Low-Rank Multi-Source Data-Mixing Laws for LLM Pretraining: Identifiable Scaling with $\tilde{O}(rK)$ Converged Runs

Liz Lemma Future Detective

January 17, 2026

Abstract

FLP and FLP-M show that downstream task performance can be predicted efficiently by (i) fitting compute-to-loss scaling laws on fully converged sampling models and (ii) mapping validation loss to performance using intermediate checkpoints. However, FLP-M is only demonstrated for binary mixtures (general text + code), while 2026-era pretraining routinely mixes many sources (web subsets, code, math, synthetic tool traces, multilingual corpora, multimodal transcripts). Naively generalizing to K sources leads to a combinatorial mixture space and an intractable number of converged runs. We propose a many-source mixing law that makes this regime statistically and computationally tractable: domain-specific validation losses obey a log-linear model whose source-by-domain sensitivity matrix is low-rank, capturing a small number of latent capability factors that mediate transfer and overlap between sources. We give (i) an experiment design for selecting sparse compute allocations that ensures identifiability, (ii) a nuclear-norm regularized estimator with finite-sample uniform prediction guarantees over the simplex of mixtures, and (iii) matching minimax lower bounds showing that $\Omega(rK)$ converged runs are necessary (for constant D). Finally, we demonstrate how the resulting loss predictor composes with FLP-M's loss-to-performance stage to forecast and optimize downstream performance over mixture ratios, substantially reducing required converged sweeps compared to pairwise/binary baselines. Empirical evaluation on 3–6 sources validates the low-rank hypothesis and shows improved prediction and mixture optimization, while simulations illustrate scaling to K≈20.

Table of Contents

2. 2. Background and source context: recap FLP/FLP-M two-stage idea; why loss is a stable intermediate; why average loss fails under mixtures; what changes in K-source setting.

3. 3. Problem formulation: define multi-source compute allocation, converged-run oracle, domain-loss targets; specify prediction and mixture-optimization tasks; evaluation metrics (uniform loss error; downstream error via Lipschitz composition).

4. 4. Model class (Low-Rank Mixing Laws): log-linear scaling with low-rank coefficient matrix; optional extensions (total-compute term, piecewise phases); discussion of interpretability (latent capability factors).

5. 5. Experiment design for identifiability: sparse allocation families; conditioning of design matrix in log-space; practical recipe for choosing runs under compute constraints; robustness to near-zero allocations.

6. 6. Estimation algorithm: nuclear-norm regularized multi-response regression; cross-validation for rank/regularization; numerical stability and constraints (monotonicity in compute).

7. 7. Theory I (Upper bounds): estimation error in Frobenius/operator norms; uniform-in-mixture prediction bound over simplex; dependence on r,K,D,n,σ,design conditioning.

8. 8. Theory II (Lower bounds): minimax sample complexity; impossibility without low-rank (or other structure); matching the upper bounds up to logs and constant factors.

9. 9. Composition to downstream prediction + mixture optimization: error propagation bound through any Lipschitz loss$\rightarrow$performance map (including FLP-M stage-2 NN); optimization formulations and when they are convex/tractable; practical algorithms (projected search / BO on simplex).

10. 10. Experiments (recommended): 3–6 sources real training; ablations on rank, design, noise; comparison to pairwise/binary laws; mixture optimization case studies; scaling simulations to K=10–20.

11. 11. Discussion and limitations: misspecification, nonstationary data, overlap and contamination, extending to continual pretraining and post-training pipelines; reproducibility artifacts (design + fitting code).

1 Introduction

Modern pretraining pipelines rarely rely on a single homogeneous corpus. Instead, we allocate a fixed compute budget across a collection of heterogeneous sources—web text, code, mathematics, synthetic data, multilingual corpora, or curated instruction traces—and we seek a mixture that performs well on a family of downstream domains. The operational question is therefore not whether to include a given source, but how to distribute compute across many sources under a simplex constraint. When the number of sources is small, one may attempt to resolve this question by direct search or by fitting a low-dimensional response surface; however, as the number of sources grows, both approaches deteriorate. Direct search becomes combinatorial, while unrestricted response-surface fitting incurs a parameter blowup and demands an infeasible number of fully-converged pretraining runs.

Two-stage procedures in the spirit of FLP/FLP-M address the cost of repeated full pretraining by learning an intermediate predictor from pretraining signals and then composing it with a (cheaper) map from those signals to downstream performance. In the binary setting (two sources), this approach can be implemented by sampling a modest number of mixtures, measuring a stable intermediate quantity (typically a converged validation loss), and regressing performance on that intermediate. Yet the binary intuition does not directly extend to the many-source case. First, the space of mixtures is now $(K-1)$-dimensional, so "probing along a line" between two endpoints no longer explores the relevant directions. Second, naive scalar summaries such as average validation loss can fail to discriminate mixtures: two allocations may have the same average loss while exhibiting sharply different losses on the specific domains that matter downstream. Third, the downstream model used in stage 2 may be well-behaved only on a restricted region of intermediate signals, and the many-source mixture simplex makes it difficult to ensure that the training runs populate that region without an explicit design.

We therefore require a principled generalization that (i) remains identifiable with a small number of converged runs, (ii) yields accurate prediction uniformly over the simplex of allocations, and (iii) produces a predictor that is amenable to subsequent optimization over mixtures. A minimal modeling choice is to view the converged log-loss on each validation domain as a function of the log-compute allocation vector, since compute allocation is multiplicative and empirical scaling laws are typically linear in log-variables. If one were to fit an unconstrained linear model separately for each domain, one would estimate a $D \times K$ coefficient matrix without additional structure. This is statistically expensive: absent further assumptions, the number of degrees of freedom scales as DK, and any method must essentially pay for each coordinate. Moreover, even if we only care about a small subset of domains, the induced coefficient vectors can vary in a correlated manner, suggesting that treating domains independently is wasteful.

Our central structural hypothesis is that the domain-by-source effects are low-dimensional: there exist only r latent factors that explain how source allocations translate into improvements across all domains simultaneously. Concretely, we posit that the $D \times K$ coefficient matrix governing log-loss responses has rank at most $r \ll \min\{D, K\}$. This assumption is both an inductive bias and an empirical regularity: many domains improve together when we allocate more compute to certain sources (e.g., code and formal math), while other domains share sensitivities to different sources (e.g., multilingual and low-resource text). Low rank captures such shared structure while permitting domain-specific intercepts and heterogeneous slopes.

We design our approach around the following desiderata.

- *Identifiability under expensive supervision.* Each fully-converged training run is treated as an oracle call producing a D-dimensional vector of domain log-losses. Since such calls are costly, we must be able to recover the mixing law from a number of runs far smaller than K times the number of domains.

- *Sample efficiency with uniform guarantees.* We require prediction error bounds that hold uniformly over all feasible allocations, not merely on the finite set of probed mixtures. Uniform control is necessary for mixture optimization, where the optimizer will seek maxima at points not previously queried.

- *Optimization readiness.* The fitted model should yield a smooth (ideally convex after a change of variables) surrogate objective over the simplex. At minimum, it must allow reliable gradient-based or projected methods, and it should make explicit any monotonicity constraints (more compute to a source should not worsen loss, within the modeled regime).

Our contributions align with these requirements. First, we formalize a mixing law in which each domain log-loss is affine in the log-allocation vector and the coefficient matrix is low rank. This directly generalizes the binary mixture view to K sources while preserving an interpretable linear structure in log-space. Second, we give an experiment design that uses sparse mixtures: each run assigns substantial compute to a small subset of sources while keeping the rest at a floor level, thereby producing a design matrix that is sufficiently diverse for low-rank recovery without requiring dense exploration of the entire simplex. The sparsity level is chosen to scale with the latent rank r, reflecting the effective dimension of the model rather than the ambient dimension K.

Third, we provide a polynomial-time estimator based on nuclear-norm regularized least squares. This estimator is the natural convex relaxation for low-rank matrix regression: it shares statistical guarantees with rank-constrained formulations while avoiding nonconvex optimization over factored parameters. The resulting predictor has a simple closed form: predicted log-losses are affine in $\log C$, hence predicted losses are positive after exponentiation. This form is important for optimization, since it makes the dependence on allocations explicit and differentiable (away from any imposed floors on C_k).

Fourth, we establish finite-sample guarantees. Under standard sub-Gaussian noise assumptions and a restricted strong convexity condition on the designed log-allocations, we bound the estimation error of the coefficient matrix and translate it into a uniform prediction bound over the entire simplex (with a small compute floor to control $\log C$). The dependence of the run complexity on D reflects a key advantage of the setting: each converged run returns D losses, so increasing the number of measured domains effectively provides parallel supervision. We complement the upper bound with a minimax lower bound showing that, even with adaptivity, one cannot in general beat the scaling in $rK/(D\varepsilon^2)$ (up to logarithmic factors) when D is constant. Thus the proposed procedure is information-theoretically near-optimal within the stated model class.

