SPREADSHEETSMITH: EVALUATING AI AGENTS ON CONSTRUCTING FINANCIAL SPREADSHEETS FROM SCRATCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Spreadsheets are the lingua franca of modern finance, driving valuation, forecast, audit, and many other core workflows of the industry. As LLM agents advance to automate knowledge work, spreadsheet construction is a natural and pertinent testbed. However, most existing spreadsheet benchmarks evaluate agents on fill-in-the-blank style tasks, where an agent performs minor edits in otherwise complete spreadsheets. This protocol fails to capture the most central and time-consuming part of the spreadsheet workflow: the modeling decisions, format design, and sheet layout regular knowledge workers need to toil through to build a workbook *from scratch*. We introduce SpreadsheetSmith, a benchmark of 101 original spreadsheet construction tasks, the hardest of which is estimated to take 33 hours for a modeler with 3+ years of experience. To holistically evaluate spreadsheet construction, we introduce 129 criteria spanning 12 categories that operationalize quality beyond the accuracy of the spreadsheet. Evaluating over 20 agents across 8 models and 4 harnesses, we find that frontier agents rarely construct spreadsheets that are both accurate and meet our quality criteria: the top agent achieves a pass rate of only 13% on the benchmark. The benchmark demonstrates a substantial gap in spreadsheet building capability of current agents.

1 INTRODUCTION

Finance is a major sector of the economy. In the U.S., government agencies attribute $1.4 trillion of GDP to the financial sector, at 4.9% of the entire economy (U.S. Bureau of Economic Analysis, 2026). Among the tools the sector relies on, spreadsheets are the lingua franca of modern finance: valuation, forecasts, audit, and numerous other key financial workflows all rest on spreadsheet software (Association for Financial Professionals, 2025; CFA Institute, 2026). As advances in large language model (LLM) agents accelerate, fueling the hope for agents to automate knowledge work, spreadsheet workflows emerge as a natural and pertinent testbed for their proficiency in financial work.

A flurry of spreadsheet benchmarks have been proposed as a result. The overwhelming majority, however, evaluate agents on fill-in-the-blank style tasks (Ma et al., 2024; Krumdick et al., 2024; Choi et al., 2025; Kundurthy et al., 2026). There, agents are asked to extract answers to specific questions or to recover a few tampered formulas in an otherwise correct spreadsheet; even SpreadsheetBench 2 (Zhu et al., 2026) derives its tasks from complete workbooks with parts removed and largely grades whether the computed values match the reference.

This evaluation protocol misses the central, mundane, and time-consuming part of financial spreadsheet workflow — the modeling decisions, format design, and sheet layout that experts toil through to build a high-quality spreadsheet *from scratch*. Figure 1 illustrates this difference. Far fewer benchmarks (Patwardhan et al., 2025; Lau et al., 2026; Yen et al., 2026) evaluate this *spreadsheet construction* capability of LLM agents.

Evaluating financial spreadsheet construction poses two main challenges. The first concerns *data availability*. Finance is broad, encompassing investment banking, private equity, corporate finance, wealth management, and more, making it costly to assemble a sufficiently large and diverse set of tasks. Exacerbating the issue, real financial data is often sensitive and proprietary, which precludes

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

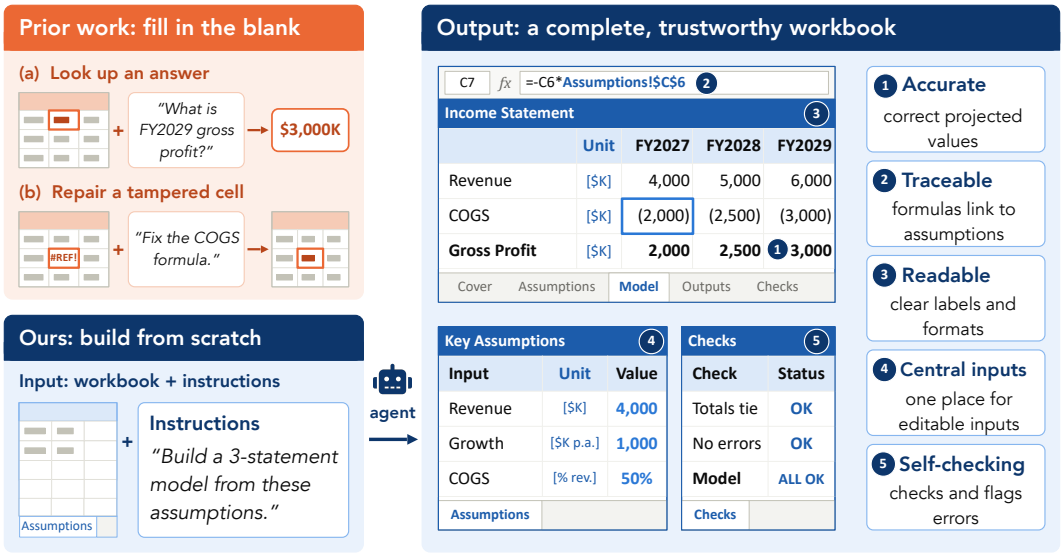

Figure 1: **Focus of SpreadsheetSmith** is to evaluate agents on constructing entire spreadsheets *from scratch*. (*Top Left*) Prior benchmarks ask agents to find answers in or correct otherwise finished workbooks. (*Bottom Left & Right*) SpreadsheetSmith instead gives agents only the inputs and instructions (like a real knowledge worker), and grades the delivered workbook on accuracy and on the quality criteria that make professional spreadsheets maintainable and reusable.

public release. Reflecting these constraints, BankerToolBench (Lau et al., 2026) focuses exclusively on investment banking, whereas MBABench (Yen et al., 2026) covers finance broadly but releases only 38 of its 202 tasks, citing intellectual-property restrictions. The second challenge concerns *evaluation*. In practice, spreadsheets serve not just as calculation devices but also as collaborative communication tools, and miscommunication can be costly. During Barclays' 2008 acquisition of Lehman Brothers assets, a junior associate accidentally led the bank to purchase 179 additional assets due to lack of clarity in the spreadsheet (Finextra Research, 2008). This communicative role of spreadsheets naturally demands additional quality criteria such as readability and auditability, beyond mere accuracy. To capture these broader quality criteria, GDPval (Patwardhan et al., 2025) employs experts to manually annotate pairwise preferences, but this approach is costly to scale and hard to reproduce. The community thus needs a public benchmark that offers both breadth and scale in its tasks, and evaluates quality criteria beyond accuracy in a transparent, scalable manner.

This paper presents SpreadsheetSmith, a fully public benchmark comprising 101 spreadsheet construction tasks that cover key financial fields including investment banking, corporate financial planning and analysis (FP&A), commercial banking, and wealth management. Similar to Lau et al. (2026) and Yen et al. (2026), SpreadsheetSmith employs a validated agentic judge to evaluate quality criteria at scale. To make quality concrete and gradable, we propose a taxonomy of 129 criteria across 12 categories, grounded in the principles of public financial modeling codes (ICAEW Excel Community, 2018); conventions these codes leave open are fixed in house standards given to every agent. For instance, `No hardcoded numbers in formulas` ensures all calculations use one coherent set of assumptions, and `No hidden rows/columns` targets the failure mode behind the Barclays incident above. Following the taxonomy of Yen et al. (2026), half of our tasks are Medium-Hard (30) or Hard (20), the tiers where frontier agents still struggle. In comparison, we have 7.5× and 10× as many difficult tasks as MBABench's public tasks (4 and 2, respectively). These tasks examine long-horizon model building ability, with estimated build times of up to **33** hours for a modeler with 3+ years of experience. Finally, we conduct a large-scale evaluation of spreadsheet construction: we evaluate over 20 agents (from selected combinations of 8 frontier models × 4 agentic harness types) for a total of 2K long-horizon trajectories, exceeding $57K in agent inference cost alone.

We find that even the leading agents pass only 13% of the tasks, whereas a substantial number of agents, with worse backbone models or harnesses, pass 0% of the time. Upon inspection, we find failure modes shared across agents, such as superficial sense checks or poor reusability and auditability (e.g., deeply nested formulas). All results suggest that current spreadsheet agents still largely fall short of delivering professional spreadsheets.

Our major contributions are as follows:

- We create and release 101 original spreadsheet construction tasks, spanning 16 task types across key fields of finance.
- We propose a taxonomy of spreadsheet quality consisting of 129 criteria across 12 categories. To enable transparent and scalable evaluation, we develop and validate a corresponding agentic judge system.
- We conduct a large-scale evaluation of over 20 agents. Our results reveal several significant shortcomings; for example, agents still fail on harder tasks, and the harness appears to substantially shape the quality of the delivered spreadsheet.
- Through qualitative analysis, we find shared and common pitfalls. For example, agents' sense checks are often superficial, as they compare results against the model's own inputs. The resulting financial models are also difficult to reuse and audit.

2 Related Work

Financial and Spreadsheet Benchmarks. Finance has long attracted interest from LLM benchmarks. Early benchmarks are predominantly QA-based, testing models on simple derivation or finance knowledge (Xie et al., 2023; 2024a; Zhang et al., 2025; Mateega et al., 2025; Lu et al., 2025). As LLM agents have improved, attention has shifted to spreadsheet manipulation that underpins several core finance workflows. Before the long-horizon capabilities of agents matured, SpreadsheetBench (Ma et al., 2024), widely used during this time, examined agents on targeted manipulations of existing spreadsheets, drawn from questions on Excel forums and graded on the resulting cell values. Many subsequent evaluations follow the same pattern, focusing on summarization, QA, and a few formula edits (Krumdick et al., 2024; Choi et al., 2025; Kundurthy et al., 2026). SpreadsheetBench 2 (Zhu et al., 2026) extends this to end-to-end workflows on large workbooks, but derives its tasks from complete workbooks with parts removed and, except for visualization tasks, grades whether the computed cell values match the reference. Constructing spreadsheets from scratch is part of GDPval (Patwardhan et al., 2025), which includes such tasks among its occupational deliverables, and is evaluated by Lau et al. (2026) for investment banking and by Yen et al. (2026) for finance in general. We build on this line of work with a fully public benchmark consisting of 101 spreadsheet construction tasks across general finance.

Agents for Professional Work and Evaluating on Quality Criteria. Agents are increasingly expected to perform economically valuable work. The first promising domain is coding (Chen et al., 2021; Jimenez et al., 2024), followed by computer use (Zhou et al., 2024; Xie et al., 2024b), legal work (Harvey AI, 2026), and other occupations from manufacturing to animation (Patwardhan et al., 2025; Sun et al., 2026). As agents' tasks become more realistic, accuracy-based evaluation no longer captures the full spectrum of desiderata in real-world work product. For example, despite advances in standard coding benchmarks, agents often produce verbose and difficult-to-maintain code (Whitfill et al., 2026; Yang et al., 2026). GDPval (Patwardhan et al., 2025) accounts for these high-level *quality criteria* in evaluation by having experts annotate pairwise preferences, at the cost of scalable evaluation. Recent coding (Lu et al., 2026) and spreadsheet benchmarks (Lau et al., 2026; Yen et al., 2026) tackle this with a mix of automatic graders and LLM-based judges. This work takes the latter approach: we first propose a taxonomy of quality criteria for financial spreadsheets, and then develop and validate a corresponding agentic judge system for reliable and scalable grading.

3 SpreadsheetSmith

What SpreadsheetSmith Evaluates. Financial modeling tasks vary widely: a 3-statement model, a discounted cash flow (DCF) valuation, and a working capital model serve drastically different pur-

poses. Yet beneath this variety, good financial models share a large common core: they should separate inputs from calculations, use transparent formulas, follow consistent formatting conventions, and include error checks so that the model is *clear* and *maintainable* to others. This commonality allows us to design a list of *universal* criteria that we apply to all tasks (e.g., `No hardcoded numbers in formulas`). In addition, we supplement the list with *conditional* criteria whose applicability depends on the specific task (e.g., `Single date driver`). This fixed criterion set keeps the grading scale consistent across tasks and types of financial model and allows the judge to be tested criterion by criterion (see Section 5.1). Our rubric thus consists of 129 criteria in total (93 universal and 36 conditional; see Figure 4).

Criteria Deep Dive. After annotating applicability, each task in SpreadsheetSmith is graded on between 105 and 126 criteria. Each criterion is further assigned to one of 12 categories and one of four importance levels, from which its weight is derived. We defer details to Appendix A. This design is *comprehensive*, as every task is checked against the common core and against the conditional criteria its model requires, and *efficient*, as a new task needs only a yes/no relevance decision per conditional criterion and a new model type only the few criteria specific to it. It is also *verifiable*: because each criterion is defined once for all tasks, the judge can be tested on each criterion individually (Section 5.1). Finally, although the applicable criteria differ between tasks, each category carries the same weight in every task, which keeps the grading scale consistent across tasks and types of financial models.

As an example, consider the conditional criterion `Single date driver` (Figure 2). This criterion applies to any task with a time component and requires a single, clearly labeled anchor date that drives all time periods and dependent calculations. A spreadsheet that satisfies this criterion allows an analyst to easily shift the entire model timeline by changing a single cell, which is particularly useful when a planned transaction is postponed. Without it, the analyst would have to painstakingly update all cells that depend on the anchor date, which risks introducing mistakes.

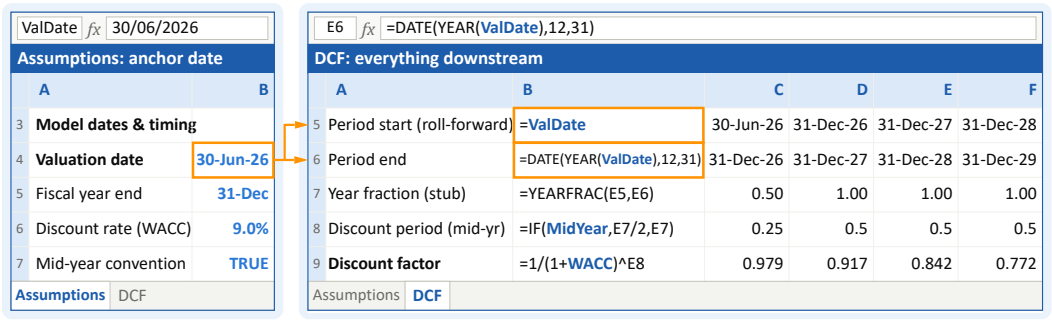

Figure 2: **Exemplar Criterion.** One labeled anchor date (*left*, `Assumptions!B4`, named `ValDate`) drives the period ends, stub period, mid-year discount periods, and roll-forwards of the DCF sheet (*right*).

How SpreadsheetSmith Evaluates. Each criterion is marked satisfied or violated. `Final calculation accuracy` is graded deterministically by comparing the agent's answers to the task's questions with the correct answers (Appendix A.4). All other criteria are graded by an agentic judge, built on GPT-5.6 Sol, because many of them require judgment in context, and fixed rules would have to anticipate every layout, such as the same table placed at B2 in one workbook and at C8 below colored headings in another. Since all tasks share one list of criteria, we were able to build the judge around this rubric; we validate it in Section 5 and describe it in Appendix E.

Metrics. Ideally, a spreadsheet satisfies all applicable criteria; we therefore report the **Pass Rate** of each agent across tasks, with a small tolerance defined below. To account for the judge's false positive rate (FPR; i.e., incorrectly flagging a satisfied criterion as violated) in grading these criteria, we require the spreadsheet to satisfy the deterministically graded `Final calculation accuracy` and to violate no more than k_i criteria for each of the four importance levels i (detailed in Appendix A.4).

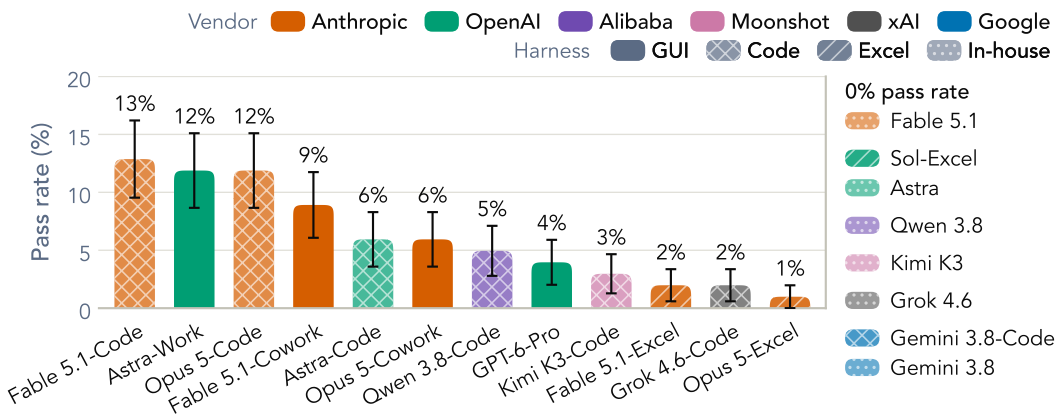

Figure 3: **Pass Rate Leaderboard.** Agents regularly fail on SpreadsheetSmith. Notably, the agentic harness appears to strongly affect how agents perform (e.g., see Fable 5.1, Opus 5 across proprietary harnesses).

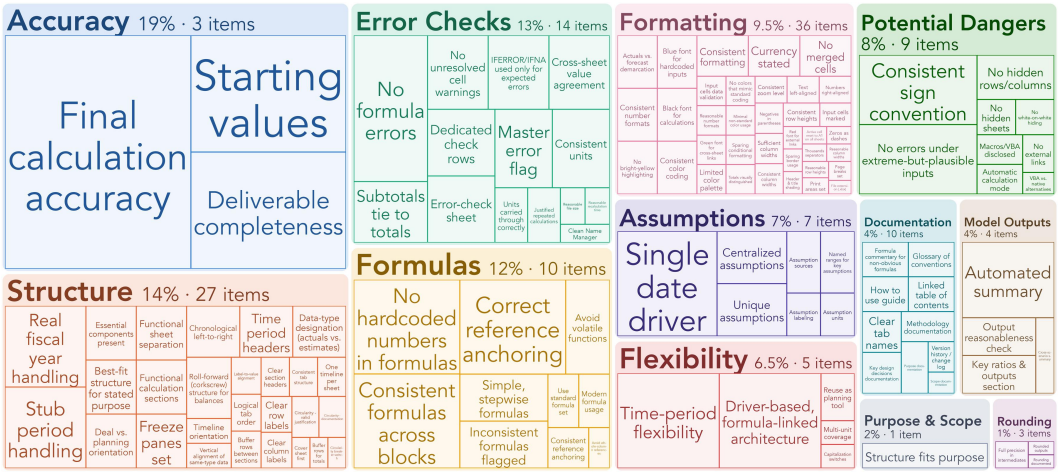

Figure 4: **The Full Rubric of SpreadsheetSmith**. We introduce a taxonomy of 129 criteria that captures what makes a financial spreadsheet trustworthy in practice.

We choose k_i by weighing criteria importance in practice, and so that a perfect spreadsheet would be deemed a pass with high probability. We adopt $k_1 = 1$ out of the 11 *ultra important* criteria other than `Final calculation accuracy`, $k_2 = 2$ out of 41 *highly important*, $k_3 = 3$ out of 52 *important*, and $k_4 = 2$ out of 24 *not absolutely crucial*; alongside the judge's empirical FPR of 1.61% (see Section 5), they result in

$$\mathbb{P}(\text{Pass} \mid \text{Perfect}) = \prod_{i=1}^{4} \mathbb{P}\left(\text{NumViolated}_i \leq k_i\right) = \prod_{i=1}^{4} F_{\text{Bin}}(k_i;\, N_i, \text{FPR}) = 0.944,$$

where $F_{\text{Bin}}(k; N, p) = \sum_{j=0}^{k} \binom{N}{j} p^j (1-p)^{N-j}$ is the binomial CDF. Note that the pass rule above is intentionally lenient; combined with the judge's higher false negative rate (Section 5), this means the pass rate will accept a workbook with genuine violations, and hence should be read as an upper bound on the percentage of acceptable workbooks.

Since a substantial number of agents have a 0% pass rate, we also report the **Score**, the sum of the weights of the criteria an attempt satisfies (Appendix A.2). Note that since many rubric criteria are related to structure or format of the spreadsheet, it can be deceptively high even for an unusable spreadsheet; submitting the input workbook unchanged would generally score around 50%. Appendix A.3 details the pass rule and why we do not report a human baseline.

Agentic Harnesses. We evaluate each model through up to four harnesses. **GUI**: the LLM vendor's own web pages (e.g., Claude Cowork). **Code**: Claude Code for Anthropic models and the open-sourced Codex CLI for all others. **Excel**: the proprietary Excel add-ins by OpenAI and Anthropic. **In-house**: a custom agentic harness that lets LLMs operate on spreadsheets via `openpyxl`. We defer details of these harnesses to Appendix G. Note that since we do not have direct access to the designs of proprietary harnesses, we do not aim to disentangle the effect of model and harness. The results serve to (1) provide practically useful observations on performance of existing agents and (2) conditioned on the same model, demonstrate that harness has an effect on agent performance.

Agent Prompt. Public financial modeling codes agree on principles but leave many conventions open, such as sign convention or number formats (ICAEW Excel Community, 2018). Firms fix these in internal guidelines, and spreadsheet agents increasingly receive such guidelines as context (Microsoft, 2026; Anthropic, 2025). We mirror this setting: every agent receives house standards, a compact markdown file written in the style of a firm's internal guidelines. Stating the target convention ensures that agents are not penalized for conventions they could not infer, and lets the judge check conformance to a known specification rather than decide whether an alternative is acceptable, which keeps verdicts comparable across agents. The rubric itself is not disclosed.

4 Task Creation

The task set targets the core work of financial modelers, building a complete model end to end, across a wide range of spreadsheet models used in finance, from company valuation and 3-statement models to FP&A, debt schedules, instrument pricing, personal finance, and business analytics. With 16 task types across seven fields, it is broader than any comparable benchmark (Figure 6). About half of the tasks (50 of 101) are Medium-Hard or Hard, with estimated build times of up to 33 hours (Appendix B.2). We release the full task set, including the golden solutions.

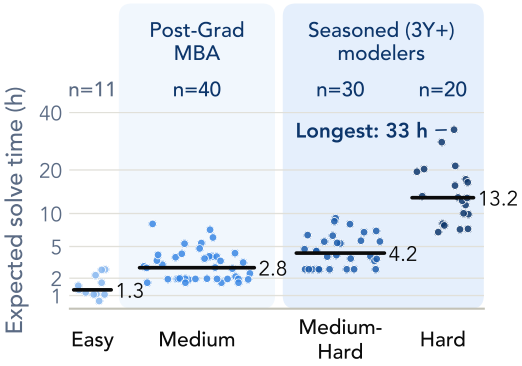

Figure 5: **Difficulty and Expected Build Time of the 101 Tasks in SpreadsheetSmith**. About half of the tasks are Medium-Hard or Hard, demanding in both modeling skill (x-axis) and estimated build time (y-axis).

