

Introduction

Proteomics researchers often rely on visualizations to evaluate evidence, compare hypotheses, and support analytical decisions across complex datasets. Mass Dynamics 3.0 (MD 3.0) already offers a broad catalogue of interactive plots covering the most common proteomics workflows, but researchers' questions don't always fit predefined views, and building a new visualization requires engineering effort, statistical know-how, and weeks of turnaround, leaving many scientifically interesting questions unexplored.

Recent AI advances now **enable natural-language-driven visualization generation**, offering a way to significantly reduce the latency between a scientific questions and decision-relevant evidence. A researcher could simply describe the figure they need and receive an interactive, publication-ready plot built against their own data in minutes rather than weeks. However, when such systems are integrated directly into analytical workflow, maintaining scientific integrity and reproducibility becomes essential.

Here, we present a **benchmark-driven evaluation of AI-assisted visualization** generation for MD 3.0, assessing AI agents' ability to reproduce expert-curated visualizations, modify existing ones, generate entirely new plot types, and correctly reject requests that fall outside the scope of visualization. This framework lays the groundwork for **rapidly expanding MD 3.0's visualization capabilities** while preserving analytical rigour, freeing expert time for deeper scientific innovation, and putting custom, trustworthy visual analytics within reach of every researcher, regardless of programming background.

Agentic Pipeline

MD 3.0 employs a **two-stage agentic architecture** powered by Anthropic's Claude¹.

The **Planning Agent** (architect) reads a user's **natural-language request** together **with a specifications document** that encodes coding patterns, data schemas, statistical conventions, and visual style rules. It emits a feature list of 15+ concrete, testable requirements (e.g., "box must display Q1, median, Q3, and Tukey fences"), turning vague questions into engineering blueprints.

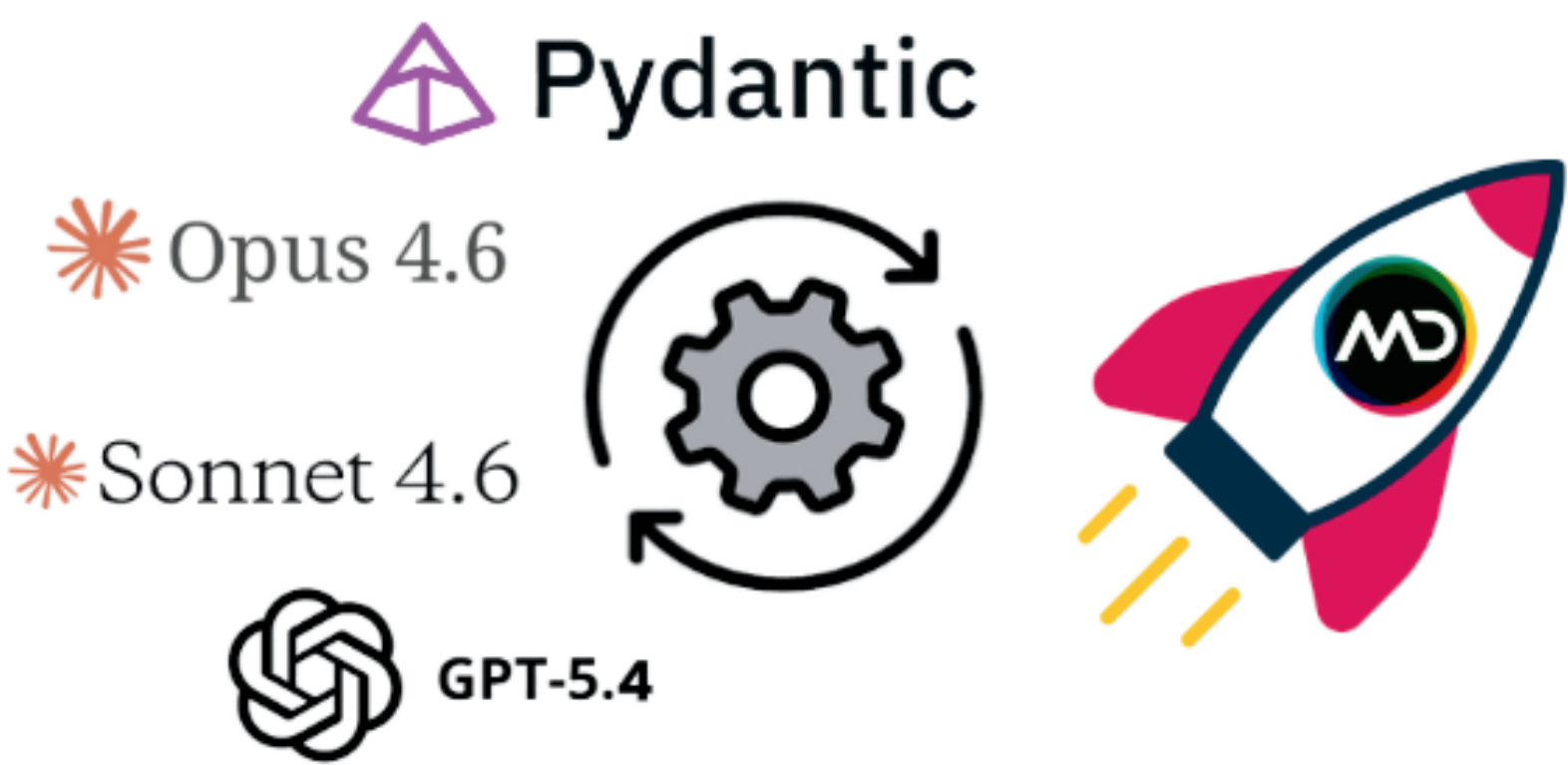
The **Coding Agent** (builder) **implements features sequentially**: writes code and unit tests, runs them on samples, fixes defects, and advances only when each item passes. Items remain pending until satisfied.