Finally, we connect loss prediction to downstream performance prediction. Stage 2 in FLP-M can be viewed abstractly as a map from the vector of domain log-losses to a scalar performance metric (or a vector of metrics). When this map is Lipschitz on the relevant region of loss space—as is the case for linear predictors and for small neural networks with bounded weights on bounded inputs—uniform loss-prediction error transfers directly to uniform performance-prediction error. This composition principle justifies treating converged loss as the stable intermediate signal: the stage-1 predictor can be trained with a small number of expensive runs, while stage 2 can be fit with cheaper labels and then safely composed with stage 1.

In addition to prediction, we are ultimately interested in choosing an allocation. The low-rank log-linear model yields an optimization-ready surrogate: optimizing a downstream objective over the simplex can be carried out by projected methods, and in special cases (e.g., concave objectives in the negative log-losses) the problem becomes convex after transforming variables to log-allocations. When the downstream map is unrestricted, global optimization can be computationally hard; our framework isolates this difficulty from the estimation problem by providing accurate and efficiently evaluable predictions, so that mixture search can proceed empirically with clear statistical guarantees on the underlying response model.

In summary, we provide a mathematically explicit route from binary mixture heuristics to a many-source regime: we replace scalar summaries by a vector of domain losses, impose low-rank structure to avoid combinatorial sample complexity, design sparse mixture experiments to satisfy the needed conditioning, and obtain uniform prediction guarantees suitable for subsequent mixture optimization and downstream composition. The result is a sample-efficient, optimization-compatible generalization of FLP-M that scales with the intrinsic latent dimension rather than the number of sources.

2 Background and source context

Two-stage mixture selection procedures arise from a pragmatic constraint: a single fully-converged pretraining run can be orders of magnitude more expensive than any auxiliary measurement performed during or after training. The FLP paradigm exploits this asymmetry by introducing an *intermediate* quantity that is (i) inexpensive to measure for a trained model and (ii) predictive of downstream performance. One then spends a small number of expensive runs to learn how the intermediate quantity varies with the mixture, and a larger number of cheap labels to learn how downstream performance varies with the intermediate quantity. The resulting composite predictor can be optimized over mixtures without requiring an exhaustive search over fully-converged runs.

In the binary mixture setting, the intermediate quantity is often taken to be a validation loss measured on a fixed held-out distribution. Let us write $C \in \mathbb{R}_+^2$ for the compute allocated to each of two sources under a fixed budget, and let $L(C)$ denote a converged validation loss for a particular reference domain. The stage-1 task is to fit an empirical relation $C \mapsto L(C)$ from a modest number of mixtures (often sampled along the line segment joining the two extreme allocations), while stage 2 fits a function h such that downstream performance is approximated by $h(L)$. This decomposition is attractive precisely because stage 2 can be trained using substantially cheaper supervision: for example, one may label many intermediate checkpoints, many prompt suites, or many downstream tasks, while only rarely paying the cost of full pretraining. The essential idea is to avoid repeatedly paying for the same expensive signal (the dependency of the converged model on mixture) when one merely wishes to reweight the downstream objective.

The principal reason that validation loss is a natural intermediate is stability. First, converged losses tend to be low-variance compared to downstream metrics that depend on finicky evaluation pipelines, decoding choices, or task-specific idiosyncrasies. Second, loss is *dense* in the sense that a single trained model yields many loss evaluations across domains essentially for free once one has fixed the evaluation suites. Third, loss is connected to training dynamics: at least within a pretraining regime and architecture class, decreases in relevant domain losses tend to correlate with improvements in a wide range of downstream measures. In particular, if we model downstream performance as a Lipschitz (or otherwise regular) function of a vector of domain losses, then small uniform errors in the loss predictor translate directly into small errors in performance prediction. This is the sense in which loss can serve as an information-bearing sufficient statistic for mixture selection, while downstream metrics are treated as a readout from that statistic.

The FLP-M variant emphasizes this point by allowing the intermediate to be *multi-dimensional*. Rather than compressing the model state into a single scalar loss, we track a vector of losses across multiple validation domains. This is not merely a refinement: it addresses a concrete failure mode of scalar summaries. If a mixture trades off performance across domains, then the average loss can hide meaningful variation. To make this precise, suppose we evaluate losses on D domains and consider two allocations C and C'. It may happen that

$$\frac{1}{D}\sum_{d=1}^{D}\log L_d(C) \approx \frac{1}{D}\sum_{d=1}^{D}\log L_d(C'),$$

while $\log L(C)$ and $\log L(C')$ differ substantially coordinatewise. Any downstream objective that effectively reweights domains—either explicitly, as in a mixture of tasks, or implicitly, as in a complex evaluation suite—will distinguish between C and C' even though the average loss does not. Thus a scalar intermediate can be non-identifiable with respect to the decisions we actually care about: different allocations can map to the same scalar, yet yield different downstream performance. The vector of domain losses is a minimal remedy, as it preserves the degrees of freedom needed to represent such tradeoffs.

Even when a scalar summary is not exactly constant across mixtures, it can still be statistically inefficient. If the true relationship between downstream performance and domain losses is approximately linear in a small number of directions in $\mathbb{R}^D$, then collapsing $\mathbb{R}^D$ to $\mathbb{R}$ discards information that could have reduced variance in stage-2 fitting. Conversely, keeping D losses allows stage 2 to discover which domains matter and how they interact, while stage 1 supplies a predictor that can be queried at arbitrary mixtures. In this sense FLP-M is better viewed as a representation-learning step: we choose a representation of the trained model (domain losses) that is cheap, stable, and sufficiently expressive for downstream prediction.

The many-source setting changes the geometry of the problem in a more fundamental way. With two sources, the feasible set of allocations is one-dimensional, and a small number of probes along the mixture line can cover the relevant range. With K sources, the feasible set is a $(K-1)$-dimensional simplex, and the naive extension of line probing becomes meaningless: sampling along a few edges explores only a negligible portion of the simplex volume, and it fails to excite most directions in the mixture space. Moreover, simple gridding is infeasible: even a coarse discretization at resolution m scales as m^{K-1}. Consequently, the stage-1 step must be reformulated as an experiment design problem in high dimension, where each expensive run provides a vector response and we must choose allocations to make the resulting regression well-posed.

A second change is that the relevant inputs are *multiplicative.* Compute allocation is naturally constrained by a budget, and the effect of scaling a source by a constant factor is empirically closer to additive in log-space than in raw compute. This motivates modeling the response as a function of the log-allocation vector. In the binary case, this viewpoint is often implicit: one draws mixtures by varying a ratio and then fits a smooth curve. In the K-source case, the log map offers a uniform coordinate system in which the simplex constraint becomes a structured subset and linear models become interpretable as elasticities of loss with respect to compute. The transformation also clarifies the need for a compute floor: without bounding C_k away from zero, $\log C_k$ is unbounded below and no uniform control is possible.

A third change concerns the role of *domain structure.* The stage-1 mapping from allocations to the vector of domain losses can be viewed as a D-output regression problem with K inputs. If we fit each domain independently, we must estimate DK slope parameters plus intercepts. In the regime where K is large and oracle calls are expensive, this is not tenable. Yet domain losses are not arbitrary: empirical experience suggests that domains share sensitivities to sources, as when several domains all benefit from additional code data or when multilingual domains exhibit correlated responses to multilingual corpora. This motivates a structural prior that couples domains and reduces effective dimension. The low-rank hypothesis used later is one such coupling: it asserts that the $D \times K$ matrix of source effects lies near a low-dimensional subspace, so that observing losses on many domains in each run provides a form of multi-task supervision.

Finally, optimization considerations impose additional constraints on what constitutes a useful intermediate model. Since the ultimate goal is to recommend an allocation, we require that the stage-1 predictor be evaluable and optimizable over the simplex. A black-box regressor fitted only on observed mixtures may interpolate poorly off-support, and it may admit pathological behavior that misleads an optimizer. By contrast, models that are affine in log-allocations yield smooth objectives and transparent monotonicity properties (e.g., more compute to a source should not systematically worsen loss within the modeled regime). In the many-source case, this is not a cosmetic preference: the optimizer will necessarily explore mixtures that were never run, and thus the stage-1 model must come with *uniform* error control on a continuous domain, not merely pointwise fit on a finite training set.

In summary, the two-stage idea remains the correct cost decomposition in the K-source regime, but each of its components must be strengthened. Stage 1 must handle high-dimensional inputs and exploit shared structure across domains; stage 2 must consume a vector of intermediate signals rather than a single average; and the composition must support reliable optimization over a simplex. These requirements lead naturally to (i) modeling log-losses as structured functions of log-allocations, (ii) designing sparse, information-rich mixtures for the expensive runs, and (iii) fitting a predictor whose error can be controlled uniformly so that downstream composition and mixture optimization are statistically meaningful.

