Bringing Domain Intelligence into Business Modeling: Executable Function Discovery via LLM-Guided Symbolic Search

Tianyu CUI[A], Pengfei ZHANG[B], Yaohui JIN[C], Yi LU[D] and Jialiang LU[A]
[A]SJTU Paris Elite Institute of Technology
[B]Artefact, Shanghai, China
[C]Shanghai Jiao Tong University
[D]China Europe International Business School

Business modeling systems estimate parameters after transformation functions are specified, but narrow expert-defined families limit heterogeneous responses. We present symbolic function discovery by Large Language Model (LLM) agents, constrained for business use and evaluated on Marketing Mix Modeling (MMM) Adstock discovery. The workflow calibrates time and non-time seeds, separates stochastic structural proposal from system-controlled settlement, and searches channel-specific formulas in a best-available-state global context. Candidate selection combines predictive metrics with agent-derived, domain-informed proxies for curve-shape rationality and parameter clarity. On the public Robyn dataset, the search expands executable response families beyond the calibrated fixed-form reference while incrementally improving global fit; the integrated workflow also outperforms PySR and LLM-SR under a controlled global-residual comparison. Post-discovery qualitative inspection finds Search and Print shapes consistent with established MMM and advertising literature. These results support MMM Adstock discovery; transfer to other business-modeling settings remains future work.

Marketing Mix Modeling, Large Language Model Agents, Symbolic Regression, Response Function Discovery, Explainable AI

1. Introduction

Business modeling systems often face a function-specification bottleneck: they can efficiently estimate parameters once a transformation structure is fixed, but selecting that structure remains a high-impact manual decision. Marketing Mix Modeling (MMM) provides a concrete instance. Existing open-source frameworks such as Robyn and Meridian provide strong estimation and optimization pipelines, but they depend on a small set of expert-defined response families, primarily geometric carryover and Hill-type saturation, to model how marketing channel effects carry over time [1, 2]. The response function acts as a transformation that converts raw media exposure into carried-over effect signals before entering the additive model. When channels exhibit different temporal response patterns, such as short-lived effects in a performance channel and long-memory effects in a brand channel, forcing them into the same parametric template can distort attribution and downstream budget allocation.

Similar bottlenecks can arise when modeling backends are effective after a functional form is supplied but expose only a narrow menu of structures. We use MMM Adstock discovery as the evaluation domain because it offers well-instrumented backends, heterogeneous channel behavior, and established conventions for interpreting lag-response patterns. The framework is designed around three requirements that may recur in such settings: discovered functions must be executable in the existing backend, their parameters must be identifiable from noisy observational data, and their response behavior must remain amenable to domain interpretation. This paper tests these requirements in MMM; transfer to other function-specification settings remains a future-work hypothesis.

Large Language Models (LLMs) offer a path to program discovery: FunSearch couples LLM proposals with systematic evaluation, while LLM-SR combines LLM priors with evolutionary equation search [3, 4]. Unlike physical equation recovery, observational MMM data need not identify a unique channel-response law. We therefore treat discovery as generating executable response-shape hypotheses, not recovering causal truth: LLMs propose structures from context and feedback, while system services search parameters, validate execution, settle candidates, and archive traceable artifacts.

The Data Scientist and Data Analyst (DS–DA) workflow has three design principles. Calibrated initialization establishes temporal and media-side references before formula search. Channel-wise decomposition evaluates one lag kernel at a time while holding the global context frozen. Layered evaluation separates authority: predictive fit governs settlement and best-state replacement, domain proxies enter archive ranking, and feedback guides later proposals, while system services control validation and state transitions. The workflow calibrates seeds in Phase 1 and Phase 1.5 and discovers formulas in Phase 2. In the reported Robyn run, the workflow expands executable lag-kernel families beyond the fixed-form reference and modestly improves full-data reconstruction fit; PySR and LLM-SR provide integrated-workflow, not component-level, references.

Our main contributions are:

1. We formulate channel-wise Adstock discovery as an instance of business-constrained function discovery and develop an LLM-guided search that explores parameterized structures beyond predefined backend families while enforcing executability and identifiability.
2. We introduce a staged calibration procedure that decomposes temporal baseline effects, media priors, and channel-specific transformation discovery, mitigating cold-start issues and confounding between global trends and channel-level patterns.
3. We implement a layered candidate-evaluation mechanism: predictive fit governs numeric settlement and best-state replacement, agent-derived domain proxies rank settled archive entries, and diagnostic feedback guides subsequent proposals, allowing executable response-shape hypotheses to be preserved and compared without equating agent ratings with human validation.

2. Related Work

Our work studies LLM-guided symbolic discovery for function-specification bottlenecks in business modeling, with MMM Adstock discovery as the current implementation and evaluation instance. To ensure discovered functions satisfy practical constraints, we integrate symbolic search with agentic diagnostic workflows. The evaluated system gap is specific: existing MMM pipelines can estimate parameters once response functions are specified, but they provide limited support for discovering executable, channel-specific response functions with structured validation and feedback.

2.1. MMM and Response-Function Specification

MMM systems have matured into repeatable automated workflows, but their automation largely operates within user-selected or curated response families. Classical market response modeling has long treated functional form and lag structure as substantive modeling choices [5]. Bayesian marketing models provide uncertainty-aware estimation and prior regularization for noisy observational data [6], and open-source MMM systems such as LightweightMMM, Robyn, and Meridian make attribution, response-curve analysis, and budget optimization more accessible [1, 2, 7]. These systems turn MMM from a bespoke econometric exercise into a repeatable modeling workflow: users can prepare

aggregated data, estimate channel effects, inspect response curves, and evaluate budget scenarios within a maintained backend.

Their common limitation is that automation largely operates after the response family has been selected. Hyperparameter search can tune decay or saturation settings, and Bayesian estimation can express uncertainty around parameters, but the candidate transformation space remains anchored to a small menu of predefined forms. In practice, these workflows commonly operationalize carryover and saturation through expert-chosen families such as geometric Adstock and Hill-type response curves. This design is stable and interpretable, but it leaves the upstream choice of functional form outside the automated search loop. Prior marketing research shows that lag and carryover effects vary across media and communication settings [8,9], while Bayesian MMM work treats carryover and shape choices as modeling assumptions that affect attribution [10]. Our framework addresses this specification problem by treating Adstock discovery as a constrained symbolic search task: rather than selecting from a fixed formula library, it proposes parameterized structures that are evaluated under domain constraints including executability, parameter identifiability, and curve-shape plausibility.

