Cambridge Judge Business School

AI in Manufacturing: Identifying Procurement Cost-Saving Opportunities Using Linear Pricing Performance Analysis

An experimental comparison of frontier AI models and staged agentic workflows for reliable procurement analysis

Ujjwal Pandey
Visiting Associate, Cambridge Judge Business School and Founder of OptiSpend AI

Prerak Mody
PhD in AI-driven image analysis, Leiden University

July 2026

UNIVERSITY OF CAMBRIDGE
Judge Business School

Contents

Acknowledgements **4**

1. Executive summary **5**

2. Introduction: the problem **5**

3. The AutoGen Motors benchmark **6**

3.1 Why a synthetic dataset 6
3.2 The dataset 7
3.3 The planted inefficiencies 8
3.4 The LPP family 9

4. The experiment: two approaches **10**

4.1 The frozen task brief 11
4.2 Models and repeated trials 11
4.3 Making the single-prompt approach runnable 11
4.4 Scoring 12

5. Approach 1: everything in one prompt **12**

5.1 Headline results 12
5.2 Two models that never reached the problem 13
5.3 Contaminating the baseline 13
5.4 The wrong price model 14
5.5 Reasoning itself out of the answer 15
5.6 Invented and misread evidence 15
5.7 The unexamined time window 15
5.8 The deeper problem: inconsistency 15
5.9 What every failure has in common 16

6. Approach 2: the staged pipeline **16**

6.1 From failures to requirements 16
6.2 The dividing rule 17
6.3 The seven stages 17
6.4 Evidence trails and exceptions 19
6.5 What this is not 19

7. Approach-2 results 20

7.1 Setup 20
7.2 The headline 20
7.3 Why the model stops mattering 21
7.4 The Haiku divergence 22
7.5 Robustness under model failure 22
7.6 Economics 22

8. Head-to-head and discussion 23

8.1 What actually changed 23
8.2 The quality-assurance parallel 25
8.3 What the result means 25

9. Limitations and roadmap 25

9.1 Limitations 25
9.2 Roadmap 26

10. Conclusion 26

References 27

Appendix A — Dataset composition 28

Appendix B — Planting method 30

Appendix C — The task brief (verbatim) 32

Appendix D — Scoring rubric and full score sheet 33

Appendix E — Provider configuration and run environment 37

Acknowledgements

The authors are deeply grateful to Professor Jaideep Prabhu of Cambridge Judge Business School, who mentored and guided this work throughout. His advice shaped the structure of the paper from its earliest outline, and his careful questioning at every stage sharpened both the argument and its presentation.

1. Executive summary

Manufacturing companies rarely lose money through one big sourcing mistake. They lose it through thousands of small ones: a part priced differently at two plants, an invoice creeping above its contract, an urgent order placed at a premium. Each looks too small to investigate. Together, they quietly erode margin. The data needed to catch these inefficiencies usually exists, but it is scattered across ERP extracts, contracts, engineering drawings and spreadsheets, and no procurement team has the bandwidth to examine every decision. Most of them therefore go unnoticed.

This paper asks whether AI can take on that examination work and, more importantly, whether its answers can be trusted. To find out, we built a synthetic manufacturer: AutoGen Motors, a fictional carmaker with six plants, 44,318 purchase-order lines and 46 engineering drawings, into which we planted seven classes procurement inefficiencies representing EUR 6.10 million in potential annual savings. Because we planted them ourselves, we know exactly what a detection system should find, something that is impossible to know on real customer data.

We then studied one inefficiency in depth: **linear pricing performance (LPP)**, which asks a simple question of every part: is it priced fairly for what it physically is? Within a family of genuinely comparable parts, price should scale with mass. Two parts were planted above that line, together worth EUR 276,007 a year. One instance was deliberately designed to evade cross-plant variance analysis: both plants paid the same inflated price, so benchmarking one plant against the other would not reveal the inefficiency.

Two approaches were tested on identical data. **Approach 1** reflects the natural instinct of a team with access to a powerful model: hand everything over in a single prompt and ask it to find the overpriced parts. **Approach 2** is a staged pipeline, the design behind OptiSpend AI, in which ordinary code does everything that has one computable answer, and the model is used only where judgement is genuinely needed: categorizing files into types and reading engineering drawings to fill in missing data.

The results are stark. Under Approach 1, only 8 of 24 runs across eight frontier models found both planted parts (outliers), and only one model did so reliably. The failures were confident and silent: fabricated masses, contaminated baselines, and savings claims that ranged from a fraction of the true value to 380 percent above it. Under Approach 2, seven of the eight models found both parts with zero false positives, returned an identical answer every single time, and did so for as little as $0.0032 per investigation, roughly 700 times cheaper than the only reliable single-prompt result.

The conclusion of this study is that trust in application of agentic AI in procurement is not a property of the model. It is an engineering property of the workflow built around the model. The rest of this paper shows how we reached that conclusion, and what the workflow looks like.

2. Introduction: the problem

Discrete manufacturers do not lose margin only through a few large sourcing mistakes. Leakage also accumulates through thousands of smaller decisions: a carryover part priced differently across plants, an allocation drifting from its negotiated share, an invoice staying above contract, a part priced out of line with its physical characteristics, or an urgent purchase made under production pressure. Each case can look too small to revisit. Together, they create a structural margin problem.

The data usually exists. It is spread across ERP systems, purchase orders, bills of material, contracts, quotations, drawings, specifications and spreadsheets. Procurement teams can investigate a known issue

once it is escalated; the harder task is deciding which of thousands of decisions deserve expert attention in the first place. Human attention is the scarce input. Deloitte's 2025 Global CPO Survey reports that even the most digitally mature procurement functions spend roughly two-thirds of their time on non-strategic work while managing about $20 million of spend per procurement employee [1]. More decisions per person means less attention per decision.

OptiSpend AI's working estimate, based on manufacturing category experience and more than 60 interviews across four global regions, is that better detection and execution can recover or protect 2–4 percent of addressable direct-material spend in discrete manufacturing, a range consistent with published estimates of procurement value leakage [2], [3]. This paper does not attempt to prove that percentage. It does something narrower and more rigorous: it builds a benchmark in which the recoverable value is known by construction, so that detection performance can be measured exactly.

Traditional spend analytics is useful for visibility, but often stops at dashboards and variance views. A dashboard can show that two prices differ; it rarely establishes whether the difference is justified by specification, volume, contract terms or engineering history. Large language models make this deeper analysis newly tractable, because they can interpret unstructured technical and commercial material at scale. The term frontier models refers to highly capable, general-purpose AI models that can perform a wide range of tasks at or beyond the level of the most advanced models available. But access to a capable model does not, by itself, create a trustworthy procurement workflow. In the controlled trials reported here, frontier models given the full dataset and asked to find planted pricing inefficiencies sometimes succeeded, and also failed in specific, measurable and silent ways.

This paper follows one inefficiency from planting to detection: **linear pricing performance (LPP)** [4], which tests whether comparable parts are priced consistently once their legitimate physical cost drivers are accounted for. The paper makes four contributions. It builds the AutoGen Motors benchmark with seven classes of ground-truthed inefficiency (Section 3); it measures the single-prompt approach across eight models and 24 scored runs (Section 5); it builds and evaluates a staged pipeline on the same data (Sections 6 and 7); and it proposes an evaluation method in which precision, repeatability and evidence completeness are treated as engineering requirements rather than reporting niceties (Section 8).

3. The AutoGen Motors benchmark

3.1 Why a synthetic dataset

Trust requires measurement, and measurement requires an answer key. On real customer data, recall is not measurable: if a detection system runs across a live dataset and returns no findings, there is no way to know whether the data is clean or the system is blind. A capability claim made only against live data can describe what a system surfaced, but never what it failed to find.

The AutoGen Motors (AGM) benchmark was built to make that inefficiency known. Inefficiencies are planted before evaluation, with their mechanism, location, nature of inefficiency and value recorded in a separate ground-truth ledger. The dataset is built in two layers. The agent-facing layer contains everything a system may inspect: purchase-order extracts, item masters, contracts, drawing registers and engineering drawings, with nothing labelling or hinting at any planted inefficiency. The ground-truth layer, which no system under evaluation ever sees, exists only to score them. This lets us clearly see what each system found, missed or got wrong.

Synthetic data does not reproduce every form of real-world disorder, and Section 9 discusses what it leaves out. But it captures the properties that matter for this test: fragmentation across plants, missing data, revision drift, currency heterogeneity, and the need to join structured (i.e. csv/excel files) and unstructured (i.e. imaging) evidence. It also reflects a commercial reality: no early-stage vendor should expect unrestricted access to a manufacturer's full procurement data for experimentation.

3.2 The dataset

The benchmark models a fictional premium automotive manufacturer operating across six plants, six countries and three currencies. It was deliberately built not as a clean central database but as a fragmented multi-plant dataset, the form in which procurement data usually reaches a real analysis team. Each plant extract carries the imprint of the system that produced it: for e.g. "Bremerhaven" exports German column headers, semicolons and decimal commas; "Valencia" exports Spanish headers; "Debrecen" truncates descriptions at 24 characters; "Greer" abbreviates descriptions and writes dates month-first and "Oxford" omits the currency column because sterling is implied. None of this is exotic. It is what a multi-plant dataset that has grown locally over time looks like. Table 1 summarises the benchmark at a glance.

Table 1. The AGM benchmark at a glance.

Dimension	Content
Plants / countries / currencies	6 / 6 / 3 (EUR, GBP, USD)
Vehicle models and platforms	14 models on 3 platforms, ~705,000 vehicles per year
Suppliers / unique parts	210 / ~925
Purchase-order lines	44,318 (April 2025 – June 2026, 15 months)
Annualisation window	July 2025 – June 2026 (12 months)
eBOM lines / frame contracts	3,033 / 210
Drawing-register rows / engineering drawings	951 / 46

The only identifier common across all plants is the **drawing number (or DRW)**, which represents the part as an engineered object rather than a purchasing record. That common identity exists, but it is deliberately damaged. The clearest example is "DRW-70207", purchased at two plants under entirely unrelated local identifiers. Elsewhere, one plant concatenates the revision into the reference, another substitutes a space for the hyphen, and three plants leave a share of references blank. In two cases, identity can be recovered only by matching supplier and supplier part number across plants. Entity resolution is therefore not a preprocessing convenience; it is the gate every analysis must pass. Figure 1 shows the drawing-number spine and the identity damage elsewhere in the dataset.