3 Problem formulation

We consider K pretraining data sources (e.g., web, code, math, synthetic, multilingual) under a fixed total compute budget $C_{\mathrm{tot}} > 0$. A *multi-source compute allocation* is a vector $C = (C_1, \dots, C_K) \in \mathbb{R}_+^K$ satisfying the budget constraint

$$\sum_{k=1}^{K} C_k \;=\; C_{\mathrm{tot}}.$$

The feasible set is the simplex

$$\Delta(C_{\mathrm{tot}}) \;=\; \Big\{ C \in \mathbb{R}_+^K : \sum_{k=1}^{K} C_k = C_{\mathrm{tot}} \Big\}.$$

Because our predictors will be expressed in log-allocation coordinates, we must control behavior near the boundary of the simplex. Concretely, we assume either an explicit compute floor $C_k \geq C_{\min} > 0$ for all k (implemented by reserving $KC_{\min}$ compute and distributing the remainder), or an equivalent soft flooring mechanism; in either case we work with a bounded log-allocation map

$$x(C) \;=\; \log(C) \in \mathbb{R}^K,$$

where $\log(\cdot)$ is applied elementwise and is understood to include the chosen floor in practice. This boundedness is not a modeling convenience but a prerequisite for uniform control: without a lower bound on C_k, one has $\inf_{C \in \Delta(C_{\mathrm{tot}})} x_k(C) = -\infty$ and no estimator can attain a finite worst-case error over $\Delta(C_{\mathrm{tot}})$.

We fix D validation domains, indexed by $d \in [D]$, and interpret a domain as a held-out distribution together with a well-defined token-level loss evaluation. For each allocation $C \in \Delta(C_{\mathrm{tot}})$ we write $L_d(C) > 0$ for the *fully-converged* loss on domain d attained by pretraining under allocation C until the relevant notion of convergence (fixed token budget, compute budget, or an agreed stopping rule). We collect these into the vector

$$L(C) \;=\; (L_1(C), \dots, L_D(C)) \in \mathbb{R}_+^D, \qquad \log L(C) \;=\; (\log L_1(C), \dots, \log L_D(C)) \in \mathbb{R}^D.$$

The key operational asymmetry that motivates our setting is that, for a given trained model, evaluating $L_d(C)$ across many d is comparatively cheap, whereas obtaining the model itself at convergence for a new allocation C is expensive. Hence each converged run yields a dense D-dimensional response at the price of a single oracle call.

We formalize this via a *converged-run oracle* $\mathcal{O}$. On input an allocation $C \in \Delta(C_{\text{tot}})$, the oracle executes one converged pretraining run under C and returns noisy log-loss measurements

$$\mathcal{O}(C) \;=\; y \in \mathbb{R}^D, \qquad y \;=\; \log L(C) \;+\; \xi,$$

where $\xi \in \mathbb{R}^D$ captures run-to-run randomness (initialization, data order, nondeterminism, etc.) and measurement noise. We do not require a specific distributional form at this point; later guarantees will assume independent sub-Gaussian coordinates conditional on C. The algorithm is permitted to choose allocations adaptively based on past observations, although our main designs will be non-adaptive.

The stage-1 task is: given at most n oracle calls, fit a predictor that maps allocations to *all* D domain losses. Formally, after choosing allocations $C^{(1)}, \dots, C^{(n)} \in \Delta(C_{\text{tot}})$ and observing

$$y^{(i)} = \mathcal{O}\Big(C^{(i)}\Big) \in \mathbb{R}^D, \qquad i \in [n],$$

we output a function

$$\widehat{L} : \Delta(C_{\text{tot}}) \to \mathbb{R}^D_+, \qquad C \mapsto \widehat{L}(C),$$

intended to approximate $L(C)$ for *all* feasible allocations, not merely those sampled. Since mixture selection will require optimizing over a continuum of allocations, the relevant quality criterion is a *uniform* error bound rather than a pointwise or average-case fit on the design points. We therefore evaluate stage-1 prediction by the worst-case log-loss error

$$\mathcal{E}_\infty(\widehat{L}) \;=\; \sup_{C \in \Delta(C_{\text{tot}})} \big\| \log \widehat{L}(C) - \log L(C) \big\|_\infty.$$

Working in log-loss is natural for two reasons. First, it aligns with empirical scaling behaviors in which multiplicative improvements in loss correspond more closely to additive changes in log-loss. Second, it avoids an undesirable dependence on the absolute scale of $L_d(C)$ across domains: a uniform bound in $\log L$ is invariant to constant rescalings of the loss units.

In some applications we may care about a weighted or structured notion of error across domains (for example, emphasizing particular domains or grouping them). Our baseline criterion is coordinatewise control as above; any alternative that is Lipschitz with respect to $\|\cdot\|_\infty$ (or $\|\cdot\|_2$ on bounded sets) will transfer from the same type of analysis. The emphasis on uniformity also clarifies why experiment design matters: even if n is large enough to interpolate the observed points, a poor choice of allocations may leave entire regions of $\Delta(C_{\text{tot}})$ effectively unconstrained, allowing arbitrarily bad extrapolation.

The stage-1 predictor is a means to an end: recommending an allocation that performs well on downstream evaluations. To model this, we introduce an abstract *loss-to-performance map*

$$g : \mathbb{R}^D \to \mathbb{R},$$

which takes a vector of log-domain-losses and returns a scalar performance metric to be maximized (accuracy, reward, a composite benchmark score, or the negative of a cost). For any allocation C we define the true and predicted downstream objectives

$$P(C) \;=\; g(\log L(C)), \qquad \widehat{P}(C) \;=\; g(\log \widehat{L}(C)).$$

The stage-2 fitting problem (learning g from cheap supervision) is deliberately separated from stage 1 and treated as exogenous here: we assume g is given, either from prior knowledge or from an auxiliary training procedure that does not consume converged runs. Our concern is how stage-1 error propagates through the composition $g \circ \log \widehat{L}$.

To that end, we evaluate downstream prediction by

$$\mathcal{E}_P(\widehat{L}) \;=\; \sup_{C \in \Delta(C_{\mathrm{tot}})} \big|\widehat{P}(C) - P(C)\big|.$$

A sufficient regularity condition ensuring that small loss-prediction error implies small downstream error is Lipschitz continuity of g on the relevant range. Specifically, if g is L_g-Lipschitz with respect to $\|\cdot\|_2$ on a set containing $\{\log L(C) : C \in \Delta(C_{\mathrm{tot}})\}$, then

$$\mathcal{E}_P(\widehat{L}) \;\le\; L_g \cdot \sup_{C \in \Delta(C_{\mathrm{tot}})} \big\|\log \widehat{L}(C) - \log L(C)\big\|_2 \;\le\; L_g\sqrt{D} \cdot \mathcal{E}_\infty(\widehat{L}).$$

Thus, uniform control in stage 1 is a direct enabler of end-to-end uniform control after composition. In particular, the multi-domain intermediate is not an aesthetic choice: it is the object to which we will apply a stable map g, and the above inequality makes explicit how prediction fidelity at the intermediate level governs fidelity at the downstream level.

Finally, we formalize the *mixture-optimization* task. Given a stage-1 predictor $\widehat{L}$ and a loss-to-performance map g, we recommend an allocation

$$\widehat{C} \;\in\; \arg\max_{C \in \Delta(C_{\mathrm{tot}})} \widehat{P}(C) \;=\; \arg\max_{C \in \Delta(C_{\mathrm{tot}})} g(\log \widehat{L}(C)).$$

Equivalently, one may define $\widehat{C}$ by minimizing a downstream loss or a weighted combination of predicted domain losses; the distinction is not material for the formulation. What is material is that $\widehat{C}$ is chosen by optimizing a *model-based* objective over a high-dimensional simplex. Consequently, we will seek stage-1 predictors that are not only accurate but also well-behaved under optimization: they should be defined on the full simplex, respect positivity of losses, and (when appropriate) admit monotonicity constraints expressing that increasing compute on a source should not systematically degrade predicted loss in the regime under study.

Summarizing, our inputs are (K, D, C_{tot}), access to $\mathcal{O}$ for at most n calls, and (for mixture recommendation) a map g. Our outputs are a predictor $\widehat{L}$ and optionally an allocation $\widehat{C}$. The primary performance target is a prescribed uniform accuracy level $\mathcal{E}_\infty(\widehat{L}) \le \varepsilon$ with probability at least $1-\delta$, and the secondary target is small downstream error $\mathcal{E}_P(\widehat{L})$ and near-optimality of the recommended mixture with respect to P. The remaining question is which model class for $C \mapsto \log L(C)$ permits such guarantees with n far smaller than what would be required to fit D independent high-dimensional regressions; we address this by imposing shared structure across domains in the next section.