We introduced **self-evaluation** at both stages:

- the planning agent's output is scored against specification criteria;
- the coding agent similarly self-evaluates its iterations.

Independently, we ran **offline benchmarks using Pydantic Eval**² after agent runs completed, defining structured benchmark cases with typed inputs and expected outputs. Offline benchmark results are not fed back to the agent at runtime. We **evaluated three LLM backends**: Anthropic's Claude Opus 4.6, Claude Sonnet 4.6, and OpenAI's GPT-5.4³. The architecture ensures AI-generated plots adhere to the same statistical assumptions, data access patterns, and reproducibility standards as manually implemented plots.

Between benchmark rounds, the specifications document provided to the planning agent was updated based on observed failure modes. We then re-ran the same benchmark cases to measure whether targeted spec updates produced measurable improvements in subsequent agent runs, closing the loop between evaluation and prompt engineering.



References

- Anthropic. (2026). Claude (Opus 4.6) [Large language model]. <https://www.anthropic.com/claude>
- Pydantic. (2026). Pydantic Evals [Computer software]. <https://ai.pydantic.dev/evals/>
- OpenAI. (2026). GPT-5.4 [Large language model]. <https://openai.com>

Conflicts of interest

The authors **Giuseppe Infusini**, **Aaron Triantafyllidis** and **Andrew I. Webb** are co-founders of Mass Dynamics, a for-profit enterprise delivering software as a service for the processing, analysis, and sharing of proteomics data. The authors **Mansi Aggarwal** and **Sean Brady** are employees of Mass Dynamics.

Methods

Each benchmark case is structured with **typed inputs** (user request, dataset, parameters) and a **reference output manually curated** by an MD engineer or bioinformatician. Benchmark scores are computed after the agent has finished and are never fed back to the agent at runtime.