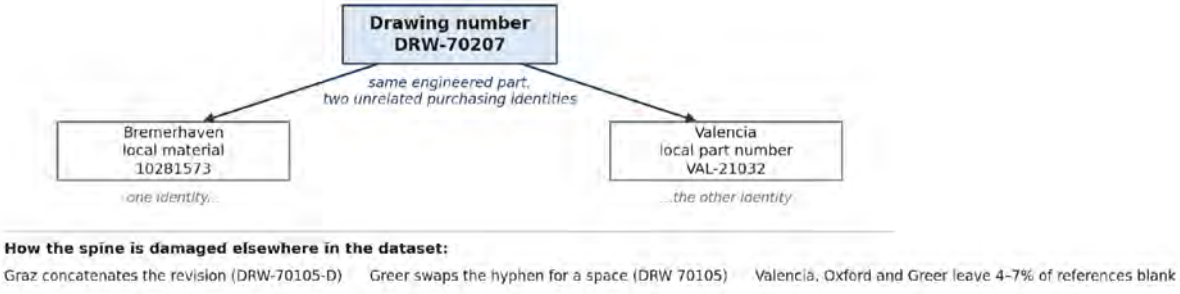

Figure 1. The drawing-number spine: one engineered part with two unrelated purchasing identities, and the deliberate identity damage elsewhere in the dataset.

The dataset is not only structured records. Forty-six engineering drawings provide the unstructured layer, carrying material specifications and part masses in their title blocks. The drawing register, the structured index of those drawings, leaves the mass field blank for most released revisions: of the 18 parts in the family of plastics studied in this paper, only five have a mass in the register. The other thirteen masses exist only on the drawings. This is deliberate. Price lives in purchase-order lines; the physical property needed to judge whether that price is fair may live only in an engineering drawing title block. Any system that solves this problem must join structured and unstructured evidence at part level.

3.3 The planted inefficiencies

The benchmark carries seven classes of planted inefficiency, totalling EUR 6.10 million over the twelve-month window. This paper examines one class in depth. The other six are present because the benchmark is intended as a general evaluation instrument, and because they make the test harder: a system looking for one type of inefficiency must not be distracted by six others. Appendix B documents how each class was planted. Table 2 lists the seven classes.

Table 2. The seven planted inefficiency classes.

Inefficiency class	Mechanism	Instances	12-month value (EUR)
Cross-plant price divergence	The same part is bought at materially different prices across plants without commercial justification.	11	1,746,033
Invoice above contract	Invoice prices exceed contracted prices in a sustained pattern.	12	721,533
Share-of-business drift	Volume migrates from the intended allocation toward a costlier supplier.	5	128,722
Commodity-index drift	Prices fail to follow contractual indexation when the underlying index falls.	10	1,606,992

Inefficiency class	Mechanism	Instances	12-month value (EUR)
Panic buying at premium	Urgent purchases under continuity pressure carry avoidable premiums.	29	53,032
Over-purchasing against plan	Ordered volumes exceed the production programme's justified requirement.	8	1,568,564
Characteristic-price divergence (LPP)	A part is priced materially above what its physical characteristics justify within a comparable family.	2	**276,007

3.4 The LPP family

LPP tests whether a system can move beyond surface price comparison. Within a genuinely comparable family, where parts share material and manufacturing process, the fair price of a part should scale with its physical characteristics, principally mass, an approach long established in automotive purchasing [4]. Fitting that relationship across the family produces a characteristic price line. Thus, a part sitting materially above the line is paying more than its physical properties justify.

The AGM dataset contains exactly eighteen injection-moulded PP-GF30 parts. Fitted across the fourteen uncontaminated members, the characteristic line is **fair price = EUR 0.498 + EUR 4.070 × mass (kg)**, with **R^2 = 0.998**: the underlying relationship is clean enough for deviations to be meaningful rather than noise. Two parts are planted above it.

- **DRW-70199**, a bumper energy-absorber retainer bought at the "Debrecen" plant, sits roughly 28 percent above its characteristic price, about EUR 136,000 of annual excess spend. This is the straightforward case: any method that establishes the family, extracts the mass and fits the line should surface it.
- **DRW-70207**, an air-intake resonator housing, is the more important case. It is bought at both "Bremerhaven" and "Valencia" plants, from the same supplier, at the same inflated price, roughly 21 percent above the line and about EUR 140,000 a year. Because the price is identical at both plants, it is invisible to cross-plant comparison, one of the most common heuristics in spend analytics. It can be found only by establishing what the part ought to cost given its material and mass.

DRW-70207 is therefore the benchmark's cleanest separation between a system that has understood the pricing logic and one that is pattern-matching on price differences. Figure 2 plots price against mass for the family, with the fitted line and the two planted outliers.

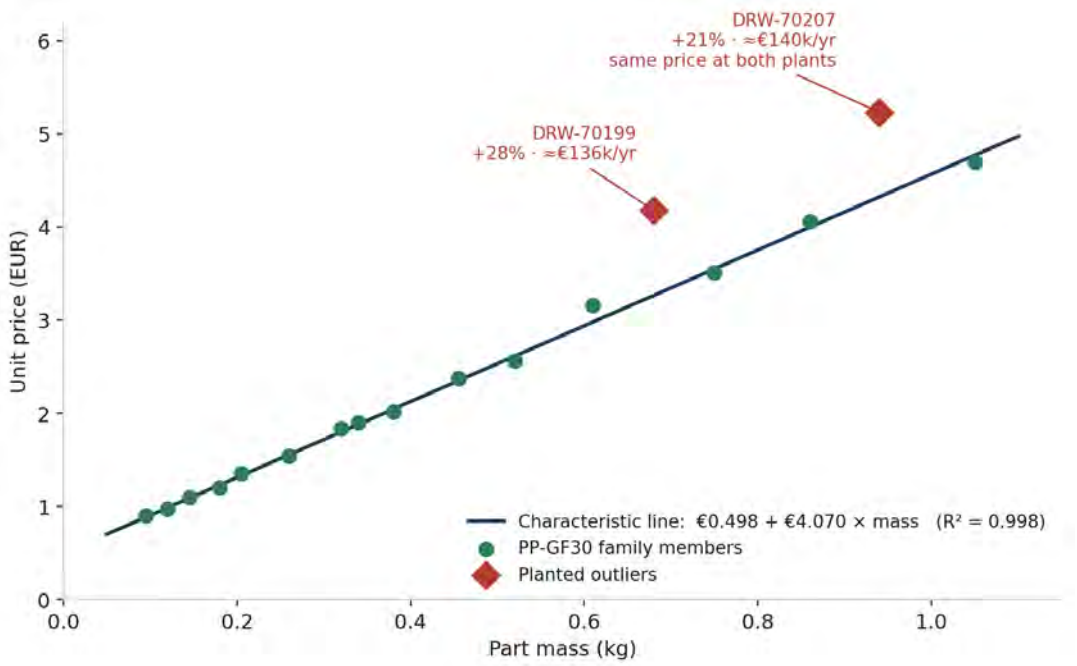

Figure 2. Price versus mass for the PP-GF30 family, with the fitted characteristic line and the two planted outliers. DRW-70213, whose mass exists nowhere in the dataset, cannot be plotted.

The family also contains designed traps, each of which defeats a shortcut and can fail silently: a superseded revision whose stale mass sits in the register while the correct mass appears only on the drawing (DRW-70105); a part whose mass exists nowhere, testing refusal to fabricate (DRW-70213); near-identical parts in other polymers, testing material-based family selection; and clean aluminium and steel families as plausible distractors. A stale, invented or wrongly matched value can still produce a confident, well-formatted saving, and in procurement a fabricated claim is more damaging than a visible error, because it consumes the team's trust. Appendix B documents each trap.

4. The experiment: two approaches

Both approaches were given the same dataset, the same commercial task and the same ground-truth scoring. The only thing that changes is the workflow. Figure 3 sets the two approaches side by side.

- **Approach 1 (single prompt).** The prepared dataset is placed into one prompt, and a frontier model is asked to identify the overpriced parts, quantify the annual excess spend and provide the evidence trail. Everything, from family selection to the final number, happens inside one act of generation.
- **Approach 2 (staged pipeline).** The same investigation is decomposed into stages. Code performs all deterministic work, while model calls are limited to classifying files, extracting values from technical documents and capturing the user's analytical intent. The chat interface allows the same workflow to be applied to any selected material or group of materials, although this capability is not evaluated in the present study.

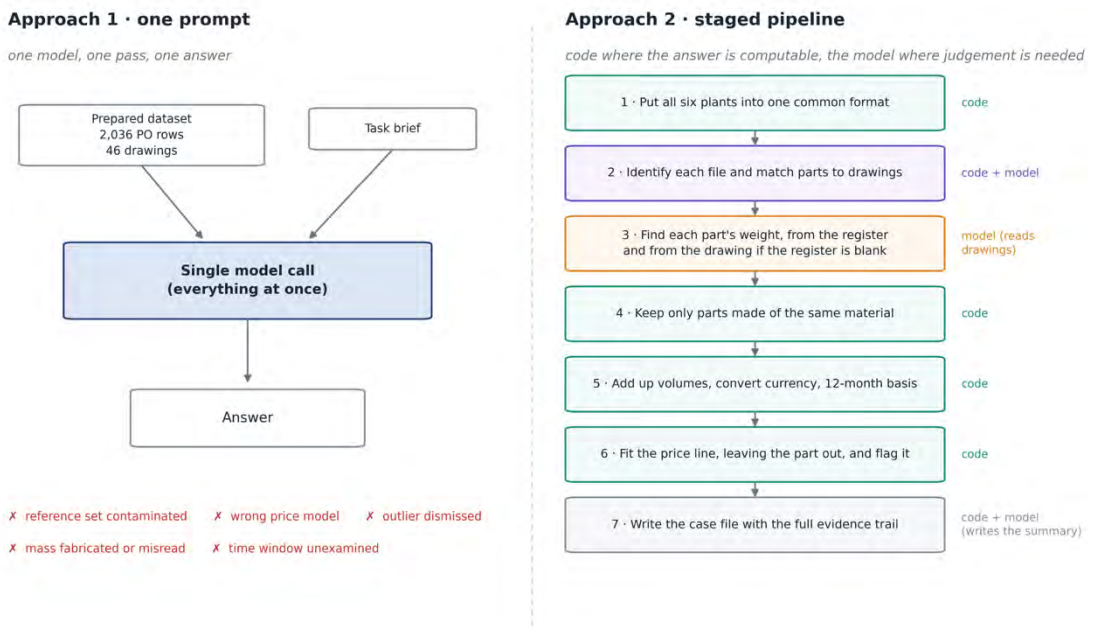

Figure 3. The two approaches, evaluated against the same dataset and the same hidden answer key.

4.1 The frozen task brief

The task brief asks for injection-moulded parts that appear overpriced for their size, the annual excess spend in EUR, and the evidence supporting each conclusion. It explains that comparison must stay within a genuinely comparable material and process family, that fair price should scale with physical characteristics such as mass, that identifiers may vary across plants, and that missing values must not be guessed. It does not name PP-GF30, the planted parts, the number of findings, or any statistical method. It was frozen before the first scored run and used byte-for-byte across every model, trial and approach. The full brief appears in Appendix C.

4.2 Models and repeated trials

Approach 1 was run across eight models from Google, OpenAI and Anthropic, spanning frontier reasoning models and lower-cost tiers, with each model run three times on identical input: 24 scored runs. Repetition matters, because a procurement team needs to know whether a result is dependable, not merely whether a model can produce it once. Provider settings and model identifiers are recorded in Appendix E. One boundary case is worth flagging now: Claude Haiku 4.5's context window accepted only 6 of the 19 prepared files, excluding all pricing data, so its three runs are reported as insufficient-context outcomes rather than detection failures.