2.2. Symbolic Regression and LLM-Guided Discovery

Symbolic regression and LLM-guided program discovery provide the methodological foundation for our work, but require adaptation for constrained settings. Genetic programming introduced expression-tree search [11], and later methods combine neural networks, inductive biases, or learned search policies with symbolic extraction [12–14]. Recent LLM-guided systems use language models as proposal mechanisms inside external evaluation loops: FunSearch pairs LLM-generated programs with systematic evaluation [3], and LLM-SR uses LLM-generated equation skeletons with downstream parameter optimization [4]. We position LLM guidance in the same broad proposal-evaluation family, but adapt it to respect domain-specific constraints: executability in the MMM backend, parameter identifiability, and curve-shape interpretability.

2.3. Agentic Diagnostics for Model Workflows

Our framework uses LLM agents as bounded diagnostic support rather than autonomous evaluators, drawing on recent progress in agentic reasoning and chart understanding. LLM agent research shows that reasoning, action, observation, and tool use can be interleaved in structured loops [15], while chain-of-thought prompting motivates decomposed reasoning for complex tasks [16]. MMM diagnostics also rely heavily on visual evidence: practitioners inspect predicted-vs-actual plots, response curves, and channel effect shapes before trusting attribution results. Chart-understanding work such as ChartQA, MatCha, and DePlot demonstrates progress on reasoning over visualized data [17–19].

Chart-based judgments, however, can be sensitive to axis conventions, aggregation choices, local noise, and overconfident textual explanations; for this reason, agent outputs should be treated as diagnostic signals rather than ground truth. This framing keeps visual and textual feedback useful while leaving final model decisions grounded in executable artifacts and reproducible evaluation evidence.

3. DS–DA Dual-Agent Team Framework

The preceding sections frame response-family specification as a constrained discovery problem rather than a parameter-estimation task. This section describes the implemented workflow that realizes this framing: calibrated seed construction, bounded agent proposal and diagnosis, and system-controlled validation, numerical settlement, archive management, and state updates, with predictive criteria

and domain-informed proxies serving distinct evaluation roles (Fig. 1).

3.1. System Architecture Overview

Agent roles and decision authority. Stochastic proposal and diagnosis are separated from system-controlled execution. *Prior DS* calibrates seed priors, while *Formula DS* proposes channel-wise structures from context, archive history, and DA feedback. *Global Prediction Analyst* and *Per-Channel Adstock Scorer* provide diagnostics and agent-derived ratings but do not settle candidates. The Knowledge Base (KB) preserves formulas, observations, and channel experience; the Orchestrator controls validation, numerical search, settlement, archive updates, and state transitions.

Inputs. The framework starts from a Meridian-based additive MMM specification, where the outcome is represented as the sum of temporal baseline effects and media effects, with controls or other related treatments included when they are present in the source data. The system calibrates a time seed in Phase 1 and may initialize Phase 2 from it directly or after freezing the time state and calibrating non-time parameters in Phase 1.5. The system also receives channel-level context stored in the KB, which provides channel metadata and prior discovery experience.

Proxy search and settlement backend. LLM outputs do not themselves become settled candidates. Proxy mode evaluates a materialized formula as an executable kernel and searches point parameters and lag length under the frozen global context. FitEngine mode enters the same candidate into Meridian's probabilistic graph for posterior estimation. The modes interoperate by freezing posterior means into numeric states and converting proxy-selected values into deterministic or tight priors for verification. Agent proposals thus enter search state only as reproducible numeric or probabilistic settlements under a shared contract.

Adstock execution semantics. For a candidate g, $x \in \{0, \ldots, L\}$ denotes an integer lag index rather than a normalized media value, and proxy search selects $1 \le L \le 15$. Declared parameters are optimized within type-specific bounds: unit-interval $[0.005, 0.995]$, positive $[10^{-6}, 30]$, nonnegative $[0, 30]$, or real $[-10, 10]$. Evaluating $g(x)$ produces a raw lag-response vector. Nonnegative vectors are divided by their sum; signed vectors by the sum of absolute values. The resulting normalized kernel forms a weighted sum of lagged, scaled media, after which the seed-inherited saturation and coefficient map the Adstocked series to the channel contribution.

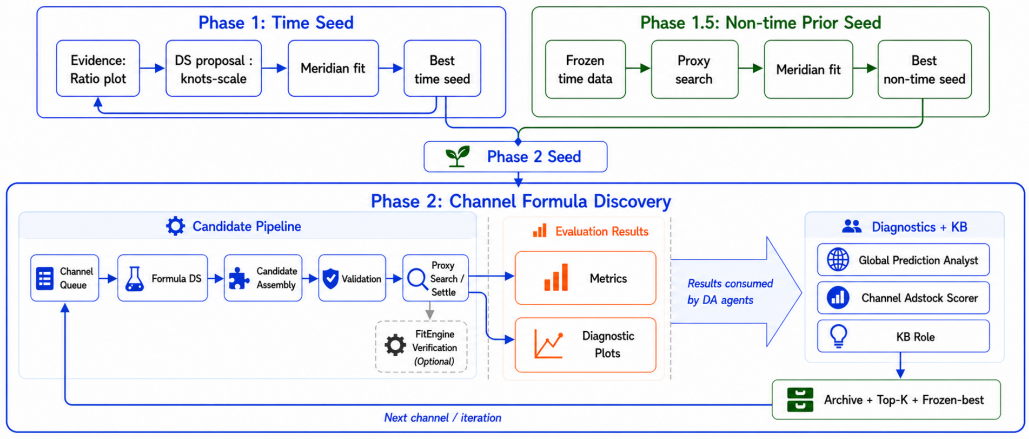

Figure 1 | Overall DS–DA architecture with configurable seed entry. The system may initialize Phase 2 from the Phase 1 time seed or, after optional non-time calibration, from the Phase 1.5 seed; the lower panel shows channel-wise discovery, diagnostic review, and archive-based selection.

Scoring, archive, and final combination search. Only settled candidates enter the archive, which stores configurations, fit and score summaries, DA feedback, diagnostics, and reproducibility paths. Selection therefore uses comparable records rather than raw LLM outputs. The archive supports best-state replacement and per-channel Top-K selection, preferring entries above the score threshold and otherwise falling back to the best available records. Optional final search enumerates per-channel proxy candidates over the seed, reconstructs full predictions, and ranks global configurations.