All tasks were created from scratch by finance experts rather than derived from existing materials, which reduces the risk of data contamination. Each task consists of an input workbook (the *starting file* in our repository) and a *golden solution* with the completed financial model. The input workbook contains the task instructions, assumptions, supporting data where needed, and questions. The instructions are phrased as practitioners would phrase such a request and were refined iteratively to remove the ambiguities we identified. The questions, which determine `Final calculation accuracy`, each have a single, definitive answer and were sampled randomly across the entire model. All tasks passed a multi-stage quality assurance process with human and LLM-based review; Appendix C describes the task design and quality assurance process in detail.

Difficulty and Task Types. To show what the benchmark covers, we manually annotated each task with a difficulty level and a task type (Figures 5 and 6). The benchmark spans levels 2–5 of a six-level scale: *Easy* tasks can typically be solved without deep financial expertise, whereas *Hard* tasks are challenging even for seasoned modelers. Task types range from DCF valuations and 3-statement models to debt schedules and personal-finance calculators. Appendix B details both annotations, including a description of each task type.

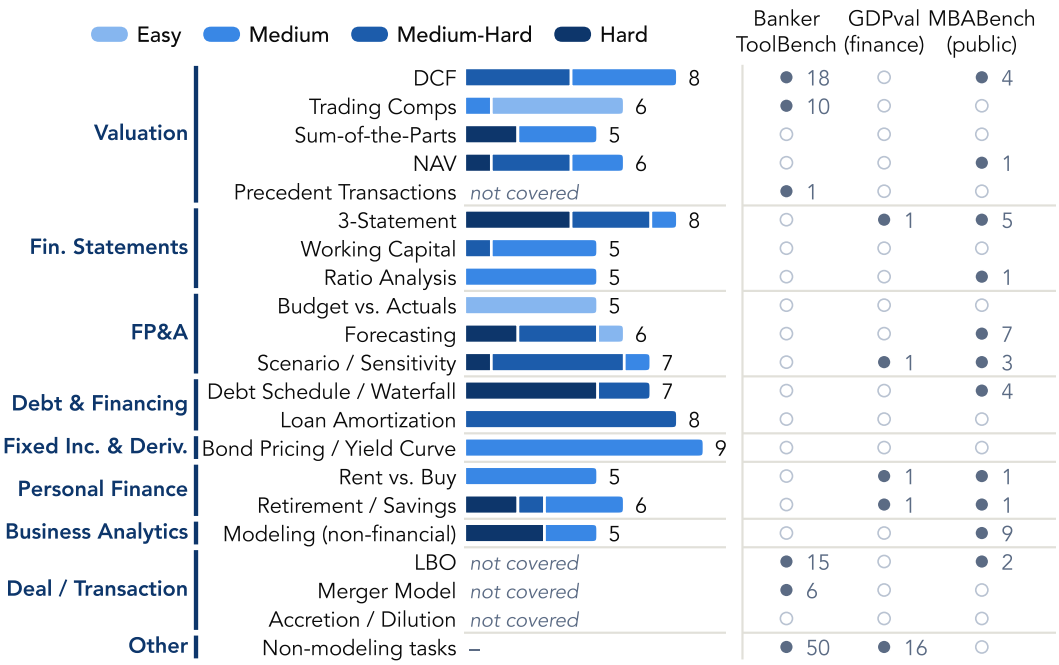

Figure 6: **Task Coverage of SpreadsheetSmith.** Compared to BankerToolBench and GDPval, we cover significantly more modeling task types. Only MBABench is comparable in coverage, but they release only 38 tasks, whereas we release all 101 tasks, with 50 Medium-Hard+ challenging ones (compared to 6 in MBABench public tasks).

5 Judge Validation

In this section, we validate the reliability of the agentic judge we employ for scalable evaluation. We approach the validation by stress-testing the judge on grading two types of attempts: (a) hand-crafted examples that satisfy or violate a specific criterion, so we can have a controlled and targeted examination of the judge's performance on individual criteria, and (b) actual attempts by evaluated agents, so we can examine the judge's performance on the attempt distribution that it sees at deploy time.

5.1 Per-Criterion Controlled Validation

We validated the agentic judge on each of the 128 criteria it grades. For each criterion, we built a pair of examples by adapting a task's golden solution: a *clean example* that clearly satisfies the criterion, and a *violating example* into which the task's author manually introduced an unambiguous but not trivially detectable violation (Appendix E.2). We ran the judge on each of the 256 examples for three repetitions. Of these, 244 (95.3%) are graded correctly in all three runs and 255 (99.6%) in at least two.

Over the 768 individual verdicts, the judge achieves 98.3% accuracy, a false positive rate of 0.26% (1 of 384 verdicts on clean examples), a false negative rate of 3.1% (12 of 384 verdicts on violations), and an F_1 of 98.3%. No category falls below 96% per-verdict accuracy on these examples.

5.2 Validation on Agent Attempts

Validation Against Experts. To evaluate the agentic judge's accuracy on real attempts of full tasks, we selected 12 attempts for annotation by human experts. Since expert annotation is extremely time-consuming, we selected the attempts from four agents spanning a range of capability (Astra-Work, Fable-Cowork, Fable-in-house, Gemini-3.8-Code). The experts graded each attempt on all applicable rubric criteria by inspecting the agent's workbook alongside the golden solution; their annotations were afterward compared with the judge's verdicts. Across them, the agentic judge

made 1,368 determinations on specific rubric criteria. Of these, the judge achieved 96.4% accuracy, an FPR of 1.61%, an FNR of 12.7% and an F_1 of 89.8%. Note that the low FPR helps the pass rate identify acceptable workbooks at a high rate.

Judge Self-Consistency. We took 90 attempts across 30 difficulty-weighted tasks from three agents, Codex (GPT-6 Astra), In-house (Fable 5.1), and Codex (Gemini 3.8 Flash), and regraded each attempt three times with the judge. Let v_a denote the variance in score for each attempt across repeated gradings. We observe the standard deviation $\sqrt{\sum_a v_a/90} = 3.1$ out of 100. Of all (attempt, criterion) pairs, 90.6% receive the same verdict in all three runs, close to the $\mathbb{P}(\mathsf{agree}) = \mathsf{acc}^3 + (1-\mathsf{acc})^3 = 89.6\%$ expected if each run were independently correct at the empirical accuracy $\mathsf{acc} = 96.4\%$. On the pass verdict, pairs of runs disagree in only 0.7% of cases. These results complement the earlier validation by speaking to the consistency of the judge.

Judge Backbone Model. We chose GPT-5.6 Sol as the judge's backbone to balance the accuracy of the judge and the cost per grading. Here, we examine how closely the Sol-based judge agrees with a stronger but more expensive model, Fable 5.1. We regraded all 303 attempts by the same set of three agents above under the Fable-based judge. On both score and pass rate, all 3 agents maintain their relative rankings. The scores from both judges have a Pearson correlation of 0.912 across attempts.

Beyond judge agreement, we share additional findings on the effect of the LLM judge. (1) **The superior judge is stricter.** Fable flags a violation in 59.7% of the assessments when the two judges disagree, scoring attempts 1.70 ± 0.38 points lower on average. This finding aligns with the qualitative observations by the experts during the initial judge selection stage of the project, where stronger judge models flagged more violations in our runs. (2) **No evidence of self-preference.** In the one comparison available, In-house (Fable 5.1) trails Codex (GPT-6 Astra) by 8.20 points under the Sol judge and by 8.53 under the Fable judge, a difference of 0.33 ± 0.67 across 303 attempts that is not statistically significant.

6 Results and Discussion

6.1 Quantitative Analysis

Pass Rate. We show the resulting leaderboard in Figure 3, and find that most attempts do not meet our pass bar. Each agent has one recorded attempt per task, following common practice in end-to-end agentic evaluation (Appendix G); we therefore report error bars to reflect the resulting uncertainty. Note that even with the same underlying model, the pass rate changes drastically (e.g., Fable 5.1 in Claude Cowork and Claude Code vs. Claude for Excel). This suggests that the harness, or the product built around it, has a major effect on the quality of the resulting spreadsheet.

Calculation Accuracy on Difficult Tasks. Here we examine agents' performance purely on whether they arrive at the correct numerical solution. Since Easy tasks are largely solved by the top agents, we primarily focus on tasks that are Medium and harder (see Figure 7). On these tasks, the top agents reach at most 54% accuracy, and on Medium-Hard and Hard tasks at most 50% (95% CI $[37\%, 63\%]$; 50 tasks).

Agent Solve Time. In general, wall-clock solve time increases with task difficulty for all agents. Among the six agents shown, those whose solve time increases least with difficulty also perform worst (see Gemini agents in the bottom middle panel of Figure 7, and how Pearson correlation in the bottom right correlates with agent performance). Agents also take far less wall-clock time than the estimated human build times, although these savings must be weighed against the share of incorrect answers.

Reasoning Effort Ablation. To evaluate the impact of reasoning effort, we ran Claude Code (Fable 5.1) at three levels of reasoning effort: Max, High, and Low. We find a positive correlation between reasoning levels and scores, with scores averaging 90.1 for Max, 88.6 for High, and 86.9 for Low.

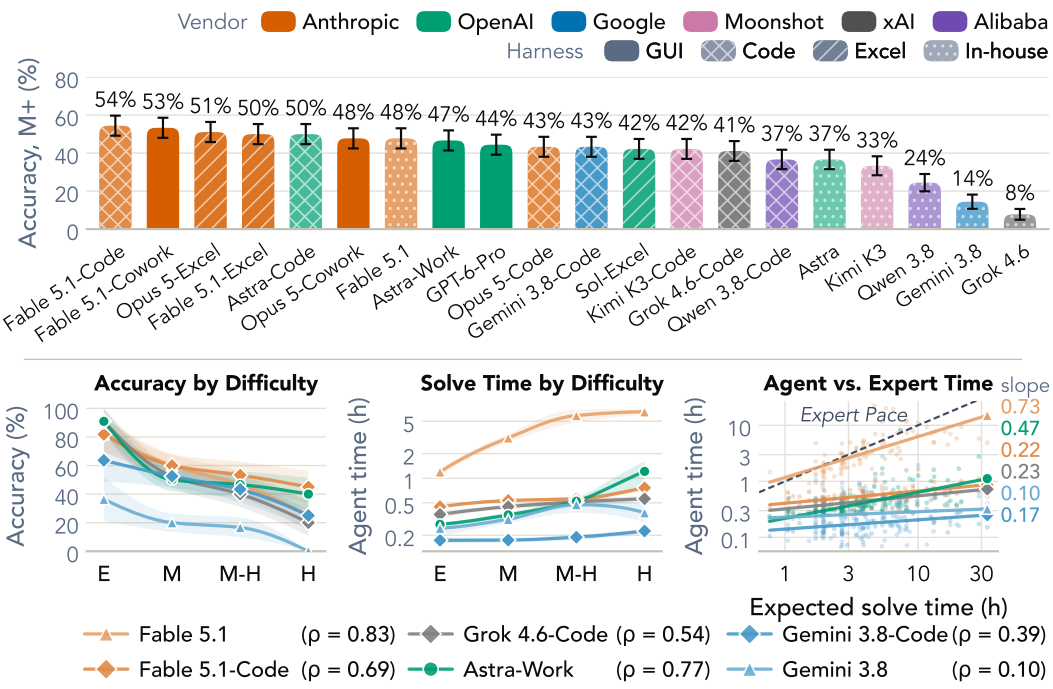

Figure 7: `Final Calculation Accuracy` **on More Difficult Subset.** *Top:* purely in accuracy, the top agent solves only 54% of the harder tasks. *Bottom Left:* Accuracy decreases significantly for harder tasks. *Bottom Mid/Right:* better agents tend to identify difficulty and spend more time accordingly. ρ denotes Pearson correlation.

6.2 Qualitative Analysis

In this section, we provide a complementary qualitative assessment of how agents fall short of professional standards. We discuss here the main shortcomings shared across agents (with more details in Appendix H) and defer agent-specific failure modes to Appendix I.

Accuracy. On more difficult tasks, the workbooks remain complete and structurally sound, but the underlying financial models are more often incorrect. Accuracy is largely stable across the commercial harnesses: in the Excel harness, the same backbone models are about as accurate as in the GUI harness, and in the Code harness, the frontier models answer about as accurately as their GUI counterparts.

Superficial Checks. A faulty financial model usually remains internally consistent and therefore passes the safety measures built into it. The agents' sense checks typically compare results against the workbook's own inputs rather than against outside benchmarks, and the master error flag may even display "OK" while individual checks fail.

Poor Reusability and Auditability. The resulting models are often hard to reuse and to audit. Timelines are often fixed to the horizon specified in the task instructions, so rerunning a financial model with a later start date or a longer duration requires manual editing. Presentation conventions such as freeze panes and color coding are applied inconsistently. On harder tasks, the logic evolves into long, nested formulas that are very hard to review, which makes it particularly difficult to pinpoint where a model goes wrong. The Excel add-ins further hurt auditability, as they omit much of the surrounding context, such as named inputs and formula commentary.

Top Agent. Even the top-ranked agent, Claude Code (Fable 5.1), which tests its outputs against outside benchmarks more often than any other agent, shows these weaknesses: it copies entire calculation engines once per scenario instead of parameterizing a single one, and it sometimes builds inflexible timelines with a fixed number of columns (Appendix I.1.1).

AI USE STATEMENT

In this work, we use generative AI tools for polishing the writing and plots, and for code development. We have not used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, clean and reformat dataset, or interpret results. We have reviewed all AI-assisted work including code and plot generation, which were checked by corresponding authors. We take responsibility for the final content of this work including text produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide code and procedure to access the data to reproduce the experiments in the following anonymous link https://anonymous.4open.science/r/SpreadsheetSmith-BCBE/.

REFERENCES

Anthropic. Advancing Claude for financial services, 2025. URL https://www.anthropic.com/news/advancing-claude-for-financial-services. Accessed: 2026-09-25.

Rahul K. Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, Johannes Heidecke, and Karan Singhal. Healthbench: Evaluating large language models towards improved human health, 2025. URL https://arxiv.org/abs/2505.08775.

Association for Financial Professionals. 2025 AFP FP&A Benchmarking Survey Report: Technology & Data. https://www.financialprofessionals.org/training-resources/resources/survey-research-economic-data/Details/afp-fpa-benchmarking-survey-report-technology, 2025. Accessed: 2026-09-01.

CFA Institute. Financial Modeling. https://www.cfainstitute.org/programs/cfa-program/candidate-resources/practical-skills-modules/financial-modeling, 2026. Accessed: 2026-09-01.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Chanyeol Choi, Jihoon Kwon, Alejandro Lopez-Lira, Chaewoon Kim, Minjae Kim, Juneha Hwang, Jaeseon Ha, Hojun Choi, Suyeol Yun, Yongjin Kim, and Yongjae Lee. Finagentbench: A benchmark dataset for agentic retrieval in financial question answering. In *Proceedings of the 6th ACM International Conference on AI in Finance*, ICAIF '25, pp. 632–637, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400722202. doi: 10.1145/3768292.3770362. URL https://doi.org/10.1145/3768292.3770362.

Finextra Research. Excel error leaves Barclays with more Lehman assets than it bargained for. https://www.finextra.com/newsarticle/19135/excel-error-leaves-barclays-with-unwanted-lehman-assets, 2008. Accessed: 2026-09-01.

Harvey AI. Harvey lab: The legal agent benchmark, 2026. URL https://github.com/harveyai/harvey-labs/tree/v1.0. Announcement: https://www.harvey.ai/blog/introducing-harveys-legal-agent-benchmark.

ICAEW Excel Community. Financial modelling code, 2018. URL https://www.icaew.com/-/media/corporate/files/technical/technology/excel-community/financial-modelling-code.ashx. Accessed: 2026-09-25.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

Michael Krumdick, Rik Koncel-Kedziorski, Viet Dac Lai, Varshini Reddy, Charles Lovering, and Chris Tanner. BizBench: A quantitative reasoning benchmark for business and finance. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8309–8332, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.452. URL https://aclanthology.org/2024.acl-long.452/.

Srivatsa Kundurthy, Clara Na, Colton Moraine, Anoushka Mohta, Case Winter, George Fang, John Ling, Emma Strubell, and Zach Kirshner. Bluefin: Benchmarking llm agents on financial spreadsheets, 2026. URL https://arxiv.org/abs/2605.30907.

Elaine Lau, Markus Dücker, Ronak Chaudhary, Hui Wen Goh, Rosemary Wei, Vaibhav Kumar, Saed Qunbar, Guram Gogia, Yi Liu, Scott Millslagle, Nasim Borazjanizadeh, Ulyana Tkachenko, Samuel Eshun Danquah, Collin Schweiker, Vijay Karumathil, Asrith Devalaraju, Varsha Sandadi, Haemi Nam, Punit Arani, Ray Epps, Abdullah Arif, Sahil Bhaiwala, Curtis Northcutt, Skyler Wang, Anish Athalye, Jonas Mueller, and Francisco Guzmán. Bankertoolbench: Evaluating ai agents in end-to-end investment banking workflows, 2026. URL https://arxiv.org/abs/2604.11304.

Eric Lu, Ben Pan, Deniz Birlikci, Sam Lee, Ray Wang, Rohan Choudhury, Fermi Ma, TC Qin, Carlo Baronio, Silas Alberti, Katie Cheng, Joseph Alessio, Claudio Costa, Martin McKeaveny, Lance Fuchia, Merlijn Vos, Tomer Nosrati, and Swyx. Introducing FrontierCode. https://cognition.com/blog/frontier-code, 2026. Accessed: 2026-09-01.

Guilong Lu, Xuntao Guo, Rongjunchen Zhang, Wenqiao Zhu, and Ji Liu. Bizfinbench: A business-driven real-world financial benchmark for evaluating llms. *arXiv preprint arXiv:2505.19457*, 2025.

Zeyao Ma, Bohan Zhang, Jing Zhang, Jifan Yu, Xiaokang Zhang, Xiaohan Zhang, Sijia Luo, Xi Wang, and Jie Tang. Spreadsheetbench: Towards challenging real world spreadsheet manipulation. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=KYxzmRLF6i.

Spencer Mateega, Carlos Georgescu, and Danny Tang. Financeqa: A benchmark for evaluating financial analysis capabilities of large language models, 2025. URL https://arxiv.org/abs/2501.18062.

Microsoft. Agents for Microsoft 365 Copilot, 2026. URL https://learn.microsoft.com/en-us/microsoft-365/copilot/extensibility/agents-overview. Accessed: 2026-09-25.

Tejal Patwardhan, Rachel Dias, Elizabeth Proehl, Grace Kim, Michele Wang, Olivia Watkins, Simón Posada Fishman, Marwan Aljubeh, Phoebe Thacker, Laurance Fauconnet, Natalie S. Kim, Patrick Chao, Samuel Miserendino, Gildas Chabot, David Li, Michael Sharman, Alexandra Barr, Amelia Glaese, and Jerry Tworek. Gdpval: Evaluating ai model performance on real-world economically valuable tasks, 2025. URL https://arxiv.org/abs/2510.04374.

Yiyou Sun, Xinyang Han, Weichen Zhang, Yuanbo Pang, Tianyu Wang, Yuhan Cao, Yixiao Huang, Chris Duroiu, Haoyun Zhang, Jeffrey Lin, Weishu Zhang, Tyler Zeng, Ying Yan, Bo Liu, Hanson Wen, Mingyang Xu, Xiaoyuan Liu, Zimeng Chen, Weiyan Shi, Amanda Dsouza, Vincent Sunn Chen, Patrick Bryant, Carl Boettiger, Yamini Rangan, Bradley Rothenberg, Kyle Steinfeld, Arvind Rao, Tapio Schneider, Georgios Yannakakis, Laure Zanna, Kaan Ozbay, Ida Sim, Tarek Zohdi, George Em Karniadakis, Jack Gallant, Teresa Head-Gordon, Yushan Li, Wenxi Deng, Tao Sun, Huiqi Wang, Zhun Wang, Justin Xu, Chris Yuhao Liu, Yafei Cheng, Rongwang Hu, Aras Bacho, Shengcao Cao, Zengyi Qin, Yixiong Chen, Hengduan Fan, Hao Liu, Lin Zeng, Shashank Muralidhar Bharadwaj, Litian Gong, Yingxuan Yang, Maojia Song, Ruheng Wang, Zongzheng Zhang, Honglin Bao, Shuo Lu, Jianhong Tu, Zhonghua Wang, Zheng Zhang, Zijiao Chen, Yanqiong Jiang, Zhendong Li, Bohan Lyu, Chang Ma, Peiran Xu, Benran Zhang, Shangding Gu, Haoyue Hua, Haoyang Li, Wanzhe Liao, Chengzhi Liu, Junbo Peng, Haoran Sun, Zechen Xu, Bo Chen, Jiayi Cheng, Yi Jiang, Keying Kuang, Yuan Li, Youbang Pan, Ziyan Rao, Alexander Schubert, Yifan Shen, Vincent Siu, Xiatao Sun, Kangqi Zhang, Xiaopan Zhang, Yuchen Zhu, Ishaan Singh Chandok, Lei Ding, Jingxuan Fan, Andrew Glover, Jiaming Hu, Yiran Hu, Wenbo Huang, Zixin Jiang, Haoran Jin, Lukas Kim, Ming Liu, Yang Liu, Alireza Rafiei, Xuhuan Shen, Kunyang Sun, Sophia Sun, Ting Sun, Eric Wang, Yixin Wang, Hanwen Xing, Sihan Xu, Yuzheng Xu, Zhongxing Xu, Zhiling Yan, Boqin Yuan, Ruiqi Zhang, Yifan Zhang, Zibo Zhao, Liana, Santanu Bosu Antu, Haoyue Bai, Carlo Bosio, Joseph Cavanagh, Patricia Cavazos-Rehg, Tianxing Chen, Xuewen Chen, Yipu Chen, Chenyu Zhu, Chen Dai, Stefano De Castro, Yunfu Deng, Kaustubh Dhole, Jiayuan Ding, Chenchen Du, Zhehang Du, Hao Fan, Run-Ze Fan, Hengyu Fu, Shi Gu, Yifan Gu, Charlie Guo, Baihe Huang, Baixiang Huang, Rimika Jaiswal, Zhihan Jiang, Ran Jin, Erin Kasson, Xin Lan, Joseph Lee, Deren Lei, Chenyu Li, Daofeng Li, Haitao Li, Hongwei Li, Jingyan Li, Xiao Li, Yi Li, Yinsheng Li, Yuangang Li, Zhixu Li, Wenyu Liang, Longtai Liao, Kevin Qinghong Lin, Andy Zeyi Liu, Che Liu, Jiaming Liu, Kaiyuan Liu, Xuan Liu, Pan Lu, Wenbo Lv, Yicheng Lyu, Qiuyang Mang, Kyle Montgomery, Yuzhou Nie, Ruoxi Ning, Jorin Overwiening, Xu Pan, Layna Paraboschi, Core Francisco Park, Justin Purnomo, Swati Rajwal, Scott Rankin, Bixuan Ren, Yiren Rong, HaoYang Shang, Ventus Shaw, Fiona Shen, Jiawei Shen, Minqi Shi, Shi Qiu, Huaxiu Yao, Tianneng Shi, Jonah So, Vladislav Susoy, Hannah Szlyk, Haocheng Wang, Jialu Wang, Wei Wang, Xinyu Wang, Zehao Wang, Dowling Wong, Angela Wu, Dehao Wu, Fangyu Wu, Mengyuan "Millie" Wu, Yu Wu, Yuchen Wu, Yuhao Wu, Qingpo Wuwu, Weihang Xiao, Yongyi Xiong, Fan Xu, Ruiling Xu, Mingxuan Yan, Benjamin Yang, Jirong Yang, Sen Yang, Xiaoli Yang, Yushi Yang, Haoran Ye, Xiaohu Yu, Zhengming Yu, Chenlong Zhang, Chi Zhang, Hanning Zhang, Hanwen Zhang, Junge Zhang, Kunpeng Zhang, Song Zhang, Wenjin Zhang, Wenshuo Zhang, Ying Zhang, Yizhi Zhang, Brian Zhao, Qijian Zhao, Yimin Zhao, Yuhaohua Zheng, Liwei Zhou, Tianyue Zhou, Sichen Zhu, Siqi Zhu, Yan Zhu, Yishu Zhu, Jierui Zuo, Chonghao Cai, Helena Casademunt, Wenjia Chen, Cheng Cheng, Nawen Deng, Rao Fu, Tianfu Fu, Yifan Han, He Ren, Zhenyu He, Qiao Jin, Langlang Li, Yuetai Li, Sylvia Liu, Lu Lu, Luqing Zhou, Subhabrata Mukherjee, Yunqi Ouyang, Yin Ren, Dawei Shi, Haoran Wu, Zhiyue Wu, Hannah Yao, Zhuoran Yi, Jenny Yu, Rhea Zhan, Hang Zhou, Blake Zhu, Junfan Zhu, Alan Yuille, Yang Liu, Russell Alan Poldrack, Jiachen Li, Zhenglu Li, Molei Tao, Jing Huang, Wenqi Shi, Costas Spanos, Lichao Sun, Chenguang Wang, Orson Xu, Zhen Dong, Hector Gomez, Aylin Caliskan, Ali Emami, Haimin Hu, Zhi Li, Lihui Liu, Murphy Niu, Yi Shao, Jianxin Sun, Mikko Tolonen, Ting Wang, Sanjiv Das, Yanjun Gao, Wenbo Guo, Erika J Schneider, Zhiyong Lu, Yian Ma, Mark Mueller, Radha Poovendran, Somayeh Sojoudi, Yinglun Zhu, and Dawn Song. Agents' last exam, 2026. URL https://arxiv.org/abs/2606.05405.