4.3 Making the single-prompt approach runnable

The raw dataset contains 44,318 purchase-order lines and measures roughly four million tokens. No tested model could hold it. Approach 1 therefore required a deterministic, blind aggregation from 44,318 lines to 2,036 part-plant rows before any model produced an answer: no part was filtered, prioritised or flagged, and order, contract and invoice prices were preserved separately. Even then, the prepared payload measured 376,842 tokens for OpenAI, 454,857 for Google and 550,497 for Anthropic - the same files, counted 46 percent higher by one provider's tokeniser than another's. Long-context research also shows

that models use long inputs unevenly, with retrieval accuracy degrading as relevant material moves toward the middle of the window [5]. This matters to the interpretation: Approach 1 was given a generous, pre-digested input, so its measured failures are conservative. Approach 2, by contrast, reads the raw 44,318-line dataset, because its workflow makes that scale possible.

4.4 Scoring

Each run is scored against the hidden ground-truth ledger on countable criteria: detection of the two planted parts, false positives, family membership, mass accuracy, trap handling, savings error, and whether every claim traces to a named file, row or drawing. The purchase data spans fifteen months while the ledger is stated on a twelve-month basis, so runs are also checked for explicit annualisation, and savings error is reported against both the EUR 276,007 twelve-month truth and its EUR 345,009 fifteen-month equivalent. Precision is weighted above recall throughout: a missed opportunity costs money, but a fabricated one can cost the credibility of the entire system. The full rubric and score sheet are in Appendix D.

5. Approach 1: everything in one prompt

Approach 1 tests the simplest plausible workflow: give a frontier model the prepared dataset, a competent brief and a large context window, and ask it to find the planted pricing inefficiencies directly. The purpose is not to show that language models cannot perform procurement analysis. They can: one model found both planted parts in all three of its trials. The finding is narrower and more important. The single-prompt approach is not reliable, and its failures are specific, diagnosable and silent. They produce confident, well-formatted answers that look like procurement analysis, and from the output alone a user would often have no way to know whether the model had found a real opportunity, missed one, fabricated one, or reasoned itself out of the correct answer.

5.1 Headline results

Across 24 scored runs, 8 identified both planted parts, 2 identified one, and 14 identified neither. Gemini 3.1 Pro found both in all three trials; no other model was reliable across repeated runs. Table 3 reports the outcome of every model and trial.

Table 3. Approach 1 results by model and trial (TP = true positives out of 2; FP = false positives).

Model	Trial 1	Trial 2	Trial 3	Summary
Gemini 3.1 Pro	2 TP / 0 FP	2 TP / 1 FP	2 TP / 0 FP	Found both parts in all three trials; the only reliable model. One trial fabricated a mass.
GPT-5.5	2 TP / 0 FP	2 TP / 1 FP	0 TP / 19 FP	Two clean trials, then a methodology switch producing nineteen false positives.

Model	Trial 1	Trial 2	Trial 3	Summary
Claude Opus 4.8	2 TP / 1 FP	2 TP / 1 FP	1 TP / 0 FP	Found both twice; misread the DRW-70199 mass in every trial; dismissed DRW-70207 in trial 3.
GPT-5.4	0 TP / 5 FP	2 TP / 0 FP	0 TP / 2 FP	Contaminated baseline, then a clean run, then the wrong anomaly class.
Claude Sonnet 5	1 TP / 0 FP	0 TP / 1 FP	0 TP / 1 FP	Found one part, dismissed the other on cross-plant grounds, then abandoned the LPP analysis.
Gemini 3.5 Flash	0 TP / 1 FP	0 TP / 2 FP	0 TP / 2 FP	No trial succeeded; one trial contaminated its baseline with DRW-70207.
Gemini 2.5 Flash-Lite	0 TP / 1 FP	0 TP / 1 FP	0 TP / 1 FP	Identical repetition loop in every trial; no completed analysis.
Claude Haiku 4.5	insufficient context	insufficient context	insufficient context	Only 6 of 19 files fit its window; it correctly declined to report findings.

The fair reading matters. The task is solvable: a frontier model can reconstruct the family, recover the drawing evidence and find both anomalies, including DRW-70207. The finding is not impossibility. It is capability without dependable control. The rest of this section examines the failure modes one by one, because each one shapes the design in Section 6.

5.2 Two models that never reached the problem

Claude Haiku 4.5 could not receive the task-relevant data: only 6 of the 19 files fit its context window, and the six purchase-order extracts and the drawing register were among those excluded. In all three trials it declined to report findings rather than fabricate analysis, stating plainly that without weight information the comparison could not be calculated. That is a context-limit finding, and the behaviour under the constraint was correct. Gemini 2.5 Flash-Lite failed differently: with no reasoning mechanism suited to the task, it produced the same 1,443-line repetition loop in all three trials, filling its output budget without ever reaching a conclusion. These two cases are separated from the analytical failures below because they did not really test the procurement reasoning problem: one lacked the data, the other never completed the reasoning.

5.3 Contaminating the baseline

To judge whether a part is overpriced, a system must first establish what a fair price looks like: in LPP, a characteristic line fitted through a genuinely comparable family. But the family contains the overpriced

parts. If the outliers are included in the reference fit, they pull the line upward, and a part cannot exceed a line it is helping to define.

GPT-5.4's first trial shows the failure cleanly: its stated reference set included both planted parts, placing the targets inside its own definition of normal. Against the resulting pooled rate, both looked unremarkable; the run missed them and flagged five innocent parts instead. Gemini 3.5 Flash made the same error with DRW-70207. The failure is structural rather than a forgotten rule: nothing in a single-prompt workflow separates the act of defining normality from the act of detecting deviation. There is no protected reference stage.

5.4 The wrong price model

Injection moulding carries a fixed cost per part: tooling amortisation, cycle time, handling. These costs do not scale with mass, so the characteristic price line has an intercept, and a consequence follows: price per kilogram falls as parts get heavier. A 0.18 kg bracket at around EUR 6.8/kg and a 0.94 kg housing at around EUR 4.6/kg can both sit exactly on the same fair-price line.

A system that benchmarks on a flat price-per-kilogram ratio inherits that structure as error: it flags light parts, which naturally look expensive per kilogram, and misses heavy outliers, which naturally look cheap. GPT-5.5's third trial is the clearest example. It adopted a flat EUR 5.30/kg benchmark, flagged nineteen parts, claimed roughly EUR 1.33 million of excess spend, about 380 percent above the true value, and missed both real outliers, which sat close to the flat line. All nineteen false positives were of exactly the predicted kind. Figure 4 contrasts a fitted line with a flat euro-per-kilogram benchmark.

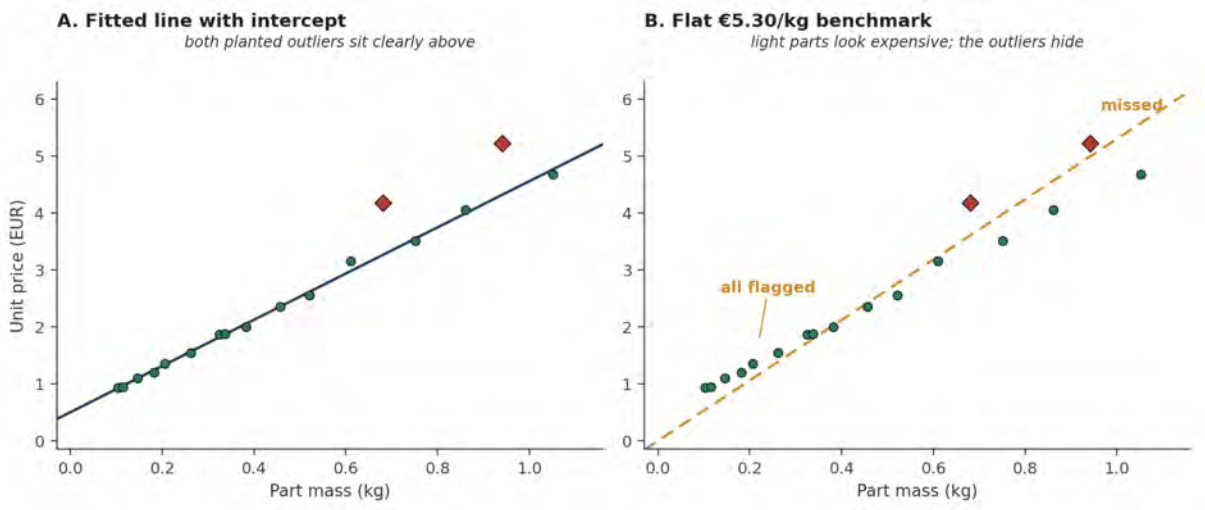

Figure 4. A fitted line preserves the fixed-cost intercept; a flat EUR/kg benchmark flags structurally normal light parts and conceals both real outliers.

This is a fair failure to report. The task brief describes fair price as scaling with weight, wording compatible with either a proportional ratio or a line with an intercept. The data resolves the ambiguity decisively: a linear fit gives $R^2 = 0.998$ and a proportional fit does not. Every run that fitted a line found both outliers; no run using a flat ratio found both. The failure is not that the brief was impossible, but that no run which chose the wrong form ever tested that choice against the data.

5.5 Reasoning itself out of the answer

DRW-70207 exists to test whether a system can move beyond cross-plant comparison, and two runs demonstrated exactly why. Claude Opus (trial 3) and Claude Sonnet (trial 1) both found the part, examined it, and then dismissed it, reasoning that because Bremerhaven and Valencia were paying the same supplier the same price, the price appeared well aligned. The heuristic is often sound: cross-plant agreement can indicate a well-managed part. But shared overcharging is indistinguishable from shared good pricing unless the system has an independent view of what the part ought to cost.

This is the most instructive failure in the experiment, because it is not a lapse. It is a system reasoning competently, from a defensible premise, to a wrong conclusion, and stating that conclusion with confidence. A procurement team reading the output would see a part examined and cleared. The EUR 140,000 stays lost, and the report says the analysis was done.

5.6 Invented and misread evidence

Gemini 3.1 Pro, the only model reliable on detection, invented a mass of 0.180 kg in one trial for DRW-70213, the part whose mass exists nowhere in the dataset, and computed a fair price, an excess and an annual saving from that fabricated value, reported alongside its correct findings with identical confidence and formatting. Claude Opus produced a different version of the same risk: it read DRW-70199's mass from the drawing as 0.580 kg instead of 0.680 kg, consistently, in all three trials, inflating the calculated excess for that part by more than 40 percent. A stable misreading is in some ways more dangerous than a one-off hallucination, because it survives a second run.