3.2. Cold-Start Seed Calibration

Formula discovery is unstable when the search loop starts from an uncalibrated additive MMM state: symbolic Adstock candidates may be forced to explain temporal baseline variation, weak media priors, or baseline-side effects rather than channel memory. Seed preparation is therefore configurable: the system establishes the time state in Phase 1 and optionally invokes Phase 1.5 for non-time calibration before channel-specific formula discovery.

Phase 1: Time-only seed calibration. In Phase 1, the system isolates the shared temporal baseline while freezing all media-side formulas and priors. It first constructs a ratio-normalized baseline target from an externally configured time-contribution ratio, giving the time-effect search a numerical anchor rather than leaving the agent to infer scale freely. The *Global Prediction Analyst* then uses the prediction plot as multimodal evidence: it compares the smoothed ratio-normalized target with the current baseline and identifies whether the time effect is missing amplitude, misplaced knots, or excessive flexibility. The *Prior DS* converts this diagnosis and recent history into an executable time-effect hypothesis by proposing a complete knot list for the shared trend and a scale adjustment for `knot_values`; the location of the prior is fixed by the numerical scaffold. Each hypothesis is still materialized as a complete Meridian run, and the selected output is the validated time-effect seed snapshot with the best baseline predictive accuracy.

Phase 1.5: Non-time seed calibration. In the optional Phase 1.5 stage, the system starts from the Phase 1 best state, freezes the calibrated knots and time-effect prior, and restores the full source data configuration. It then uses proxy search to calibrate the remaining non-time family and channel parameters under the frozen temporal scaffold. When DA guidance is invoked in this stage, the *Global Prediction Analyst* receives a segmented without-time diagnostic: both the observed and predicted series subtract the frozen time-effect curve, so the visual feedback targets the remaining non-time/media-side gap rather than reopening the time seed. The proxy optimizer supports multiple search presets (e.g., Latin hypercube with local refinement, cross-entropy method); because each candidate is evaluated through deterministic contribution curves and reconstructed full predictions, this search is lower-cost than repeatedly sampling a full probabilistic graph. The system separates the remaining target-channel effect into channel-level contribution curves and parameter states, which provide the channel-ordering and effect materials used by Phase 2's single-channel iterations. The settlement policy can either accept the best proxy seed directly or pass a small set of top proxy seeds into Meridian Engine for final best-seed selection.

3.3. Phase 2: Channel-Wise Formula Discovery

In Phase 2, the system starts from a validated upstream seed produced during Phase 1 or Phase 1.5. The reported experiments use the latter as the pre-discovery reference: the best available fit before any channel response-function shape is changed. The selected seed provides channel-level effect materials, and the system initializes baseline archive entries, business-score cache values, and best-available states from it. The objective then shifts from seed calibration to response-function discovery. The search does not select among a fixed library of Adstock formulas. Instead, the *Formula DS* generates

parameterized algebraic expressions within an allowed operator and response-mechanism space, while validation enforces parseability, declared shape parameters, and backend compatibility before settlement. This constrained space permits structures beyond predefined backend families without treating LLM output as executable by default. For each target channel, the system tests whether replacing only that channel's Adstock shape can improve the seeded model while preserving a plausible and interpretable response. The *Global Prediction Analyst* also receives a segmented current-channel diagnostic: the reference curve removes the frozen background from the observed outcome, and the candidate curve contains only the proposed channel effect. Candidate selection is therefore not driven by predictive fit alone; it also considers curve-shape rationality and parameter clarity, with the detailed scoring design introduced in Section 3.4.

Operationally, the Orchestrator follows the seed-derived contribution order so that higher-impact channels receive formula-search budget earlier. Within each channel loop, the *Formula DS* converts channel context, KB experience, and DA feedback into a candidate Adstock expression and explains the intended carryover behavior. The proxy optimizer searches the associated numeric settings and lag length, while the Orchestrator combines the active-channel proposal with frozen-best states from all inactive channels.

Algorithm 1 uses lowercase s for a seed state, uppercase S for a score, and Plateau Triggered Exploration (PTE) for score-plateau resets. The implementation searches Adstock structures only; Saturation parameters remain seed-inherited.

Algorithm 1 Phase 2: Seeded Channel-wise Formula Discovery with Optional Global Recombination

Require: Data $\mathcal{D}$; upstream seed s_0; target channels C; policies Π, Ω; knowledge base $\mathcal{K}$
Ensure: Per-channel Top-K candidates and optional global combinations
Order C by seed-derived channel contribution
Bootstrap best-available states, baseline archive entries, and business-score cache from s_0
for each channel c in the ordered target-channel list **do**
  $e_c^{(0)} \leftarrow \text{ArchiveEntry}(\text{BestAvail}_c, S_c^{(0)})$
  Initialize archive list $\mathcal{A}_c \leftarrow [e_c^{(0)}]$
  Initialize $\text{score_history}_c \leftarrow []$ and $\text{pte_cooldown}_c \leftarrow 0$
  Let ctx_c collect KB, history, DA feedback, retry, and PTE state
  while successful settlements for channel $c < N_{\max}$ **do**
    $\text{prop}_c \leftarrow \text{FormulaDS}(\text{ctx}_c)$
    Validate prop_c ($\leq R_{\max}$ retries); continue if invalid
    $\text{candidate} \leftarrow \text{Merge}(\text{prop}_c, \text{BestAvail}_{-c})$
    $\text{settle}_c \leftarrow \text{ProxySettle}(\text{candidate}, \mathcal{D}, \Omega)$; continue on failure
    $S_n \leftarrow \text{AggregateScore}(\text{settle}_c, \Pi)$
    $e_n \leftarrow \text{ArchiveEntry}(\text{settle}_c, S_n)$
    $\mathcal{A}_c$.append(e_n) and update $\mathcal{K}_c$
    score_history_c.append(S_n)
    if $\text{Better}(e_n, \text{BestAvail}_c, \Omega)$ **then**
      $\text{BestAvail}_c \leftarrow \text{FreezeForFutureSearch}(e_n, \Omega)$
    end if
    Update pte_active_c, pte_cooldown_c using recent S_{final}
  end while
end for
for each channel c **do**
  $\mathcal{B}_c \leftarrow [e \in \mathcal{A}_c \mid S_{\text{final}}(e) \geq \tau]$
  $\mathcal{T}_c \leftarrow$ Top-K($\mathcal{B}_c$ if nonempty else $\mathcal{A}_c$) by S_{final}
end for
if final global recombination is enabled **then**
  $\mathcal{G} \leftarrow \text{GlobalComboSearch}((\mathcal{T}_c)_{c \in C}, s_0)$
end if
return $(\mathcal{T}_c)_{c \in C}$ and optional $\mathcal{G}$

DA feedback closes the same channel loop rather than forming a separate stage. After settlement, the *Global Prediction Analyst* checks whether the candidate improves full-model behavior, the *Per-Channel Adstock Scorer* evaluates whether the discovered curve is plausible and parameter-clear, and the KB Builder records compact formula, observation, and experience if valuable. These signals guide subsequent *Formula DS* prompts, the scoring layer, and PTE resets, but they do not replace deterministic settlement or archive logic.