U.S. Bureau of Economic Analysis. Gross Domestic Product: Finance and Insurance (52) in the United States. https://fred.stlouisfed.org/series/USFININSNGSP, 2026. Accessed: 2026-09-01. Note: The calculation is based on data from 2024. This link shows 2.23 Trillions in GDP from Finance and Insurance. Subtracting 0.79 Trillions from Insurance obtains 1.44 Trillions in Finance. GDP percentage is calculated from 29.3 Trillions Nominal GDP.

Parker Whitfill, Cheryl Wu, Joel Becker, and Nate Rush. Many SWE-bench-Passing PRs Would Not Be Merged into Main. https://metr.org/notes/2026-03-10-many-swe-bench-passing-prs-would-not-be-merged-into-main/, 2026. Accessed: 2026-09-01.

Qianqian Xie, Weiguang Han, Xiao Zhang, Yanzhao Lai, Min Peng, Alejandro Lopez-Lira, and Jimin Huang. PIXIU: A comprehensive benchmark, instruction dataset and large language model

for finance. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=vTrRq6vCQH.

Qianqian Xie, Weiguang Han, Zhengyu Chen, Ruoyu Xiang, Xiao Zhang, Yueru He, Mengxi Xiao, Dong Li, Yongfu Dai, Duanyu Feng, Yijing Xu, Haoqiang Kang, Ziyan Kuang, Chenhan Yuan, Kailai Yang, Zheheng Luo, Tianlin Zhang, Zhiwei Liu, Guojun Xiong, Zhiyang Deng, Yuechen Jiang, Zhiyuan Yao, Haohang Li, Yangyang Yu, Gang Hu, Jiajia Huang, Xiao-Yang Liu, Alejandro Lopez-Lira, Benyou Wang, Yanzhao Lai, Hao Wang, Min Peng, Sophia Ananiadou, and Jimin Huang. Finben: a holistic financial benchmark for large language models. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024a. Curran Associates Inc. ISBN 9798331314385.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024b. URL https://openreview.net/forum?id=tN61DTr4Ed.

Tianbao Xie et al. OSWorld 2.0: Benchmarking computer use agents on long-horizon real-world tasks. *arXiv preprint arXiv:2606.29537*, 2026.

John Yang, Kilian Lieret, Jeffrey Ma, Parth Thakkar, Dmitrii Pedchenko, Sten Sootla, Emily McMilin, Pengcheng Yin, Rui Hou, Gabriel Synnaeve, Diyi Yang, and Ofir Press. Programbench: Can language models rebuild programs from scratch?, 2026. URL https://arxiv.org/abs/2605.03546.

Thomson Yen, Julian Poeltl, Harshith Srinivas Gear, Yilin Meng, Joshua Fan, Adam Shen, Yili Liu, Ali Bauyrzhan, Patrick Shea, Siri Du, Haoyang Liu, Daniel Guetta, and Hongseok Namkoong. Mbabench: Evaluating llm agents on end-to-end spreadsheet tasks in finance, 2026. URL https://arxiv.org/abs/2605.22664.

Zhihan Zhang, Yixin Cao, and Lizi Liao. XFinBench: Benchmarking LLMs in complex financial problem solving and reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 8715–8758, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.457. URL https://aclanthology.org/2025.findings-acl.457/.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=oKn9c6ytLx.

Jian Zhu, Yuzheng Zhang, Zeyao Ma, Bohan Zhang, Armin Schoepf, Daniel Woloch, Peter Yiliu Wang, Guangyu Robert Yang, Samuel Jacob, Siddharth Nagisetty, Abhiram Chundru, Jean Lin, Spencer Mateega, and Jing Zhang. Spreadsheetbench 2: Evaluating agents on end-to-end business spreadsheet workflows, 2026. URL https://arxiv.org/abs/2606.29955.

A Rubric

A.1 Rubric Design and Applicability Annotation

Motivation. Task-specific rubrics would have to restate the common core of financial models for every task; they are costly to write, inconsistent across tasks, and impractical to validate per criterion. BankerToolBench (Lau et al., 2026), for instance, uses a separate rubric for each of its 100 tasks and validates its grader only in aggregate; HealthBench (Arora et al., 2025) validates its grader per criterion only on the 34 criteria shared across conversations, not on its tens of thousands of conversation-specific ones. We instead designed a single rubric that combines universal criteria, which capture this common core, with conditional criteria, which capture what differs between models.

Criteria Creation. The rubric was developed by finance experts and covers the quality dimensions they consider essential when building and reviewing spreadsheet-based financial models. It deliberately extends beyond numerical correctness: the Accuracy category accounts for only 19% of the total weight, and the remainder goes to criteria assessing, among other things, adherence to industry-standard formatting conventions, adaptability and maintainability, and formula transparency. Each criterion is scored as satisfied or violated: it is satisfied only if no mistake is found (Appendix A.2). In the version given to the judge, each criterion consists of a name, a description, and a characterization of good and bad practice; Appendix A.4 restates these criteria in a form intended to be read, and the verbatim rubric is available in our repository. For criteria that proved prone to misinterpretation during judge validation (Section 5.1), the judge additionally receives clarifying notes, for example that a cover sheet is judged by its contents rather than by its name.

Universal and Conditional Criteria. Of the 129 criteria, 93 are universal and apply to every task; the remaining 36 are conditional and apply only to tasks whose models require the feature they check. These features include circular references (e.g., `Circularity -- valid justification`), multiple business units (`Multi-unit coverage`), alternative accounting treatments (`Capitalization switches`), stub periods (`Stub period handling`), a non-calendar fiscal year (`Real fiscal year handling`), a time dimension (e.g., `Single date driver` or `Chronological left-to-right`), multiple scenarios (`Cross-scenario summary`), and a workbook large enough to need navigation (`Linked table of contents`). Table 2 lists all conditional criteria by category; Accuracy, Formulas, Purpose & Scope, and Rounding contain only universal criteria.

Applicability Annotation. For each task, its author annotated which of the 36 conditional criteria apply to the model the task requires; universal criteria apply by definition. The annotation was completed before grading and does not depend on any agent's output. A criterion annotated as not applicable is neither presented to the judge nor scored for that task, and its weight is redistributed within its category (Appendix A.2). Between 12 and 33 conditional criteria apply to each task, so that each task is graded on between 105 and 126 criteria. Because every category contains at least one universal criterion, all 12 categories contribute to the score of every task.

Categories and Importance Levels. The experts also assigned each criterion to one of 12 categories, which group criteria by the aspect of the workbook they assess, and to one of four importance levels, which reflect how strongly a violation impairs the workbook. Appendix A.4 introduces each category, and Appendix A.2 describes how categories and importance levels are converted into weights. Of the 129 criteria, 12 are ultra important, 41 highly important, 52 important, and 24 not absolutely crucial (Table 1).

Grading. `Final calculation accuracy` is graded deterministically by comparing the agent's answers on the questions tab with those of the golden solution; a single wrong, missing, or hardcoded answer violates the criterion. The equivalence rules given in its description (Appendix A.4) disregard differences of form: an answer counts as correct if it rounds to the correct value at the required precision, differs only in display (e.g., a percentage given as a fraction, a negative number in parentheses, or a different date format), or, for costs and other outflows, differs only in sign. A different unit scale, such as thousands instead of millions, is never accepted. All other applicable criteria are graded by the agentic judge.

Relation to the House Standards. The rubric itself is not disclosed to the agents. Instead, the house standards state the conventions behind several criteria that vary between organizations, such as the change log, the linked table of contents, the checks tab with a single flag on the cover, and the sign convention, without revealing the criteria or their weights. The rubric encodes principles shared across public codes (ICAEW Excel Community, 2018); the house standards fix the conventions those codes leave open, so that criteria such as `Consistent sign convention` have a single correct target.

Advantages. This design combines the coverage of task-specific rubrics with the rigor of a single shared rubric. **Comprehensive.** Universal criteria set a common quality bar, while conditional criteria capture model-specific requirements, so every applicable rubric criterion is checked for every

task, whatever model it requires. **Efficient and extensible.** A new task requires no new criteria, only a yes/no relevance decision per conditional criterion, and even a new model type needs only the few criteria specific to it. **Verifiable.** Because each criterion is defined once for all tasks, the judge can be tested on each criterion individually (Section 5.1); and because each category carries the same weight in every task, the grading scale stays consistent across tasks and types of financial models.

Level of importance	Universal	Conditional	Total
ultra important	9	3	12
highly important	29	12	41
important	41	11	52
not absolutely crucial	14	10	24
Total	93	36	129

Table 1: Number of universal and conditional criteria per level of importance.

Category	Conditional criteria
Assumptions	Named ranges for key assumptions; Single date driver
Documentation	Linked table of contents
Error Checks	Error-check sheet; Master error flag
Flexibility	Multi-unit coverage; Capitalization switches; Reuse as planning tool; Time-period flexibility
Formatting	Actuals vs. forecast demarcation; Input cells data validation; Green font for cross-sheet links; Red font for external links; Currency stated; Print areas set; Page breaks set
Model Outputs & Executive Summary	Cross-scenario summary
Potential Dangers	Macros/VBA disclosed; VBA vs. native alternatives
Structure	Cover sheet first; Timeline orientation; Deal vs. planning orientation; Real fiscal year handling; Stub period handling; Functional sheet separation; Logical tab order; Consistent tab structure; Freeze panes set; One timeline per sheet; Chronological left-to-right; Time period headers; Data-type designation (actuals vs. estimates); Roll-forward (corkscrew) structure for balances; Circularity – valid justification; Circularity – documentation; Circularity breaker switch

Table 2: The 36 conditional criteria by category. Accuracy, Formulas, Purpose & Scope, and Rounding contain only universal criteria.

A.2 Weighting of the Rubric Criteria

The final score aggregates the 129 rubric criteria of Section 3 (93 universal and 36 conditionally applicable) into a single score, combining each criterion's score with its weight. The weighting procedure itself is a fixed formula, but the constants that go into it were set by judgment. All inputs described below, as well as the assignment of criteria to categories, were determined by our finance experts.

Base Weights. Each criterion was assigned one of four levels of importance, which determines its base weight b_i:

Level of importance	Base weight b_i
ultra important	12
highly important	5
important	2
not absolutely crucial	1

The scale is intentionally convex rather than linear: within a category, violating a single ultra important criterion outweighs violating up to five merely important ones, since the former generally

renders the workbook significantly less usable while the latter only impair its quality. Across categories, relative importance is mediated by the category weights W_c introduced below.

Discrimination Discount. Nine criteria can largely be met by retaining or restating what the input workbook already provides, such as the labeled assumptions tab or the task instructions. For these, satisfying the criterion indicates mainly that the agent preserved or carried over existing material, rather than demonstrating financial-modeling skill, so we expect them to discriminate less between strong and weak agents than criteria that must be satisfied from scratch. We retain them, because violating them is still a genuine shortcoming, but multiply their base weight by a discount factor $d_i = 0.35$. All nine are universal, so they apply to every task; the remaining 120 criteria keep their full weight ($d_i = 1$).

Normalization and Category Weights. Let $\mathcal{A}_c$ be the set of criteria in category c that apply to a given task. Within each category the discounted base weights are normalized, and the resulting in-category weight is then scaled by the fixed category weight W_c. The overall weight of criterion $i \in \mathcal{A}_c$ is

$$w_i \;=\; \underbrace{\frac{b_i\, d_i}{\sum_{j \in \mathcal{A}_c} b_j\, d_j}}_{\text{in-category weight}} \times W_c, \tag{1}$$

and the score of an attempt on a task is

$$G \;=\; \sum_{c} \sum_{i \in \mathcal{A}_c} w_i\, s_i, \qquad s_i \in \{0, 1\}, \tag{2}$$

where s_i is the score of criterion i: the judge records the mistakes it finds for each criterion, and $s_i = 1$ if it finds none and 0 otherwise.

By construction $\sum_{i \in \mathcal{A}_c} w_i = W_c$ for every category, and the category weights sum to 100% (Table 3). Neither the discrimination discount nor the exclusion of inapplicable criteria can therefore move weight between categories; both act only on the relative weights inside a category. In particular, when a conditional criterion does not apply to a task it is dropped from $\mathcal{A}_c$, the remaining criteria of that category absorb its share in proportion to their own weights, and the category still carries a total weight of exactly W_c in the final score. Every category contains at least one universal criterion, so $\mathcal{A}_c$ is never empty. Because W_c is fixed independently of how many criteria a category contains, a category with many fine-grained criteria divides its budget more finely rather than accumulating a larger share of the score: the category weights express the importance of the category itself, not of the individual checks within it. When all criteria apply, the 93 universal criteria account for 77% of the total weight, and individual criterion weights range from 0.08% (`File extension (.xlsx)`) to 10.36% (`Final calculation accuracy`).

Worked Example. Assumptions carries $W_c = 7.0\%$ and contains seven criteria (Table 3): three highly important, two important, and two not absolutely crucial. Four are discounted—two of the highly important ones and both important ones—so they enter with $5 \times 0.35 = 1.75$ and $2 \times 0.35 = 0.70$ in place of 5 and 2. If all seven apply to the task, the denominator of Equation 1 is

$$\sum_{j \in \mathcal{A}_c} b_j d_j = 1.75 + 1.75 + 0.70 + 0.70 + 1 + 1 + 5 = 11.90,$$

so a discounted highly important criterion carries $\frac{1.75}{11.90} \times 7.0\% = 1.03\%$ of the final score, a discounted important one $\frac{0.70}{11.90} \times 7.0\% = 0.41\%$, each not absolutely crucial one $\frac{1}{11.90} \times 7.0\% = 0.59\%$, and `Single date driver`, highly important but not discounted, $\frac{5}{11.90} \times 7.0\% = 2.94\%$.

The discount is visible in the first and last of these figures: `Centralized assumptions` and `Single date driver` are both highly important, yet carry 1.03% and 2.94% respectively. Had the discounted one kept its full base weight, the denominator would have been 15.15 and it would have carried 2.31%—so the discount leaves it with about 45% of the weight it would otherwise have, rather than 35%, because the reduction also shrinks the denominator it is divided by.

Two of the seven criteria are conditional. In a compact task with no global constants worth naming, `Named ranges for key assumptions` does not apply, the denominator falls to 10.90, and the remaining six absorb its share in proportion: the discounted highly important criteria rise to 1.12% and `Single date driver` to 3.21%. In both cases the category's weights still sum to 7.0%.

Category	# Criteria	W_c
Accuracy	3	19.0%
Assumptions	7	7.0%
Documentation	10	4.0%
Error Checks	14	13.0%
Flexibility	5	6.5%
Formatting	36	9.5%
Formulas	10	12.0%
Model Outputs & Executive Summary	4	4.0%
Potential Dangers	9	8.0%
Purpose & Scope	1	2.0%
Rounding	3	1.0%
Structure	27	14.0%
Total	129	100.0%

Table 3: Number of criteria and category weights W_c, in alphabetical order.

A.3 Pass Rule and Human Baseline

This section describes the standard that the pass rule reflects and explains why we do not report a human baseline, i.e., the pass rate of human modelers on the same tasks.

Standard of the Pass Rule. The pass rule requires correct results (`Final calculation accuracy`) and adherence to the modeling conventions that practitioners are trained to follow, but tolerates a limited number of violations at each importance level; this tolerance also absorbs occasional judge errors (Section 3). Several observations indicate that this standard is attainable: each criterion graded by the judge is satisfied by a golden solution, modified where necessary (Appendix E.2); the golden solutions show that every task can be solved correctly (Appendix C); and the top agents already pass some of the tasks (Section 6).

Why No Human Baseline. Like quality standards in practice, the pass rule is defined by what the work requires, not by how often a particular group of humans meets it. A human pass rate would also depend on choices for which no canonical setting exists: the experience of the modelers, the time they are given, and whether their work is reviewed before grading. Different choices could yield very different pass rates for the same tasks, so no single baseline would provide a stable reference point. Even for one setting, a reliable baseline would require several qualified modelers per task, and thus considerable resources. Instead, the benchmark relates its tasks to human capability through its annotations: the difficulty levels reflect the expertise each task demands (Appendix B.1), and the time estimates state how long a human modeler would need to build each model (Appendix B.2).

A.4 Evaluation Criteria

For readability, the criteria below paraphrase the criteria given to the judge; the verbatim rubric is available in our repository.

Each criterion is tagged with three attributes. Its level of importance is one of ultra important, highly important, important, or not absolutely crucial. A conditional criterion applies only to tasks for which it is relevant. A discounted criterion carries reduced weight because it can largely be met with what the input workbook already provides. Appendix A.2 explains how these attributes enter the weighting.

A.4.1 CATEGORY: ACCURACY

Accuracy covers whether the workbook returns the correct numbers for the task at hand, separating three parts that are frequently treated as one: correct carry-over of the task inputs, correct calculation on those inputs, and presence of every required output.

Accuracy

Correctness of outputs, inputs, and deliverable coverage

Final calculation accuracy

(ultra important · Conditional: no · Discounted: no) The final calculations must be accurate given the inputs and task instructions. Hardcoded results are not accepted, even if the value happens to be correct, because it is impossible to verify whether the result was calculated in the workbook or copied from another program; only text answers, such as Yes or No, may be entered directly. The rubric item is scored strictly: a single incorrect, missing, or hardcoded answer is enough to violate it. Differences of form, however, are not counted as errors. A numerical answer is correct if it rounds to the ground-truth value at the precision required for that answer, or deviates from it by no more than one part in a million; where both values are rounded, a deviation of one unit in the last decimal place is accepted. Whether answers are rounded as instructed is assessed under the Rounding category rather than here. A percentage may be given as a fraction or vice versa, and values are compared irrespective of their display, so that negatives in parentheses, currency symbols, thousands separators, numbers stored as text, a dash for zero, or a different date format do not matter. Answers stating that a value lies outside the model horizon, such as "N/A", are treated as equivalent to one another. For costs and other outflows, an answer that differs from the ground truth only in sign is accepted, since sign conventions vary and the house standards prescribe negative values for these items; for all other items, including net, balance, and difference figures, the sign is part of the answer. Differences in the unit scale of amounts, such as thousands instead of millions, are never accepted.

Starting values

(highly important · Conditional: no · Discounted: no) The values the model starts from must be those given in the input workbook, carried over with the correct unit and scale. Any deviation violates the criterion, whether a value differs from the input workbook or is carried over at the wrong magnitude, since an error at this point propagates into every calculation that depends on it.

Deliverable completeness

(highly important · Conditional: no · Discounted: no) Every output, schedule, and deliverable specified in the task instructions must be present and fully built out. The rubric item addresses coverage rather than correctness: a workbook that completes only part of the task, or leaves a required schedule half-built, violates the criterion regardless of how accurate the finished portions are.

A.4.2 CATEGORY: ASSUMPTIONS

Assumptions covers how the inputs that drive the model are held and presented. It asks whether the assumptions are gathered in one place and flow in a single direction into the calculations, whether each one is defined exactly once, and whether labels, units, and sources allow a reader to interpret and verify an input without tracing the formulas that use it.

Assumptions

Central, uniquely defined, and documented model inputs

Centralized assumptions

(highly important · Conditional: no · Discounted: yes) All assumptions should be summarized at the beginning of the model or in a separate assumptions sheet, rather than hidden in calculation cells or scattered throughout the model with no single source of truth. Assumptions should flow in one direction, from the assumptions sheet into the calculations and never the reverse. A clearly labeled, self-contained input block on a build sheet is permitted, provided that it is the sole source for that driver and is noted in the central assumptions sheet.