Benchmark Scenarios

- Reproduce an existing plot:** The agent must recreate a visualization - box plot - already present in MD 3.0, matching the engineering team's implementation in visual aesthetics, statistical computation, threshold lines, and protein counts.
- Modify an existing plot:** The agent must add functionality without regressing surrounding behaviour (point counts, hover data, trace identity). We tested adding a new show/hide entity-list control to ANOVA volcano plot.
- Generate a new plot absent from MD 3.0:** The agent should be able to create new visualizations - ridge plot using kernel density estimation (KDE) per condition, and an instrument QC chart that requires ingesting and reconciling data from multiple uploaded sources.
- Reject a non-visualization request:** verifying the agent returns a user-facing error when the input request is not visualisation-related.

Sample datasets are **real proteomics datasets** spanning dozens of samples across multiple experimental conditions, ensuring evaluation under realistic shapes and noise. For every generated plot, the harness compares the agent's output against the reference along several axes

Weighted Scoring

Each benchmark test is weighted (Table below), i.e. higher weight for higher severity, and the final benchmark score is computed as (earned points / total possible points) × 100. All scenarios were run against the three LLM backends under identical prompts, datasets, and scoring rubrics. For each run we recorded the final weighted score, token usage, request counts, and tool-call counts, enabling a direct comparison of quality against cost.

Tier	Weight	Examples
Critical	x10	- Agent run completed without error - Expected traces rendered; output is a valid Plotly figure - Correct number of subplots/facets - No exceptions thrown during ingestion
Statistical	x5	- Statistical information in plots show - Q1, median, Q3, mean; KDE curve shape - Plot p-values and fold-change values - Protein/feature counts per condition
Structural	x3	- Hover text fields and contents - X and Y axes labels and units - Legend entries and ordering - Trace names, subplot titles
Visual	x2	- Image-similarity score (generated vs. expected plot) - Colour palette matches MD style - Consistent font and layout
Settings	x1	- Plot parameters wiring (e.g. log-scale toggle, sample metadata selection) - Default values applied correctly - UI control labels

Table 1. Table showing examples of benchmark tests grouped by severity coded in the weighth column, where higher weight is associated to higher severity.