The KB contains iteration traces such as observations and formula history, but its cross-run role is centered on the channel-level `experience.md`. After each iteration, a dedicated KB experience agent decides whether the latest result contains reusable lessons, anti-patterns, or search directions, and rewrites this compact memory rather than appending another log entry. Because the file is stored by channel and reloaded by later Phase 2 proposals, it can preserve both successful priors and failed paths across runs.

3.4. Business-Aware Evaluation and Selection

A structured `ScoreBreakdown` exposes whether predictive fit, curve rationality, or parameter clarity favors a candidate. For a successful Phase 2 settlement entry e at channel iteration i_c, the final score

combines Machine Learning (ML) and business components:

$$S_{\text{final}}(e) = \alpha(i_c)S_{\text{ML}}(e) + (1 - \alpha(i_c))S_{\text{business}}(e). \tag{1}$$

The configurable schedule $\alpha(i_c)$ is nondecreasing and piecewise constant over the channel budget, so early iterations give more weight to business rationality while later iterations give more weight to predictive fit.

The ML score combines clipped R^2 and normalized Root Mean Squared Error (RMSE), using weights normalized over the active ML terms and $[x]_+ = \max(0, x)$:

$$S_{\text{ML}}(e) = w_1^{\text{ML}}[R^2]_+ + w_2^{\text{ML}}[1 - \text{RMSE}_{\text{norm}}]_+. \tag{2}$$

The business score is a domain-informed proxy objective, not a measure of human-expert judgment. The *Per-Channel Adstock Scorer* rates Curve Shape Rationality (CSR) and parameter clarity, while a deterministic simplicity term penalizes unnecessary sampled parameters:

$$\begin{aligned} S_{\text{business},c}(e) &= w_1^{\text{B}} S_{\text{CSR}}(e, c) \\ &\quad + w_2^{\text{B}} S_{\text{clarity}}(e), \\ S_{\text{clarity}}(e) &= \eta \frac{\log 2}{\log(1 + |P(e)|)} \\ &\quad + (1 - \eta) S_{\text{LLM-clarity}}(e). \end{aligned} \tag{3}$$

Because the system updates one channel per Phase 2 iteration, the total business score $S_{\text{business}}(e)$ is computed by replacing the active channel's cached business score and averaging over the target channels. Archive selection uses $S_{\text{final}}(e)$, stored as `score_total`; in proxy settlement, best-state replacement and final combination ranking additionally use proxy predictive quality.

After each channel exhausts its successful-settlement budget, candidates are filtered into an eligible list by the admission threshold τ:

$$\mathcal{B}_c = [\, e \in \mathcal{A}_c \mid S_{\text{final}}(e) \geq \tau \,]. \tag{4}$$

If $\mathcal{B}_c$ is nonempty, the per-channel Top-K candidates are selected from that list; otherwise, the system falls back to the best available archive entries. When final global recombination is enabled, the system forms an archive-derived pool of per-channel proxy candidates, reconstructs full predictions over the seed baseline, and ranks the combinations by proxy R^2.

Budgeted discovery beyond fit thresholds. Each Phase 2 channel search ends after its fixed successful-settlement budget rather than at an R^2 target. This choice reflects the goal of discovering formulas that are not only predictively strong but also interpretable as MMM response functions. The PTE mechanism in Algorithm 1 supports this budgeted search by monitoring recent S_{final} values; when the channel search plateaus, the next *Formula DS* prompt receives a structural-reset signal, with a cooldown to prevent immediate retriggering. The loop can then continue exploring plausible response-shape alternatives instead of prematurely converging to a local high-fit variant.

4. Experiments

The experiments follow the complete workflow described in the method section and are conducted on the same public MMM dataset. We organize them into three stages:

E1. Seed calibration. We test whether staged calibration can produce a reliable Phase 1.5 seed before response-function search.

E2. Phase 2 discovery analysis. We examine the executed channel-wise search, including selected Adstock formulas, curve rationality, and fit–business tradeoffs.

E3. Global residual baselines. Under the same Phase 1.5 seed and paid-media residual target, we compare PySR and LLM-SR as external global formula-discovery baselines for channel-wise DS–DA discovery.

4.1. Dataset and Shared Experimental Reference

All experiments use the public national MMM dataset released with Robyn [1], with revenue as the outcome and five paid media channels as the response-discovery target.

We run Phase 1 and Phase 1.5 from the initial fixed-function MMM state to obtain the shared Phase 1.5 proxy seed. All downstream tests start from this seed, which serves as the Phase 2 discovery reference and the basis for the PySR and LLM-SR paid-media residual target. This design controls data preparation and seed calibration, so later comparisons can mainly focus on formula-discovery behavior. All reported metrics (R^2, $\text{RMSE}_{\text{norm}}$, ML score) are computed on full-data deterministic reconstructions; all methods are evaluated under identical conditions so that relative comparisons reflect formula-discovery quality rather than evaluation protocol differences. With approximately 200 weekly observations, a time-series hold-out would leave a short test window subject to high variance from seasonal or promotional events.

Detailed run-level parameters, including agent-model assignments (Gemini 3.1 Flash Lite for visual DA and Claude Sonnet 4.6 for other agents), are retained in the experiment records. Stage-specific settings are reported with the corresponding results below.

4.2. Seed Calibration Results

In Phase 1, the system calibrates the time effect before any response-function search. Fig. 2 shows the full diagnostic view used to construct the selected Phase 1 time-effect seed.

The diagnostic target is constructed by multiplying the observed revenue by the configured time-contribution ratio (0.5 in this run) and smoothing that ratio-normalized series with a seven-point centered rolling median. This scaled view helps the system recover broad temporal variation and knot placement in Phase 1, rather than chase pointwise fit to the full revenue series.