In both cases the error is invisible in the output: a fabricated mass carries no marker of fabrication, and a misread one no marker of misreading. Each flows into a fair price, then a saving, then a business case. This behaviour is not incidental: analyses of model training argue that standard training and evaluation reward a confident guess over an admission of uncertainty, and that some rate of fabrication is statistically unavoidable in calibrated models [6], [7]. One designed trap, for completeness, did not fire: no run used the superseded revision-B mass for DRW-70105, because models consistently preferred the drawing over the register. That preference is correct behaviour, and the null result is reported as part of the evidence.

5.7 The unexamined time window

The purchase-order data spans fifteen months, while the ground truth is stated on a twelve-month basis. Exactly one run out of 24, Gemini 3.1 Pro's third trial, recognised the fifteen-month window and applied a 12/15 scaling factor. The other 23 took the prepared quantity at face value and produced savings inflated by 25 percent before any analytical mistake had been made at all. The finding here is not the arithmetic; it is that the assumption was never surfaced. The runs produced savings figures precise to the euro without stating that the period basis had not been checked. The number that reaches a CFO carries a decimal point and no doubt, and this error compounds silently with every other error in this section.

5.8 The deeper problem: inconsistency

The strongest result of Approach 1 is not any single failure. It is inconsistency across repeated trials on identical input. GPT-5.5 produced two clean trials and then switched methodology entirely. GPT-5.4 contaminated its baseline, then ran clean, then analysed the wrong anomaly class. Claude Sonnet found one outlier and then abandoned the LPP analysis altogether. Only Gemini 3.1 Pro detected both parts

three times out of three, and even it fabricated a mass in one of those runs. Part of this variation is a property of the serving infrastructure itself: identical requests can be computed differently depending on concurrent load, so it cannot be removed by prompt or temperature settings alone [8].

A procurement team does not run an analysis three times and take the mode, the strategy that self-consistency decoding formalises in research settings [9]. It runs it once and acts on what comes back. On that basis, GPT-5.5 is a model that can find the real EUR 276,007 opportunity or invent EUR 1.33 million of false opportunity, depending on the run, and the output gives no reliable indication which has happened. Aggregate accuracy is therefore the wrong measure on its own. What matters is whether a single run can be trusted, and for six of the eight models tested, it could not. The 24 runs also cost about $45 in total, between $0.07 and $4.91 each, and since a single attempt cannot be trusted, the honest cost of a usable answer is some multiple of that, plus the human work of reconciling divergent runs, performed by exactly the scarce expert attention the system was meant to conserve.

5.9 What every failure has in common

Every failure in this section is silent. A contaminated baseline produces a clean-looking benchmark. The wrong price model produces a confident table of findings. A dismissed outlier produces a part examined and cleared. A fabricated mass produces a precise saving. An unexamined time window produces a number to two decimal places. None announces itself as an error, and a system whose errors are invisible cannot be supervised. In procurement, where one fabricated saving can damage the credibility of every finding that follows it, invisible error is not just a quality problem. It is a deployment problem.

The failures are not random, and that is the useful part. Each one points to a specific structural requirement: separate the reference set from the candidates; derive the statistical form from the data; test the physical-cost hypothesis rather than a price-comparison heuristic; refuse to proceed on values that do not exist; derive the time basis explicitly; and use deterministic computation wherever a deterministic answer exists. That reading is consistent with independent taxonomies of agentic-system failures, in which most documented failure modes trace to system design rather than to model capability [10]. Section 6 describes the architecture those requirements produce.

6. Approach 2: the staged pipeline

6.1 From failures to requirements

It is tempting to read Section 5 as a report card on the models. It is more useful to read it as a shopping list: each failure tells you what a design would have to do to make that failure impossible rather than merely unlikely. Table 4 maps each failure to the design requirement it implies.

Table 4. Approach 1 failures and the design requirements they imply.

What went wrong in Approach 1	What the design therefore has to do
The overpriced parts were included in the set used to define a fair price	Establish the reference set separately from the candidates being judged
A flat euro-per-kilogram rate was assumed	Derive the statistical form from the data instead of assuming it

What went wrong in Approach 1	What the design therefore has to do
Two plants paying the same price was read as proof the price was right	Detect against the physical hypothesis, not price-comparison heuristics
A mass was invented; another was misread, three times over	Make extraction a separate, verifiable step; halt on missing values
Fifteen months of data reported as if it were twelve	Derive the time basis deterministically from the dates in the data
Same model, same prompt, three different answers	Make everything mechanical deterministic; confine the model to judgement

Read the right-hand column again. Only one line, the one about the physical hypothesis, asks anything of the model's reasoning. The other five ask something of the workflow. That is the central claim of this paper: the single-prompt approach does not fail because frontier models reason poorly. Several of them reasoned very well. It fails because a single undivided pass has nowhere to put the discipline. There is no separate place to establish a reference set, to check that an extracted mass came from somewhere real, or to derive a time basis and show the working. Everything happens at once, inside one act of generation, and so nothing can be checked.

6.2 The dividing rule

The pipeline reserves the language model for steps that genuinely need judgement and does everything else in ordinary code: a predefined workflow rather than a self-directing agent [11], in the tradition of program-aided approaches that have the model interpret the problem while a deterministic runtime computes the answer [12]. The test is simple enough for any reader to apply to their own workflow. A step belongs in code if it has one computable correct answer: parsing six plant schemas, converting currencies, summing volumes, fitting a line, computing a residual. A step belongs to the model if it needs interpretation that no rule reliably captures: reading a material specification out of a drawing title block, deciding whether an unfamiliar file is a purchase-order extract or a bill of materials. Approach 1 collapsed those two categories into a single call, and every failure in Section 5 happened at the seam between them.

The cost of separating them is rigidity, and it is worth stating plainly: a staged pipeline can only run the investigations it has stages for, while a single prompt will attempt anything you ask of it. In procurement, where a fabricated finding taken to a supplier costs more than a finding never made, we argue this is a trade worth making. But it is a trade, and we present it as one.

6.3 The seven stages

Seven stages, each existing because something specific went wrong in Approach 1. Figure 5 sets them out, with the failure each stage removes.

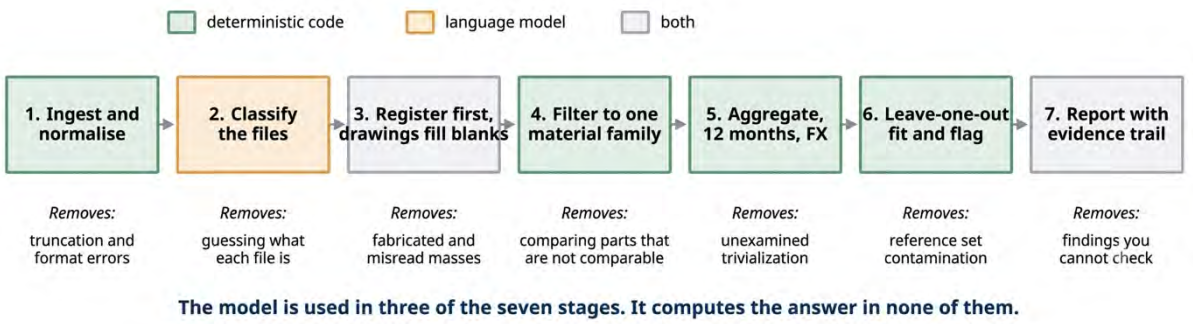

Figure 5. The seven stages, and the Approach 1 failure each one is designed to remove.

- **Ingest and normalise (code).** Six plants, six schemas, three currencies, 44,318 purchase-order lines, turned into one canonical transaction table. This removes silent format errors, and removes the need to fit the raw dataset into a context window at all.
- **Classify the files (model).** The pipeline is handed a folder, not a labelled dataset, and something must decide which file is which. No rule does this reliably across customers, so the model does it. As Section 7.4 shows, this is the one stage where the choice of model still makes a difference.
- **Register first, drawings fill the blanks (code plus a vision model).** Where the drawing register carries a mass, that value is used. Where it is blank, and only then, the drawing image is read. Anything still unresolved is raised as an exception rather than quietly dropped. This removes the two worst Approach 1 behaviours at once: inventing a mass that exists nowhere, and confidently misreading one that does.
- **Filter to one material family (code).** Only parts sharing material and manufacturing process are compared, which removes comparisons between parts that are not comparable.
- **Aggregate over a derived window, with monthly FX (code).** The window is computed from the dates actually present in the data, and each transaction is converted at the rate for its own month. This removes the unexamined annualisation that produced a 25 percent error in Approach 1.
- **Fit and flag, leaving one out (code).** The regression that defines a fair price and the residual test that flags departures from it, described below.
- **Report with the evidence trail (code plus model prose).** Every number in the report is assembled by code. The model writes the sentences around those numbers; it cannot change one.

Stage 6 carries the paper's main methodological contribution: each candidate part is judged against a line fitted **without it**. This sounds like a technicality; it is not. Our first version trimmed outliers against a line fitted on everything, then refitted until it settled, which looks perfectly reasonable and fails in a specific, quiet way. A mildly anomalous part pulls the line toward itself, which suppresses its own residual, which keeps it under the flag threshold, which keeps it inside the reference set, where it goes on pulling the line. The part protects itself, a failure known in the robust-statistics literature as the masking effect [13]. In our own pre-fix batch this happened to DRW-70207 in every run across seven models: a residual of 11.2 percent against a line it had helped to fit, comfortably under a 15 percent bar, invisible. Leave-one-out removes the possibility by construction, because a part can no longer influence the line it is measured against. Figure 6 shows the effect.

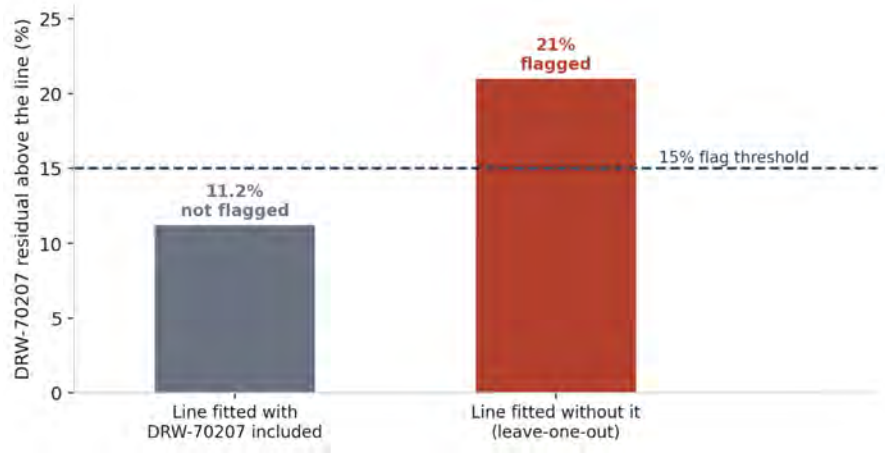

Figure 6. Self-protecting contamination. Fitted with DRW-70207 included, its residual sits under the flag threshold; fitted without it, the part is clearly flagged.

Notice what the model is no longer doing. It reads drawings and labels files. It does not choose the reference set, compute the fair price, or calculate the saving. The answer is produced by code that runs the same way every time.