4 Model class: low-rank mixing laws

We now specify the structural assumption that makes stage-1 prediction feasible with substantially fewer than KD effective degrees of freedom. The guiding principle is that, while each validation domain d may respond differently to shifts in the pretraining mixture, these responses are not arbitrary: domains share latent "capability" directions, and each data source contributes to these directions with a source-specific profile. Mathematically, this leads to a log-linear model whose domain-by-source coefficient matrix is low rank.

4.1 Log-linear mixing in log-allocation coordinates

For an allocation $C \in \Delta(C_{\mathrm{tot}})$, we posit that the converged validation loss on domain d satisfies

$$\log L_d(C) \;=\; a_d + \theta_d^\top x(C) + \varepsilon_d, \qquad x(C) = \log(C) \in \mathbb{R}^K, \tag{1}$$

where $a_d \in \mathbb{R}$ is a domain intercept, $\theta_d \in \mathbb{R}^K$ is a vector of source sensitivities for domain d, and ε_d captures the residual that remains after fitting the parametric trend. As described in the problem formulation, $x(C)$ is bounded by enforcing $C_k \ge C_{\min}$ (or a soft floor), and $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_D)$ is treated as conditionally mean-zero with sub-Gaussian coordinates.

The interpretation of (1) is an *elasticity model.* Formally, in the noiseless limit, the partial derivative

$$\frac{\partial}{\partial \log C_k} \log L_d(C) \;=\; \theta_{d,k}$$

is constant: increasing compute on source k by a multiplicative factor α decreases $\log L_d$ by $\theta_{d,k} \log \alpha$ (and thus scales L_d by a factor $\alpha^{\theta_{d,k}}$). In regimes where loss behaves approximately as a power law in effective data/compute, this is the natural local approximation.

Because we impose a fixed total compute constraint, there is an equivalent parameterization in terms of mixture proportions. Writing $p_k = C_k/C_{\text{tot}}$ so that $p \in \Delta(1)$ and $\log C_k = \log p_k + \log C_{\text{tot}}$, we may rewrite

$$\log L_d(C) \;=\; \underbrace{(a_d + \log C_{\text{tot}} \cdot \mathbf{1}^\top \theta_d)}_{=:a'_d} \;+\; \theta_d^\top \log p \;+\; \varepsilon_d.$$

Thus, at fixed C_{tot}, modeling in $\log C$ or in $\log p$ differs only by an intercept shift. We use $\log C$ to keep notation aligned with settings where C_{tot} may vary, but the identifiability of slopes is driven by variation in proportions.

4.2 Low-rank shared structure across domains

The decisive structural assumption is that the matrix of coefficients

$$\Theta \;=\; \begin{bmatrix} \theta_1^\top \\ \vdots \\ \theta_D^\top \end{bmatrix} \in \mathbb{R}^{D\times K}$$

has rank at most $r \ll \min\{D, K\}$. Equivalently, there exist factors $U \in \mathbb{R}^{D\times r}$ and $V \in \mathbb{R}^{r\times K}$ such that

$$\Theta \;=\; UV, \qquad \text{and hence} \qquad \log L(C) \;=\; a + U\big(Vx(C)\big) + \varepsilon, \tag{2}$$

with $a = (a_1, \dots, a_D)$. In this form, the map $x(C) \mapsto Vx(C) \in \mathbb{R}^r$ computes r latent "capability coordinates" induced by the allocation; the domain-specific map $u_d^\top(\cdot)$ (row $u_d^\top$ of U) then converts these coordinates into a log-loss prediction for domain d.

This low-rank hypothesis is not merely a convenience for analysis; it encodes the claim that domains co-vary under mixture shifts. Concretely, if one domain benefits from increasing code relative to web, we expect other programming-adjacent domains to move in a correlated way. The rank bound asserts that such correlated movements are generated by a small number of latent directions, rather than requiring a distinct source-sensitivity vector θ_d for each domain.

We emphasize that (2) is *not* an identifiability statement about the factors themselves: the decomposition $\Theta = UV$ is non-unique since $(UQ)(Q^{-1}V) = UV$ for any invertible $Q \in \mathbb{R}^{r\times r}$. Our estimation target is the product Θ, which is identifiable from linear measurements under suitable design conditions; interpretability of U and V requires additional conventions (e.g., orthogonality, nonnegativity, sparsity, or a rotation chosen post hoc). The nuclear-norm approach adopted later treats Θ as the primitive object and avoids committing to a particular factorization during estimation.

4.3 Monotonicity and sign structure (optional)

In many pretraining regimes it is reasonable to impose that additional compute on any single source should not worsen loss on any domain, at least locally and after controlling for other effects. In the log-linear model this corresponds to

$$\theta_{d,k} \;\leq\; 0 \qquad \text{for all } d \in [D],\; k \in [K]. \tag{3}$$

We treat (3) as optional: it can be violated if sources are heterogeneous and interact (e.g., adding a low-quality source could degrade performance if it displaces higher-quality data under a hard budget), or if the model is asked to extrapolate beyond the regime where the fitted linear trend is valid. Nevertheless, when monotonicity is believed to hold approximately, it can be incorporated as a convex constraint in the nuclear-norm regression without altering the basic sample-complexity scaling, and it can substantially stabilize mixture optimization by preventing spurious "anti-learning" directions.

A weaker and often more realistic structure is that monotonicity holds only after projecting onto latent factors, i.e., that V has mixed signs but the composite Θ satisfies (3) on average across relevant domains. Our theory does not require either variant, but our estimator can accommodate coordinatewise constraints when desired.

4.4 Extensions: total-compute scaling and enriched feature maps

The preceding discussion fixes C_{tot} and models only how *mixing* affects outcomes. In practice one may also wish to compare different total compute budgets. A minimal extension adds a separate total-compute term,

$$\log L_d(C) \;=\; a_d + b_d \log C_{\text{tot}} + \theta_d^\top \log p + \varepsilon_d, \qquad p = C/C_{\text{tot}}. \tag{4}$$

If empirical evidence suggests that all domains share a common compute-scaling exponent, one may further restrict $b_d \equiv b$; if not, the vector $b \in \mathbb{R}^D$ introduces only D additional parameters and does not change the essential burden relative to estimating Θ. The analysis in later sections extends to (4) by augmenting the design matrix with an additional column $\log C_{\text{tot}}$ and treating (b, Θ) as a concatenated coefficient matrix, with low-rank structure applied only to the Θ block.

A different extension concerns *phase changes* in scaling. Empirically, loss curves can exhibit regime shifts (e.g., data-limited to compute-limited) or varying marginal returns as compute increases. Within our framework, such behavior can be captured by replacing the linear map $x(C) \mapsto \Theta x(C)$ with a piecewise-linear or basis-expanded model that remains low rank after feature expansion. For example, fix knots $\tau_{k,1} < \tau_{k,2} < \cdots$ in log-compute and define hinge features

$$\phi_{k,j}(C) \;=\; \big(\log C_k - \tau_{k,j}\big)_+,$$

then model

$$\log L_d(C) \;=\; a_d + \sum_{k=1}^{K} \theta_{d,k} \log C_k + \sum_{k=1}^{K} \sum_{j} \gamma_{d,k,j}\, \phi_{k,j}(C) + \varepsilon_d,$$

with a low-rank assumption on the enlarged coefficient array (θ, γ) when reshaped appropriately. This increases the effective feature dimension from K to $K(1+J)$, and correspondingly increases the number of runs needed for uniform prediction, but it preserves the core mechanism: shared structure across domains is exploited through low rank rather than fitting each domain independently. We will keep the base model (1) as our principal object, noting that these enrichments are available when the linear approximation is insufficient.

4.5 Interpretability via latent capability factors

Although our estimation target is Θ, the low-rank hypothesis implicitly posits a small set of latent factors. When we choose a particular factorization (e.g., via a truncated SVD of $\widehat{\Theta}$), each row of V can be read as a *source profile* describing how that latent factor is synthesized from log-compute across sources, and each row of U becomes a *domain loading* describing which factors matter for that domain. In favorable cases these factors align with semantically meaningful axes (e.g., "code reasoning," "mathematical formalism," "multilinguality"), and the model gives a compact account of why domains move together under mixture reallocation.

We stress, however, that such interpretations require care: (i) the factors are only identified up to rotations unless we impose additional structure; (ii) the meaning of a factor can drift with the chosen feature map and compute floor; and (iii) the model is descriptive at the level of losses and does not, by itself, establish causality about data composition. For our purposes the central benefit of the low-rank view is operational: it reduces the statistical burden of learning $C \mapsto \log L(C)$ uniformly over a simplex. Having fixed the model class, we next address the design question: how to choose a small number of allocations so that the induced log-space design matrix is sufficiently informative to recover Θ and to support uniform prediction over all feasible mixtures.