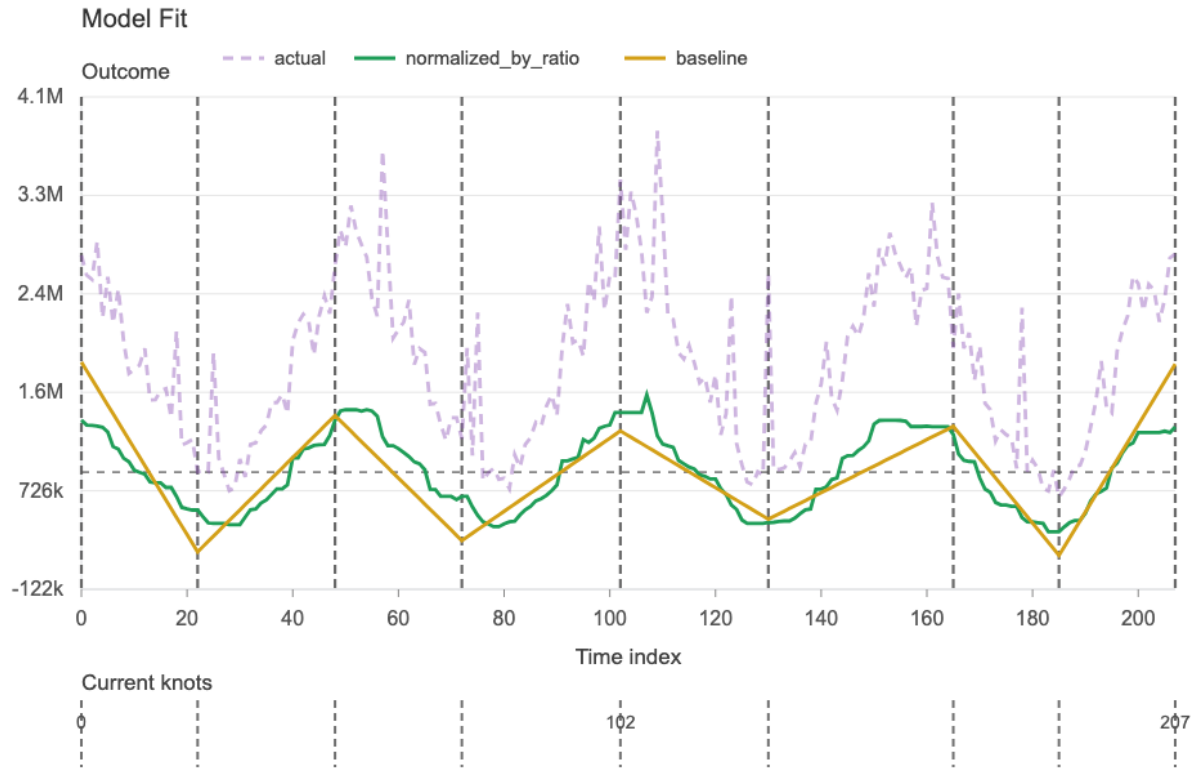

Figure 2 | Full Phase 1 diagnostic view for the selected time-effect seed. Purple is observed revenue, green is the ratio-normalized smoothed target, and orange is the fitted baseline.

To reduce irrelevant visual information for the AI agent, the actual DA prompt consumes a cleaner view (Fig. 3) that retains only the scaled target and the current baseline, allowing the *Global Prediction Analyst* to focus on target–baseline alignment.

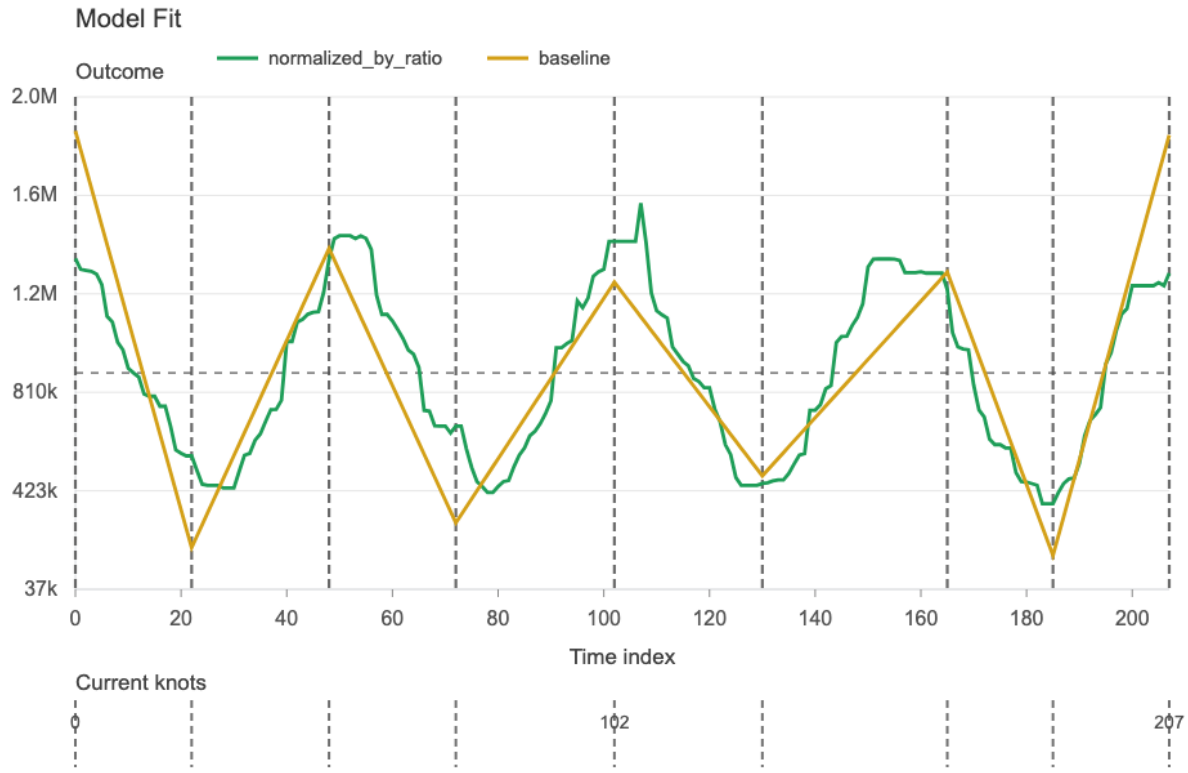

Figure 3 | Reduced Phase 1 DA prompt view. This is the actual image input for the *Global Prediction Analyst* during Phase 1 seed calibration.