6.4 Evidence trails and exceptions

Every finding leaves the pipeline as a case file rather than a number: the source rows it was built from, the extracted attributes with their provenance recorded as register or drawing, the fitted line with its membership and goodness of fit, the derived time window, the currency method, and the assumptions made along the way. Values that cannot be resolved become named exceptions, not silent omissions. DRW-70213 is the example to hold on to: its mass exists nowhere, and in Approach 1 one model invented a value and built a finding on it. The pipeline cannot do this; the part leaves the analysis as a named exception, and the category manager can see both that it was excluded and why. Nothing is actioned automatically. The output is a case file for a human buyer to take into a supplier conversation, built so that every step can be checked first. This is also the shape of oversight that emerging regulation asks of high-risk AI systems: output that a person can interpret, override and halt [14].

6.5 What this is not

The pipeline is not general: it runs the investigations it has stages for, and no others. It is not autonomous: it produces a case file, and a person decides what to do with it. And it is not a claim that language models are unsuited to procurement; it is a claim about where in a workflow they belong. The disclosure is that the pipeline was developed against the benchmark on which it is evaluated. The residual flag threshold was pre-registered at 15 percent before any scored run, and no fact specific to any planted instance appears in any skill or configuration file, but the same team built both, and the reader should weigh the results with that in mind.

7. Approach-2 results

7.1 Setup

Same frozen brief, same dataset, same scoring and the same ground truth as Approach 1. Eight models, three trials each, 24 scored runs, all of which completed. Within any single run, the model that classifies the files is the same model that reads the drawings. All calls were routed through OpenRouter, which means the exact deployment a provider served on a given day was outside our control; the served model string was recorded for every run, and the pipeline's skills and configuration were frozen before the first scored run. One asymmetry bears repeating because it is the architectural point: Approach 2 reads the raw 44,318-line extracts that would not fit into any Approach 1 context window.

7.2 The headline

Seven of the eight models found both planted parts, with zero false positives, in all three trials. More than that: the seven produced the same answer. Not a similar answer, the same one. The same fitted line, the same family membership, the same flags and the same money figure, in all 21 runs. The recovered line, **fair price = EUR 0.521 + EUR 4.066 × mass**, sits almost exactly on the hidden reference line of EUR 0.498 + EUR 4.070 × mass, which no part of the pipeline ever reads. The identified annual saving was EUR 292,534 against a ground truth of EUR 276,007, an overstatement of 6.0 percent. For contrast, the same eight models on the same task under Approach 1 averaged 0.75 correct findings out of a possible two. Table 5 reports the results by model; Figure 7 compares detection accuracy across the two approaches and Figure 8 places the recovered line against the hidden reference.

Table 5. Approach 2 results by model. Every figure traces to per-trial run artefacts; the ground truth was never visible to the pipeline.

Model	Found (of 2)	False positives	Fitted line	Saving (EUR)	Same in all 3 trials
Gemini 3.1 Pro	2	0	4.0655x + 0.5209	292,534	yes
Gemini 3.5 Flash	2	0	4.0655x + 0.5209	292,534	yes
Gemini 2.5 Flash-Lite	2	0	4.0655x + 0.5209	292,534	yes
GPT-5.5	2	0	4.0655x + 0.5209	292,534	yes
GPT-5.4	2	0	4.0655x + 0.5209	292,534	yes
Claude Opus 4.8	2	0	4.0655x + 0.5209	292,534	yes
Claude Sonnet 5	2	0	4.0655x + 0.5209	292,534	yes
Claude Haiku 4.5	2	2	4.2730x + 0.4715	254,083	yes
Ground truth	2	0	4.0697x + 0.498	276,007	n/a

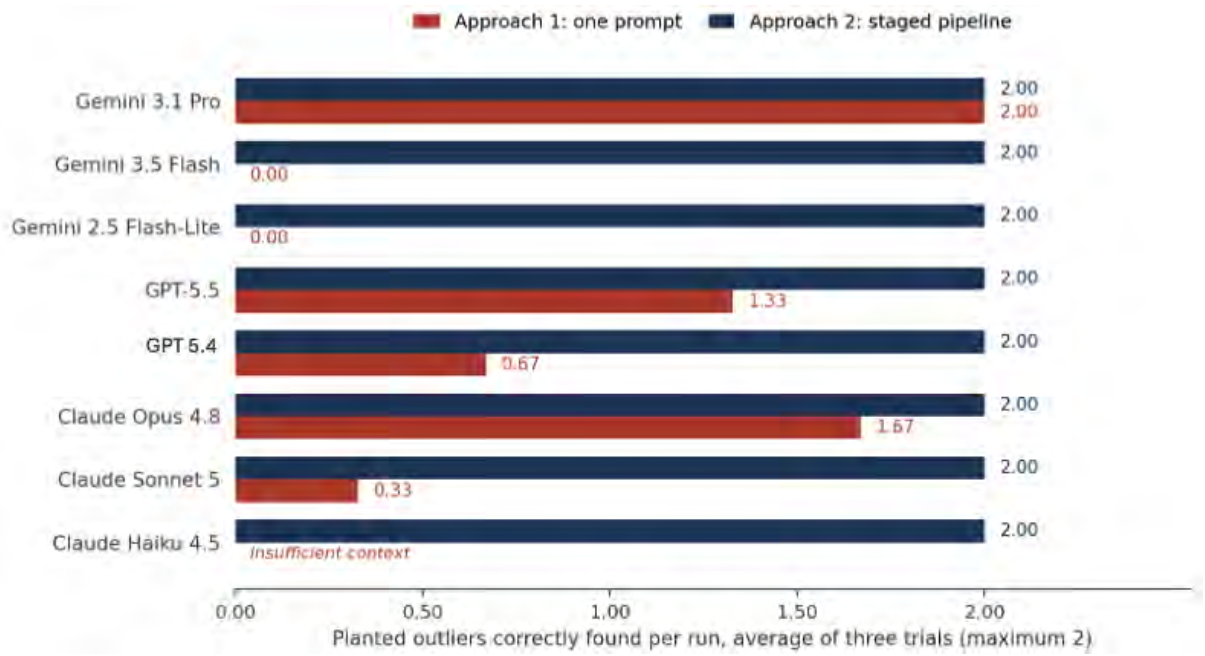

Figure 7. Detection accuracy for the same eight models under the two approaches.

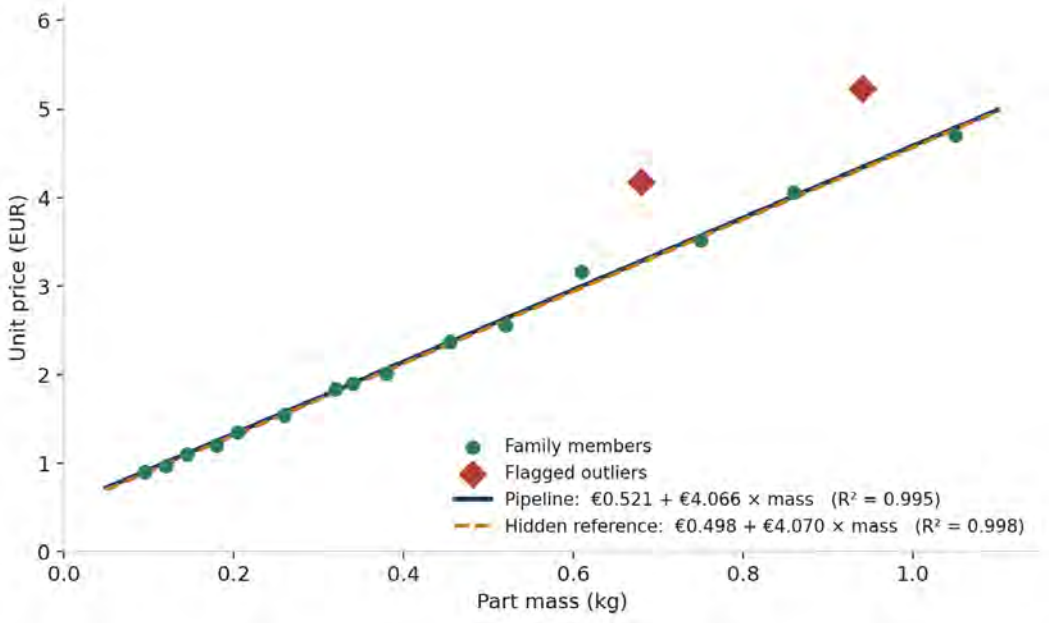

Figure 8. The characteristic line recovered by the pipeline against the hidden reference line. The slopes differ by 0.004; the difference is not visible.

7.3 Why the model stops mattering

This is the result to take away. Across the seven agreeing models, cost per investigation varies by a factor of 29, latency by a factor of nearly five and token use by a factor of six. The answer does not vary at all. This follows directly from the design rather than from any property of the models: the model does not compute the answer. It reads a title block and labels a file. The regression, the reference set, the residuals and the money are fixed code, so they return the same values whichever model served those two narrow jobs.

The vision step deserves separate mention, because it is exactly where a sceptical reader would expect variation to creep in. Both target masses, 0.68 kg for DRW-70199 and 0.94 kg for DRW-70207, appear only on drawing title blocks, not in the register. Across the 42 trials run in this study, every model extracted both correctly, without a single misread. Under Approach 1, by comparison, Claude Opus 4.8 misread one of those same masses and carried the error straight into its savings figure. Constraining the vision task to one value from one title block turns an unreliable step into a reliable one.

7.4 The Haiku divergence

One model did not join the consensus, and it is worth reporting rather than smoothing over. Claude Haiku 4.5 flagged both planted parts correctly, but also flagged two correctly priced parts, consistently, in all three trials. Its fitted line differs, and is built on 13 family members where every other model used 15; its saving of EUR 254,083 is 7.9 percent below ground truth. The cause is not vision: both extra parts had their masses read correctly. The cause sits at the one stage where model judgement still enters the pipeline: file classification. Haiku labels the input files differently, which changes what enters aggregation, which changes the population the line is fitted on. The divergence is therefore not a hole in the finding but a map of the remaining exposure: one stage out of seven is still model-sensitive, and it is a step that a schema contract or a single human confirmation would close.

7.5 Robustness under model failure

An accidental result proved instructive. In two of Claude Opus 4.8's three trials the provider's API failed mid-run, and in one trial none of the seven model calls succeeded at all. The approach 2 architecture nonetheless produced the correct answer - the same fitted line, the same two flagged parts, the same savings figure. It did so because the part masses were read directly from the drawings' embedded PDF text layer, which requires no model, and because every step after extraction is deterministic code. We report the analytical result but exclude Opus's cost, latency and token figures from all comparisons: a run in which no model call succeeded is not a representative measure of what that model costs.

7.6 Economics

Token volume is where the architectural choice shows up on an invoice. Under Approach 1, every question re-ingests the whole dataset: 376,842 to 550,497 input tokens per run, at $0.07 to $4.91. Under Approach 2, code reduces 44,318 purchase-order lines to a family of roughly twenty comparable parts before the model sees anything, so a complete investigation costs 6,884 to 39,273 tokens, at $0.0032 to $0.0931. The fairest comparison is between the two designs at their most reliable. Under Approach 1, the only model that found both parts in all three trials cost a mean of $2.29 per run. Under Approach 2, the cheapest model to do the same cost $0.0032: roughly 700 times less, for an answer that was also identical every time it was asked.