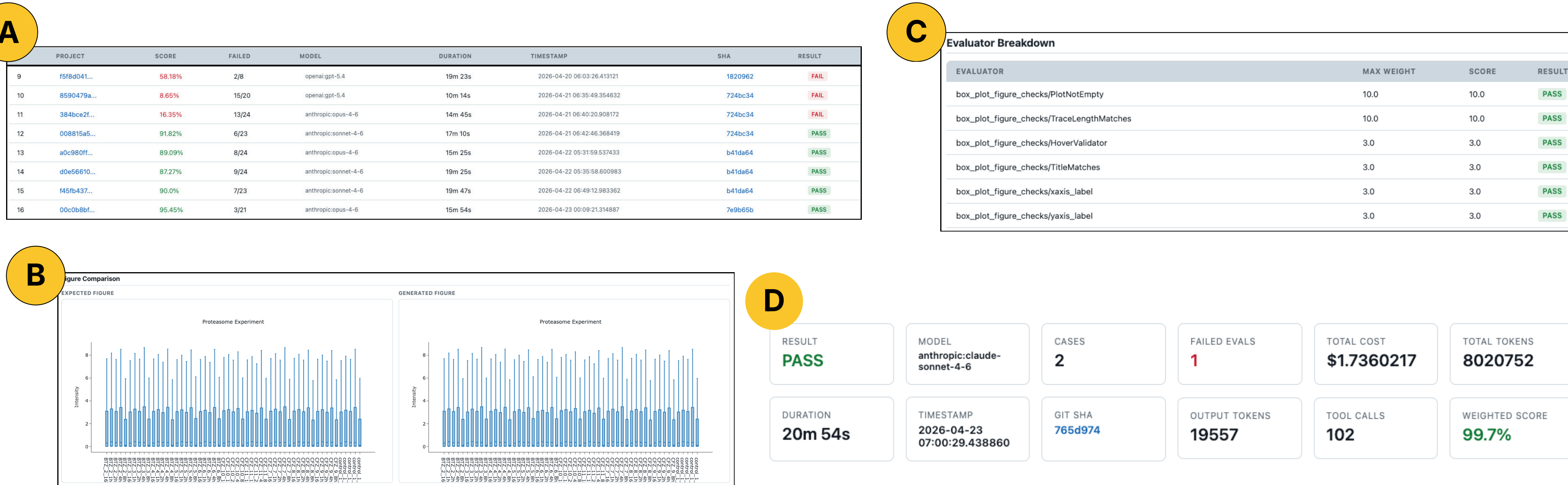


Figure 1. Components of the benchmark report. Each agent run produces a structured HTML report combined with a cross-run trends view, enabling both detailed inspection of individual runs and comparison of performance over time. **(A)** Run history listing every historical run of a benchmark scenario, with score, model backend, duration, commit SHA, and pass/fail outcome, supporting cross-run comparison across models and spec versions. Linking each run to a specific commit makes it possible to attribute any improvements or regressions to the exact spec, prompt, or code change that produced them; **(B)** Side-by-side rendering of the expected (manually curated reference) and generated (agent-produced) figures, enabling direct visual inspection of statistical, structural, and aesthetic differences. **(C)** Breakdown showing each individual check for a given plot (e.g. trace presence, hover content, axis labels), its severity weight, the points earned, and the pass/fail outcome; (D) Each run's summary panel for a given plot reporting key metrics for the run.

Results

Across the four benchmark classes, Claude Opus 4.6 produced the highest-quality outputs; Claude Sonnet 4.6 trailed only modestly and at lower token cost; GPT-5.4 consumed far fewer tokens but struggled on statistical correctness and specification adherence, particularly for the new and modified plot scenarios.

Scenario	Opus 4.6	Sonnet 4.6	GPT-5.4	Avg Cost Per Run
Box plot (reproduce existing plot)	~99.7%	~99.7%	0–3%	\$2.64
ANOVA volcano with show/hide entities (modify plot)	~97.7%	~97.7%	13–93%	\$1.72
Ridge plot (new plot)	83–97%	79–83%	5–25%	\$2.19
Instrument QC chart (new plot)	~90%	~90%	0–60%	\$2.41
Non-visualization request	100%	100%	100%	\$0.04

Updating the specifications document between benchmark rounds produced visible jumps in the benchmarking results. Two representative examples:

- Codifying statistical and unit conventions:** Early box plot runs failed in two ways: pre-computed summary statistics (mean, quartiles) were silently dropped from the final plot due to a Plotly export, and threshold lines in ANOVA volcano plot (e.g. for fold-change, FDR) were drawn at the raw input value, instead of being logged for log-scaled axes, landing in the wrong position. We added explicit rules in the agent specifications covering both. Box plot statistical-tier scores rose from ~1% to ~99.7% on Anthropic backends.
- Standardising plot layout:** Ridge plot runs showed inconsistent sizing, margins, and hover behaviour compared to MD's manually created plots. We updated the spec to require the same shared layout scaffolding used by existing MD modules, and pointed the planning agent at a reference implementation already in the codebase. Ridge plot scores on Opus 4.6 rose from ~7–25% on early runs to a consistent 94–98%.

These results show that benchmark-driven AI visualization generation can safely reduce the latency between scientific questions and decision-relevant evidence, freeing expert time and enabling researchers to explore custom visualizations without compromising statistical rigor or reproducibility.

Agent-Ready Omics

Since this work began, the platform has been extended beyond visualization. Mass Dynamics 3.0 now runs a hosted **Model Context Protocol (MCP) server**, allowing external AI tools to upload and process data, build datasets and set metadata, and run analyses directly against a researcher's own experiments. These operations run on live data, under the same validation and record described above. The benchmark reported here evaluates MD 3.0's internal planning and coding agents; the MCP server is a separate interface to the same platform, in which the researcher confirms what each analysis compares.

Keen to try Mass Dynamics MCP with your own data?



Simply scan the QR code to get started.