5 Experiment design for identifiability

The statistical burden in stage 1 is governed less by the optimization method than by the *informativeness* of the queried allocations. Writing the responses in matrix form,

$$Y \ = \ \mathbf{1}a^{\top} + X\Theta^{\top} + E, \tag{5}$$

we see that all estimators of Θ (convex or not) ultimately rely on the geometry of the design matrix X whose ith row is $x(C^{(i)})^{\top} = \log(C^{(i)})^{\top}$. Accordingly, our design goal is to choose allocations $\{C^{(i)}\}_{i=1}^{n} \subset \Delta(C_{\text{tot}})$ so that X satisfies a restricted invertibility property on low-rank directions, while remaining practical under pretraining constraints (e.g. hard floors, limited ability to realize extreme mixtures, and numerical stability of log-features).

5.1 Design objectives and the role of log-space

Two features distinguish our setting from standard linear regression. First, we do not observe scalar labels but D domain losses per run; this is favorable, because each oracle call yields D independent (or weakly dependent) measurements and thus effectively multiplies sample size by D in concentration bounds. Second, the covariates $x(C) = \log C$ are restricted to a curved manifold induced by the simplex constraint $\sum_k C_k = C_{\text{tot}}$. The latter implies that "orthogonal" designs in Euclidean space are not directly available, and that naive grids over the simplex can produce highly collinear columns once mapped through $\log(\cdot)$.

A useful guiding principle is that the identifiability of Θ is driven by *relative* shifts among sources. In particular, allocations that are nearly uniform in C tend to cluster in log-space (small variation in each $\log C_k$), leading to small effective signal-to-noise for Θ. Conversely, designs that explore a range of proportions (including moderately imbalanced mixtures) create larger variation in $\log C_k$ and can substantially improve conditioning—subject to not pushing any C_k so close to zero that the model becomes numerically or semantically unstable.

5.2 Sparse allocation families

We advocate a family of *sparse* allocations, in which each run concentrates most compute on a small support of sources while holding all other sources at a fixed floor. Concretely, fix a floor $C_{\min} > 0$ (or an equivalent soft floor used only inside the log map), and choose an integer sparsity level $s \ll K$. For each run i:

1. Sample a support set $S^{(i)} \subset [K]$ with $|S^{(i)}| = s$ (e.g. uniformly without replacement, or via weighted sampling to emphasize important sources).
2. Allocate a residual budget $C_{\text{res}} := C_{\text{tot}} - (K - s)C_{\min}$ over $S^{(i)}$ by drawing proportions $p^{(i)} \in \Delta(1)$ from a well-spread distribution (e.g.

Dirichlet($\alpha\mathbf{1}$) with $\alpha \in [0.5, 5]$), and set

$$C_k^{(i)} = \begin{cases} C_{\min} + C_{\text{res}}\, p_k^{(i)} & k \in S^{(i)}, \\ C_{\min} & k \notin S^{(i)}. \end{cases}$$

The resulting log-features have two desirable properties. First, within the active support, $\log C_k$ varies meaningfully across runs, enabling estimation of source sensitivities. Second, outside the support, features are pinned at $\log C_{\min}$, which reduces variance contributed by rarely used sources and makes each run "target" a small subset of coordinates.

The choice $s \approx O(r)$ is natural: since the coefficient matrix has rank r, each response direction lies in an r-dimensional latent subspace, so it is plausible that $O(r)$ active coordinates per run suffice to excite these directions while keeping designs diverse across runs. More conservative choices such as $s \approx O(r \log K)$ can improve coverage and conditioning at the cost of less extreme allocations.

Sparse designs also admit a practical interpretation: in many training pipelines, it is easier (and safer) to create mixtures that emphasize a few sources than to finely tune small fractions across dozens of sources. The floor $C_{\min}$ represents the engineering requirement that every source either be absent or appear at a minimum viable sampling rate, avoiding degenerate data loader behavior and preventing $\log C_k$ from diverging.

5.3 Conditioning and restricted strong convexity in log-space

To obtain uniform prediction guarantees from nuclear-norm regression, we require that X not annihilate low-rank directions in Θ. A convenient sufficient condition is a restricted strong convexity (RSC) inequality of the form

$$\frac{1}{n}\|XM^\top\|_F^2 \;\geq\; \kappa\|M\|_F^2 \qquad \text{for all } M \in \mathbb{R}^{D\times K} \text{ with } \operatorname{rank}(M) \leq 2r, \tag{6}$$

for some $\kappa > 0$ depending on the design distribution and the boundedness of $x(C)$. At an intuitive level, (6) asserts that distinct low-rank coefficient matrices induce distinct mean responses on the queried allocations.

Sparse randomized designs promote (6) for two reasons. First, random supports diversify which coordinates of $x(C)$ vary across runs, reducing the chance that any column (source) is nearly constant across the design. Second, drawing proportions from a Dirichlet distribution yields continuous variability *within* each support, making the conditional covariance of $\log C$ on that support non-degenerate. Although the rows of X are not isotropic sub-Gaussian vectors (because of the simplex constraint and the log transform), one can still obtain RSC-type statements by bounding the row norms $\|x(C^{(i)})\|_2$ (via $C_{\min}$) and by ensuring that the empirical Gram matrix $(1/n)X^\top X$ has no extremely small eigenvalues on the span of coordinates that are activated with nontrivial probability.

There is one subtlety specific to log-space: if $C_{\min}$ is excessively small, then $\log C_{\min}$ is very negative, and columns corresponding to "inactive" sources contribute large-magnitude constants to $x(C)$. This does not, by itself, prevent identifiability (constants can be absorbed into intercept-like terms), but it can degrade numerical conditioning by inflating $\|x(C)\|_2$ and thus the effective noise scale in the regression. Two standard remedies are: (i) choose $C_{\min}$ to reflect a realistic minimum sampling rate rather than an arbitrarily tiny number; and/or (ii) *center* the log-features across runs by replacing X with $X(I - \frac{1}{K}\mathbf{1}\mathbf{1}^\top)$ (or, equivalently, include an explicit column for $\sum_k \log C_k$ and treat it separately), thereby removing large common offsets. Either approach preserves the estimand Θ up to a harmless reparameterization of intercept terms and typically improves the conditioning of the optimization problem.

5.4 A practical recipe for choosing runs

We summarize an implementation-oriented recipe that we have found to align with the preceding principles.

1. **Select a floor.** Choose $C_{\min}$ so that each source at the floor contributes a non-negligible number of samples per epoch (or per training window), and ensure $C_{\text{tot}} > (K - s)C_{\min}$.

2. **Pick a sparsity level.** Set $s \in \{r, 2r, 4r\}$ as a starting point; increase s if certain sources must co-occur for data pipeline reasons, or if preliminary designs yield unstable estimates.

3. **Ensure coverage.** Generate supports so that each source appears in at least m runs (e.g. $m \gtrsim c \log K$), either by rejection sampling or by deterministic balancing (block designs).

4. **Diversify proportions.** Within each support, draw p from Dirichlet$(\alpha\mathbf{1})$ with α near 1 for broad coverage; use smaller α to create more imbalanced mixtures if the model remains valid in that regime.

5. **Check conditioning.** After forming X, compute simple diagnostics such as column variances, pairwise correlations, and the smallest singular value of a centered/normalized version of X. If diagnostics are poor, resample supports/proportions.

6. **Freeze the design (non-adaptive).** In the regime where converged runs are expensive, we prefer committing to a non-adaptive randomized design that meets conditioning criteria; adaptivity can be layered on later if additional runs become available.

This recipe is intentionally conservative: its purpose is to ensure that the subsequent estimator is not asked to extrapolate far beyond the convex hull of observed log-allocations, which is precisely the failure mode that leads to unstable mixture recommendations.

5.5 Robustness to near-zero allocations

The low-rank log-linear model is only as credible as the regime in which it is fit. Allocations with C_k extremely close to zero pose three distinct issues: (i) $\log C_k$ becomes large in magnitude, creating high leverage points; (ii) the semantics of "including" a source at vanishing rate can differ from excluding it (tokenization artifacts, curriculum schedules, and mixing implementation details); and (iii) the marginal effect of adding a tiny amount of a source may be nonlinear, contradicting constant elasticity.

For these reasons, we treat robustness to near-zero allocations primarily as a *design* constraint rather than an estimation trick. Enforcing a hard $C_{\min}$ keeps the feature domain bounded, yielding uniform bounds on $\|x(C)\|_2$ that propagate directly into uniform prediction guarantees. When an engineering stack requires the possibility of true zeros (sources absent), a practical compromise is to use a soft floor only in the feature map, e.g. $x_k(C) = \log(C_k + C_{\text{floor}})$ with $C_{\text{floor}} \ll C_{\text{tot}}$, while still implementing $C_k = 0$ in training. This sacrifices exact parametric correctness but often improves stability and maintains approximate identifiability.