That cheapest model, Gemini 2.5 Flash-Lite, is worth dwelling on. Under Approach 1 the very same model found nothing at all, producing an identical repetition loop in every trial. Nothing about the model changed between the two conditions; only the workflow around it did. Two consequences follow, and the second matters more. Cost per investigation falls far enough that an analyst can ask many questions rather than rationing them. And because the model is no longer doing the analysis, the pipeline works with small models: a manufacturer that cannot send purchase-order data to a frontier API can run this on a

model small enough to host inside their own network, and get the same answer. Table 6 gives cost and token volume per run, and Figure 9 plots cost against detection accuracy.

Table 6. Cost and token volume per run, by model and approach. Approach 1 costs are the mean of three trials; Claude Opus 4.8 is omitted for the reason in Section 7.5. Haiku's Approach 1 token count is low only because 13 of 19 files did not fit its window.

Model	Approach 1: input tokens	Approach 1: cost	Approach 2: total tokens	Approach 2: cost	Saving factor
Gemini 3.1 Pro	454,857	$2.29	15,420	$0.0732	31x
Gemini 3.5 Flash	454,857	$1.26	14,229	$0.0496	25x
Gemini 2.5 Flash-Lite	454,857	$0.07	30,768	$0.0032	22x
GPT-5.5	376,842	$4.71	8,440	$0.0589	80x
GPT-5.4	376,842	$1.48	6,884	$0.0196	76x
Claude Sonnet 5	550,497	$1.67	39,273	$0.0931	18x
Claude Haiku 4.5	118,490	$0.12	27,627	$0.0326	4x

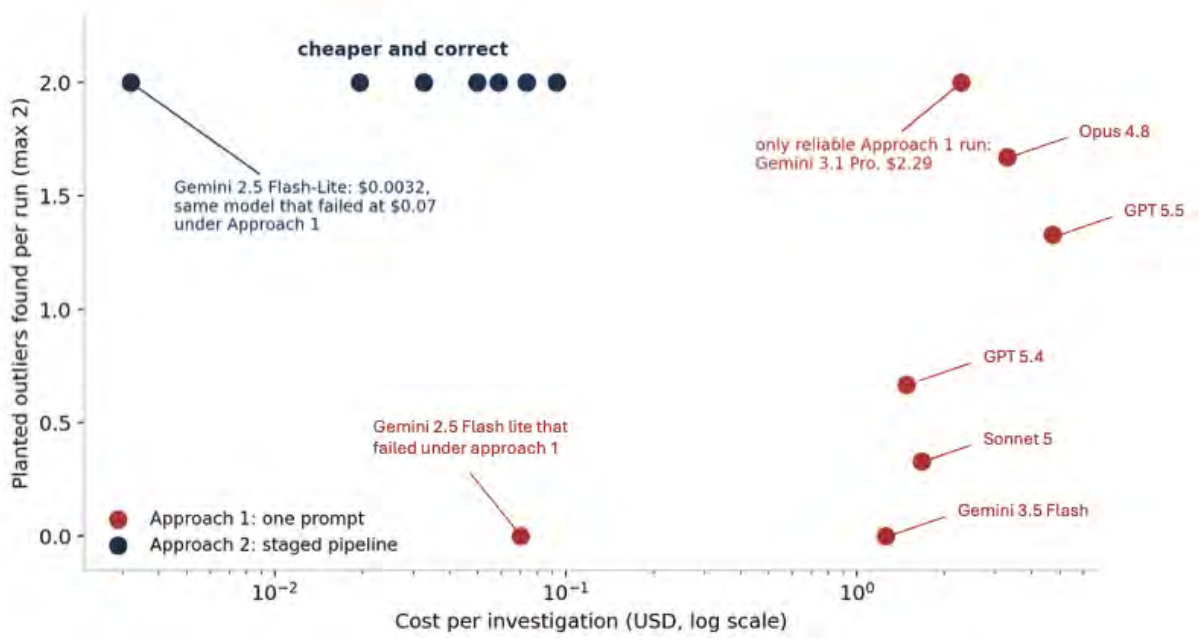

Figure 9. Cost against detection accuracy. The staged workflow moves the same models toward the upper-left corner: cheaper and correct.

8. Head-to-head and discussion

8.1 What actually changed

The comparison is controlled: dataset, commercial question, model set and scoring are held constant, and the only thing that changes is where judgement is allowed to enter the workflow. Approach 1 asks the model to define the family, gather evidence, choose the statistical method, calculate savings and reach a

conclusion in a single step. Approach 2 narrows model judgement to bounded tasks and makes the reference set, regression, residual test, annualisation and money calculation explicit and repeatable. The staged pipeline is not better because it uses more AI steps. It is better because it uses AI in the right places and gives each remaining model decision a visible boundary. For example Reading a mass from a drawing's title block is a judgement; fitting a line through 18 parts is arithmetic. Approach 1 did both in one pass, so a misread digit and a bad regression reached the reader as a single confident figure. Approach 2 records the extracted mass with its source and computes the line in deterministic code, so when a finding is wrong, you can tell which step produced the error. Table 7 summarises the comparison, and Figure 10 shows the signed error in each claimed saving.

Table 7. Head-to-head summary.

Metric	Approach 1: one prompt	Approach 2: staged pipeline
Data presented to the system	Pre-aggregated 2,036 rows; 376,842–550,497 tokens per run	Raw 44,318-line dataset; 6,884–39,273 tokens per investigation
Both planted parts found	8 of 24 runs	24 of 24 runs
Clean repeatability	1 of 8 models in all 3 trials; even it fabricated one mass	7 of 8 models identical across all trials
False positives	39 across the scored set	0 for the 7-model consensus; 2 per Haiku run
Savings error	−100% to +380%	+6.0% (consensus); −7.9% (Haiku)
Cost per investigation	$0.07–$4.91; $2.29 for the only reliable model	$0.0032–$0.0931
Evidence and control	Variable evidence, silent assumptions, no protected stages	Provenance-rich case files, deterministic calculations, named exceptions
Human role	Reconcile divergent analyses; hunt hidden errors	Review a bounded case file before supplier action

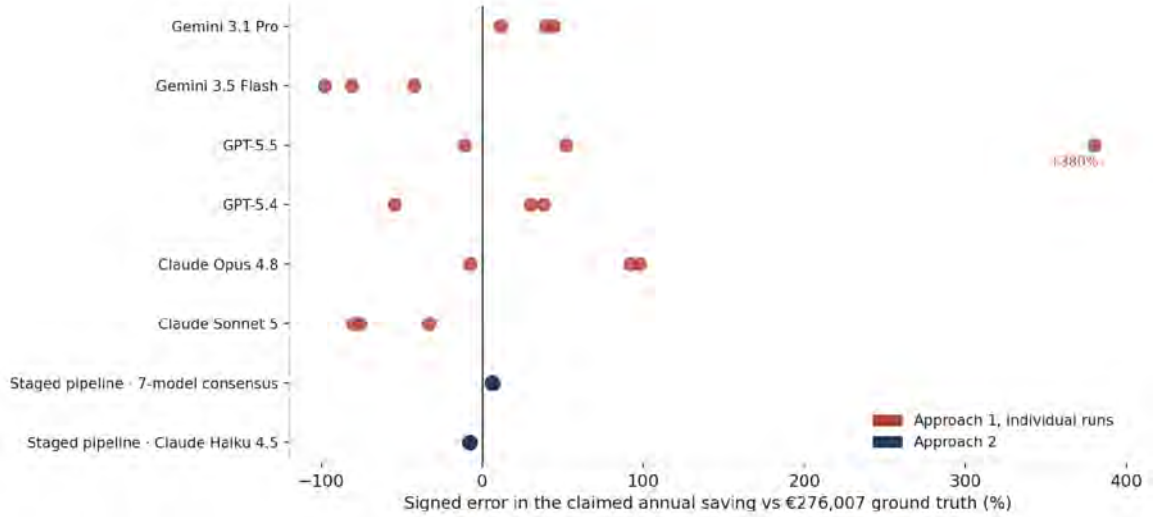

Figure 10. Signed error in the claimed annual saving against the EUR 276,007 ground truth.

8.2 The quality-assurance parallel

A useful parallel is quality assurance in radiotherapy contouring [15]. Automated output there is not trusted because it is accurate on average; it becomes usable when the workflow makes deviations visible before a consequential decision is released. Procurement savings claims need the same discipline. Approach 2 does not eliminate error; it localises it. The 6.0 percent overstatement, the stale-revision bug and Haiku's classification divergence are visible, repeatable and attributable to named stages. That is a materially different risk profile from a workflow in which the same model can alternate between a correct EUR 276,007 opportunity and EUR 1.33 million of false opportunity without signalling which answer to trust.

8.3 What the result means

The experiment supports three conclusions. First, frontier models can solve the procurement reasoning problem, but a single prompt does not make that capability dependable. Second, deterministic structure can make smaller and cheaper models commercially useful, by removing the work they should not be doing. Third, evidence quality and repeatability are not reporting features added after the analysis; they are part of the analytical method itself.

9. Limitations and roadmap

9.1 Limitations

- **Synthetic benchmark.** AGM captures fragmentation, missing references, revision drift, currency heterogeneity and structured-unstructured joins. It does not reproduce every scanned-drawing failure, free-text convention, ERP customisation or piece of tribal knowledge in a live manufacturing dataset.
- **Development and evaluation coupling.** The pipeline was developed against the benchmark on which it is evaluated. The 15 percent threshold was pre-registered and no instance-specific facts appear in the pipeline, but the same team built both, and out-of-sample datasets are required before broader performance claims.

- **Limited repetition and provider control.** Three trials per model is enough to expose instability, not to estimate production failure rates precisely; and calls were routed through OpenRouter, so the exact deployment served on a given day was outside the authors' control.
- **Known pipeline defects.** The consensus saving is 6.0 percent high; DRW-70105 is assigned the superseded revision-B mass rather than the released revision-D mass; and Haiku's file classification creates two false positives. These are retained because a trustworthy evaluation should expose the pipeline's defects, not only its wins.
- **Scope.** The study evaluates one LPP investigation. It does not show that the same pipeline detects the other six planted classes, executes a supplier negotiation, or realises savings in a live organisation. Approach 1 was also given a generous pre-aggregated input, making its reported failures conservative.

9.2 Roadmap

The immediate engineering priorities are bounded, because the relevant failure stages are already visible: revision-aware aggregation, a schema contract or one-time human confirmation for file classification, and tracing the remaining 6.0 percent savings difference to a specific cause. The next evaluation step is out-of-sample validation on new synthetic datasets with different parts, suppliers, schemas and planted values, frozen before any pipeline tuning, with an expanded model set and higher trial counts. The commercial step is a controlled customer pilot, run first in observation mode against historical decisions, with findings reviewed by procurement and engineering teams before any supplier action, and success measured by validation rate, analyst time saved, time to evidence and realised savings. Only then should the skill library expand to the other planted classes in AGM.