Unique assumptions

(highly important · Conditional: no · Discounted: yes) Each assumption is defined only once. Duplicates or repetitions in cells or worksheets violate the criterion, since a value stored in multiple locations may be updated in one of those locations but overlooked in the others.

Assumption labeling

(important · Conditional: no · Discounted: yes) Each assumption must have a clear, descriptive label identifying what it represents. Vague labels such as "Rate 1" or "Input" do not satisfy the criterion, as they name a cell without explaining the quantity it holds.

Assumption units

(important · Conditional: no · Discounted: yes) Each assumption must display an explicit unit of measure, such as %, $, multiple, or days, so the reader can interpret values without guessing.

Assumption sources

(not absolutely crucial · Conditional: no · Discounted: no) Each assumption references a specific, verifiable source, for example a numbered section of the task instructions or a page of a company filing, dated where a date is meaningful, so that reviewers can verify the input and judge whether it has gone stale. Where no external source exists, the assumption must be labeled explicitly as a judgment or estimate and carry an owner and a date; a bare reference to management does not satisfy the criterion.

Named ranges for key assumptions

(not absolutely crucial · Conditional: yes · Discounted: no) Truly global and stable inputs, such as the tax rate or the valuation date, should use named ranges so that formulas can be read without tracing cell references. Names must be self-explanatory and follow a consistent convention. The rubric item penalizes overuse as well as absence, since naming everything hides the underlying references and clutters the Name Manager.

Single date driver

(highly important · Conditional: yes · Discounted: no) A single, clearly labeled anchor date, that is, a valuation date, as-of date, or model start date, should drive discount periods, stub periods, mid-year conventions, and roll-forwards across the model.

A.4.3 Category: Documentation

Documentation covers what the workbook tells a reader who did not build it. It spans the statements that frame the model as a whole, that is, its purpose, scope, methodology, design decisions, and revision history, and the navigational and explanatory material that makes the sheets usable, from tab names and a linked table of contents to a glossary of conventions and commentary on individual formulas.

Documentation

Recorded purpose, conventions, and guidance for the reader

Formula commentary for non-obvious formulas

(important · Conditional: no · Discounted: no) Formulas that are not obvious or are complex should be accompanied by an explanation, such as a cell comment, a note, or an explanation in an adjacent comment column. The explanation must give the business rationale or the methodology behind the formula; a note that restates what the formula computes, such as "calculates margin", does not satisfy the criterion.

Glossary of conventions

(important · Conditional: no · Discounted: no) The workbook documents in one place the colors, abbreviations, and conventions it uses, so that a reader can decode the sheets rather than reconstruct the conventions from context.

How to use guide

(important · Conditional: no · Discounted: no) A "How to use this model" tab or section should walk users through the inputs they may change, how to select scenarios, and where to read the outputs, explaining how to operate the model from the user's perspective.

Linked table of contents

(important · Conditional: yes · Discounted: no) The model should include a table of contents with working hyperlinks to each worksheet. For very small workbooks, roughly three sheets or fewer beyond the tabs provided in the input workbook, a table of contents may reasonably be omitted.

Clear tab names

(important · Conditional: no · Discounted: no) Every tab has a clear, descriptive, professional name that states what the sheet contains. Default names such as "Sheet1" or "Sheet3" are unacceptable, as are cryptic abbreviations not explained in a glossary.

Purpose documentation

(highly important · Conditional: no · Discounted: yes) The model should include documentation that clearly articulates its intended purpose and the business question it answers.

Scope documentation

(important · Conditional: no · Discounted: yes) Documentation must explicitly list what the model covers, what it excludes, and the boundary assumptions that justify each exclusion.

Key design decisions documentation

(important · Conditional: no · Discounted: no) Documentation explains the key design decisions and the reasoning behind them, for instance the choice of valuation methodology, the granularity of the build, or the consolidation approach.

Version history / change log

(not absolutely crucial · Conditional: no · Discounted: no) The model should include a version history or change log recording what changed, when, and by whom, either on a dedicated tab or in a clearly labeled section of the cover sheet.

Methodology documentation

(important · Conditional: no · Discounted: no) Methodology documentation should explain the modeling approach, conventions, and assumptions for stakeholders and users. It may sit outside the workbook, provided the model points to it.

A.4.4 CATEGORY: ERROR CHECKS

Error Checks covers whether the model is internally consistent and whether it demonstrates that consistency to the reader. It combines arithmetic consistency, that is, subtotals matching the totals they roll into and a given value holding wherever it appears, with the absence of visible errors and warnings, disciplined use of error suppression, consistent units across the workbook, and the dedicated check rows, check sheet, and master flag that surface failures without manual tracing. A smaller set of issues relates to workbook hygiene, specifically file size, recalculation time, and the Name Manager.

Error Checks

Internal consistency, visible errors, and built-in validation

Subtotals tie to totals

(highly important · Conditional: no · Discounted: no) Subtotals must be correct sums of their line items and must tie to the totals into which they roll. The check runs against omission and double counting alike; a line item left outside a sum range, or captured twice within it, violates the criterion even where the resulting figure looks plausible.

No formula errors

(ultra important · Conditional: no · Discounted: no) The model must be free of visible Excel errors like #REF!, #N/A, #DIV/0!, #VALUE!, #NUM!, #NULL!, #NAME?, and, in modern Excel, #SPILL! and #CALC!.

No unresolved cell warnings

(highly important · Conditional: no · Discounted: no) The model should contain no unresolved cell warnings. Such warnings point to numbers stored as text, cells omitted from ranges, or inconsistent formulas, and left standing they undermine trust in the model even where the underlying figure is correct. Warnings that are known false positives, such as a year deliberately stored as text, should be dismissed or explained rather than left visible.

IFERROR/IFNA used only for expected errors

(highly important · Conditional: no · Discounted: no) Error handling such as IFERROR or IFNA should suppress only errors the model produces intentionally, for example a lookup returning #N/A in early periods before the data exists. Genuine errors must remain visible for investigation, so blanket suppression that masks them violates the criterion.

Reasonable file size

(not absolutely crucial · Conditional: no · Discounted: no) File size should be proportionate to the model's content. As a guideline, a typical single-entity operating or valuation model rarely exceeds 10 MB, and sizes above 25 MB usually indicate oversized used ranges, legacy formatting, embedded objects, or model bloat. A larger file is acceptable where the content justifies it, such as an embedded data extract, provided the reason is documented.

Reasonable recalculation time

(not absolutely crucial · Conditional: no · Discounted: no) A full recalculation in under five seconds is a reasonable benchmark for a typical model, and noticeable lag when inputs change impedes interactive use and scenario analysis. Long recalculation times commonly follow from volatile functions such as INDIRECT, OFFSET, or NOW, whole-column references, array formulas over large ranges, or overly complex formula chains.

Cross-sheet value agreement

(highly important · Conditional: no · Discounted: no) A value for a specific item must match everywhere that item appears across sheets and outputs. Discrepancies of this kind,

Error Checks *(continued)*

for instance net income differing between the income statement and the cash flow statement, are treated as serious errors rather than presentational inconsistencies.

Clean Name Manager

(not absolutely crucial · Conditional: no · Discounted: no) The Name Manager should contain only current, actively referenced defined names, with orphaned, outdated, or broken entries removed before delivery.

Dedicated check rows

(highly important · Conditional: no · Discounted: no) Dedicated check rows should validate key calculations, such as totals tying or the balance sheet balancing, so that internal consistency is visible at a glance. The rubric item is violated where the reader must re-derive totals or trace formulas manually because no row reports whether its check holds.

Error-check sheet

(highly important · Conditional: yes · Discounted: no) A dedicated error-check sheet or summary should centralize the integrity tests, among them balance checks, cash ties, and roll-forward checks, and flag failures automatically. Checks that exist only on the calculation sheets, without being summarized centrally, do not satisfy the criterion, since a reviewer must then tab through the entire workbook to assess integrity.

Master error flag

(highly important · Conditional: yes · Discounted: no) A master error flag on the cover sheet or summary sheet should immediately indicate whether a check has failed, and conditional formatting should be used to ensure that the error status is unmistakably clear.

Consistent units

(highly important · Conditional: no · Discounted: no) Units must be consistent across the model, with no mixing of scales, for example all figures in millions or all in thousands. Where a different unit is genuinely appropriate, such as a price per unit, a headcount, or a per-share figure, it must be labeled explicitly.

Units carried through correctly

(important · Conditional: no · Discounted: no) Units defined on the assumptions sheet must carry through unchanged to the calculation and output sheets. A scale change mid-model violates the criterion, except where the conversion is needed to bring all units on a calculation sheet to the same magnitude.

Justified repeated calculations

(important · Conditional: no · Discounted: no) The same calculation should not be computed in more than one place without a clear reason. Where a figure has to appear on several tabs, it is shown by cell reference rather than re-derived, so that each quantity retains a single point of calculation.

A.4.5 Category: Flexibility

Flexibility describes the extent to which the model can accommodate changes without having to be rebuilt. The key considerations here are whether the architecture is driver-based and formula-driven, whether time periods can be adjusted, extended, or re-granularized without manual effort, and whether the structure anticipates situations that would otherwise necessitate a redesign: the consolidation of multiple business units, a switch between capitalization and expense recognition, and the use of the model beyond its original intended purpose.

Flexibility

Capacity to absorb changes without rebuilding

Multi-unit coverage

(not absolutely crucial · Conditional: yes · Discounted: no) Where multiple business units, divisions, or locations are in scope, the model should use a segmented structure with consolidation logic and, where applicable, inter-company eliminations and transfer pricing.

Capitalization switches

(not absolutely crucial · Conditional: yes · Discounted: no) Where the task warrants alternative accounting treatment, switches should allow expenditures to be capitalized or expensed without rebuilding the model.

Reuse as planning tool

(important · Conditional: yes · Discounted: no) Where a model built to evaluate a business opportunity is likely to be reused afterwards, for instance as an ongoing planning tool, the design should support that second life rather than suit one-time evaluation alone.

Time-period flexibility

(ultra important · Conditional: yes · Discounted: no) Assumptions and time periods must be able to be updated, extended, or subdivided without requiring manual rework. The rubric item is violated where updating the periods means rebuilding parts of the model by hand.

Driver-based, formula-linked architecture

(ultra important · Conditional: no · Discounted: no) The model should be driver-based and formulaic, so that inputs can be changed and new line items added quickly. Reliance on hardcoded values or rigid structures that force manual rework to update or extend the model violates the criterion.

A.4.6 Category: Formatting

Formatting refers to the visual conventions that the workbook uses to indicate the function of each cell and the magnitude of each number. Many of these elements relate to industry-standard color coding, that is, blue for hardcoded values, black for calculations, green for cross-sheet links, and red for external references, as well as the need to exercise restraint so that additional colors, conditional formatting, borders, and shading do not conflict with this color coding. The remaining points concern number formats and finance conventions for negative numbers, zeros, separators, and currencies; the alignment and dimensions of rows and columns; and sheet- and workbook-level settings, such as zoom level, merged cells, print settings, and file format.

Formatting

Color-coding conventions, number formats, and visual consistency

Actuals vs. forecast demarcation

(highly important · Conditional: yes · Discounted: no) Actuals, which are historical and fixed, and forecasts, which are formula-driven from editable assumptions, should be separated by labeling, formatting, or a vertical divider, so that the point of transition between them is obvious.

Input cells marked

Formatting *(continued)*

(highly important · Conditional: no · Discounted: yes) Input cells must be visually marked, by color, shading, or border, so that a user sees immediately where values may safely be entered and which cells carry formulas.

Input cells data validation

(important · Conditional: yes · Discounted: no) Key input cells should use data validation to constrain entries to sensible ranges, types, or lists, so that an out-of-range entry is rejected or flagged at the point of entry rather than breaking the model silently.

Reasonable number formats

(important · Conditional: no · Discounted: no) Number formats should be sensible and professional: no excess decimals, currency symbols where they are needed, and thousands separators applied.

Consistent number formats

(highly important · Conditional: no · Discounted: no) Comparable values must carry the same number format throughout the model, including the number of decimals displayed.

No bright-yellow highlighting

(highly important · Conditional: no · Discounted: no) Bright yellow highlighting used during construction to flag placeholder, TBD, or work-in-progress items must be removed from the final deliverable, since it signals unfinished work. Yellow used intentionally for a defined purpose, such as conditional formatting on error-check rows, a documented convention for marking input cells, or flagging assumptions that require periodic review, remains acceptable provided the glossary or conventions guide explains it; undocumented yellow cannot be told apart from a leftover marker.

Blue font for hardcoded inputs

(highly important · Conditional: no · Discounted: no) Hardcoded inputs must use blue font, following the industry-standard color-coding convention.

Black font for calculations

(highly important · Conditional: no · Discounted: no) Formulas and calculations must use black font, following the industry-standard color-coding convention.

Green font for cross-sheet links

(important · Conditional: yes · Discounted: no) Cells that link to other sheets should use green font, following the industry-standard color-coding convention.

Red font for external links

(not absolutely crucial · Conditional: yes · Discounted: no) Cells containing external links, that is, references to other workbooks, should use red font, so that they cannot be mistaken for internal formulas or cross-sheet links.

Consistent color coding

(highly important · Conditional: no · Discounted: no) The color-coding scheme (blue inputs, black formulas, green cross-sheet links, and red external links) must be applied consistently across the entire model, so that the purpose of a cell is clear from its color alone. Inconsistent coding is worse than none, because it makes color an unreliable signal.

Consistent formatting

Formatting *(continued)*

(highly important · Conditional: no · Discounted: no) Fonts, sizes, alignment, borders, and section styling should be consistent for comparable content across all sheets. The comparison is between sheets, not only within one.

Limited color palette

(important · Conditional: no · Discounted: no) Colors that go beyond the usual standard colors of blue, black, green, and red—whether used as text color, background color, or tab color—should form a small, consistent color palette rather than a wide range of shades.

No colors that mimic standard coding

(important · Conditional: no · Discounted: no) Additional colors must not closely resemble the blue, black, green, or red of the standard convention, since a near match misleads the reader about the type of the cell.

Minimal non-standard color usage

(important · Conditional: no · Discounted: no) The number of cells and elements carrying color beyond the standard coding should be limited, so that additional formatting does not clutter the sheet or distract from the intentional coding scheme.

Sparing conditional formatting

(important · Conditional: no · Discounted: no) Conditional formatting should be confined to error-check rows, data-validation indicators, error flags, and threshold alerts, and should not override the standard color-coding convention.

Sparing border usage

(not absolutely crucial · Conditional: no · Discounted: no) Borders should be used sparingly, to partition sections and separate key areas of the model; applied throughout, they produce a cluttered appearance.

Header & title shading

(not absolutely crucial · Conditional: no · Discounted: no) Cell shading is reserved for headers, titles, and documented markers such as input cells, so that headings are visually distinct from data rows without shading spreading across whole sections.

Totals visually distinguished

(important · Conditional: no · Discounted: no) Totals and subtotals should be bolded, underlined, or otherwise set apart from the individual items that make them up.

Consistent zoom level

(important · Conditional: no · Discounted: no) Every sheet should use the same zoom level, commonly 75%, 80%, or 100%, so that the model does not jump in scale as the reader moves between tabs.

Active cell reset to A1 on all sheets

(not absolutely crucial · Conditional: no · Discounted: no) Before delivery, the cursor should be parked at cell A1, or at the designated home cell of the sheet, on every tab, so that each one opens at its intended starting position.

Text left-aligned

(important · Conditional: no · Discounted: no) Body text and row labels should be left-aligned. Column headers above numeric columns, such as period labels, are the standard exception and may be right-aligned or centered so that they sit over the values beneath them.

Formatting *(continued)*

Numbers right-aligned
(important · Conditional: no · Discounted: no) Numerical values should be right-aligned within their cells, consistently throughout the model.

Negatives in parentheses
(important · Conditional: no · Discounted: no) Negative values should be displayed in parentheses rather than with a minus sign, following finance convention.

Zeros as dashes
(not absolutely crucial · Conditional: no · Discounted: no) Zero values should be displayed as dashes rather than as the digit zero, following finance convention.

Thousands separators
(not absolutely crucial · Conditional: no · Discounted: no) Monetary and other large quantity values should carry a thousands separator. Values that conventionally carry none, namely years, percentages, and valuation multiples, are excluded, and applying a separator to them, as in a year shown as 2,024, is itself a violation.

Currency stated
(highly important · Conditional: yes · Discounted: no) The currency used in the model must be stated explicitly as a unit, and stated per block or column, which matters in particular where FX is involved. Values without a stated currency are ambiguous.

Sufficient column widths
(important · Conditional: no · Discounted: no) Column widths must be sufficient to display all values in full, with no truncated text and no ### in place of a number.

Reasonable column widths
(not absolutely crucial · Conditional: no · Discounted: no) Columns should be no wider than their content requires, so that several full-width columns fit on screen at once, and long text should be wrapped rather than accommodated by widening the column.

Consistent column widths
(important · Conditional: no · Discounted: no) Calculation columns on a sheet should share a consistent width. Header, text, and unit columns may be wider.

Consistent row heights
(important · Conditional: no · Discounted: no) Row heights should be consistent within each tab, with header rows permitted to be taller.

Reasonable row heights
(not absolutely crucial · Conditional: no · Discounted: no) Rows should by default be one line high. Taller rows are acceptable but should be used sparingly.

No merged cells
(highly important · Conditional: no · Discounted: no) Merged cells interfere with the spreadsheet's grid logic, resulting in incorrect formulas, failed sorting, and reference errors. Center Across Selection should be used instead, as it achieves the same visual effect for horizontal merges while preserving grid integrity.

Print areas set

Formatting *(continued)*

(not absolutely crucial · Conditional: yes · Discounted: no) Print areas should be defined on each tab so that printing or export to PDF captures the relevant content and nothing else, rather than empty space or extraneous material.

Page breaks set

(not absolutely crucial · Conditional: yes · Discounted: no) Page breaks should be placed so that each tab prints legibly, whether to PDF or hard copy, without splitting tables, sections, or key content awkwardly across pages.

File extension (.xlsx)

(important · Conditional: no · Discounted: yes) The workbook should be saved in a standard Excel format, that is, .xlsx, or .xlsm where VBA or macros are present. Legacy or non-standard formats, such as .xls, .csv, or an export from another spreadsheet application, violate the criterion.

A.4.7 CATEGORY: FORMULAS

Formulas covers how the calculations themselves are written: which functions are used, how much logic a single cell is allowed to carry, whether the same logic is expressed identically wherever it recurs, and whether references are anchored so that a copied formula pulls the intended cells. The rubric places greater emphasis on readability and verifiability than on conciseness, since a formula that a reviewer cannot follow cannot be verified.

Formulas

Function choice, formula construction, and referencing

Use standard formula set

(important · Conditional: no · Discounted: no) The model should keep to the formula set that is standard in financial modeling and avoid uncommon constructions such as FILTERXML, WEBSERVICE, or recursive LAMBDAs where standard alternatives exist.

Modern formula usage

(important · Conditional: no · Discounted: no) Modern functions should be preferred where they improve robustness or readability and backward compatibility is not a constraint: e.g., XLOOKUP over VLOOKUP and IFS or SWITCH over deeply nested IFs. INDEX with (X)MATCH remains a fully acceptable professional standard and is not penalized. Unless the task instructions state otherwise, the client is assumed to run a current version of Excel; where an older version must be supported, or where a legacy construction computes more efficiently in a very large model, the legacy equivalent is acceptable and the trade-off should be documented. The typical violation is a fragile legacy construction with no such reason given, most often VLOOKUP with a hardcoded column index or IFs nested many levels deep.

Avoid volatile functions

(highly important · Conditional: no · Discounted: no) Volatile functions, among them INDIRECT, OFFSET, TODAY, NOW, RAND, RANDBETWEEN, INFO, and CELL, recalculate on every change in the workbook and should be avoided where possible. Their use is acceptable where clearly justified, for instance for displaying the current date.

No hardcoded numbers in formulas

Formulas *(continued)*

(ultra important · Conditional: no · Discounted: no) Formulas must reference assumption cells rather than embed constants, as when a tax rate is written directly into a multiplication. The only accepted exceptions are fixed conversion constants, such as 12 months, 365 days, or 100 for a percentage conversion, and technical remnants that are not true inputs.

Simple, stepwise formulas

(highly important · Conditional: no · Discounted: no) Formulas should be simple, readable, and broken into logical steps through helper columns, helper rows, or LET. As a practical guideline (the three-clause rule), a formula nesting more than about three functions deep should be split, although some complexity is acceptable where splitting would bloat the structure of the model. Monolithic formulas that obscure the logic and make debugging difficult violate the criterion, as does the absence of helper columns where they are clearly required.

Consistent formulas across blocks

(ultra important · Conditional: no · Discounted: no) Formulas should be identical across a row, that is, along the timeline, and across similar blocks, so that copying works as intended and any exception is immediately obvious.

Inconsistent formulas flagged

(highly important · Conditional: no · Discounted: no) Where formulas must be inconsistent, for example in the first period of a roll-forward, the cells should be flagged visually, whether by distinctive fill, an exception-marker column, a cell comment, or an error-check flag, so that a reviewer can distinguish an intentional break from an error.

Correct reference anchoring

(ultra important · Conditional: no · Discounted: no) Absolute, relative, and mixed references ($) must be anchored correctly. Mis-anchoring is a dangerous defect because a copied formula then pulls the wrong data silently, without producing any visible Excel error.

Consistent reference anchoring

(important · Conditional: no · Discounted: no) Anchoring conventions ($) should be applied consistently across comparable formulas and blocks.

Avoid whole-column references

(not absolutely crucial · Conditional: no · Discounted: no) Whole-column and whole-row references, especially in array formulas, SUMPRODUCT, or user-defined functions, can force Excel to evaluate all 1,048,576 rows for every formula. Defined ranges, trimmed references, table columns, or named ranges should be used instead.

A.4.8 Category: Model Outputs & Executive Summary

Model Outputs & Executive Summary covers the results layer of the model, that is, the metrics it reports and the page on which they are presented. It asks whether the output metrics relevant to the model's purpose are computed in a dedicated section, whether the results are checked for plausibility against outside reference points as part of the model rather than alongside it, and whether the summary recalculates with the inputs rather than simply presenting a manually created snapshot.

Model Outputs & Executive Summary

Reported results, key metrics, and their plausibility

Output reasonableness check

(important · Conditional: no · Discounted: no) Outputs should hold up under a directional sense check against intuition, industry benchmarks, comparable companies or transactions, or third-party estimates, carried out in an explicit sense-check section of the model. The metrics to be checked follow from the model type: growth and margins for an operating model, IRR and money multiple for a transaction model, debt service coverage for project finance, payback and NPV for a capex case, implied multiples for a valuation.