Table 1 | Phase 1 Time-Effect Seed Calibration

Stage	R^2	$\text{RMSE}_{\text{norm}}$	S_{ML}
First Phase 1 settled seed	0.6240	0.2405	0.6918
Selected Phase 1 time seed	0.7463	0.1975	0.7632

Table 1 confirms that the first successful Phase 1 settlement has limited predictive quality, indicating that formula discovery would start from a poorly calibrated additive MMM if run directly. The selected Phase 1 seed improves the temporal baseline.

The system freezes the temporal scaffold and calibrates the five paid channels under fixed response-function templates in Phase 1.5. The best seed ($R^2 = 0.7805$, $\text{RMSE}_{\text{norm}} = 0.1837$) is the calibrated fixed-form reference for Phase 2 (Table 2).

4.3. Channel-Wise Formula Discovery

Phase 1.5 optimizes five paid channels as a fixed-form media block; Phase 2 then searches each channel's Adstock shape under that seed. The Phase 1.5 contribution shares set the order: Television (TV) 45.7%, Facebook 24.2%, Out of Home (OOH) 18.3%, Search 7.1%, and Print 4.7%.

For the reported Phase 2 run, each channel receives 50 successful settlements. The configurable scoring setup uses $\alpha(i_c) = 0.3/0.5/0.7$ over channel iterations 1–15/16–35/36–50, $w^{\text{ML}} = (0.7, 0.3)$, $w^{\text{B}} = (0.5, 0.5)$, and a clarity-mixture coefficient $\eta = 0.5$; the archive retains $K = 3$ candidates per

channel using threshold $\tau = 0.8$. Rather than treating the archive as a leaderboard of five channel winners, we use it to identify representative response shapes that differ materially from geometric Adstock and remain interpretable after settlement. Table 2 compares the Phase 2 selected combination against the fixed-form baseline. Under Meridian's fixed 14.54M budget, it also reports optimized Return on Investment (ROI).

Table 2 | Discovery vs. Fixed-Form Baseline

Method	R^2	$\text{RMSE}_{\text{norm}}$	ROI
Fixed-form Adstock	0.7805	0.1837	15.33
DS–DA dual-agent team	**0.8035**	**0.1738**	**15.41**

Relative to this reference, the combination raises R^2 by 0.0230 and reduces $\text{RMSE}_{\text{norm}}$ by 0.0099 while expanding executable response families, showing that structural flexibility does not sacrifice global fit.

Fig. 4 and Fig. 5 show two representative cases selected from this archive analysis. During Phase 2, the *Formula DS* generates formulas within the constrained expression space rather than selecting them from a literature-derived candidate library. We therefore use MMM and advertising literature after discovery to assess whether the resulting lag-response behavior admits a domain interpretation. Such consistency supports business plausibility at the response-shape level; it does not establish that an exact algebraic form is unique or previously unreported.

Within the reported run, 36/50 Search settlements fall into the delayed or short-memory shape class. This concentration is descriptive; it does not establish reproducibility across independent pipeline runs. Here, a response-shape family refers to candidate formulas with similar overall lag-response trends, even when their exact response magnitudes, peak positions, or tail strengths differ. The grey curves in Fig. 4 illustrate this family of related delayed or short-memory shapes, while the orange curve is the high-scoring representative adopted in the final sequential combination:

$$g_{\text{search}}(x) = \lambda x \exp(-\lambda x). \tag{5}$$

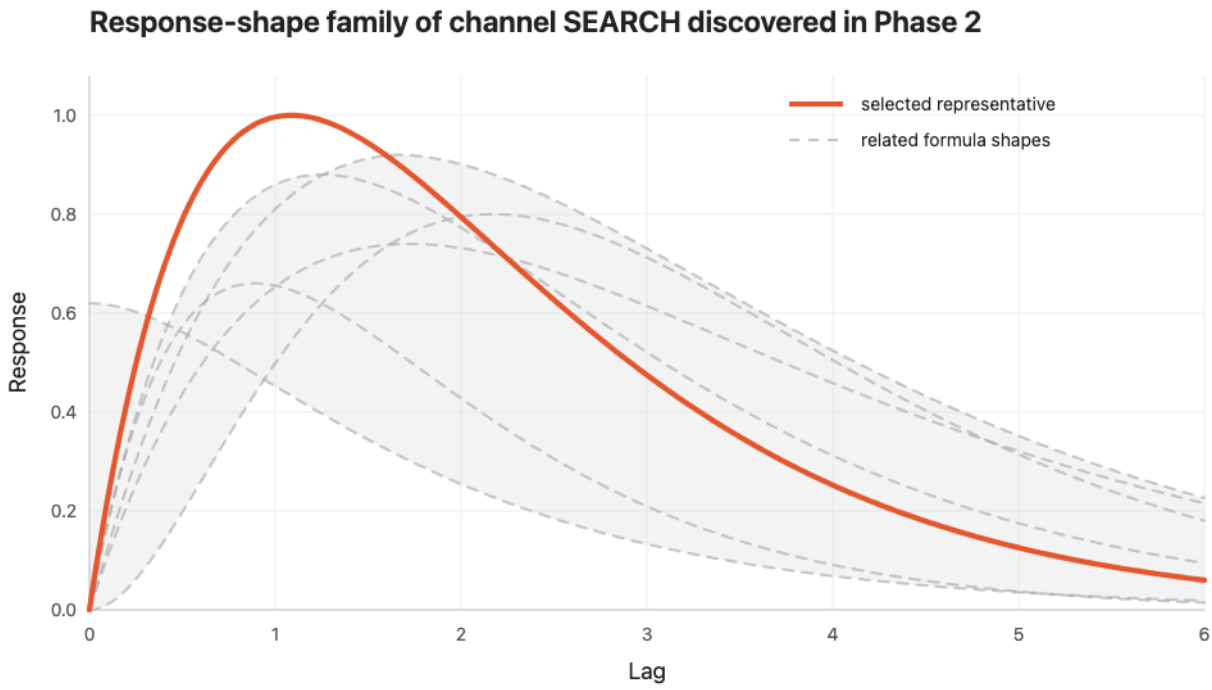

Figure 4 | The orange curve is the selected representative; grey dashed curves illustrate related formula shapes found during search.

This family-level finding suggests that Search is better represented by delayed or short-memory response shapes than by a fixed immediate-peak geometric decay. The interpretation is consistent with contemporary MMM practice, where carryover is modeled through delayed or flexible Adstock rather than only a geometric curve; for example, Jin et al. model carryover and shape effects, PyMC-Marketing provides delayed and Weibull-PDF Adstock transforms, and Robyn's Weibull PDF option permits peaks after the first period [10, 20, 21].