10. Conclusion

This study began with a simple question: can a frontier language model be handed a realistic procurement dataset and trusted to find the planted pricing inefficiencies directly? The answer is qualified. The models were capable, but the single-prompt workflow was not dependable: across 24 runs, the same task produced misses, false positives, fabricated evidence, unexamined assumptions and swings of several hundred percent in the claimed saving.

The staged pipeline changed the result without changing the underlying models. By separating identity, extraction, family selection, time basis, regression, flagging and reporting, it made the mechanical steps deterministic and the remaining judgement visible. Seven models then returned the same clean answer across 21 runs, at a fraction of the token volume and cost. AGM is synthetic and the pipeline still contains known defects, so the result is not field proof. It is evidence for an architectural principle: trust in agentic procurement is an engineering property, created through bounded judgement, reproducible calculation, explicit provenance, visible exceptions and human release. The next test is whether that discipline survives live customer data.

References

1. Deloitte. 2025 Global Chief Procurement Officer Survey: Agents of Change — Procurement's Big Bet on Digital. Deloitte Development LLC, 2025. https://www.deloitte.com/content/dam/assets-zone3/us/en/docs/services/consulting/2025/us-deloitte-2025-global-cpo-survey.pdf
2. Erriquez, M., Khushalani, S., Lietke, B., Anderson, R. and Shariff, S. Mitigating Procurement Value Leakage with Generative AI. McKinsey & Company, 9 April 2025. https://www.mckinsey.com/capabilities/operations/our-insights/mitigating-procurement-value-leakage-with-generative-ai
3. Sukharevsky, A., West, A., Catania, C. and Grande, D. How Finance Teams Are Putting AI to Work Today. McKinsey & Company, 3 November 2025. https://www.mckinsey.com/capabilities/strategy-and-corporate-finance/our-insights/how-finance-teams-are-putting-ai-to-work-today
4. Newman, W. R. and Krehbiel, T. C. Linear performance pricing: a collaborative tool for focused supply cost reduction. Journal of Purchasing and Supply Management, 13(2), 2007, pp. 152–165. https://www.econbiz.de/10003549894
5. Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F. and Liang, P. Lost in the Middle: How Language Models Use Long Contexts. Transactions of the Association for Computational Linguistics, 12, 2024, pp. 157–173. https://aclanthology.org/2024.tacl-1.9/
6. Kalai, A. T., Nachum, O., Vempala, S. S. and Zhang, E. Why Language Models Hallucinate. arXiv:2509.04664, September 2025. https://arxiv.org/abs/2509.04664
7. Kalai, A. T. and Vempala, S. S. Calibrated Language Models Must Hallucinate. Proceedings of the 56th Annual ACM Symposium on Theory of Computing (STOC), 2024. https://arxiv.org/abs/2311.14648
8. He, H. and Thinking Machines Lab. Defeating Nondeterminism in LLM Inference. Thinking Machines Lab: Connectionism, 10 September 2025. https://thinkingmachines.ai/blog/defeating-nondeterminism-in-llm-inference/
9. Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A. and Zhou, D. Self-Consistency Improves Chain of Thought Reasoning in Language Models. International Conference on Learning Representations (ICLR), 2023. https://arxiv.org/abs/2203.11171
10. Cemri, M., Pan, M. Z., Yang, S., Agrawal, L. A., Chopra, B., Tiwari, R., Keutzer, K., Parameswaran, A., Klein, D., Ramchandran, K., Zaharia, M., Gonzalez, J. E. and Stoica, I. Why Do Multi-Agent LLM Systems Fail? NeurIPS Datasets and Benchmarks Track, 2025. https://arxiv.org/abs/2503.13657
11. Anthropic. Building Effective Agents. Anthropic Engineering, 19 December 2024. https://www.anthropic.com/engineering/building-effective-agents
12. Gao, L., Madaan, A., Zhou, S., Alon, U., Liu, P., Yang, Y., Callan, J. and Neubig, G. PAL: Program-aided Language Models. ICML, 2023. https://arxiv.org/abs/2211.10435
13. Rousseeuw, P. J. and Hubert, M. Anomaly detection by robust statistics. WIREs Data Mining and Knowledge Discovery, 8(2), 2018, e1236. https://arxiv.org/abs/1707.09752
14. European Union. Regulation (EU) 2024/1689 (Artificial Intelligence Act), Article 14: Human Oversight. Official Journal of the European Union, June 2024. https://artificialintelligenceact.eu/article/14/
15. The Royal College of Radiologists. Guidance on Auto-Contouring in Radiotherapy. London: The Royal College of Radiologists, November 2024. https://www.rcr.ac.uk/media/rqjlnlny/rcr-auto-contouring-in-radiotherapy-2024.pdf

Appendix A — Dataset composition

The dataset has two layers. The agent-facing layer (agent_estate/) is everything a system under evaluation may see; nothing in it labels, flags or hints at a planted inefficiency. The ground-truth layer (ground_truth/) is used only for scoring and is never shown to any system. Table A1 gives the plants and their volumes.

Table A1. Plants and volumes.

Code	Plant	Country	Currency	PO lines	Item-master rows
BRV	Bremerhaven	Germany	EUR	7,965	338
GRZ	Graz	Austria	EUR	7,506	352
VAL	Valencia	Spain	EUR	6,855	281
OXF	Oxford	United Kingdom	GBP	8,007	370
DEB	Debrecen	Hungary	EUR	5,094	239
GRE	Greer	United States	USD	8,891	421
			Total	44,318	2,001

Purchase-order extracts. Six files, one per plant, each carrying the imprint of a different source system: German, Spanish and terse-abbreviated header languages; semicolon and comma separators; decimal commas; four date conventions; one plant with no currency column (GBP implied); one plant truncating descriptions at 24 characters. Every extract holds the same price triplet (order / contract / invoice) so that no pricing signal is lost.

Item masters. Six files carrying the drawing-number spine, damaged differently at each site: revision concatenated into the reference at Graz (DRW-70105-D); a space substituted for the hyphen at Greer (DRW 70105); roughly 4–7 percent of references left blank at Valencia, Oxford and Greer. Every item master exposes a supplier part number, the secondary identity path when the drawing reference is blank.

Reference data. supplier_master.csv (210 rows), contract_master.csv (210), drawing_register.csv (951), ebom.csv (3,033), production_plan.csv (1,260), commodity_index.csv (45) and fx_rates.csv (15). Two properties of the drawing register are load-bearing: mass_kg is blank for most released revisions (of the eighteen PP-GF30 family members, only five carry a register mass), and superseded revisions are present, with status distinguishing RELEASED from SUPERSEDED. Table A2 summarises the forty-six engineering drawings.

Table A2. Engineering drawings (46 in total), each carrying material and mass in the title block. Manufacturing process is deliberately not stated; a system must infer it from the material specification.

Group	Count	Purpose
PP-GF30 comparable family	18 (+1 superseded revision of DRW-70105)	The set the LPP analysis operates on

Group	Count	Purpose
Material twins	10	Near-identical titles, different polymer (ABS, PA66-GF35, unfilled PP, PC/ABS)
Die-cast context family	8	A clean second family (HPDC aluminium)
Steel distractors	9	A different commodity entirely

Approach 1 aggregated files. Because the raw dataset (~4M tokens) exceeds every available context window, the six PO extracts were deterministically collapsed to one row per part per plant: 2,036 rows, blind group-by, no filtering or flagging, prices kept in local currency and the three price types kept separate. The quantity column carries the full fifteen-month volume, a labelling artefact disclosed in Section 4.4.

Ground-truth files (never agent-facing): the master part registry, the identity link table, the ledger of all 77 planted instances with their twelve-month EUR values, the solved LPP reference assembly, a generation manifest with seed and SHA-256 hashes, and an integrity report of eleven automated checks, all passing. The benchmark is generated deterministically from a seeded process.

Appendix B — Planting method

This appendix documents how the inefficiencies were planted, which must be public for the benchmark to be usable. It does not document how any system detects them. All values are twelve-month EUR deltas within the annualisation window (July 2025 – June 2026), computed from counterfactual baselines stored at the moment of planting. Planted part sets are disjoint across classes, with one designed exception noted below. Table B1 gives the planting mechanism for each class, and Table B2 the two planted LPP instances.

Table B1. Planting mechanism by class.

Class	Planting mechanism
1. Cross-plant price divergence (11, EUR 1,746,033)	One plant's unit price multiplied by 1.08–1.18, held for the full window.
2. Invoice above contract (12, EUR 721,533)	For a run of 4–8 months starting between July and December 2025, invoice price set 3–8% above contract; all three prices agree outside the run.
3. Share-of-business drift (5, EUR 128,722)	Cheaper supplier's share declines linearly from 70% to 35%; the alternate supplier carries a 6–10% premium.
4. Commodity-index drift (10, EUR 1,606,992)	The steel index peaks at 118 (Sep 2025) and falls to ~101 (Mar 2026); ten indexed parts have prices frozen at the October level while contracts state an INDEXED basis.
5. Panic buying at premium (29 lines, EUR 53,032)	29 extra PO lines with 1–3 day lead times (normal 28–55), a 15–35% premium, blank contract price, and distributor suppliers.
6. Over-purchasing against plan (8, EUR 1,568,564)	Ordered quantity multiplied by 1.30–1.60 against plan-implied demand, every month.
7. Characteristic-price divergence, LPP (2, EUR 276,007)	Two parts lifted above the family's characteristic line; see Table B2.

Table B2. The two planted LPP instances.

Part	Description	Mass	Plants	Uplift	12-month value
DRW-70199	Retainer, bumper energy absorber	0.680 kg	DEB	+28%	EUR 135,531
DRW-70207	Housing, air intake resonator	0.940 kg	BRV + VAL	+21%	EUR 140,476

DRW-70207 carries the identical inflated price at both plants, from the same supplier, and is therefore invisible to cross-plant comparison. One designed overlap exists: DRW-70186 is both an LPP family member and a class-1 cross-plant instance; it is excluded from the reference regression so the fair line stays uncontaminated. Table B3 lists the designed traps within the family.

Table B3. Designed traps within the LPP family.

Trap	Part(s)	What it tests
Stale revision	DRW-70105	Register carries a mass only for superseded rev B (0.420 kg); the current rev D mass (0.455 kg) exists only on the drawing. Tests revision checking.
Missing mass	DRW-70213	Mass absent from both register and drawing. Tests refusal to fabricate.
Material twins	10 parts	Near-identical titles in other polymers. Tests material-based family selection.
Blank drawing reference	DRW-70193 (GRZ), DRW-70133 (GRE)	Reference blank at one plant; recoverable only via supplier + supplier part number. Tests secondary identity resolution.
Distractor families	8 die-cast + 9 steel	Clean comparable sets of a different process or commodity. Tests family-selection precision.