Key ratios & outputs section

(important · Conditional: no · Discounted: no) The model should include a dedicated key-ratios section computing the output metrics relevant to its purpose, for example margins, returns, leverage and coverage, growth rates, valuation multiples, or transaction returns.

Cross-scenario summary

(not absolutely crucial · Conditional: yes · Discounted: no) A summary should display the results of all scenarios or cases in one view, rather than leaving them buried in separate sheets.

Automated summary

(highly important · Conditional: no · Discounted: no) The summary page must update automatically when inputs change. Values that are hardcoded or pasted in rather than formula-linked do not propagate and leave the summary to go stale.

A.4.9 Category: Potential Dangers

Potential Dangers covers the properties of a workbook that mislead a reviewer or break under use: content that is hidden rather than shown, dependencies on files outside the workbook, code that adds risk without adding capability, and settings under which displayed values need not reflect the current inputs. Two items go beyond the file itself to the model's behavior, namely the consistency of its sign convention and its stability when assumptions are pushed to the edge of the plausible range.

Potential Dangers

Concealment, fragility, and unsafe workbook settings

No hidden sheets

(important · Conditional: no · Discounted: no) The file should contain no hidden or very hidden sheets; any legitimate one must be disclosed and justified.

No hidden rows/columns

(highly important · Conditional: no · Discounted: no) The file should contain no hidden rows or columns other than those collapsed within a grouping. Groupings should be used where content needs to be collapsed.

No white-on-white hiding

(important · Conditional: no · Discounted: no) No content should be concealed by formatting, in particular white font on a white background or a number format that renders a populated cell blank.

No external links

Potential Dangers *(continued)*

(highly important · Conditional: no · Discounted: yes) The file should be self-contained, without links to other workbooks, unless the task instructions require them. Such links break when the linked file moves, produce #REF! errors, and hide dependencies.

Macros/VBA disclosed

(important · Conditional: yes · Discounted: no) Macros and VBA add complexity, security risk, and maintenance burden, and should be avoided. Any that are present must be disclosed, reviewed, and justified.

VBA vs. native alternatives

(not absolutely crucial · Conditional: yes · Discounted: no) Where VBA is present, it should handle tasks that native Excel features cannot perform, rather than duplicating them—for example, by using loops where an array formula would suffice, or by creating custom dialog boxes where data validation would be enough.

Consistent sign convention

(ultra important · Conditional: no · Discounted: no) One sign convention, such as costs negative and revenues positive, must hold across the entire model, with particular attention to EBITDA add-backs, cash flow statement conventions, and the treatment of expenses on the income statement. The same item shown with different signs in comparable places, for instance a cost entered as positive on one sheet and negative on another, violates the criterion.

No errors under extreme-but-plausible inputs

(ultra important · Conditional: no · Discounted: no) The model should still work when key assumptions are changed substantially, for instance a useful life tripled, a valuation date shifted by twelve months, or revenue halved. Stress of this kind exposes hidden breaks, balance sheet imbalances, divide-by-zero errors, broken roll-forwards, and nonsensical results such as negative shares outstanding.

Automatic calculation mode

(important · Conditional: no · Discounted: no) The workbook should be delivered with calculation set to automatic, since manual mode means the displayed values need not reflect the inputs currently shown. Where intentional circularity requires it, iterative calculation must be enabled and that requirement stated.

A.4.10 Category: Purpose & Scope

Purpose & Scope holds a single item, which judges the model as a whole against the job it was built to do. Where the other categories examine components, this one asks whether the overall architecture is the right one, given that a model can satisfy every local convention and still be built at the wrong level of detail for its purpose.

Purpose & Scope

Fit between model architecture and stated purpose

Structure fits purpose

(ultra important · Conditional: no · Discounted: no) The overall structure, that is, the calculation flow, the number of worksheets, and the level of detail, should be appropriate to the model's stated purpose. An illogical flow, too many or too few sheets, or a level of detail mismatched to the question being answered violates the criterion.

A.4.11 CATEGORY: ROUNDING

Rounding covers where in the calculation chain precision may be given up and how the resulting convention is communicated. It separates the internal treatment, where full precision must survive to the end of the chain, from the presentation of the final figures, where rounding to a sensible precision conveys the certainty the numbers actually carry.

Rounding

Precision retained, presented, and documented

Full precision in intermediates

(highly important · Conditional: no · Discounted: no) Intermediate calculations must retain full precision, with rounding applied only to the final outputs presented to users. Display formatting, such as showing no decimals, is acceptable because it leaves the underlying value untouched. Rounding inside the model is legitimate only where the quantity is genuinely discrete, for instance headcount, units sold, or share counts, or where a contract or the task instructions specify it; unjustified rounding compounds precision errors through the model.

Rounded outputs

(important · Conditional: no · Discounted: no) Final outputs should be rounded to a sensible, readable precision that communicates the appropriate level of certainty, rather than displayed at full, false precision. The task may dictate the precision to be used.

Rounding documented

(important · Conditional: no · Discounted: no) Rounding conventions should be stated in visible proximity to the rounded figures, in the manner of a heading that gives the unit and the rounding step. A convention that is wrong, undocumented, or stated far from the figures it governs violates the criterion.

A.4.12 CATEGORY: STRUCTURE

Structure covers the architecture of the workbook and the layout within each sheet. At workbook level it asks which components are present, how functions are separated across tabs, how the tabs are ordered, and whether the architecture matches the entity's fiscal calendar and the model's deal or planning orientation. At sheet level it asks how the timeline is laid out and labeled, how calculations are grouped and headed, how labels align with the values they describe, and how balances and circular references are constructed.

Structure

Sheet architecture, timeline layout, and section organization

Essential components present

(highly important · Conditional: no · Discounted: no) The model should include all essential components: cover sheet, table of contents, summary or dashboard, assumptions, scenarios where applicable, calculation sheets, financial statements where applicable, error-check sheet, and output data. The components scale with the size and purpose of the model, so in a small model several may reasonably sit on fewer sheets, provided each function remains clearly delineated.

Cover sheet first

Structure *(continued)*

(not absolutely crucial · Conditional: yes · Discounted: no) The cover sheet should be the first tab, setting the context and orienting the user on opening, and typically stating the model's title, purpose, owner, version, and date.

Timeline orientation

(important · Conditional: yes · Discounted: no) Within calculation tabs the timeline should run horizontally as standard, with periods across columns and line items down rows, applied consistently. Period columns should align across tabs, so that a given column refers to the same period everywhere in the workbook, absent a shift in period length; misalignment between sheets creates link risk. A vertical timeline is acceptable where the content justifies it, typically a register or data table whose records need several attribute columns visible at once and whose periods are few enough to scan, but it should not be the default, since it forces vertical scrolling to compare periods and breaks the copy-across formula convention.

Vertical alignment of same-type data

(important · Conditional: no · Discounted: no) Columns and rows carrying the same type of information should be aligned vertically across the sheet, and the arrangement of the different calculations should follow a logical flow.

Label-to-value alignment

(important · Conditional: no · Discounted: no) Subtitles, categories, and calculation names should be aligned with the values they describe, so that the reader can associate each label with its data without ambiguity.

Best-fit structure for stated purpose

(highly important · Conditional: no · Discounted: no) Beyond being acceptable, the structure should be the best fit for the model's specific requirements, its audience, and its foreseeable future uses. The rubric item is violated only where a clearly better alternative exists and can be articulated, and where the chosen structure creates avoidable friction.

Deal vs. planning orientation

(highly important · Conditional: yes · Discounted: no) The calculation architecture should reflect whether the model serves one-time deal evaluation or ongoing planning, since the two differ in data-refresh cadence, flexibility, and depth.

Real fiscal year handling

(ultra important · Conditional: yes · Discounted: no) The model must follow the entity's real fiscal calendar, including non-calendar fiscal years such as July to June. Where the task instructions specify no fiscal calendar, a calendar-year assumption is acceptable provided it is stated.

Stub period handling

(ultra important · Conditional: yes · Discounted: no) Stub periods, that is, partial periods at the start or end of a forecast, must be calculated with a day-count convention, such as act/365, act/360, or 30/360, appropriate to the model's purpose, the task instructions, and the jurisdiction.

Functional sheet separation

(highly important · Conditional: yes · Discounted: no) Cover, inputs, calculations, analyses, and outputs should each sit on their own sheet, with no mixing of functions on a single sheet unless there is a clear structural reason.

Logical tab order

Structure *(continued)*

(important · Conditional: yes · Discounted: no) Tabs should be ordered to follow the flow of the model, from cover and table of contents, through assumptions and inputs, calculations, financial statements, analyses and scenarios, to outputs and summary, and finally the error checks.

Functional calculation sections

(highly important · Conditional: no · Discounted: no) Calculations should be split into clearly labeled functional sections, for example revenue, COGS, operating expenses, capital expenditure, debt, and equity, rather than run together as one undifferentiated block.

Buffer rows for totals

(not absolutely crucial · Conditional: no · Discounted: no) Buffer rows or columns should sit above or beside totals, so that new line items can be inserted without breaking the sum formulas.

Buffer rows between sections

(important · Conditional: no · Discounted: no) Major sections should be separated by buffer rows rather than running together.

Clear section headers

(important · Conditional: no · Discounted: no) Each major component of the model should carry a clear, descriptive section header.

Consistent tab structure

(important · Conditional: yes · Discounted: no) Similar tabs should share a layout, using the same columns for headings, sub-headings, units, and subtotals.

Freeze panes set

(highly important · Conditional: yes · Discounted: no) Freeze panes should keep row labels and the timeline visible while scrolling. They should not be set excessively either: no more than about two thirds of a regular screen view should be taken up by the frozen area, so that enough of the calculations remains visible.

One timeline per sheet

(important · Conditional: yes · Discounted: no) Each worksheet should contain a single timeline, since several on one sheet create ambiguity about the period a calculation refers to. Where a sheet genuinely needs a second periodicity, such as an annual summary beneath a quarterly build, the blocks must be clearly separated, labeled, and justified.

Chronological left-to-right

(highly important · Conditional: yes · Discounted: no) The columns should be arranged chronologically from left to right, with the earliest time periods on the left and those furthest in the future on the right.

Time period headers

(highly important · Conditional: yes · Discounted: no) Header rows should state the time period of each column, so that the reader knows immediately which period a column of data represents.

Data-type designation (actuals vs. estimates)

(highly important · Conditional: yes · Discounted: no) Header rows should indicate whether the values in a column are actuals, estimates or forecasts, or another type of data, whether

Structure *(continued)*

by appending A and E to the period label or by a separate designation row. Without it the reader cannot separate historical from projected data.

Clear row labels

(important · Conditional: no · Discounted: no) Row labels should be clear, descriptive, professional, correctly spelled, and expressed in standard financial terminology appropriate to the target audience.

Clear column labels

(important · Conditional: no · Discounted: no) Column labels should be clear, descriptive, professional, correctly spelled, and expressed in standard financial terminology appropriate to the target audience.

Roll-forward (corkscrew) structure for balances

(highly important · Conditional: yes · Discounted: no) Balance items that move over time, among them debt, cash, PP&E, equity, provisions, and share count, should be built as explicit roll-forwards: opening balance, additions, deductions, closing balance, with each movement on its own line and each period's closing balance feeding the next period's opening balance. A compressed single formula, or an opening balance that does not equal the prior closing balance, violates the criterion.

Circularity – valid justification

(important · Conditional: yes · Discounted: no) Any circular reference in the model must rest on a clear and valid modeling reason, such as an average debt balance, a revolver sweep, an iterative WACC, or interest on average cash. An unjustified circular reference points to a formula error or poor design.

Circularity – documentation

(important · Conditional: yes · Discounted: no) Any intentional circular reference must be documented, by cell comment or model note, explaining its purpose, the mechanics of the circularity, and how it resolves. Undocumented circularity leaves a reviewer unable to tell the intentional from the accidental.

Circularity breaker switch

(not absolutely crucial · Conditional: yes · Discounted: no) Each intentional circular reference should have a circuit-breaker switch for debugging, that is, a flag cell that toggles the circularity off and replaces the circular formula with zero or a static value.

B Task Annotation

Tasks of this kind are costly to create: each is authored from scratch by a finance expert, requires a long build time (Appendix B.2), and passes several review rounds (Appendix C). Individual difficulty levels and task types therefore contain limited numbers of tasks. We accordingly focus difficulty-related findings on the harder levels jointly and report confidence intervals where we discuss a single level (Section 6), and we use task types to describe the benchmark's coverage rather than to compare agents across types.

B.1 Task Difficulty

B.1.1 Procedure and Weighting

Every task was scored in each of four categories: Scope, General Modeling, Finance Knowledge, and Implementation in Excel. For all four categories, a numerical score between 1 and 6 was as-

signed based on the classes described in the subsections below. The scores were then weighted, and the weighted sum was rounded to the nearest whole number to determine the overall difficulty. Halves were rounded up, so a weighted sum of 3.5 yielded level 4 (Medium-Hard). One task reached a rounded score of 6 (Very Hard); since a level containing a single task would not be representative, we reclassified it as Hard, so that no task is labeled Very Hard.

Difficulty Category Weights

Scoring procedure with four categories

Category	Weight
Scope	$\frac{1}{3}$
General Modeling	$\frac{1}{3}$
Finance Knowledge	$\frac{1}{6}$
Implementation in Excel	$\frac{1}{6}$

The rounded weighted sum maps to a difficulty level:

1	Very Easy
2	Easy
3	Medium
4	Medium-Hard
5	Hard
6	Very Hard

B.1.2 SCOPE

Scope measures the breadth and structural complexity of the task (how many components, steps, or interdependent pieces the task contains). While it is correlated with the task solving duration, it is not derived from it alone, since it takes into account not only the extent but also the complexity.

Scope Difficulty Levels

Breadth and structural complexity of the task

1 — Very Easy

Single step or very small task, usually one output. This class is used if the task is very small or even only a single calculation with a single number as the required output.

2 — Easy

Small but multi-step task; limited tabs or exhibits. This class is used if the task is still not extensive, but at least involves multiple steps to reach the solution.

3 — Medium

Standard professional case exercise. This class is used for a standard task involving multiple steps and at least 30 minutes of solving time in a focused context.

4 — Medium-Hard

Broader professional case exercise. This class is used for more extensive tasks involving at least 1 hour of solving time in a focused context.

5 — Hard

Complex, multi-part case with interdependent components. This class is used for problems that take longer than an hour to solve and are also highly complex, involving several interdependent parts.

Scope Difficulty Levels *(continued)*

6 — **Very Hard**
Very complex case with many moving parts and interdependencies. This class is used for very complex tasks requiring several hours of deep work to solve as well as involving several complex, highly interdependent parts.

B.1.3 General Modeling

General Modeling measures the technical difficulty of the quantitative work itself (the sophistication of the calculations, logic, and model architecture required). The scale captures the depth of analytical rigor: how linked, layered, and error-prone the calculations are, and how much modeling knowledge and experience are required to solve the task.

General Modeling Difficulty Levels

Technical difficulty of the quantitative work

1 — **Very Easy**
Basic arithmetic / direct formula application. This class is used if the modeling is straightforward and might even be done efficiently with pen and paper or using a calculator.

2 — **Easy**
Straightforward financial calculations, limited dependencies. This class is used if the modeling is still straightforward, but several calculations are based on each other.

3 — **Medium**
Multiple linked calculations and assumptions. This class is used for a standard financial model that includes multiple interrelated calculations and assumptions.

4 — **Medium-Hard**
Non-trivial structure. This class is used if the modeling is beyond standard and, for example, requires modeling across several interrelated tabs.

5 — **Hard**
Advanced model logic; high risk of errors without careful checks. This class is used for complex tasks in which the calculations are highly interrelated.

6 — **Very Hard**
Sophisticated architecture and/or novel approach design. This class is used for highly complex tasks that require extensive experience in financial modeling to solve, given the complexity and interdependence of the task.

B.1.4 Finance Knowledge

The category Finance Knowledge determines how much (and to what depth) finance knowledge is needed to solve the task. This determination also incorporates potential assistance given in the task instructions, while not completely dismissing the need for finance knowledge if the instructions provide such assistance.

Finance Knowledge Difficulty Levels

Depth of finance knowledge required to solve the task

1 — **Very Easy**

No finance knowledge necessary. This class is used if no finance knowledge is needed to solve the task.

2 — **Easy**

Basic finance knowledge necessary. This class is used if very basic finance knowledge is needed, such as basic principles of compounding.

3 — **Medium**

Standard finance knowledge necessary. This class is used if finance knowledge acquired within the regular scope of a business degree is needed, i.e., the finance knowledge is necessary but also sufficient to complete a trivial (integrated) 3-statement model.

4 — **Medium-Hard**

Extensive finance knowledge necessary (topic is still relatively regular). This class is used if the finance knowledge is still not niche, but not acquired via a regular business degree (e.g., transaction models like LBOs).

5 — **Hard**

More exotic topic with needed prior finance knowledge. This class is used if the topic is more exotic and work experience in the specialized field is necessary to gain the needed knowledge.

6 — **Very Hard**

Very niche topic on an advanced level (extended topic knowledge necessary). This class is used if the topic is not only very specialized, but also the needed knowledge is deep, which is typically acquired only through extensive specialized work experience.

B.1.5 Implementation in Excel

Regardless of the overall model architecture, converting the input data into a viable solution in conjunction with financial expertise requires a certain level of proficiency in Microsoft Excel. While some tasks require only basic arithmetic in Excel, others cannot be completed without a solid understanding of the latest features.

Excel Implementation Difficulty Levels

Level of Excel proficiency required

1 — **Very Easy**

No Excel necessary / very basic (arithmetic). This class is used if only basic arithmetic in Excel is needed.

2 — **Easy**

Basic Excel knowledge necessary (e.g., lookups). This class is used if foundational formulas like lookups are needed to solve the task.

3 — **Medium**

Moderate Excel knowledge necessary (nested formulas starting to get involved). This class is used if functions are not only used but also combined into nested formulas. In general,

Excel Implementation Difficulty Levels *(continued)*

this level corresponds to the standard proficiency of a seasoned and experienced Excel user in a business setting.

4 **— Medium-Hard**

Intermediate Excel knowledge necessary or knowledge of financial Excel formulas. This class is used if either knowledge of the financial formulas in Excel is necessary to solve the task or if knowledge exceeding the standard proficiency of a business user is needed (such as the usage of Data Tables or the FILTER function).

5 **— Hard**

Advanced and modern Excel knowledge a plus (e.g., array manipulation). This class is used if advanced and novel Excel knowledge is needed to solve the task properly and in an efficient manner, though the task might still be solvable without such knowledge.

6 **— Very Hard**

Advanced and modern Excel knowledge needed (e.g., array manipulation). This class is used if the task is not solvable without knowledge of advanced functions.

B.1.6 Exemplary Tasks

Exemplary Tasks by Difficulty Level

Illustrative tasks for each difficulty level

Very Easy

No task was classified as Very Easy.

Easy — `Pharma` (task 97)

Small classical comparables valuation with given comparable companies and provided data. Involves only a few steps with linked but straightforward calculations. Only standard finance knowledge is required, as well as moderate Excel knowledge.

Medium — `EasyDCF` (task 20)

Small DCF model with given assumptions and iterative WACC determination. The task is Medium in all four categories: a standard case exercise with multiple linked calculations and assumptions, requiring standard finance knowledge and nested Excel formulas.

Medium-Hard — `DayMonthYear` (task 79)

A very extensive DCF model with several submodels and two ways of calculating the company value. The model requires building a fully fledged preceding 3-statement model. Complex with a non-trivial structure due to the combinatorial nature. Solving the task requires both extensive finance knowledge and intermediate Excel skills.

Hard — `DailyDetail` (task 77)

A very complex and extensive debt waterfall model as used in corporate treasury including interest, FCF forecasting, and specific payout rules. It involves several interdependencies and multiple scenarios. The interconnected model and the model type (financing waterfall) make it very prone to errors. Solving the task requires finance knowledge beyond the standard level, due to the interest rate conventions involved, and advanced knowledge of the newest Excel functions is a plus.

Exemplary Tasks by Difficulty Level *(continued)*

Very Hard

No task is classified as Very Hard; the only task that reached this level was reclassified as Hard (see Appendix B.1.1).

B.2 Time Estimates

We decided to use time estimates determined by humans after we concluded that initial attempts with AI-based time estimates were not reliable enough. The (human) time required to complete each task was estimated immediately after its creation.

The estimate covers the total time required to complete the task, including answering the questions and performing initial plausibility checks and quality assurance. The reference was a modeler with moderate experience—approximately 3+ years in financial modeling—who had already created similar models but might still need to double-check some more complex aspects or consult how-to sources.

For every model, the time was broken down into the following pieces:

- Reading and interpreting the instructions
- Understanding the given assumptions
- Understanding the structure the model should have
- Planning the layout / model architecture
- Building the model (see details below)
- Answering questions and cross-checking
- Debugging / sanity checks (initial QA before handing off to superiors)

A step may also have an estimated duration of 0 minutes if it is not required for certain (particularly small) models.

The main part of the estimate—and generally the most time-consuming part—is model building. The time required for model building was estimated and broken down individually for each task. The model development process was divided into model-specific sub-steps; a time range was then estimated for each sub-step, with the average value of each range used as the basis for the time estimate, and all sub-step estimates were added together.

B.3 Task Types

The following entries describe the different financial models that we created as tasks. The task type was specified directly during the creation process.

Valuation Models

Methods for estimating the value of a company or asset

DCF (Discounted Cash Flow)

Estimates the intrinsic value by forecasting future cash flows and discounting them to present value using a risk-adjusted discount rate. This method is highly dependent on assumptions and focuses on the company's intrinsic ability to generate cash flow over time. It is one of the most commonly used valuation methods.

Comparables (Trading Comps)

Valuation Models *(continued)*

Values a company based on the market valuation of similar publicly traded companies. It relies on metrics such as EV/EBITDA or P/E ratio to perform a relative valuation to determine the value of the company being evaluated.

Sum-of-the-Parts Valuation

Breaks down a diversified company into individual business segments and values each separately. The total enterprise value is the sum of all segment valuations. The valuation methods used might be different for each part.

NAV (Net Asset Value) Modeling

Values a company based on the fair value of its underlying assets minus liabilities. Frequently used for asset-intensive companies such as real estate or mining firms, and for investment funds.

Financial Statement Models

Models integrating the three financial statements and related analyses

3-Statement Model

Integrates the income statement, balance sheet, and cash flow statement into a fully interconnected financial model. It serves as the foundation for most financial analyses and other modeling tasks.

Working Capital Modeling

Forecasts short-term operating assets and liabilities (receivables, payables, inventory) as a standalone liquidity analysis. Differs from the complete 3-statement model in that it does not necessarily include the income statement or long-term items.