Within the same run, 39/50 Print settlements produce the short-memory envelope family. Fig. 5 shows a family with a short early shoulder, a steep middle-lag compression, and a weak long tail after the fourth lag. The selected representative can be summarized as

$$g_{\text{print}}(x) = \exp\left[-\left(\frac{x}{T}\right)^{\beta}\right]\left(a + (1-a)\exp(-\lambda x)\right). \tag{6}$$

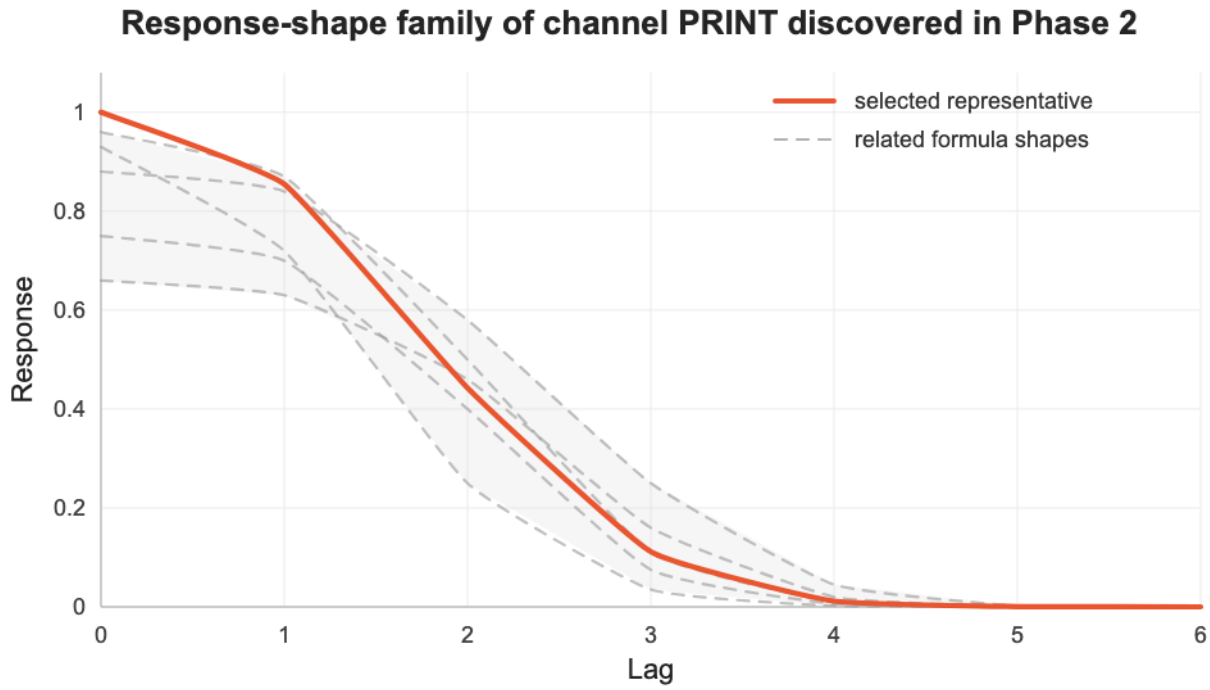

Figure 5 | The orange curve is the selected representative; grey dashed curves illustrate related formula shapes found during search.

where the first factor forms a Weibull-like decay envelope and the second factor controls the remaining tail. This family therefore differs from a single-rate geometric decline: it allows Print to retain a limited carryover shoulder while quickly suppressing later residual mass. The interpretation is consistent with evidence that advertising carryover varies across communication forms and model specifications [9]. It is also compatible with direct-response evidence from weekly and monthly magazines and customer mailings, where response over time is modeled as a decaying curve rather than an instantaneous effect [22]. We therefore interpret the Print candidate as an executable short-memory carryover shape, not as proof of a unique causal mechanism for print advertising.

Beyond per-iteration formula selection, the KB accumulates cross-iteration experience that steers future proposals away from known failure modes. To illustrate, the Facebook channel stores the compact experience record shown in Fig. 6.

Facebook channel experience memory

Durable lesson	"Pure exponential decay is the strongest **simple baseline** . . . score_simplicity=1.0."
Anti-pattern	"**Low-decay attractor trap**: **all fifty** proxy searches fall into an equivalent low-decay region . . . any tail-suppression parameter is **bypassed by proxy** . . ."
Direction	"Single-parameter exponential + `lag_length`=3 or 4 . . . **physical truncation is the only tail-removal path** . . ."

Figure 6 | Recorded experience excerpt for channel Facebook.

The Facebook example shows this behavior: a simple exponential baseline remains useful, the low-decay attractor is marked as an anti-pattern, and future search is redirected toward shorter lag windows instead of another tail-suppression variant.

These channel-level findings illustrate a secondary output of the framework: discovered formulas and their shape families surface channel-specific response hypotheses that a fixed-template system cannot express. Together with the diagnostic annotations and anti-patterns accumulated in the knowledge base, they persist across runs as reusable modeling assets.

4.4. Comparison with Global Discovery Baselines

The baseline study evaluates PySR [23] and LLM-SR [4] in a controlled global-discovery setting using the same Phase 1.5 paid-media residual target. PySR represents evolutionary symbolic regression for closed-form equation search, while LLM-SR represents LLM-guided symbolic regression. The LLM-SR run uses Claude Sonnet 4.6, matching our text-generation agents, but it has no visual DA component for prediction or curve diagnostics.

The PySR baseline uses a broad-math operator setting with 10000 iterations, and the LLM-SR baseline uses a 2500-sample budget. Since these baselines are not designed for channel-level discovery, they do not use the DA business-rationality score, channel-wise frozen-best replacement, or the Top-K archive; Table 3 therefore reports the common predictive metrics under the controlled residual target.

Table 3 | Global Paid-Media Residual Comparison

Method	R^2	$\mathrm{RMSE}_{\mathrm{norm}}$	S_{ML}
PySR	0.6940	0.2169	0.7207
LLM-SR	0.7132	0.2100	0.7362
DS–DA dual-agent team	**0.8035**	**0.1738**	**0.8103**

Table 3 shows that the integrated channel-wise workflow achieves the strongest predictive result under the shared residual target. Compared with LLM-SR, it improves R^2 by 0.0903 and reduces $\mathrm{RMSE}_{\mathrm{norm}}$ by 17.2%. This comparison supports the integrated workflow in the controlled task, but it does not isolate the individual contribution of channel decomposition, staged calibration, business-aware scoring, or frozen-best settlement.