Benign noise. So that detection cannot proceed by flagging every price movement, 30 percent of non-planted parts receive a −1.5 percent long-term-agreement reduction in January 2026, FX rounding is applied at plant level, and purchase orders are split into random lots. **Verification.** Eleven automated integrity checks run before the dataset is emitted, including: planted sets are disjoint; invoice exceeds contract only on class-2 lines; non-planted parts show <3.5 percent cross-plant spread while class-1 parts show >6 percent; the LPP regression achieves $R^2 \geq 0.95$ with clean residuals within ±8 percent and outlier residuals ≥ 15 percent; and the PP-GF30 material census returns exactly eighteen parts. All eleven pass.

Appendix C — The task brief (verbatim)

Frozen before the first scored run and used byte-for-byte identically across every model, every trial and both approaches. No prompt was tuned to any model.

> *You are analysing procurement data from AutoGen Motors, a vehicle manufacturer with 6 plants in 6 countries. The data comes from different plant systems, so formats, currencies, part numbers and languages vary.*
>
> *Your task: find injection-moulded parts that are being bought at a price above what is fair for their size, and quantify the wasted spend.*
>
> *Approach the fair price by comparing genuinely comparable parts: parts made of the same material by the same process. Within such a family, a part's fair price scales with its weight. A part priced well above the family's price-for-its-weight is overpriced.*
>
> *Deliver:*
>
> *1. The specific parts you judge to be overpriced, identified by drawing number.*
>
> *2. For each, the excess unit price and the estimated excess spend per year (in EUR), showing how you calculated it.*
>
> *3. The evidence trail: which files and rows support each conclusion.*
>
> *Notes:*
>
> *- Prices are in each plant's local currency; convert to EUR using the provided fx rates.*
>
> *- The same physical part may appear under different local part numbers at different plants.*
>
> *- Some parts share similar descriptions but are made of different materials; these are not comparable and should not be mixed into the same family.*
>
> *- If a value you need is missing, say so rather than guessing.*
>
> *Work through the analysis internally. In your response, provide ONLY the final result: the overpriced parts (by drawing number), each part's excess unit price and estimated annual excess spend in EUR with the calculation shown for THAT part, and the specific files and rows that evidence each finding. Do not narrate your examination of every part you considered; report conclusions, not your full working.*

The brief provides only the commercial logic a competent analyst would already hold; it does not name PP-GF30, the planted parts, the number of findings, any statistical method, or the data's time window. The final output-format paragraph was added during setup and is disclosed: without it, reasoning models filled their visible output budget narrating per-part comparisons and never reached a conclusion; it constrains the reporting format, not the analysis, identically for every model. Finally, the phrase "scales with its weight" is a known ambiguity, compatible with either a proportional ratio or a line with an intercept; the data resolves it, and Section 5.4 discusses the runs that never tested their choice against the data.

Appendix D — Scoring rubric and full score sheet

Benchmark AGM-BM-3.0.0. Ground truth: *ground_truth_ledger.csv* (rows C7-01, C7-02) and *lpp_reference_assembly.csv*; no model under evaluation had access to either. Each run is scored on countable criteria: parts flagged, true and false positives, family membership (e.g. Injection Moulding or Casting), wrongly admitted twins, fair-price method, baseline contamination, mass accuracy, trap engagement, annualisation, savings error against both time bases, and evidence traceability. Precision is weighted above recall, and a fabricated finding is recorded as a distinct failure rather than folded into generic inaccuracy. Table E1 states the ground-truth targets and Table E2 gives the full score sheet.

Table E1. Ground-truth targets. Reference line, fitted on the 14 uncontaminated members: fair price = EUR 0.498 + EUR 4.070 × mass (kg), R^2 = 0.998.

Drawing	Description	True mass	Mass source	Plants	12-month excess (EUR)	15-month equivalent (EUR)
DRW-70199	Retainer, bumper energy absorber	0.680 kg	Drawing title block only	DEB	135,531	169,414
DRW-70207	Housing, air intake resonator	0.940 kg	Drawing title block only	BRV + VAL	140,476	175,595
Combined					276,007	345,009

Table E2. Full score sheet, 24 runs. TP = true positives (max 2); FP = false positives; Fam = family members correctly used; Contam = planted part inside the run's own reference set. Method: L = fitted line, I = interpolation, F = flat EUR/kg ratio, — = no LPP analysis.

#	Model	Trial	TP	FP	Fam	Method	Contam	Savings (EUR)	Error vs 12m	Evidence
1	Gemini 3.1 Pro	1	2	0	3	L	N	384,476	+39.3%	yes
2	Gemini 3.1 Pro	2	2	1	5	I	N	397,474	+44.0%	yes
3	Gemini 3.1 Pro	3	2	0	3	L	N	307,580	+11.4%	yes
4	Gemini 3.5 Flash	1	0	1	4	F	N	5,497	−98.0%	partial
5	Gemini 3.5 Flash	2	0	2	2	F	Y	51,955	−81.2%	partial
6	Gemini 3.5 Flash	3	0	2	0	—	N	158,425	−42.6%	partial
7	Gemini 2.5 Flash-Lite	1	0	1	4	—	N	36,400	n/a	no
8	Gemini 2.5 Flash-Lite	2	0	1	4	—	N	36,400	n/a	no
9	Gemini 2.5 Flash-Lite	3	0	1	4	—	N	36,400	n/a	no
10	GPT-5.5	1	2	0	6	L	N	245,751	−11.0%	yes
11	GPT-5.5	2	2	1	8	I	N	419,792	+52.1%	yes
12	GPT-5.5	3	0	19	4	F	N	1,325,133	+380.1%	yes
13	GPT-5.4	1	0	5	14	F	Y	359,443	+30.2%	yes
14	GPT-5.4	2	2	0	10	L	N	380,558	+37.9%	yes
15	GPT-5.4	3	0	2	0	—	N	125,699	−54.5%	partial

#	Model	Trial	TP	FP	Fam	Method	Contam	Savings (EUR)	Error vs 12m	Evidence
16	Claude Opus 4.8	1	2	1	11	L	N	528,984	+91.7%	yes
17	Claude Opus 4.8	2	2	1	13	L	N	544,755	+97.4%	yes
18	Claude Opus 4.8	3	1	0	6	F	N	255,072	−7.6%	yes
19	Claude Sonnet 5	1	1	0	17	I	N	184,708	−33.1%	yes
20	Claude Sonnet 5	2	0	1	0	—	N	66,200	−76.0%	yes
21	Claude Sonnet 5	3	0	1	0	—	N	53,528	−80.6%	yes
22–24	Claude Haiku 4.5	1–3	—	—	—	—	—	0	n/a	n/a

Haiku's runs are recorded as insufficient context, not detection failures: only 6 of 19 files fit its 200k window, with no purchase orders and no drawing register, so the task was not testable, and the model correctly reported the missing data rather than fabricating findings. Totals: 8 runs found both parts, 2 found one, 14 found neither; 39 false positives in all, 19 of them from a single run. Table E3 sets out method against outcome, and Table E4 trap engagement across the 24 runs.

Table E3. Method versus outcome. The single strongest predictor of success is the functional form chosen for the fair-price benchmark.

Fair-price method	Runs	Found both	Found one	Found neither
Fitted line (L)	7	7	0	0
Interpolation between neighbours (I)	3	2	1	0
Flat EUR/kg ratio (F)	5	0	1	4
No LPP analysis performed	6	0	0	6
Insufficient context (Haiku)	3	—	—	—

Table E4. Trap engagement across the 24 runs.

Trap	Runs caught	Outcome
DRW-70207: same inflated price at two plants	2 (Opus T3, Sonnet T1) examined it and cleared it	Fired
Baseline contamination	2 (GPT-5.4 T1, Gemini 3.5 Flash T2) put a planted part in their own reference set	Fired
DRW-70213: mass absent everywhere	1 (Gemini 3.1 Pro T2) invented 0.180 kg and computed a saving from it	Fired
Material twins	5 runs admitted twins (max 3 in one run)	Fired
DRW-70105: stale revision-B mass in register	0: models consistently preferred the drawing over the register	Did not fire
15-month window vs 12-month ledger	23 of 24 runs took the quantity at face value	Fired

Appendix E — Provider configuration and run environment

Table F1 records the models and settings as run.

Table F1. Models and settings as run.

Provider	Model	API identifier	Context	Reasoning setting	Max output
Google	Gemini 3.1 Pro Preview	gemini-3.1-pro-preview	~1M	dynamic thinking	65,536
Google	Gemini 3.5 Flash	gemini-3.5-flash	~1M	dynamic thinking	65,536
Google	Gemini 2.5 Flash-Lite	gemini-2.5-flash-lite	~1M	none	65,536
OpenAI	GPT-5.5	gpt-5.5-2026-04-23	~1M	reasoning_effort: high	max_completion_tokens
OpenAI	GPT-5.4	gpt-5.4	~1M	reasoning_effort: high	max_completion_tokens
Anthropic	Claude Opus 4.8	claude-opus-4-8	1M	extended thinking, effort high	128,000
Anthropic	Claude Sonnet 5	claude-sonnet-5	1M	extended thinking, effort high	128,000
Anthropic	Claude Haiku 4.5	claude-haiku-4-5-20251001	200k	none	16,384

Temperature was 0.0 where the API accepts it; reasoning models reject the parameter, and Anthropic requires 1.0 when extended thinking is enabled. Claude Sonnet 5 required a 128,000-token output ceiling after three earlier attempts at lower ceilings exhausted their budget without completing. GPT-5.5's first trial ran at the provider default before reasoning_effort was set explicitly; it is retained and disclosed, and it completed successfully.

Payload. Every Approach 1 run received 19 CSV files and 46 drawing images as base-64 PNGs, assembled in a fixed priority order. The identical payload counted as 376,842 tokens for OpenAI, 454,857 for Google and 550,497 for Anthropic: a 46 percent spread from tokeniser differences alone, meaning that "fits in the context window" is a property of the dataset, the model and the tokeniser together. An empirical ×1.54 correction was required for Anthropic budget estimation after tiktoken undercounts caused early payload rejections. Claude Haiku 4.5's 200k window admitted only 6 of the 19 files, dropping all purchase-order data and the drawing register, which is why its runs are scored as insufficient context.

Logging. Every run appends an immutable row recording run identifier, timestamp, approach, served model string, reasoning setting, files included and dropped, token counts, finish reason, cost, latency and raw-response path, with payload manifests recording exact file lists and SHA-256 hashes. Thirty-three additional runs were discarded before scoring and remain in the log, tagged by phase: pre-fix truncations where reasoning consumed the output budget (14), API payload and rate-limit failures (9), Sonnet output-ceiling exhaustions (3), Haiku payload rejections before the token correction (3), one GPT-5.4 trial run before reasoning was configured, and three smoke tests. None appears in the 24-run score sheet.

Cambridge Judge Business School
University of Cambridge
Trumpington Street
Cambridge
CB2 1AG
United Kingdom

T +44(0)1223 339700
enquiries@jbs.cam.ac.uk
www.jbs.cam.ac.uk