Ratio Analysis

Applies predefined financial ratios to existing financial statements to interpret performance rather than forecast it. Unlike a 3-statement model, it does not provide forecasts but is used for diagnosis and benchmarking.

FP&A Models

Financial planning and analysis models

Budget vs. Actuals Variance Analysis

Compares planned financial performance with actual results to identify variances. The purpose is to help management understand the factors driving performance and improve forecast accuracy.

Forecasting Model

Projects a company's future financial performance—revenue, expenses, cash flow, and other key metrics—based on historical trends, business drivers, and stated assumptions. It serves as the baseline plan for budgeting and strategic decision-making.

Scenario / Sensitivity Analysis

FP&A Models *(continued)*

Scenario analysis defines a set of specific, named cases (base/bull/bear scenarios) to delineate the range of possible outcomes. Sensitivity analysis isolates individual variables to demonstrate the extent to which a single assumption influences the outcome. Both are built on a base model and are often not created as standalone models.

Debt & Financing Models

Financing structure and capital models

Debt Schedule / Waterfall Modeling

Tracks debt levels, interest, and repayments over time, often across multiple tranches. Waterfall structures show how cash is distributed among creditors and shareholders.

Loan Amortization Schedule

Breaks down loan repayments over time into principal and interest. This provides transparency regarding the repayment structure and the outstanding balance over time.

Fixed Income & Derivatives Pricing

Valuation of financial instruments and options

Bond Pricing / Yield Curve Analysis

Calculates the value of fixed-income securities based on interest rates and cash flows. Yield curves are used to analyze market expectations and discount rates across various maturities.

Personal Finance Models

Personal finance and investment planning models

Rent vs. Buy Analysis

Compares the financial implications of renting versus buying a property. This takes into account costs, appreciation, and the opportunity cost of capital.

Retirement / Savings Calculator

Projects future savings based on contributions, returns, and time horizon. It helps individuals plan for long-term financial goals such as retirement, or plan their finances more generally.

Business & Data Analytics

Non-financial and operational analytical models

Non-Financial or Non-FP&A Model

An analytical model outside the traditional financial realm, such as operational, marketing, or product analyses. Focuses on data-driven insights rather than purely financial results, but is nevertheless model-driven.

C TASK DESIGN

All tasks in our benchmark were created from scratch by finance experts and are not derived from existing materials. This reduces the risk of data contamination, i.e., the risk that tasks or their solutions appear in the training data of the models underlying the evaluated agents, although the public release may expose future models to them. Each task consists of two files: an input workbook and a golden solution. The input workbook contains three or four sections on separate tabs: task instructions (analogous to a prompt), assumptions, supporting data (for some tasks only), and the questions on which the evaluation of `Final calculation accuracy` is based.

The instructions are as detailed as necessary while remaining concise. We refined the instructions iteratively to remove the ambiguities we identified and to support a fair comparison of agents. We deliberately refrained from including examples or explicit formulas, as this would not reflect how practitioners currently formulate such requests. Where terminology or conventions are not uniformly defined in finance, we followed prevailing industry practice in both the instructions and the golden solution. None of the questions are open-ended: every question used to assess `Final calculation accuracy` has a single, definitive answer, which is numerical in most cases.

The number of accompanying assumptions varies by task type and can reach several hundred. They are provided in tabular form with clear labels and, in some cases, additional descriptions. In accordance with industry practice, we expect agents to reference these assumptions in their financial models rather than hardcode their values. Where necessary, tasks also include supporting data, such as sales prices, trading data, or financial statements; in all but a few cases, this data was generated synthetically. The questions tab, whose answer cells are left blank, is used for our main rubric criterion: `Final calculation accuracy`. The questions were sampled randomly across the entire financial model, so that they cover its different parts. Consequently, some questions are easy to answer, while others require all instructions to be implemented correctly.

The golden solution contains the tabs of the input workbook as well as the completed financial model. Depending on the task requirements and on what is reasonable for the model, the financial model spans one or more worksheets. In the golden solution, the questions tab contains references to the corresponding answers in the financial model and serves as the ground truth for evaluating the agents' `Final calculation accuracy`. The golden solutions define the ground truth for `Final calculation accuracy`; for reasons of efficiency, they are reference computations rather than presentation-ready deliverables and are therefore not graded against the remaining criteria. Their scores and pass rates therefore cannot serve as a reference for the agents' attempts, since the golden solutions may, for instance, omit documentation or formatting that the house standards require of the agents.

All tasks underwent a multi-stage quality assurance process. We iteratively refined the instructions and the tasks themselves to remove ambiguities, and verified the correctness of the golden solution models and answers through both human and LLM-based review. For the latter, an LLM was given the golden solution together with the instructions and asked to check whether the instructions had been implemented and the model built correctly. A task was included in the benchmark only after at least two review runs in which no errors were found.

D EXPERT TEAM AND EFFORT

Our experts comprise four people with finance training: three practitioners with industry experience and one accomplished competitor and task author in financial modeling and spreadsheet competitions. All annotators have hands-on experience building the class of models the benchmark covers; the practitioners and the competitor in particular possess notably greater spreadsheet and industry expertise. Developing the benchmark, including the rubric, the tasks, the judge, and the evaluation pipeline, took approximately 1,845 person-hours, excluding the agent runs themselves.

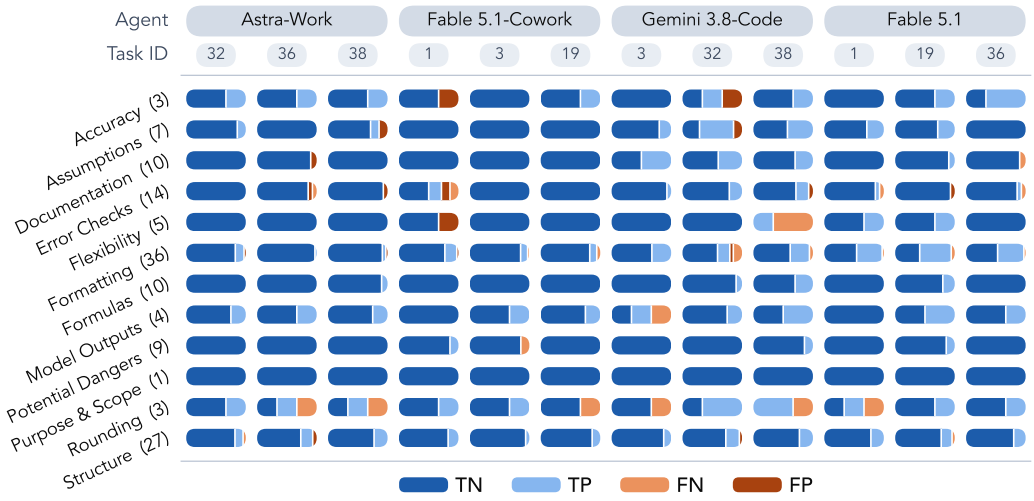

Figure 8: **Annotations on Judge Correctness.** TP: mistake caught correctly by judge. TN: attempt meeting the criterion recognized by judge. FN: mistake exists but not flagged. FP: mistake flagged but not present. The judge agrees with experts' annotations at high accuracy. In particular, FPR is far lower than FNR, which allows the judge to recognize a perfect attempt as passing at high rate. The higher FNR aligns with the pass rate design to be more lenient, but agents nonetheless fail to pass most tasks despite the leniency.

E JUDGE

E.1 AGENTIC JUDGE DETAILS

The judge is an agentic grader built on GPT-5.6 Sol. For each grading it receives the attempt workbook, the task's golden solution, and the input workbook, serialized sheet by sheet with openpyxl, and can read any range of any of them on demand. It is given the criteria applicable to the task (Appendix A.1) and works through them in a single conversation, recording a pass or fail verdict with located mistakes for each, until every criterion has a verdict. `Final calculation accuracy` is not judged this way but checked deterministically against the golden answers (Appendix A.4). Grading of criteria that would typically involve observation of the rendered workbook, such as `Sufficient column widths` or `No hidden sheets`, relies on the creation of a properties block, which is included in what the judge receives. The properties block is created by leveraging openpyxl functionality to log stored settings and metadata about the state of the workbook that is more comprehensive than just values and formulas. The judge relies on this properties block to make inferences about the formatting of the workbook. For criteria that typically would require sensitivity-testing inputs, such as `No errors under extreme-but-plausible inputs`, the judge must reason about the architecture of the workbook and formula structure to determine if any potential criteria violations exist.

E.2 PER-CRITERION VALIDATION EXAMPLES

For each of the 128 criteria graded by the judge, we constructed one pair of validation examples from a single task to which the criterion applies. Where necessary, we first modified the task's golden solution so that it clearly satisfies the criterion; the resulting workbook is the *clean example*. The author of the task then derived the *violating example* by manually perturbing the clean example so that it violates the criterion. Each perturbation was designed to make the violation unambiguous, so that the correct verdict is not in doubt, but not trivially detectable, so that the judge must inspect the workbook to find it. We then ran the judge three times on each of the 256 examples, grading each only on the criterion it was built to test; Section 5.1 reports the results.

F MODELS

Vendor	Model
Anthropic	Claude Fable 5.1
Anthropic	Claude Opus 5
OpenAI	GPT-6 Astra
OpenAI	GPT-5.6 Sol
Google	Gemini 3.8 Flash
SpaceXAI (formerly xAI)	Grok 4.6
Moonshot AI	Kimi K3
Alibaba	Qwen 3.8 Max

Table 4: Models evaluated. In agent names throughout the paper, GPT-6 Astra is shortened to Astra, GPT-5.6 Sol to Sol, and Gemini 3.8 Flash to Gemini 3.8; a suffix (-Cowork, -Work, -Code, -Excel) names the harness and a bare model name denotes the in-house harness. The in-house harness suffix is usually omitted when it is clear from context. GPT-5.6 Sol also serves as the judge backbone (Appendix E.1).

G AGENTIC HARNESSES

We use "harness" to mean everything around the model that a practitioner would receive as one product: the agent scaffold, its tools, how it reads and writes the workbook, its default settings, and its workflow (e.g., the Excel add-ins edit the open workbook in place). Our harness comparisons therefore measure the effect of the product as deployed, not of any single component. Note that the main claims only state that harness has an effect, and we do not attribute the resulting differences to any specific component of the harness.

Common protocol. Every attempt starts from the task's input workbook, the house standards, and a short instruction to build the model in a new workbook and record its answers on the workbook's question tab. The rubric is never shown.

Failures of our own infrastructure (browser, container, or provider outages) are retried and not recorded. Each agent thus has one recorded attempt per task. This follows common practice in end-to-end agentic evaluation: SWE-bench (Jimenez et al., 2024), WebArena (Zhou et al., 2024), OSWorld (Xie et al., 2024b), OSWorld 2.0 (Xie et al., 2026), and SpreadsheetBench 2 (Zhu et al., 2026) likewise evaluate each model once per task, as long-horizon attempts are costly to repeat. We therefore prioritized the number of agents evaluated over repeated runs.

All agent attempts are saved and graded as .xlsx files. The GUI and Excel harnesses run proprietary products that expose no version identifiers; we therefore record the date of every attempt and release these dates with the benchmark, so results for these harnesses are dated snapshots of the products. We evaluate models in several harnesses to observe effects of model and harness, only to identify that effects from both exist, not claiming that the effect of any component is consistent and transferable (likely not the case, since model and harnesses are inherently coupled). Note that when Gemini 3.8 Flash performs poorly in Codex, we do not conclude from this single result that the model itself is weak. Instead, we rely on the observation that Gemini 3.8 Flash passes no task in either of the two harnesses it was evaluated in.

GUI. A Chrome session logged into the consumer product is controlled programmatically: upload the task files, set the mode and reasoning effort, send the prompt, wait for the reply, download the produced workbook. The products' usage caps are respected by pausing runs; a reply that is a quota message rather than a workbook is treated as an infrastructure failure and retried.

Code. Anthropic models are run in Claude Code. OpenAI and all other models with the exception of Anthropic models are run in Codex. Claude Code and Codex run non-interactively inside a pinned container image with a firewall that resolves only the vendor's API endpoint. The container holds the task files and the prompt as a text file; the agent may install and run any code it likes, and

the image ships LibreOffice for formula recalculation. Reasoning effort is pinned per agent at the highest tier the product offers for that model. We recognize that Codex is likely tuned for OpenAI's models. However, we note that our main claim is not that each harness has a specific and consistent effect, but merely that it has an effect, so this tuning does not constitute a confounding factor. As a result, switching the coding harness from Anthropic models to other vendors also does not constitute confounding, since we are only comparing on the same model, the effect from different harnesses.

Excel. In the web-based version of Microsoft Excel, Excel Online, the pipeline opens a copy of the task's input workbook, opens the add-in panel offered by Anthropic or OpenAI, selects and verifies the model or reasoning-effort setting, attaches the remaining task files with the first prompt, and downloads the workbook the add-in edited in place. It is the only harness in which the agent works directly inside the spreadsheet application.

In-house. A custom harness built by the research team. A single loop calls the model's API with 21 spreadsheet tools (file, worksheet, cell and formula edits, formatting, structure inspection, search); every formula edit triggers a LibreOffice recalculation whose results are returned to the model. Each attempt is capped at 40 model calls to limit inference cost. The API endpoints integrated into the harness include Anthropic, OpenAI, and TensorBlock. Third-party libraries such as openpyxl are used to extract and serialize the spreadsheet data in a CSV format, which is fed to the LLM API. The prompt is included in the input sent to the API.

H Qualitative Analysis — General Agent & Harness Analysis

This section describes patterns that apply to multiple agents, harnesses, or difficulty levels, rather than to individual agents; agents are mentioned by name only as examples. It covers the 14 main agent–harness combinations at the highest reasoning-effort setting on all 101 tasks, graded by the agentic judge.

Summary. Agents usually build structurally sound models, and their structural completeness holds even on the hardest tasks. What falls with difficulty is correctness: agents hand in complete, tidy workbooks with more wrong answers. The integrity checks created by the agents tend to verify internal consistency rather than plausibility; industry-standard presentation conventions are applied inconsistently; and more complex tasks result in large, deeply nested workbooks. Finally, the harness also plays a role: while agents in the Excel harness match their GUI counterparts on answers but lose scaffolding and polish, the frontier models in the Code harness match their GUI counterparts of the same models on answers and score, though not necessarily on pass rate.

H.1 What Agents Generally Do Well

Sound structure. Agents deliver spreadsheets with live, driver-based formulas, correctly anchored references, centralized assumptions, a single date anchor, and a cover stating the model's purpose. This structural completeness holds as tasks get harder: the agents keep handing in complete workbooks with live answers, proper roll-forwards, and dedicated check rows, and blank answer sheets remain rare at every difficulty level.

Near-perfect on standard mechanics. Accuracy is highest when a task is based on well-known mechanics: bond pricing, NAV roll-ups, comparables, ratio and budget-versus-actuals analysis, and rule-based non-financial tasks. Here, most agents answer all or nearly all questions correctly, even on Hard tasks; on the Hard task `LogisticsCategorization` (task 34), for example, the stored answers of every agent match the golden solution.

Documenting purpose and architecture. Agents routinely open the workbook with a cover page or summary that outlines the purpose of the model and the business question and explains the methodology. They use clear and descriptive tab names. A reader who did not build the model can therefore find their way around without a walkthrough. A live, formula-based output summary is consistently produced by the GUI agents, the Excel add-ins, and the two frontier coding agents, whereas the other Codex agents often leave it as hardcoded text.

Consistent signs and units, nothing hidden. Workbooks rarely contain hidden or sensitive elements: no external links, no hidden sheets or rows, and no white-on-white text. They keep one sign convention and one unit scale throughout and often document the convention; monetary blocks carry $k, $m, or USD tags, and scale conversions are explicit. This reduces common causes of hidden errors and order-of-magnitude errors when a model is shared with another user.

Check sheets and master error flag as standard. Nearly every agent ships a dedicated check sheet of formula-driven pass/fail rows. The rows are typically summarized under a master error flag on the cover page or in the summary. The weak point is the roll-up: a COUNTIF summary over the status column skips rows with errors, causing the master error flag to show "OK" even though the checks fail. Such checks allow an auditor to immediately verify internal consistency, but a false "all clear" finding is worse for a client than no feedback at all.

Financial model style basics. Agents consistently follow the basic visual guidelines for a financial model: hardcoded values in blue font, often on a pale background fill; thousands separators for large numbers; a uniform format for similar values; shaded title and header areas; totals in bold; and no merged cells.

H.2 What Agents Generally Do Poorly

Fixed-horizon builds that must be manually extended. Timelines are often tailored precisely to the length specified in the instructions, with a fixed number of rows or columns or a base year incorporated into the timeline formulas, so that any reasonable change in duration, start date, or rate results in the model being shortened or in errors. The problem rarely occurs with Easy tasks but becomes common from the Medium level onward. It follows the model more than the harness: Fable 5.1 in Claude Code and Claude Cowork, and Kimi K3 and Gemini 3.8 Flash in Codex, often fix the horizon, whereas GPT-6 in ChatGPT Pro, GPT-6 Astra in Codex, GPT-5.6 Sol in ChatGPT for Excel, and Opus 5 in Claude Cowork usually drive it from one anchor date. This matters in practice, because models are routinely rerun with new data and conditions.

Sheets not set up for reading. Workbooks are built for the agent, not for a reader who flips through them, prints them out, and forwards them. Long schedules routinely ship without freeze panes or with so many rows frozen that they fill most of the screen (Figure 9), page breaks are almost never set, and print areas are often missing. On the long timelines of harder tasks, period columns are left too narrow, so whole blocks later display balances as ### (Figure 10).

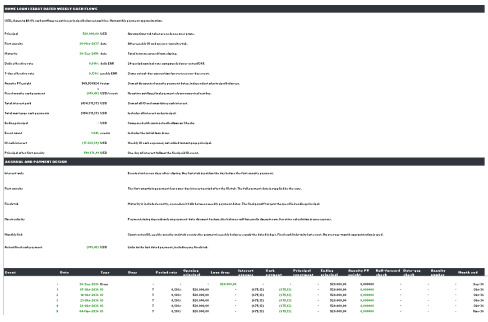

Figure 9: Excessive freeze panes: the frozen area ends only at the thin black line, so the sheet cannot be scrolled without zooming out, here to 40%. *ChatGPT Pro (GPT-6) on the task* `DetailedLoan` *(task* 16*, attempt* 2064*).*

Sense checks reconcile but do not benchmark. Most agents add a block labeled "Sense Checks", but in almost all cases, the financial model is only validated against itself: the results are compared with the underlying assumptions or with trivial constraints that any functioning model satisfies. While these blocks confirm the arithmetic, they say nothing about plausibility; therefore, a workbook can pass every test and still yield an implausible valuation or cash position.

Implied value per share — every metric x statistic side by side (Instructions B17)									
			Peer multipl				Implied value (€ per share)		
Metr (€m)	es)	tile	ge	an	tile	tile	ge	an	tile
Pric	####	0	#	#	#	#	#	#	#
EV/S	####	1	#	#	#	#	#	#	#
EV/E	####	1	#	#	#	#	#	#	#
P/E	####	0	#	#	#	#	#	#	#
P/FC	####	0	#	#	#	#	#	#	#
(buffer row inside the metric block)									

Figure 10: Columns that are too narrow display ### instead of their values. *Claude for Excel (Opus 5) on the task* `SAP` *(task* 51*, attempt* 2286*).*

Inputs left unguarded. Important input fields are rarely protected in a way that actually prevents incorrect entries. Although the GUI agents and most other agents in the Code harness add data validation rules, they disable error warnings, so the cell accepts any value; the Excel add-ins leave the warnings enabled but either skip validation or limit it to drop-down menus, thereby ignoring dates, interest rates, and maturities; and the weakest coding-CLI model does not add any validation at all. Named ranges for global drivers are also rare outside a few agents.

Color legend declared, then drifting. Nearly every agent uses the convention of blue, black, and green and documents it in a legend, but the distinction between black (calculations on the same sheet) and green (references to other sheets) drifts within a sheet (Figure 11). Once the color no longer corresponds to the cell type, a reviewer can no longer use it to track relationships, and a legend that contradicts the worksheet or is used inconsistently is more misleading than no legend at all.

		0	30	31	32
		Dez 28	Jun 31	Jul 31	Aug 31
		FY2028A	Jun-31E	Jul-31E	Aug-31E
Category	EAG p.a.	FY2028A			
		35.934.209,00	**40.111.313,82**	**40.195.668,00**	**40.351.197,74**
Non-Operating Items	2,5%	1.186.865,75	1.262.313,39	1.264.963,47	1.267.619,11
Non-Operating Items	2,5%	(12.248.784,50)	(13.027.425,15)	(13.054.774,69)	(13.082.181,64)

Figure 11: Font color drifting within a row: comparable same-sheet calculations switch from green to black partway across the row. *Claude for Excel (Fable 5.1) on the task* `RelativeDevelopment` *(task* 48*, attempt* 1556*).*

Finance display rules applied inconsistently. The basic number formatting is mostly correct, but each agent occasionally violates the financial presentation rules: negative numbers are displayed with a minus sign instead of in parentheses (Figure 12), exact zeros are shown as 0.00 instead of as a dash, and blocks of numbers are centered or left-aligned instead of right-aligned.

Line item	Unit							
Year index (0 = first model year)	#	-	1	2	3	4	5	6
Calendar year (key)	year	2030	2031	2032	2033	2034	2035	2036
Period	label	**FY2030E**	**FY2031E**	**FY2032E**	**FY2033E**	**FY2034E**	**FY2035E**	**FY2036E**
Cash receipts								
Revenue (cash in)	US$k	1.318.171,71	1.362.567,81	1.420.261,92	1.442.323,29	1.486.296,74	1.532.182,38	1.583.029,32
Operating payments								
COGS	US$k	-992.353,83	-1.019.839,87	-1.052.089,80	-1.072.951,22	-1.102.518,79	-1.137.321,44	-1.173.253,15
Selling & marketing	US$k	-39.545,15	-40.877,03	-42.607,86	-43.269,70	-44.588,90	-45.965,47	-47.490,88
Rent & Utilities	US$k	-29.150,00	-29.458,00	-30.341,74	-31.251,99	-32.189,55	-33.155,24	-34.806,62
Travel expenses	US$k	-1.800,00	-1.840,50	-1.881,91	-1.924,25	-1.967,55	-2.011,82	-2.057,09
Other expenses	US$k	-47.368,36	-48.810,52	-50.489,03	-51.628,57	-53.194,30	-55.021,77	-56.913,91
Salaries	US$k	-13.344,00	-13.710,96	-14.629,86	-14.475,43	-14.873,51	-15.282,53	-15.702,80
Payroll taxes & social security	US$k	-2.963,15	-3.044,41	-3.128,13	-3.210,84	-3.299,14	-3.520,64	-3.486,68
Accounting & IT	US$k	-890,17	-869,04	-848,41	-828,27	-808,61	-789,41	-770,67

Figure 12: Negative numbers displayed with a minus sign instead of in parentheses. *Claude Code (Fable 5.1) on the task* `OnlyOperational` *(task* 94*, attempt* 1924*).*

Large, nested models on harder tasks. Starting with Medium-Hard tasks, the agents incorporate the logic into long single-cell formulas instead of helper rows and build brute-force engines; for example, Claude Cowork (Fable 5.1) answered the Hard task `FixedDetailed` (task 84, attempt 1624) correctly, but with nested IF, SUMIFS, and INDEX formulas across roughly 150,000 cells. Such models are not only inefficient but also very hard to review. At the same time, formula notes and output summaries become less frequent, and the answer cells of workbooks built in the Code harness increasingly display errors such as #NAME? or #VALUE!.