Having specified a design family that yields an informative and numerically stable X, we now turn to the estimation procedure. In the next section we fit (a, Θ) via nuclear-norm regularized multi-response regression, and we discuss how to select the regularization level (or effective rank) and incorporate optional convex constraints such as monotonicity.

6 Estimation via nuclear-norm regularized multi-response regression

Given an experiment design $\{C^{(i)}\}_{i=1}^{n}$ and corresponding oracle responses $y^{(i)} = \log L(C^{(i)}) + \xi^{(i)} \in \mathbb{R}^D$, we form the design matrix $X \in \mathbb{R}^{n\times K}$ with rows $x^{(i)\top} = \log(C^{(i)})^\top$ (with an explicit floor inside the logarithm) and the response matrix $Y \in \mathbb{R}^{n\times D}$ with rows $y^{(i)\top}$. Estimation in stage 1 amounts to fitting the affine map $x \mapsto a + \Theta x$ under the structural prior $\text{rank}(\Theta) \le r$. Since r is typically unknown and the noise is non-negligible, we prefer a convex surrogate that (i) encourages low rank, (ii) couples information across the D responses, and (iii) is stable under modest misspecification.

6.1 Convex formulation

We estimate (a, Θ) by nuclear-norm regularized least squares:

$$(\widehat{a}, \widehat{\Theta}) \in \arg \min_{a \in \mathbb{R}^D,\ \Theta \in \mathbb{R}^{D \times K}} \frac{1}{2n} \left\| Y - \mathbf{1} a^\top - X \Theta^\top \right\|_F^2 + \lambda \|\Theta\|_*, \tag{7}$$

where $\|\Theta\|_* = \sum_j \sigma_j(\Theta)$ is the nuclear norm and $\lambda > 0$ is a regularization level. The fitted predictor is then, for any allocation $C \in \Delta(C_{\text{tot}})$,

$$\log \widehat{L}(C) = \widehat{a} + \widehat{\Theta} \log(C), \qquad \widehat{L}(C) = \exp\bigl(\log \widehat{L}(C)\bigr),$$

with exponentiation performed elementwise. We emphasize that downstream optimization and evaluation are numerically more stable in log-space; in particular, we recommend storing and composing $\log \widehat{L}(C)$ rather than $\widehat{L}(C)$ whenever possible.

The objective (7) is jointly convex and directly exploits the multi-response structure: the same covariates X explain all domains, while the nuclear norm couples domains by favoring a shared low-dimensional latent factorization. In contrast, fitting each domain separately by an ℓ_2 or ℓ_1 penalty ignores the assumed rank structure and typically requires more runs for comparable uniform accuracy.

6.2 Centering, intercept handling, and feature scaling

Although (7) includes an explicit intercept a, practical performance improves if we eliminate avoidable numerical ill-conditioning in X. Two issues arise in log-space: (i) a large common offset (e.g. many coordinates pinned at $\log C_{\min}$) can inflate $\|x(C)\|_2$ without adding identifying variation; and (ii) columns of X may have markedly different variances due to unequal activation frequencies under sparse designs.

A simple remedy is to center and optionally standardize the features before fitting. Let $\bar{x} = (1/n) \sum_{i=1}^n x^{(i)}$ and define the centered design $\widetilde{X}$ with rows $\widetilde{x}^{(i)} = x^{(i)} - \bar{x}$. Writing (5) in terms of $\widetilde{X}$ yields the same model with a reparameterized intercept, and the estimator can be computed by first fitting Θ on centered covariates and then recovering $\widehat{a}$ as the mean residual:

$$\widehat{a} = \frac{1}{n} \mathbf{1}^\top \bigl(Y - \widetilde{X}\, \widehat{\Theta}^\top\bigr). \tag{8}$$

If some sources are rarely activated, we additionally scale columns by empirical standard deviations, i.e. use $\widetilde{X} S^{-1}$ with $S = \operatorname{diag}(s_1, \dots, s_K)$ and $s_k^2 = (1/n) \sum_i \widetilde{X}_{ik}^2$ (with a small lower cap to avoid division by very small numbers). This scaling changes the implicit prior induced by $\|\Theta\|_*$, so we apply the inverse transformation to report coefficients in the original coordinates.

6.3 Optimization algorithms

Problem (7) is a standard composite minimization problem: a smooth quadratic loss plus a non-smooth convex regularizer. We solve it by proximal gradient methods (e.g. FISTA). Denote the smooth part by

$$f(a, \Theta) = \frac{1}{2n} \left\| Y - \mathbf{1}a^\top - X\Theta^\top \right\|_F^2.$$

Given $(a^{(t)}, \Theta^{(t)})$, we take a gradient step on f and apply the proximal operator of $\lambda\| \cdot \|_*$ to Θ. The proximal step is singular value thresholding: if G is the gradient-updated matrix and $G = U\Sigma V^\top$ is an SVD, then

$$\mathrm{prox}_{\eta\lambda\|\cdot\|_*}(G) = U(\Sigma - \eta\lambda I)_+ V^\top,$$

where $(\cdot)_+$ thresholds singular values at zero. The intercept update is cheap and can be done either by an explicit gradient step or by the closed form (8) at each iteration (or every few iterations) when using centered features. When D and K are large, we compute only a truncated SVD (to a rank slightly larger than the effective rank), which is sufficient because singular values below the threshold do not contribute to the prox output.

In regimes where we have a credible bound r and want a smaller memory footprint, a non-convex factorization $\Theta = UV$ with $U \in \mathbb{R}^{D\times r}$, $V \in \mathbb{R}^{r\times K}$ and a ridge penalty $(\gamma/2)(\|U\|_F^2 + \|V\|_F^2)$ is also viable; it corresponds to a variational form of the nuclear norm. We treat this as an implementation option rather than our primary estimator, since (7) provides a clearer interface to theory and hyperparameter selection.

6.4 Choosing λ or an effective rank

We require a principled method for selecting λ (or, equivalently, selecting an effective rank through the spectrum of $\widehat{\Theta}$). A purely theoretical choice $\lambda \asymp \sigma B\sqrt{(D + K)/n}$ is useful as a default, but in practice the noise level and conditioning constants are only approximately known. We therefore recommend a small grid search over λ coupled with cross-validation, subject to the constraint that n is small and each run is expensive.

We partition the n runs into K_{fold} folds (typically $K_{\text{fold}} \in \{3, 5\}$), fit (7) on $K_{\text{fold}} - 1$ folds, and evaluate prediction error on the held-out fold using a domain-weighted metric in log-space, e.g.

$$\mathrm{CV}(\lambda) = \frac{1}{|\mathcal{I}_{\text{val}}|} \sum_{i\in\mathcal{I}_{\text{val}}} \sum_{d=1}^{D} w_d \Big(\widehat{a}_d(\lambda) + \widehat{\theta}_d(\lambda)^\top x^{(i)} - Y_{id}\Big)^2, \qquad w_d \geq 0, \ \sum_d w_d = 1.$$

Weights w_d may reflect downstream importance or measurement reliability. To maintain comparability across folds under sparse designs, we stratify folds so that each source appears with similar frequency in training and validation subsets. Having selected $\widehat{\lambda}$, we refit on all runs to obtain the final $(\widehat{a}, \widehat{\Theta})$.

If we wish to target a hard rank r rather than a penalty level, we may choose λ by cross-validation and then define $\widehat{r}$ by thresholding the singular values of $\widehat{\Theta}$ at a noise-dependent level, or simply by taking the smallest r explaining a desired fraction of nuclear norm. This rank extraction is used only for interpretability and for accelerating subsequent computations; the predictor remains $\log \widehat{L}(C) = \widehat{a} + \widehat{\Theta} \log C$.

6.5 Convex constraints and numerical safeguards

The log-linear parameterization implies that increasing C_k (holding other coordinates fixed) should not increase loss; in our model this corresponds to $\Theta_{d,k} \leq 0$ for all (d, k). Because the simplex constraint couples coordinates, monotonicity is not an absolute physical law for finite-budget reallocations; nonetheless, the sign constraint is a useful inductive bias and often prevents pathological extrapolations when optimizing mixtures. We incorporate it by solving the constrained variant

$$\min_{a,\Theta} \frac{1}{2n} \left\| Y - \mathbf{1}a^\top - X\Theta^\top \right\|_F^2 + \lambda \|\Theta\|_* \quad \text{s.t.} \quad \Theta \leq 0 \text{ (entrywise)}. \tag{9}$$

This remains convex. Algorithmically, we use a projected proximal method: after the singular value thresholding step, we project onto the nonpositive orthant by $(\Theta)_{d,k} \leftarrow \min\{\Theta_{d,k}, 0\}$. While the nuclear-norm proximal map and orthant projection do not commute, alternating them yields a convergent scheme for the sum of convex functions via standard splitting methods (e.g. ADMM), and in practice a small number of inner iterations suffices.