5. Conclusion

This paper presented an LLM-guided symbolic search design for business-constrained function discovery, evaluated on MMM Adstock discovery. It separates seed calibration from formula discovery and restricts LLMs to structural proposal and diagnosis; system services control validation, settlement, and state updates. The Phase 2 workflow explores beyond predefined backend families while keeping candidates executable and auditable. Selection combines predictive metrics with agent-derived, domain-informed proxies for curve-shape rationality and parameter clarity.

On the public Robyn dataset, the search expands executable response families beyond the calibrated fixed-form reference while incrementally improving global fit. Post-discovery inspection finds representative Search and Print shapes consistent with established MMM and advertising literature, without claiming unique algebraic forms. The integrated workflow also outperforms PySR and LLM-SR under the shared residual target, but the comparison does not isolate component contributions. Limitations include one dataset, a modest budget, full-data deterministic evaluation, agent-derived proxies without independent human validation, no pipeline repeats, and the causal ambiguity of observational MMM. The 36/50 and 39/50 counts show only within-run family concentration. Future work should extend discovery beyond Adstock to saturation and other components, validate stability and human agreement on broader datasets, and develop the KB into auditable long-term memory for reusable lessons, verified industry experience, and cross-task warm starts.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant No. 72432007, which was awarded to one of the authors, Yaohui Jin.

References

[1] Meta Marketing Science, "Robyn: Continuous and semi-automated MMM," https://github.com/facebookexperimental/Robyn, 2022.

[2] Google Meridian Marketing Mix Modeling Team, "Meridian: Marketing mix modeling," https://github.com/google/meridian, 2026, software library, version 1.5.3.

[3] B. Romera-Paredes, M. Barekatain, A. Novikov, M. Balog, M. P. Kumar, E. Dupont, F. J. Ruiz, J. S. Ellenberg, P. Wang, O. Fawzi *et al.*, "Mathematical discoveries from program search with large language models," *Nature*, vol. 625, no. 7995, pp. 468–475, 2024.

[4] P. Shojaee, K. Meidani, S. Gupta, A. Barati Farimani, and C. K. Reddy, "LLM-SR: Scientific equation discovery via programming with large language models," *arXiv preprint arXiv:2404.18400*, 2024.

[5] D. M. Hanssens, L. J. Parsons, and R. L. Schultz, *Market response models: Econometric and time series analysis*. Kluwer Academic Publishers, 1990.

[6] P. E. Rossi, G. M. Allenby, and R. McCulloch, *Bayesian statistics and marketing*. John Wiley & Sons, 2005.

[7] Google Research, "LightweightMMM: Lightweight Bayesian marketing mix modeling," https://github.com/google/lightweight_mmm, 2021, software library.

[8] D. Berkowitz, A. Allaway, and G. D'Souza, "Estimating differential lag effects for multiple media across multiple stores," *Journal of Advertising*, vol. 30, no. 4, pp. 59–65, 2001.

[9] C. Köhler, M. K. Mantrala, S. Albers, and V. K. Kanuri, "A meta-analysis of marketing communication carryover effects," *Journal of Marketing Research*, vol. 54, no. 6, pp. 990–1008, 2017.

[10] Y. Jin, Y. Wang, Y. Sun, D. Chan, and J. Koehler, "Bayesian methods for media mix modeling with carryover and shape effects," *Google Inc*, 2017, available at: https://research.google/pubs/pub46001/.

[11] J. R. Koza, *Genetic programming: On the programming of computers by means of natural selection*. MIT Press, 1992.

[12] S.-M. Udrescu and M. Tegmark, "AI Feynman: A physics-inspired method for symbolic regression," *Science Advances*, vol. 6, no. 16, p. eaay2631, 2020.

[13] M. Cranmer, A. Sanchez-Gonzalez, P. Battaglia, R. Xu, K. Cranmer, D. Spergel, and S. Ho, "Discovering symbolic models from deep learning with inductive biases," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 17 429–17 442.

[14] B. K. Petersen, M. Landajuela, T. N. Mundhenk, C. P. Santiago, S. K. Kim, and J. T. Kim, "Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients," in *International Conference on Learning Representations (ICLR)*, 2021. [Online]. Available: https://openreview.net/forum?id=m5Qsh0kBQG

[15] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing reasoning and acting in language models," in *International Conference on Learning Representations (ICLR)*, 2023, arXiv preprint arXiv:2210.03629.

[16] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 24 824–24 837.

[17] A. Masry, D. X. Long, J. Q. Tan, S. Joty, and E. Hoque, "ChartQA: A benchmark for question answering about charts with visual and logical reasoning," in *Findings of the Association for Computational Linguistics: ACL 2022*, 2022, pp. 2263–2279. [Online]. Available: https://aclanthology.org/2022.findings-acl.177/

[18] F. Liu, F. Piccinno, S. Krichene, C. Pang, K. Lee, M. Joshi, Y. Altun, N. Collier, and J. Eisenschlos, "MatCha: Enhancing visual language pretraining with math reasoning and chart derendering," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023, pp. 12 756–12 770. [Online]. Available: https://aclanthology.org/2023.acl-long.714/

[19] F. Liu, J. Eisenschlos, F. Piccinno, S. Krichene, C. Pang, K. Lee, M. Joshi, W. Chen, N. Collier, and Y. Altun, "DePlot: One-shot visual language reasoning by plot-to-table translation," in *Findings of the Association for Computational Linguistics: ACL 2023*, 2023, pp. 10 381–10 399. [Online]. Available: https://aclanthology.org/2023.findings-acl.660/

[20] PyMC Labs, "Guide to adstock transformations in PyMC-Marketing," https://www.pymc-marketing.io/en/latest/notebooks/mmm/adstock_functions_guide.html, 2026, accessed 2026-05-10.

[21] Meta Marketing Science, "Robyn: Adstock transformations," https://facebookexperimental.github.io/Robyn/docs/features/, 2025, accessed 2026-05-10.

[22] C. L. Bauer, "Direct response advertising: Forecasting responses over time," *Journal of Direct Marketing*, vol. 1, no. 4, pp. 38–50, 1987.

[23] M. Cranmer, "Interpretable machine learning for science with PySR and SymbolicRegression.jl," 2023.