip()
            if value:
                by_model[r["model"]][value] += 1
        print("  %s" % flag)
        for model in sorted(by_model):
            counts = by_model[model]
            total = sum(counts.values())
            detail = ", ".join("%s: %d/%d" % (v, counts[v], total) for v in
    ↪ sorted(counts))
            print("    %-26s %s" % (model, detail))

    print()
    print("Wrote %s (%d rows)" % (args.out, len(records)))
    if warnings:
        print()
        print("Warnings:")
        for w in warnings:
            print("  " + w)
    print()
    print("Note: precision is undefined at t1_lean (no decoys present) and is")
    print("reported blank rather than as 100 per cent.")


if __name__ == "__main__":
    main()
```

chart_study2.py

Draws the Study 2 response-total bar chart, matching the layout, figure size and colours of the Study 1 chart so the two figures can be read together. Read-only on the score data.

```
"""
chart_study2.py -- Study 2 companion to the bar chart in deanonymise_scores.py.
READ-ONLY on your data.

Reads scores_deanonymised_study2.csv and writes

    scores_by_model_tier_study2.png

showing every response's total on the 0-12 scale, grouped by model, with tier
as colour.

Layout, figure size, bar width, gap and tier colours match Study 1's chart so
the two figures sit together in the report. Tier colours are reused in tier
order (t1, t2, t3) rather than by meaning -- the tiers are not the same thing
between studies.

RUN
    python3 chart_study2.py
"""

import csv
import pathlib

import matplotlib
matplotlib.use("Agg")          # no display needed; just write the PNG
import matplotlib.pyplot as plt

IN_CSV    = pathlib.Path("scores_deanonymised_study2.csv")
OUT_CHART = pathlib.Path("scores_by_model_tier_study2.png")

TIERS = ["t1_lean", "t2_diluted", "t3_heavy"]

# same three colours as Study 1, in tier order
TIER_COLOUR = {
    "t1_lean":    "#4C72B0",
    "t2_diluted": "#55A868",
    "t3_heavy":   "#C44E52",
}


def main():
    if not IN_CSV.exists():
        print(f"Need {IN_CSV} in this folder.")
        return

    rows = []
    with IN_CSV.open() as fh:
        for row in csv.DictReader(fh):
            rows.append({
                "response_id": row["response_id"],
                "model": row["model"],
                "tier": row["tier"],
                "total": int(row["total"]),
            })

    if not rows:
        print("Nothing to plot.")
        return

    models = sorted({r["model"] for r in rows})

    fig, ax = plt.subplots(figsize=(13, 6.5))
    pos = 0.0                                  # float x-position for each bar
```

```
    gap = 1.2                                   # visual gap between model groups
    xticks, xlabels = [], []
    tier_seen = set()

    for m in models:
        model_start = pos
        for t in TIERS:                         # tiers stay in this fixed order,
            cell = [r for r in rows             # so all bars of one tier sit
↪ together
                    if r["model"] == m and r["tier"] == t]
            cell.sort(key=lambda r: r["response_id"])
            if not cell:
                continue
            xs = [pos + i for i in range(len(cell))]
            heights = [r["total"] for r in cell]
            ax.bar(xs, heights, width=0.85,
                   color=TIER_COLOUR[t],
                   label=t if t not in tier_seen else None)
            tier_seen.add(t)
            pos += len(cell)
        # model label centred under its group
        xticks.append((model_start + pos - 1) / 2)
        xlabels.append(m)
        pos += gap  # gap between models

    ax.set_xticks(xticks)
    ax.set_xticklabels(xlabels, fontsize=11)
    ax.set_ylabel("Total score (max 12)")
    ax.set_ylim(0, 12)
    ax.set_yticks(range(0, 13, 2))
    ax.set_title("Response totals by model and tier")
    ax.legend(title="tier", loc="upper right", framealpha=0.95)
    ax.grid(axis="y", linestyle=":", alpha=0.4)
    ax.set_axisbelow(True)
    ax.margins(x=0.01)
    for spine in ("top", "right"):
        ax.spines[spine].set_visible(False)
    fig.tight_layout()
    fig.savefig(OUT_CHART, dpi=250)
    print(f"Wrote {OUT_CHART}")
    print(f"\n{len(rows)} responses plotted. Input CSV was not modified.\n")


if __name__ == "__main__":
    main()
```

build_response_notes.py

Writes every response's full text into its own Markdown file, organised into one directory per model, for reference during and after scoring.