Two further safeguards improve robustness. First, we clip the feature map by enforcing a floor $C_k \geq C_{\min}$ inside $\log(\cdot)$, even if the training pipeline can implement $C_k = 0$; this prevents extreme leverage. Second, we bound prediction ranges by working in log-loss space and, if necessary, clipping $\log \widehat{L}_d(C)$ to a conservative interval determined from observed validation losses. This does not affect the fitted coefficients but avoids numerical overflow when exponentiating and prevents downstream mixture optimization from being driven by implausible extrapolations.

The estimator (7) (or (9)) completes stage 1: it produces a low-rank, multi-domain predictor that can be queried cheaply at arbitrary allocations $C \in \Delta(C_{\text{tot}})$. We now analyze its estimation and uniform prediction error under the design and noise assumptions.

7 Theory I (Upper bounds): estimation and uniform prediction over mixtures

We analyze the estimator $\widehat{\Theta}$ produced by (7) (or its monotone variant (9)) under the log-linear mixing model

$$Y = \mathbf{1}a^\top + X\Theta^\top + E,$$

where $E \in \mathbb{R}^{n\times D}$ has independent, mean-zero, σ^2-sub-Gaussian entries (conditional on X). Since our downstream use-case is to query the fitted map at arbitrary allocations $C \in \Delta(C_{\text{tot}})$, the relevant target is a *uniform* bound on the prediction error $\log \widehat{L}(C) - \log L(C)$, not merely an average error on the training allocations.

7.1 Design regularity and bounded log-features

Uniform control over mixtures requires that $\log(C)$ remain bounded on the feasible set. We therefore enforce a floor $C_k \geq C_{\min} > 0$ inside the logarithm and define the (deterministic) feature bound

$$B \;:=\; \sup_{C\in\Delta(C_{\text{tot}})} \|\log(C)\|_2 \;\leq\; \sqrt{K}\,\max\{|\log C_{\min}|,\;|\log C_{\text{tot}}|\}. \tag{10}$$

The bound (10) is crude but sufficient: our theory degrades only logarithmically in the ratio $C_{\text{tot}}/C_{\min}$.

We also require a restricted strong convexity (RSC) property of the quadratic loss along low-rank directions. Writing $\Delta = \widehat{\Theta} - \Theta$, define the prediction seminorm

$$\|\Delta\|_{\text{pred}}^2 \;:=\; \frac{1}{n}\|X\Delta^\top\|_F^2.$$

We assume that there exists $\kappa > 0$ such that

$$\frac{1}{n}\|X\Delta^\top\|_F^2 \;\geq\; \kappa\|\Delta\|_F^2 \qquad \text{for all } \Delta \in \mathcal{C}_{2r}, \tag{11}$$

where $\mathcal{C}_{2r}$ is the usual cone of matrices whose components orthogonal to the tangent space at Θ are dominated by the tangent component (equivalently, the cone arising from nuclear-norm decomposability and rank-r structure). For the sparse randomized designs described earlier, (11) holds with high probability once n is large enough that the resulting X is well-conditioned on $O(r)$-sparse supports; in particular, κ can be bounded away from 0 up to logarithmic factors under mild incoherence conditions.

7.2 Estimation error in Frobenius and operator norms

We now state an upper bound that makes explicit the dependence on rank, dimensions, noise, and design conditioning. For notational simplicity we present the result for centered covariates (so that the intercept is updated as in (8)); the same rates hold with an explicit intercept in the optimization.

Theorem 7.1 (Upper bound for nuclear-norm regression)**.** *Assume* $\operatorname{rank}(\Theta) \le r$*, the entries of* E *are independent* σ^2*-sub-Gaussian,* $\|\log(C^{(i)})\|_2 \le B$ *for all* i*, and the design matrix satisfies the RSC condition* (11) *with parameter* $\kappa > 0$*. Choose the regularization level*

$$\lambda \;\ge\; c\,\sigma B\sqrt{\frac{D+K+\log(1/\delta)}{nD}}, \tag{12}$$

for a universal constant $c > 0$ *(the normalization reflects that we have* nD *scalar loss measurements across runs and domains). Then with probability at least* $1-\delta$*,*

$$\|\widehat{\Theta}-\Theta\|_F \;\le\; \frac{C}{\kappa}\,\sigma B\sqrt{\frac{r(D+K)+r\log(1/\delta)}{nD}}, \tag{13}$$

and hence, since $\operatorname{rank}(\widehat{\Theta}-\Theta) \le 2r$*,*

$$\|\widehat{\Theta}-\Theta\|_{\mathrm{op}} \;\le\; \|\widehat{\Theta}-\Theta\|_F \;\le\; \frac{C}{\kappa}\,\sigma B\sqrt{\frac{r(D+K)+r\log(1/\delta)}{nD}}. \tag{14}$$

Moreover the intercept estimate $\widehat{a}$ *obtained by* (8) *satisfies*

$$\|\widehat{a}-a\|_2 \;\le\; C\,\sigma\sqrt{\frac{D+\log(1/\delta)}{n}} \;+\; \frac{C}{\sqrt{n}}\|X(\widehat{\Theta}-\Theta)^\top\|_F. \tag{15}$$

We sketch the proof, emphasizing the steps that later feed into uniform mixture control. First, we write the *basic inequality* comparing the objective at $(\widehat{a},\widehat{\Theta})$ and (a,Θ):

$$\frac{1}{2n}\|Y-\mathbf{1}\widehat{a}^\top - X\widehat{\Theta}^\top\|_F^2 + \lambda\|\widehat{\Theta}\|_* \;\le\; \frac{1}{2n}\|Y-\mathbf{1}a^\top - X\Theta^\top\|_F^2 + \lambda\|\Theta\|_*.$$

Expanding $Y = \mathbf{1}a^\top + X\Theta^\top + E$ and rearranging yields a bound on the prediction error $\|X(\widehat{\Theta}-\Theta)^\top\|_F^2$ in terms of the stochastic term $\langle X^\top E, \widehat{\Theta}-\Theta\rangle$ and the nuclear-norm difference. Second, we control the stochastic term by duality:

$$\frac{1}{n}\langle X^\top E, \Delta\rangle \;\le\; \left\|\frac{1}{n}X^\top E\right\|_{\mathrm{op}} \|\Delta\|_*,$$

and we choose λ to dominate $\|(1/n)X^\top E\|_{\mathrm{op}}$ with probability $1-\delta$. The bound (12) follows from standard matrix concentration for sums of sub-Gaussian rank-one matrices (each run contributes $x^{(i)}e_i^\top$), together with $\|x^{(i)}\|_2 \le B$ and the normalization by nD scalar measurements.

Third, nuclear-norm decomposability and $\operatorname{rank}(\Theta) \le r$ imply that Δ lies in the cone $\mathcal{C}_{2r}$ and that $\|\Delta\|_* \le 4\sqrt{2r}\|\Delta\|_F$. Plugging these inequalities into the basic inequality yields an upper bound on the prediction error of the form

$$\frac{1}{n}\|X\Delta^\top\|_F^2 \;\le\; C\lambda\sqrt{r}\,\|\Delta\|_F.$$

Finally, RSC (11) converts prediction error to parameter error, giving $\kappa\|\Delta\|_F^2 \le C\lambda\sqrt{r}\|\Delta\|_F$ and hence (13). The operator-norm bound (14) follows from $\|\Delta\|_{\mathrm{op}} \le \|\Delta\|_F$ (or, slightly more sharply, $\|\Delta\|_{\mathrm{op}} \le \|\Delta\|_F/\sqrt{2r}$ when one tracks rank).

7.3 Uniform-in-mixture prediction bounds over the simplex

We now translate parameter error into a uniform prediction bound over $C \in \Delta(C_{\mathrm{tot}})$. For any such C,

$$\log \widehat{L}(C) - \log L(C) = (\widehat{a} - a) + (\widehat{\Theta} - \Theta)\log(C)$$

(up to the noise ε in the generative model, which is already absorbed into the oracle measurements). Using $\|v\|_\infty \le \|v\|_2$ and $\|(\widehat{\Theta} - \Theta)\log(C)\|_2 \le \|\widehat{\Theta} - \Theta\|_{\mathrm{op}}\|\log(C)\|_2$, we obtain

$$\sup_{C\in\Delta(C_{\mathrm{tot}})} \|\log \widehat{L}(C) - \log L(C)\|_\infty \;\le\; \|\widehat{a} - a\|_2 \;+\; B\,\|\widehat{\Theta} - \Theta\|_{\mathrm{op}}. \tag{16}$$

Combining (16) with Theorem 7.1 yields the desired scaling: up to conditioning constants and logarithmic factors,

$$\sup_{C\in\Delta(C_{\mathrm{tot}})} \|\log \widehat{L}(C) - \log L(C)\|_\infty \;\lesssim\; \sigma\sqrt{\frac{D}{n}} \;+\; \frac{\sigma B^2}{\kappa}\sqrt{\frac{r(D+K)}{nD}}.$$

In the regime where the intercept is well-estimated (e.g. after centering, or when n is moderately large), the dominant term is the second, which exhibits the expected *low-rank* dependence $\sqrt{r(D+K)}$ and the *per-run multi-domain* benefit of D scalar measurements. In particular, to achieve a target uniform error ε it suffices (up to log factors) to take

$$n \;=\; \widetilde{O}\!\left(\frac{r(D+K) + \log(1/\delta)}{D\,\varepsilon^2}\right), \tag{17}$$

which matches the heuristic "nD measurements for $r(D+K)$ degrees of freedom" once we account for the conversion from average prediction error to a uniform bound over the bounded log-simplex.