I Qualitative Analysis — Agent-Specific Analysis

This section describes the strengths and weaknesses of selected models; patterns shared by nearly all agents are described in Appendix H and are not repeated. Note that we focus our attention mostly on models actually performant on the benchmark, and this section is not meant to review all agents. The points are selective: each describes one of the most pronounced strengths or weaknesses of a model, measured against the model's own profile, against the other agents, or both. They are therefore not a ranking across models: a Good point for a lower-ranked model does not necessarily mean that it does better on that point than a higher-ranked model, unless the text says so explicitly. Likewise, a point that is not listed for a model or harness does not mean that the model is free of that weakness or lacks that strength there, only that it is not among the model's most pronounced traits. For models run in more than one harness, the parentheses after each point name the harnesses in which the model stands out on that point; the point may also apply to the other harnesses, only less prominently. A concluding paragraph then separates the effects of the model from those of the harness: we attribute a pattern to the model if it stands out in several harnesses or if other models in the same harness behave differently, and to the harness only if the same model demonstrably behaves differently in another harness or if all models in that harness share the pattern.

I.1 Claude

I.1.1 Fable 5.1 (Claude Code, Claude Cowork, Claude for Excel)

In Claude Code and Claude Cowork, Fable 5.1 is narrowly the most accurate model in the field and the one most likely to test its outputs against outside benchmarks, and it delivers well-sourced, well-documented workbooks. In Claude for Excel, it answers nearly as accurately and builds some of the leanest engines in the field, but it falls behind on finishing. Its weak spots are oversized text columns and rows, "rounded to" notes on figures that are only formatted, copied engines and deeply nested formulas, and timelines that are less flexible than those of the other leading agents.

Good:

- **Sense checks against outside benchmarks** (Claude Code, Claude Cowork). The model adds a sense-check block to its check sheet that tests headline outputs, such as margins, coverage ratios, or bond prices, against explicit external or judgment ranges, often with an owner and a date, whereas nearly every other model only reconciles its outputs with its own inputs. However, the ranges are sometimes still derived from the financial model's own inputs. In the task `CashNiagara` (Claude Code, task 7, attempt 1874), debt service, distributions, interest, cash, and repayments are compared with explicit judgment ranges, and in the task `DoctorvsTrade` (Claude Cowork, task 18, attempt 1327), the tax, debt, savings, net worth, and crossover outputs are checked against explicit benchmark ranges.
- **Sourced assumptions and documented formulas** (Claude Code, Claude Cowork). The model cites a source for almost every input, either a cell of the input workbook or the relevant part of the task instructions, gives its own judgment calls an owner and a date, and explains non-obvious formulas in cell comments or adjacent methodology notes. For example, in the task `PlasticParts` (Claude Code, task 43, attempt 1495), every input cites its location in the input workbook or the task instructions, and every judgment input names an owner and a date; in the task `HouseorFlat` (task 86), both Claude Code (attempt 1861) and Claude Cowork (attempt 1625) explain their complex formulas in comments or adjacent methodology rows.

- **Named key inputs** (Claude Code, Claude Cowork). The model gives key global drivers, such as the anchor date, tax rate, and case selector, self-explanatory defined names and uses them in its formulas. This is evident in the task `BasicGrowth` (Claude Cowork, task 2, attempt 1268), where the Name Manager lists `Anchor_Date`, `Tax_Rate`, `Model_Years`, and `Case_Selected`, and in the task `TheLiquidityLoop` (Claude Code, task 62, attempt 1814), where the key drivers are named and their input cells validated.
- **Self-documenting workbooks with a master error flag** (Claude Code). The workbook opens with a genuine cover stating purpose, owner, version, scope, and usage, followed by a hyperlinked table of contents, and it includes all standard components through to the checks and the change log; the check sheet rolls up into a formula-linked master error flag on the cover or summary that turns red on failure. In the task `CustomerConcentration` (Claude Code, task 12, attempt 1876), for instance, the cover gives title, purpose, version, date, owner, scope, approach, usage, and a legend, and in the task `FiveAutomotive` (Claude Code, task 22, attempt 1775), the flags on both the cover and the summary link to the check sheet.
- **Freeze panes and print areas** (Claude Code, Claude for Excel). The model freezes the headers of long sheets at sensible positions without freezing too much, and Claude Code also defines print areas on every tab, including the tabs inherited from the input workbook. The task `Bosch` (Claude Code, task 3, attempt 1252) illustrates this: every tab has a print area, and in the task `Nike` (Claude for Excel, task 93, attempt 1662), the top five rows of the long calculation sheets are frozen.
- **Compact engines** (Claude for Excel). The model solves a problem with one parameterized engine instead of copying logic: native iterative calculation, or one shared FCFF build. For example, in the task `RelativeDevelopment` (Claude for Excel, task 48, attempt 1556), native iteration replaces the 30 and 41 copied rows in which Claude Code and Claude Cowork unroll the WACC calculation.

Bad:

- **Oversized text columns and rows** (Claude Code, Claude Cowork). The model puts long texts, such as the cover, table of contents, checks, and change log, into extremely wide single columns or very tall wrapped rows; Claude Code, at the same time, often leaves some numeric and timeline columns too narrow, so they display ###. This is evident in the task `PercentageDevelopment` (Claude Cowork, task 96, attempt 1633), where the cover column is 150 characters wide, the purpose column of the table of contents 120 characters, and the check-label column 92 characters, and in the task `DebtDeclineDynamics` (Claude Code, task 14, attempt 1935), where the text columns are 120 to 130 characters wide, while the loan amount on the schedule sheet displays as ###.
- **"Rounded to" notes for display formats** (all three harnesses). The model labels sheets as "rounded to $0.01" although the formulas carry full precision and only the number formats limit what is shown; the figures are displayed to, not rounded to, that precision. In the task `CorpBond` (Claude Cowork, task 10, attempt 1298), "rounded to $0.01" labels sit above thousands of schedule formulas that use no rounding function, and in the task `BreakingDowntheBalance` (Claude for Excel, task 4, attempt 1269), none of the 12,483 schedule formulas under such a label rounds.
- **Copied engines and nested formulas** (Claude Code, Claude Cowork). The model tends toward brute force: it copies entire engines instead of creating a single parameterized engine or using a data table; furthermore, Claude Cowork writes monolithic formulas, even though helper rows would make the financial model easier to review. In the task `CashNiagara` (Claude Code, task 7, attempt 1874), for instance, the entire 277×86 waterfall is copied three times, and in the Easy-level task `MeridianOutdoor` (Claude Cowork, task 88, attempt 1620), the commentary text formulas repeat nested IFERROR, TEXT, ABS, and IF calls.
- **Fixed-width timelines** (Claude Code, Claude Cowork). The model sometimes creates the timeline with a fixed number of columns or references fixed ranges, so extending the time period or changing the start date requires manually copying columns and editing ranges.

For example, in the task `DetailedLoan` (Claude Code, task 16, attempt 1893), the timeline is fixed at 1,008 columns, and the cover instructs users to copy the last column to the right for a longer horizon.

- **Negatives with minus signs** (Claude Code). The model often displays negative numbers with a leading minus sign instead of in parentheses, even for final answers and summary outputs. This is evident in the task `Harrington` (Claude Code, task 31, attempt 1949), where answers are displayed as $-2{,}820.0000$ and -7.62%.
- **Unexplained formulas and unguarded circular loops** (Claude for Excel). Non-obvious formulas, such as array MEDIAN(IF()), MMULT, and SUMPRODUCT/INDEX/MATCH blocks, carry no cell note or adjacent explanation, and some circular models switch on iterative calculation without a breaker switch, a stated reason for the circularity, or a note on how the loop resolves. The task `ProductionPlanning` (Claude for Excel, task 44, attempt 1544) illustrates this: the SUMPRODUCT/INDEX/MATCH and MMULT blocks carry no explanatory note, and in the task `PlasticParts` (Claude for Excel, task 43, attempt 1541), the circular financing and cash block has no breaker, justification, or note.
- **Misused font colors** (Claude for Excel). The reserved blue input and green link fonts appear in cells that are neither input fields nor links, such as text in change logs, fixed labels, legends, and even entire blocks of same-sheet calculations. This is evident in the task `FruitJuice` (Claude for Excel, task 28, attempt 1496), where same-sheet calculations are in the green reserved for cross-sheet links, and in the task `FundFun` (Claude for Excel, task 29, attempt 1502), where the entire change log is in input blue.

Model vs. harness: Most of these points stand out in both Claude Code and Claude Cowork and therefore belong to the model, including the "rounded to" notes; Claude Code merely documents rounding correctly more often and sets print areas more consistently. The Excel add-in changes the picture: compared with the other two harnesses, the model builds leaner engines but explains fewer formulas and misuses font colors more often. Since Opus 5 shows the same shift toward lean but sparsely documented engines in Claude for Excel, and all agents in the Excel harness blur the color code, these differences appear to stem largely from the harness.

I.1.2 Opus 5 (Claude Cowork, Claude for Excel)

Opus 5 in Claude Cowork is a near-frontier builder whose edges over the field are sense checks against outside benchmarks, named global drivers, and accurate rounding notes. In Claude for Excel, its engines are unusually lean, but it leaves the finishing layer undone. In both harnesses, its check flags are less reliable and its cell formatting is worse than those of the other leading models.

Good:

- **Accurate rounding notes** (both harnesses). The notes state that figures are "shown to" a given precision while full precision is carried, and they name the questions tab as the only place where ROUND is applied. In the task `CashNiagara` (Claude Cowork, task 7, attempt 3260), the precision notes on the summary and annual cash-flow sheets read "shown to" and state that full precision is retained, and in the task `Telecom` (Claude for Excel, task 59, attempt 2343), a note separates display precision from the ROUND applied on the questions tab.
- **Sense checks and named drivers** (Claude Cowork). The check sheets often include a block that tests headline outputs against dated outside benchmarks or judgment ranges, and key global inputs, such as anchor dates, horizon length, case selector, and core rates, routinely get clear defined names. For example, in the task `TheInterestGame` (Claude Cowork, task 60, attempt 3650), the installment, effective annual rate, interest burden, debt service, and yield are tested against outside benchmark ranges, and in the task `TBondII` (Claude Cowork, task 56, attempt 3635), ten global drivers are named.
- **Insert-safe totals and an anchored horizon** (Claude Cowork). Subtotal ranges include labeled buffer rows or columns, so that line items can be inserted without breaking the sums, and timelines are usually prebuilt across the required horizon and driven by one anchor date and a length driver. In the task `DailyDetail` (Claude Cowork, task 77, attempt 3673),

for instance, open buffer rows precede the totals, and in the task `CashConversionLab` (Claude Cowork, task 6, attempt 3235), the 60-quarter horizon runs from one anchor.

- **Compact engines and freeze panes** (Claude for Excel). Where other agents copy the same logic across dozens of case sheets or huge iteration grids, the model usually builds one compact engine and points everything else at it; it also freezes the header rows and label columns of long calculation and valuation sheets without freezing too much. This is evident in the task `FundRebalancing` (Claude for Excel, task 85, attempt 2452), where one parameterized engine does the work, and in the task `ApfelInc` (Claude for Excel, task 1, attempt 2001), where the long tabs are frozen at sensible positions.

Bad:

- **Check flags that cannot be trusted** (both harnesses). Claude Cowork's check formulas often perform arithmetic on whole ranges or nest a multi-cell IF or ABS inside SUMPRODUCT or MAX without array entry; they return #VALUE! and pass the error into the status cells meant to flag problems. Claude for Excel's master error flag is often plain text or fixed green without conditional formatting, so it cannot turn red when a check fails. For example, 18 check formulas return #VALUE! in the task `Fixings` (Claude Cowork, task 23, attempt 3472), and the summary of the task `LiquiditySplatter` (Claude for Excel, task 33, attempt 2133) reads "ALL CHECKS PASS" while the debt waterfall contains #NAME? errors.

- **Loose cell formatting** (both harnesses). Cross-sheet formulas appear in black despite the workbook's own green-link legend or switch between green and black; in Claude Cowork, numbers are often centered and titles merged. For example, in the task `FlightPlan` (Claude Cowork, task 24, attempt 3375), each shortest-path sheet uses eight merged title ranges instead of Center Across Selection, and in the task `GermanLife` (Claude for Excel, task 30, attempt 2092), cross-sheet formulas are black on one sheet but green on the summary and tax sheets.

- **Columns too narrow** (Claude Cowork). Number and date columns are sometimes narrower than their content, so summary results, reconciliation values, or an entire date column display as ###, more often than for any other agent. A case in point is the task `Bosch` (Claude Cowork, task 3, attempt 3290), where a column that is too narrow makes four summary outputs display as ###.

- **Blocks below "End Sheet" markers** (Claude Cowork). The model adds new model-control, scenario, and derived-driver blocks below the "End Sheet" marker of the inherited assumptions sheet instead of placing them in the declared structure of the sheet or on a separate sheet. In the task `Office` (Claude Cowork, task 40, attempt 3636), for instance, the case and scenario controls and the anchor date follow the marker in row 121.

- **Unfinished navigation, print setup, and formula notes** (Claude for Excel). The table of contents on the summary sheet is plain text without hyperlinks, print areas are often undefined, and non-obvious logic, such as nested revolver waterfalls, array reconciliations, and nested lookups, often carries no explanation. In the task `QuarterlyRatios` (Claude for Excel, task 47, attempt 2258), all eight sheets are listed, but none is linked; in the task `CoastalAggregates` (Claude for Excel, task 8, attempt 2020), no sheet has a print area; and in the task `DebtFlow` (Claude for Excel, task 15, attempt 2038), the revolver draw-and-repay and vesting-offset formulas are left unexplained.

Model vs. harness: The accurate rounding notes, the unreliable check flags, and the loose formatting stand out in both harnesses and belong to the model; that Fable 5.1 mislabels rounding in the same two harnesses underlines this. Several differences are observed to follow the harness: the sense checks, named drivers, and buffer rows that stand out in Claude Cowork are rare in the Excel add-ins, including Opus 5 in Claude for Excel; its anchored timelines are also rare in Claude for Excel, although GPT-5.6 Sol builds such timelines in ChatGPT for Excel; and its formula notes thin out there, as they do for Fable 5.1; its compact engines in Claude for Excel likewise mirror those of Fable 5.1 in that harness.

I.2 GPT

I.2.1 GPT-6 ASTRA (CHATGPT WORK, CODEX)

In both harnesses, GPT-6 Astra builds polished, print-ready, well-documented workbooks with a working master error flag, but it never names its key inputs, places its own blocks below inherited "End Sheet" markers, and tests its outputs only against the workbook itself.

Good:

- **Print-ready page setup** (both harnesses). The model defines print areas on every tab and places manual page breaks at section or table boundaries on multi-page sheets. The task `LogisticsCategorization` (ChatGPT Work, task 34, attempt 1335) illustrates this: every multi-page sheet has manual page breaks, and so do the long sheets in the task `8Options` (Codex, task 69, attempt 1748).
- **Workbooks with source citations, notes, and a linked table of contents** (both harnesses). Inputs cite their source sheet or cell, the model's own judgment calls carry an owner and a date, non-obvious formula blocks, such as WACC iteration, interpolation, asset vintages, and tax timing, carry cell notes or nearby methodology notes, and the table of contents holds working hyperlinks to every sheet. For example, in the task `FiveAutomotive` (ChatGPT Work, task 22, attempt 1413), block notes and cell comments explain the WACC iteration, interpolation, debt, and asset-vintage blocks, and in the task `TwoNOne` (Codex, task 101, attempt 1736), notes cover debt timing, the 30/360 convention, tax, the waterfall, and the terminal value, and the contents tab links to every sheet.
- **Master error flag wired to the checks** (both harnesses). Every workbook gets a master error flag on the summary sheet that is linked to the roll-up of the check sheet and conditionally formatted to turn red on failure. This is evident in the task `TwoNOne` (ChatGPT Work, task 101, attempt 1545), where the flag is tied to the count of failed checks, and in the task `CashConversionLab` (Codex, task 6, attempt 1842), where the formula-linked flag turns red on failure.
- **Formulas that hold up under stress** (ChatGPT Work). The formulas usually guard the edge cases, such as zero rates, schedule capacity, whole payment counts, selector validity, and zero-machine or negative-cash states, so plausible input changes rarely produce #DIV/0!, #N/A, or impossible balances. In the task `LargeBalance` (ChatGPT Work, task 87, attempt 1593), no break was found when the duration and working-capital drivers were stressed.
- **Enforced validation and one house style** (Codex). The model puts validation with active error alerts and sensible bounds on key dates, rates, counts, and selectors, and it gives all built sheets one font and one title, header, and total style, with blue inputs, black formulas, green cross-sheet links, and negatives in parentheses. In the task `ApfelInc` (Codex, task 1, attempt 1831), for instance, the key editable drivers carry validation with active alerts, and in the task `NextEraEnergy` (Codex, task 39, attempt 1719), the font colors follow the convention on every built sheet.
- **Timelines that roll from one input** (Codex). The model prebuilds the full required horizon from a central anchor date and period controls, so dates and periods roll forward without edits to the formulas. A case in point is the task `OnlyOperational` (Codex, task 94, attempt 1826), where all dates of the 14-year timeline follow the central drivers, with no need for manual extensions to change the horizon.

Bad:

- **No named ranges** (both harnesses). The model never defines names, so global drivers, such as anchor dates, horizons, tax rates, WACC, and case selectors, are referenced only by cell address. ChatGPT Work also adds data validation to key inputs but switches the error alerts off, so invalid entries pass without warning. For example, in the task `ThreeCars` (task 65), both ChatGPT Work (attempt 1406) and Codex (attempt 1755) reference the valuation date, the tax rate, and WACC only by cell address.

- **Content below "End Sheet" markers** (both harnesses). The model appends its own controls, inputs, derived drivers, and notes below the "End Sheet" marker of the inherited assumptions sheet, so live inputs sit outside the sheet's declared structure; ChatGPT Work also stacks second timelines or date-helper series onto sheets that already have a main timeline. The task `DoctorvsTrade` (task 18) illustrates this: ChatGPT Work (attempt 1403) continues live inputs and interpretation notes past the marker, and Codex (attempt 1845) adds 42 rows of controls and derived timing after it.
- **Sense checks that never look outside** (both harnesses). The model builds sense-check blocks, but these compare the outputs only with figures from the same workbook or with generic arithmetic bounds and often state outright that there is no external benchmark. This is evident in the task `QuarterlyRatios` (ChatGPT Work, task 47, attempt 1468), where only internal break-even checks are run, next to a note that there are no external benchmarks, and in the task `EasyRev` (ChatGPT Work, task 81, attempt 1565), where the "same-sample benchmark" is another forecast built from the same inputs.
- **"Rounded to" labels for display formats** (ChatGPT Work). Headers and notes often state that values are "rounded to 0.01" or "to 0.1%", while the formulas keep full precision and only the displayed precision is as stated. In the task `CFQuarterly` (ChatGPT Work, task 74, attempt 1425), for instance, a "rounded to $0.01mm" label sits above thousands of formulas without ROUND.
- **Oversized, formula-heavy files** (Codex). The model prefills huge capacity grids with thousands of inactive formula rows or columns and adds duplicate copies of source data, so even small models end up relatively large and carry a heavy recalculation load. In the task `LeaseorKeys` (Codex, task 32, attempt 1817), about 122,000 formulas are copied across 1,200 columns, and the checks contain very large ISERROR arrays.

Model vs. harness: The page setup, documentation, and master error flag, as well as the missing names, the blocks below "End Sheet" markers, and the inward-looking sense checks, stand out in both harnesses and belong to the model; ChatGPT Pro, running GPT-6, shows most of them as well. Where the two harnesses are observed to differ, the difference concerns how strictly the model applies conventions: in Codex, it keeps validation alerts on, describes rounding correctly, and prebuilds its timelines from one anchor, whereas in ChatGPT Work, it switches the alerts off, labels formatted figures as rounded, and sometimes fixes the length of its timelines. In Codex, the prebuilt timelines come at the price of oversized files. The robust formulas of ChatGPT Work and the house style of Codex, by contrast, stand out in one harness only.

I.2.2 GPT-6 (CHATGPT PRO)

GPT-6 in ChatGPT Pro is a disciplined, convention-heavy builder that leads the field on flexible timelines, honest rounding statements, and a uniform house font, but it overbuilds its workbooks with brute-force grids and duplicated engines, skips freeze panes, and rarely tests its outputs against outside benchmarks. Its page breaks, blocks below "End Sheet" markers, and inward-looking sense checks also mark GPT-6 Astra in both of its harnesses, which points to the model family rather than the harness.

Good:

- **Flexible timelines.** The model builds the full required horizon up front and drives it from one central anchor date and period inputs, so that changing the forecast length within that horizon requires no manual column copying or range edits. For example, in the task `OnlyOperational` (task 94, attempt 2644), the 2030–2043 timeline follows the central date and period controls, and in the task `SingleForever` (task 53, attempt 2428), the formula-driven timeline responds to the horizon inputs without edits.
- **Visible master error flag.** The model almost always rolls every check up into one master pass/fail cell on the summary, linked to the check sheet and conditionally formatted to turn red on failure. A case in point is the task `CashConversionLab` (task 6, attempt 2019), where the master flag is conditionally formatted.
- **Honest rounding statements.** The model keeps display precision separate from actual rounding: calculation sheets are labeled "shown to $0.01" and keep full-precision formulas,

and only the final answer block applies ROUND. The task `NestQuest` (task 38, attempt 2333) illustrates this: full-precision calculations, the "shown to $0.01" display, and the explicitly rounded answers are kept distinct.