```
"""
build_response_notes.py -- write every response's FULL TEXT into its own
Markdown note, organised into one folder per model. Obsidian-native layout:

    response_notes/
        Claude Opus 4.8/
            blind_resp_007.md
            blind_resp_018.md
            ...
        Claude Sonnet 5/
            ...
```

```
        Gemini 2.5 Flash/
            ...
        Gemini 3.1 Pro/
            ...

Each note is headed with the response's model, tier and total, carries your
scoring note as a quote block, then the full response text.

Reads (all read-only):
    scores_deanonymised.csv   (model / tier / total / your notes)
    to_score/resp_NNN.txt     (the actual response text)

Adjust TEXT_DIR / TEXT_PATTERN below if your response files live elsewhere or
are named differently.

RUN
    python3 build_response_notes.py
"""

import csv
import pathlib

CSV_FILE     = pathlib.Path("scores_deanonymised.csv")
TEXT_DIR     = pathlib.Path("to_score")          # folder holding the response
    ↪ texts
TEXT_PATTERN = "{rid}.txt"                        # e.g. resp_007.txt
OUT_ROOT     = pathlib.Path("response_notes")     # top-level output folder

TIER_ORDER = {"blind": 0, "informed": 1, "false_label": 2}

# prettify the raw API model strings -> also used as folder names
MODEL_PRETTY = {
    "anthropic/claude-opus-4.8": "Claude Opus 4.8",
    "anthropic/claude-sonnet-5": "Claude Sonnet 5",
    "gemini-2.5-flash":          "Gemini 2.5 Flash",
    "gemini-3.1-pro-preview":    "Gemini 3.1 Pro",
}


def safe(name):
    """Make a string safe to use as a folder/file name."""
    for ch in '/\\:*?"<>|':
        name = name.replace(ch, "-")
    return name.strip()


def main():
    if not CSV_FILE.exists():
        print(f"Can't find {CSV_FILE}. Run this in your project root.")
        return

    rows = list(csv.DictReader(CSV_FILE.open()))
    rows.sort(key=lambda r: (r["model"],
                             TIER_ORDER.get(r["tier"], 9),
                             r["response_id"]))

    OUT_ROOT.mkdir(exist_ok=True)

    written = 0
    missing = []

    for r in rows:
        rid   = r["response_id"]
```

```
        model = MODEL_PRETTY.get(r["model"], r["model"])
        tier  = r["tier"]
        total = r["total"]
        note  = r.get("notes", "").strip()

        txt_path = TEXT_DIR / TEXT_PATTERN.format(rid=rid)
        if txt_path.exists():
            body = txt_path.read_text(encoding="utf-8", errors="replace").strip
    ↪ ()
        else:
            body = "*(response text not found)*"
            missing.append(str(txt_path))

        model_dir = OUT_ROOT / safe(model)
        model_dir.mkdir(exist_ok=True)

        note_path = model_dir / f"{tier}_{rid}.md"

        parts = [f"# {rid} \u2014 {model} \u2014 {tier} (total {total})\n"]
        if note:
            parts.append(f"> **Scoring note:** {note}\n")
        parts.append(body + "\n")

        note_path.write_text("\n".join(parts), encoding="utf-8")
        written += 1

    n_models = len({MODEL_PRETTY.get(r['model'], r['model']) for r in rows})
    print(f"Wrote {written} notes into {OUT_ROOT}/ across {n_models} model
    ↪ folders.")
    if missing:
        print(f"\nCouldn't find {len(missing)} text file(s) -- check TEXT_DIR /
    ↪ "
              f"TEXT_PATTERN. First few:")
        for m in missing[:5]:
            print("   ", m)


if __name__ == "__main__":
    main()
```