- **Uniform, print-ready sheets.** The model restyles every sheet it builds to a single font, Arial 10, with matching section bands, headers, and total styles, and it places manual page breaks at section or period boundaries on sheets that print to several pages. In the task `ConstantFlow` (task 9, attempt 2036), for instance, all built sheets use Arial 10 throughout, and in the task `TheInterestGame` (task 60, attempt 2505), the multi-page sheets carry manual page breaks.
- **Sourced assumptions and annotated formulas.** The model adds a source column that cites the input workbook for each input, tags its own judgment calls with an owner and a date, and places methodology notes beside non-obvious blocks, such as WACC iteration, depreciation vintages, and cash-tax timing. This is evident in the task `FruitJuice` (task 28, attempt 2237), where every input on the assumptions sheet carries a citation, and every judgment an owner and a date.

Bad:

- **Overbuilt, bloated workbooks.** The model tends toward brute force: for small cases, it creates reserve calculation grids with more than a thousand rows, duplicates entire calculation modules instead of parameterizing a single one, expands logic into replicated matrices, and adds duplicate register and appendix blocks, resulting in inefficient workbooks that are difficult to review and modify. For example, in the task `CorpBond` (task 10, attempt 2046), tens of thousands of reserve-grid formulas serve a case with 40 payments, and in the task `CashNiagara` (task 7, attempt 2039), the entire waterfall engine is duplicated (17,521 formulas) where one selector-driven engine would be sufficient.
- **No sense checks against outside benchmarks.** The model's sense-check sections contain only internal reconciliations, comparisons with the workbook's own historical figures, or trivial bounds. In the task `EasyRev` (task 81, attempt 2610), the error metrics MAE and MAPE are checked only against another forecast built in the same workbook.
- **Missing freeze panes.** Very wide monthly schedules and long check and summary sheets usually ship without freeze panes, so row labels and period headers scroll out of view; when the model does freeze panes, it sometimes locks an excessive block of rows. This is evident in the task `PastaInc` (task 41, attempt 2406), where the integrated statements span 149 columns with no frozen labels or headers, and in the task `CashConversionLab` (task 6, attempt 2019), where the frozen rows take up most of a regular screen view, leaving little room to see the actual financial model.
- **Content below "End Sheet" markers.** When extending an inherited assumptions sheet, the model keeps its "End Sheet" marker and appends new controls, drivers, and even live calculation blocks below it, so that this content sits outside the sheet's declared structure. In the task `LeaseorKeys` (task 32, attempt 2319), live controls and calculations follow the inherited marker.
- **Erratic row heights and misaligned text.** The model occasionally generates tables whose body rows alternate between many custom heights, with tall detail rows, narrow spacer rows, and very tall outliers, and it sometimes aligns narrative text and labels to the right or centers them. For example, in the task `MonthlyLiquidity` (task 37, attempt 2365), body rows range from 18 to 62 points, with one row at 310 points, and in the task `DoctorvsTrade` (task 18, attempt 2077), the labels and narrative text on the summary are centered instead of left-aligned.

I.2.3 GPT-5.6 SOL (CHATGPT FOR EXCEL)

GPT-5.6 Sol in ChatGPT for Excel is a tidy, well-scaffolded builder that keeps its columns readable, sets print areas, and drives its timelines from start-date inputs, but it slips on conventions that most agents handle: rounding labels, font colors, buffer rows above totals, a glossary, and lean schedules. Its drifting font colors and totals without buffer rows also mark both Claude for Excel agents

(Fable 5.1 and Opus 5) and are thus likely effects of the Excel add-in harness; its readable column widths and complete print areas set it apart from them.

Good:

- **Readable widths and print areas.** The model wraps long narrative text instead of stretching columns, so that its own sheets stay within a readable width, and it usually defines a print area on every tab. This is evident in the task `EasyDCF` (task 20, attempt 1324), where no built column is excessively wide, and all long text is wrapped.
- **Timelines driven by start-date inputs.** The model usually builds the full required horizon in advance and drives period labels and dates from a central start-date and period input, so that the timeline can be rolled forward without edits to the formulas. In the task `DebtFlow` (task 15, attempt 1313), for instance, the full 30-year horizon is prebuilt with labels that follow the drivers.
- **Complete scaffold with a live summary.** The model reliably delivers the full workbook skeleton: a summary as the first tab that doubles as a cover with owner, version, and date, followed by a table of contents, a check sheet, and a change log, and the figures on the summary are live formulas. The task `CashConversionLab` (task 6, attempt 1280) illustrates this: the values and check status on the summary link to the assumptions, working-capital, and check sheets.

Bad:

- **Wrong rounding notes.** The model routinely labels sheets as "rounded to $0.01" although the formulas keep full precision and only the display format is limited; on some tasks, it also wraps intermediate values in ROUND before they feed later calculations. For example, in the task `BasicGrowth` (task 2, attempt 1267), sheets are labeled "rounded to $0.01", yet none of their roughly 9,900 formulas uses a rounding function.
- **Unreliable font colors.** In a fair number of tasks, the model colors same-sheet calculations green, the color its own legend reserves for cross-sheet links, and mixes black and green for the same kind of formula, so that the font color no longer reveals to the reader what type of cell it is. A case in point is the task `RevForecast` (task 49, attempt 1390), where same-sheet calculations are green in three columns and black in a fourth.
- **No buffer rows above totals.** The model puts totals directly below the last line item with fixed sum ranges, so that a row inserted above a total falls outside the sum. In the task `Office` (task 40, attempt 1366), total revenues and gross asset value sit directly below fixed SUM ranges, although every other agent satisfied this criterion on the task.
- **Oversized schedules that recalculate slowly.** As the flip side of its prebuilt timelines, schedules run far past the active horizon, with thousands of mostly blank formula rows that SUMIFS, lookups, or whole-column references then rescan, which makes the recalculation load avoidably heavy. In the task `PaydownPathways` (task 42, attempt 1368), for instance, a schedule of 5,200 periods, only 180 of them active, is rescanned by every yearly SUMIFS, LOOKUP, and AVERAGEIFS row.
- **No glossary of conventions.** The cover and summary explain scope and usage but sometimes give no glossary for the color conventions and abbreviations, such as FCF, EAR, and $mm, although the house standards require one. For example, in the task `SmallRiver` (task 98, attempt 1546), neither the blue/black/green convention nor abbreviations such as FCF, EAR, and $mm are explained.

I.3 SpaceXAI

I.3.1 Grok 4.6 (Codex)

Grok 4.6 in Codex is a structurally tidy builder that names its key inputs, freezes its long sheets, and documents rounding carefully, but it clutters its sheets with merged cells, centered numbers, and boxed grids, fills its summary tabs with hardcoded numbers, and occasionally ships workbooks broken by #NAME? errors.

Good:

- **Named key inputs.** The model almost always gives key global inputs, such as dates, rates, case selectors, and unit conversions, short and descriptive defined names. This is evident in the task `EasyDCF` (task 20, attempt 2353), where the Name Manager holds a full set of named DCF inputs, among them `ValDate`, `Rf`, `Beta`, `MRP`, and `TaxRate`.
- **Freeze panes on long sheets.** The model keeps titles, column headers, and period labels frozen at sensible positions on long calculation sheets. The task `Proportionality` (task 46, attempt 2377) illustrates this: the headers and labels of the long documentation and calculation sheets are frozen, and none of the freezes is excessive.
- **Honest rounding documentation.** The model's notes separate figures that are only "shown to" a precision from answers that are actually rounded with ROUND, and they sit next to the outputs. A case in point is the task `CashNiagara` (task 7, attempt 2259), where the annual sheet reads "shown to $0.01m, full precision held".

Bad:

- **Merged cells and centered blocks.** The model routinely merges sheet titles, section headers, and whole cover or summary blocks instead of using Center Across Selection, although the house standards prohibit merged cells, and it centers whole blocks of numbers in statements, schedules, and ratio tables as well as body paragraphs on the cover and summary. In the task `CorpBond` (task 10, attempt 2387), the cover contains 30 merged ranges and the summary 36, and in the task `RoadtoZero` (task 50, attempt 2329), long scope and methodology paragraphs are centered.
- **Boxed grids and oversized rows.** The model inserts thin borders on all four sides of nearly every cell within large calculation blocks, which duplicates the gridlines and creates a dense grid, and it leaves the narrative and header rows much taller than necessary, some even several times the regular row height. For example, in the task `MarketBalanced` (task 35, attempt 2180), virtually every cell of the calculation engine is boxed, and in the task `ConstantFlow` (task 9, attempt 2249), a single row is 359.7 points tall.
- **Summaries with hardcoded numbers.** The model often fills the summary tab with typed snapshot figures without formulas, so the summary never updates when inputs change. In the task `Bosch` (task 3, attempt 2088), for instance, the summary contains no formulas, and sales, margins, and balance-sheet figures are typed snapshots.
- **Deeply nested formulas.** The model writes long, monolithic formulas that nest IF, INDEX/MATCH, or SMALL chains four or more levels deep, notably in hand-built percentile interpolation and growth formulas, and leaves them as single blocks even where helper rows already exist. This is evident in the task `TMobile` (task 57, attempt 2354), where a monolithic percentile formula chains IF, OR, ABS, INT, and SMALL, although helper rows exist.

I.4 MOONSHOT AI

I.4.1 KIMI K3 (CODEX)

Kimi K3 in Codex is an agent that gets parts of the workbook plumbing right, with working table-of-contents links, clean sheet separation, honest rounding notes, and compact rows, but unlike the frontier agents, it ships hardcoded summary tabs, unsourced assumptions, and hard-wired timelines, and leaves inherited clutter and half-applied style rules behind.

Good:

- **Honest rounding notes.** The notes on the cover and sheets describe precision correctly: intermediate values carry full precision, most figures are only displayed to two decimals, and ROUND is applied to the answer cells alone. The task `TBondII` (task 56, attempt 2714) illustrates this: a note states that values are displayed to $0.01 and only the answers are rounded.

- **Working table of contents.** The model puts HYPERLINK formulas to every worksheet on the cover or a separate contents tab. For example, in the task `Bosch` (task 3, attempt 2085), functional HYPERLINK formulas open every sheet.
- **Clean functional sheet separation.** The model keeps the cover, inputs, calculations, outputs, checks, and answers on separate sheets or in clearly labeled blocks and does not append content below an inherited "End Sheet" marker. A case in point is the task `CashNiagara` (task 7, attempt 2662), where each function has its own sheet.
- **Compact rows.** The model keeps the rows it builds at a sensible height; the only tall rows are inherited instruction rows. In the task `CashNiagara` (task 7, attempt 2662), for instance, only the inherited instruction row reaches about 60 points; all other rows are lower.

Bad:

- **Static summary tab.** The tab named "Summary" is often typed narrative or a build plan without formulas, sometimes with input figures written into the prose, so nothing on it updates when inputs change. In the task `DailyCash` (task 13, attempt 2633), the summary consists only of hardcoded constants.
- **Unsourced assumptions.** Input blocks, and assumptions the agent adds itself, such as benchmarks, screening bands, forecast periods, and growth drivers, often carry no source citation and no judgment label with an owner and a date. The task `TheLiquidityEngine` (task 61, attempt 2972) illustrates this: the forecast-period, growth, averaging, and link assumptions have neither a source nor an owner or date.
- **Hard-wired timelines.** Timelines are often built from typed start dates, embedded years, constant year indices, or a fixed number of copied columns instead of flowing from one date or horizon driver, so rolling the model forward or changing its length requires manual edits. For example, in the task `PastaInc` (task 41, attempt 2596), a hardcoded first date is copied across 145 fixed monthly columns, and in the task `ConstantFlow` (task 9, attempt 2683), the engine is filled to exactly 300 columns and does not size itself from the model-length driver on the assumptions sheet.
- **Own style conventions not applied.** The model documents green as the color for cross-sheet links but leaves whole blocks of cross-sheet formulas in black, and it uses number formats that show negatives with a minus sign rather than in parentheses. This is evident in the task `OperationsWeekly` (task 95, attempt 2757), where whole aggregation blocks that link to the daily cash-flow sheet are black instead of the documented green.
- **No freeze panes on large sheets.** Long daily engines and very wide schedules usually have no frozen headers or label columns, and in one task, Kimi K3 even removed freeze panes that the input workbook already had. In the task `LogisticsCategorization` (task 34, attempt 2820), engines of 14,601 rows have no frozen header row.

I.5 GOOGLE

I.5.1 GEMINI 3.8 FLASH (CODEX)

Gemini 3.8 Flash in Codex is a lean, bare-bones builder whose files stay compact and sensibly sized. However, it ships static summaries, rigid hardcoded timelines, a master error flag that can never turn red, builder-only documentation, unsourced assumptions, and unfrozen, over-boxed sheets.

Good:

- **Sensible column widths and row heights.** The model's columns have a sensible width and its rows a normal height. In the task `DailyDetail` (task 77, attempt 2104), for instance, no built column is excessively wide.
- **Proportionate builds.** The model sizes its schedules to the base case and avoids brute-force replication, such as copied WACC iteration blocks or thousands of prefilled spare rows, so its files stay proportionate to their content; the flip side is the rigid timelines listed

below. A case in point is the task `SpendingPlan` (task 54, attempt 2095), where the workbook takes only 488 KB for a scenario dataset of 7,500 rows.

- **Totals clearly set off.** Totals, subtotals, and closing balances are consistently bold, often with shading or double bottom borders. For example, in the task `DetailedLoan` (task 16, attempt 2113), totals and ending balances are bold and often shaded.

Bad:

- **Rigid, hardcoded timelines and drivers.** Forecast years are typed labels and schedules are filled to a fixed length, so changing the start date, horizon, or payment frequency requires manual rework, and event months, thresholds, and weights are typed into formulas instead of being linked to a driver. The task `DebtDeclineDynamics` (task 14, attempt 2119) illustrates this: the engine is fixed at 60 periods and does not resize when the frequency or duration changes, and in the task `Fixings` (task 23, attempt 2112), the years 2026–2035 are typed labels, and the period count on the assumptions sheet does not drive the schedule.
- **Static summaries and check flags.** The summary or cover is often a static build plan or narrative with almost no formulas, in which headline outputs, and even case inputs and dates, are retyped as constants; the master error flag on the check sheet has a static fill without conditional formatting, so a failing check can never trigger a visual alarm. This is evident in the task `Proportionality` (task 46, attempt 2158), where the summary holds 61 constants and no formulas, and in the task `Bosch` (task 3, attempt 2082), where the master flag has a static blue fill and no conditional formatting rule.
- **Formulas that break under plausible stress.** Tax-rate, growth, and return formulas divide without a guard, so a plausible stress, such as pretax or net income falling to zero, throws #DIV/0! into the valuation, and some logic is pinned to fixed levels and misbehaves when an input is halved. In the task `EasyDCF` (task 20, attempt 2181), for instance, the unguarded ratio of tax to pretax income returns #DIV/0! when pretax income is stressed to zero, and the error flows into the average tax rate and the valuation.
- **Thin documentation and unsourced assumptions.** The notes describe how the model was built, but not how to use it: there is almost never a guide to which inputs are safe to edit or where to read the outputs, scope notes omit exclusions, complex nested formulas carry no comments, and assumption blocks carry no source, owner, or date, even for the model's own judgment calls. In the task `BasicGrowth` (task 2, attempt 2141), the navigation section only lists the tabs, and nothing explains which inputs are editable or how to use the compounding selector, and in the task `BreakingDowntheBalance` (task 4, attempt 2147), the input block has no source field and no owner or date for any driver.
- **Unfrozen, over-boxed, inconsistent sheets.** The model almost never freezes panes or sets print areas, even on very long or wide calculation sheets, boxes every cell of long schedules in thin borders, switches fonts between its own Arial and the inherited Aptos Narrow, and is loose on color conventions. For example, in the task `GermanLife` (task 30, attempt 2125), a calculation sheet of about 155 rows by 80 columns has no frozen panes.

J Finance Glossary

This appendix defines the finance and spreadsheet terms used in this paper for readers without a finance background. Financial model types, such as DCF valuations or 3-statement models, and the fields of finance they belong to are described in Appendix B.3 and are not repeated here.

Fields of Finance

Fields of finance named in the paper beyond those in Appendix B.3

Commercial banking
Granting loans to and accepting deposits from businesses and individuals; these models include loan plans and credit analyses.

Fields of Finance *(continued)*

Corporate finance
The management of a company's finances: its investment, financing, and dividend decisions.

Investment banking
Advising companies on mergers, acquisitions, and capital raising, with valuation and transaction models playing a central role.

Private equity
Investments in privately held companies, often as part of leveraged buyouts.

Wealth management
Financial planning and investment advice for individuals and households, for example, with regard to retirement or home ownership.

Financial Statements and Metrics

Accounting terms and key financial figures

Actuals, forecasts, and period labels
Actuals are historical, reported figures; forecasts are projections calculated from assumptions over the forecast horizon, that is, the time period covered by a model. Period labels indicate the type of period with a suffix, for example FY2024A (actual), FY2025E (estimate), or Q1-26B (budget).

Annual report (10-K)
The annual financial report that publicly traded U.S. companies file with the Securities and Exchange Commission; a typical source of input data for financial models.

Capital expenditure (capex)
Expenditures for long-lived assets, such as property, plant, and equipment (PP&E), which are capitalized, that is, recognized as assets and depreciated over time, rather than expensed immediately.

Consolidation
Combining the financial statements of several business units or companies into a single set of statements, eliminating the transactions between them (inter-company eliminations).

EBITDA
Earnings before interest, taxes, depreciation, and amortization; a common measure of operating profitability. Depreciation and amortization spread the costs of long-lived assets over their useful lives.

Financial statements
The income statement (revenue, expenses, and profit over a period), the balance sheet (assets, liabilities, and equity at a point in time), and the cash flow statement (cash inflows and outflows over a period).

Fiscal year

Financial Statements and Metrics *(continued)*

The 12-month period that a company uses for its financial statements, which does not necessarily coincide with the calendar year (e.g., July through June).

Free cash flow (FCF)

The cash a company generates after taxes, capital expenditures, and investments in working capital; it is available to the company's investors.

Working capital

Funds tied up in day-to-day operations, primarily accounts receivable and inventory, net of accounts payable.

Valuation and Financing

Valuation methods, financing terms, and period conventions

Day-count convention

The rule for counting days when calculating interest or discounting over a partial period, such as act/365 (actual days in a 365-day year) or 30/360 (30-day months in a 360-day year).

Deal models

Models for transactions, not covered by SpreadsheetSmith (Figure 6): a leveraged buyout (LBO) model for a predominantly debt-financed acquisition; a merger model with an accretion/dilution analysis, which shows how an acquisition affects the acquirer's earnings per share; and precedent transactions, in which a company is valued based on the prices paid in similar past transactions.

Deal vs. planning model

A model designed to evaluate a one-time transaction, as opposed to a model that is used repeatedly for ongoing planning.

Enterprise value and multiples

Enterprise value is the value of a company's business operations to all of its investors. Multiples such as EV/EBITDA or the price-to-earnings ratio (P/E) relate a value to a financial metric and are used to compare similar companies.

Mid-year convention

Discounting the cash flows of a period as if they occurred in the middle of the period rather than at its end.

NPV and IRR

The net present value is the sum of an investment's discounted cash flows, including the initial investment; the internal rate of return is the discount rate at which the net present value is zero.

Revolver and cash sweep

A revolving credit facility is a line of credit that a company draws on and repays as needed; with a cash sweep, excess cash is automatically used to repay debt.

Roll-forward (corkscrew)

Valuation and Financing *(continued)*

A schedule for a balance such as debt or cash: the opening balance plus additions minus reductions equals the closing balance, which becomes the opening balance of the next period.

Stub period

A partial period at the beginning or end of a forecast, for example when the valuation date falls within a fiscal year; its length is determined with a day-count convention.

Tranche

A separate portion of a company's debt with its own terms, such as interest rate, maturity, and priority in repayment.

WACC

Weighted average cost of capital: the combined rate of return required by equity and debt investors, which is used as the discount rate in a DCF valuation.

Spreadsheet Modeling Practice

Spreadsheet conventions and Excel features

Buffer row

An empty row above a total or between sections, so that new line items can be inserted without breaking the sum formulas.

Cases and switches

A case is a consistent set of assumptions, such as a base case or a worst case; a switch or selector is an input cell that selects the active case or turns a model feature on or off, for example whether expenditures are capitalized or expensed. With a switch, a single parameterized calculation engine serves all cases instead of one copy of the engine per case.

Cell references and anchoring

In a reference such as `$B$4`, the $ signs fix the column and the row when a formula is copied (an absolute reference); if only one of the two is fixed, the reference is mixed. If references are anchored incorrectly, copied formulas silently refer to the wrong cells.

Circular reference

A formula that depends directly or indirectly on its own result, for example interest calculated on an average debt balance, which in turn depends on the interest paid. Excel can resolve it through iterative calculation; a breaker switch turns the circularity off for debugging.

Color coding

The industry-standard convention of showing hardcoded inputs in blue, calculations in black, links to other sheets in green, and links to other workbooks in red.

Cover sheet

The first sheet of a workbook, which introduces the model: its title, purpose, scope, and conventions.

Data validation and conditional formatting

Spreadsheet Modeling Practice *(continued)*

Excel features that restrict what can be entered in a cell and that format cells automatically based on their values.

Error checks
Formulas that test the integrity of a model, for example whether the balance sheet balances or whether the cash balance matches the cash flow statement; a master error flag shows in one cell whether any check has failed.

Error values
Results such as #REF! (invalid reference), #N/A (value not available), or #DIV/0! (division by zero) that Excel displays when a formula cannot be evaluated.

Freeze panes, print areas, and page breaks
Settings that keep the headers visible while scrolling and that specify which cells are printed and where pages break.

Grouping
Collapsing rows or columns behind an outline button; unlike hiding, grouping leaves a visible indication that content is there.

Hardcoding
Entering a fixed value instead of a formula or cell reference. Hardcoded inputs are allowed as long as they are marked as such; numbers entered directly into formulas or calculation cells are an error, since they cannot be traced back to or updated from the inputs.

Inputs, assumptions, and drivers
Inputs and assumptions are values that are entered directly rather than calculated, such as a growth rate or a tax rate; a driver is an input on which many calculations depend, such as the valuation date, that is, the date as of which values are measured.

Merged cells
Cells merged into a single cell, which interferes with formulas, sorting, and references; Center Across Selection achieves the same look without merging the cells.

Named range
A descriptive name assigned to a cell or range (e.g., `TaxRate`) that formulas can use instead of a cell address.

Sense check
Comparing key results with external reference points, such as industry benchmarks or comparable companies, to identify implausible results.

Sign convention
A consistent rule for signs, for example costs and outflows negative, revenues and inflows positive.

Units and currency
The scale and currency in which amounts are stated, for example $mm for millions of U.S. dollars.

VBA and macros
Visual Basic for Applications, Excel's programming language, and the automated routines written in it.

Spreadsheet Modeling Practice *(continued)*

Volatile functions
Functions such as OFFSET, INDIRECT, or NOW that recalculate on every change in the workbook, which slows down large models.