Two remarks are in order. First, the dependence on B is unavoidable for uniform bounds: without a floor $C_{\min}$, $\log(C)$ is unbounded near the simplex boundary and no estimator can control $\sup_C$ error from finitely many samples. Second, the design enters only through κ (and through the concentration used to set λ); sparse randomized allocations are thus acceptable provided they yield an X that is sufficiently diverse in log-space. This is precisely the role of the "well-spread Dirichlet on supports" condition in our design subroutine: it prevents X from being nearly rank-deficient when restricted to the low-rank tangent cone, which would otherwise inflate the factor $1/\kappa$ in (13) and, through (16), lead to unstable mixture extrapolation.

8 Theory II (Lower bounds): minimax sample complexity and necessity of structure

We complement the upper bounds of Section 8 with information-theoretic lower bounds showing that, up to logarithmic factors (and benign conditioning constants), our run complexity is unimprovable under the stated modeling assumptions. The key point is that each converged run produces a *vector* of D domain losses, hence n runs yield nD scalar measurements; nevertheless, the unknown coefficient matrix $\Theta \in \mathbb{R}^{D\times K}$ contains on the order of $r(D+K)$ degrees of freedom under a rank-r constraint. The lower bounds formalize that, even allowing adaptive choice of allocations, one cannot beat $n = \Omega(rK/(D\varepsilon^2))$ in the worst case (for constant D), and that without low-rank (or other structural) restrictions the required number of runs scales linearly with K (and with KD if all domains are to be predicted simultaneously).

8.1 A minimax lower bound under rank-r mixing

To state a clean minimax bound, we work with a bounded feature class consistent with the floor $C_k \geq C_{\min}$ imposed for uniform prediction. Namely, we consider designs whose log-features satisfy $\|x(C)\|_2 \leq B$, where B is the deterministic bound in (10). We view an algorithm as an interactive procedure that (possibly adaptively) selects allocations $C^{(1)}, \dots, C^{(n)}$ (equivalently, feature vectors $x^{(i)} = \log(C^{(i)})$) and receives oracle responses

$$y^{(i)} \;=\; a+\Theta x^{(i)}+\xi^{(i)} \in \mathbb{R}^D, \qquad \xi^{(i)} \text{ independent, mean-zero, } \sigma^2\text{-sub-Gaussian.}$$

Since a is a nuisance parameter, we may fix $a = 0$ in the lower bound (or absorb it by centering) without weakening the conclusion.

Theorem 8.1 (Minimax run complexity under low rank)**.** *Fix $B, \sigma > 0$ and let $\mathcal{M}(r)$ denote the class of instances with* $\operatorname{rank}(\Theta) \leq r$ *and $\|\Theta\|_F \leq R$ for a fixed radius R (and arbitrary intercept). Consider any (possibly adaptive) algorithm that makes n oracle calls and outputs a predictor $\widehat{L}$, equivalently* $\log \widehat{L}(C) = \widehat{a} + \widehat{\Theta}\log(C)$*. Then there exists a universal constant $c > 0$ such that, for some instance in $\mathcal{M}(r)$, with probability at least* $1/3$,

$$\sup_{C\in\Delta(C_{\text{tot}})} \big\| \log \widehat{L}(C) - \log L(C)\big\|_\infty \;\geq\; c\,\sigma\sqrt{\frac{rK}{Dn}}.$$

In particular, achieving uniform error at most ε requires

$$n \;=\; \Omega\!\left(\frac{rK}{D\,\varepsilon^2}\right).$$

We outline the proof in the standard *packing + Fano* style, emphasizing the origin of the factor D and the dependence on rK. We construct a finite set of hypotheses $\{\Theta^{(1)}, \dots, \Theta^{(M)}\} \subset \mathcal{M}(r)$ such that (i) the matrices are well-separated in operator norm (or in their induced predictions on bounded features), and (ii) the Kullback–Leibler divergence between the transcript distributions induced by two distinct hypotheses is small when n is small. Fano's inequality then implies that no estimator can reliably identify the correct hypothesis, which in turn forces nontrivial prediction error.

For the packing, it suffices to embed an $r \times K$ sign matrix into $\mathbb{R}^{D\times K}$ by choosing an isometry $U \in \mathbb{R}^{D\times r}$ with $U^\top U = I_r$ and setting $\Theta = UV$ with $V \in \mathbb{R}^{r\times K}$. By a Varshamov–Gilbert argument, there exists a set $\{V^{(j)}\}_{j=1}^{M}$ with $M \geq \exp(c_0 rK)$ and with pairwise separation $\|V^{(j)} - V^{(\ell)}\|_F \geq c_1 \alpha\sqrt{rK}$ for a scale $\alpha > 0$ chosen below. Setting $\Theta^{(j)} = UV^{(j)}$ yields $\operatorname{rank}(\Theta^{(j)}) \leq r$ and preserves Frobenius separation. Moreover, for any feature vector x with $\|x\|_2 \leq B$,

$$\|\Theta^{(j)}x - \Theta^{(\ell)}x\|_2 \;\geq\; \|\Theta^{(j)} - \Theta^{(\ell)}\|_{\mathrm{op}} \cdot \|x\|_2 \;\gtrsim\; \alpha\sqrt{K}\,\|x\|_2,$$

after choosing the packing so that $\|\Theta^{(j)} - \Theta^{(\ell)}\|_{\mathrm{op}} \gtrsim \alpha\sqrt{K}$ (which can be arranged by normalizing $V^{(j)}$ appropriately).

For the divergence calculation, condition on the algorithm's (possibly adaptive) choice of features $x^{(1)}, \dots, x^{(n)}$. Under two hypotheses Θ and Θ', the oracle responses across runs and domains form an $n \times D$ matrix with independent sub-Gaussian noise added to the mean $X\Theta^\top$ (up to transpose conventions). In the Gaussian case (and similarly for sub-Gaussian noise by standard comparison arguments), the KL divergence between the two induced transcript distributions is bounded by

$$\mathrm{KL}(P_\Theta \,\|\, P_{\Theta'}) \;\leq\; \frac{1}{2\sigma^2}\sum_{i=1}^{n} \|\,(\Theta{-}\Theta')x^{(i)}\,\|_2^2 \;\leq\; \frac{1}{2\sigma^2}\sum_{i=1}^{n} \|\Theta{-}\Theta'\|_{\mathrm{op}}^2\,\|x^{(i)}\|_2^2 \;\leq\; \frac{nB^2}{2\sigma^2}\,\|\Theta{-}\Theta'\|_{\mathrm{op}}^2.$$

Crucially, although each run returns a D-dimensional vector, the divergence accumulates proportionally to the *total number of scalar coordinates*, effectively scaling like nD once one expresses $\|\Theta - \Theta'\|_{\mathrm{op}}^2$ in terms of average per-domain separation. Choosing the packing scale α so that $\mathrm{KL} \lesssim \log M$ when $n \ll rK/(D\alpha^2)$ yields a regime in which hypotheses are hard to distinguish. Fano's inequality then forces a constant probability of misidentification, which implies that for some pair of hypotheses the prediction error on a suitably chosen bounded feature vector x (hence on a corresponding allocation C) is at least on the order of $\sigma\sqrt{rK/(Dn)}$. Translating this into $\sup_C \|\cdot\|_\infty$ yields Theorem 8.1.

We note that adaptivity does not improve the rate: the above divergence bound already conditions on the (random, history-dependent) chosen design and hence applies to any adaptive strategy. Intuitively, adaptivity cannot manufacture more than nD noisy scalar constraints, while a rank-r matrix with K columns contains $\Theta(rK)$ effective degrees of freedom once D is treated as constant.

8.2 Impossibility without low-rank (or comparable) structure

The rank constraint is not merely a convenient sufficient condition for efficient estimation; it is necessary (up to replacement by some other strong structural prior such as sparsity) to avoid a linear blowup in the number of required runs. We formalize this by exhibiting families of instances in which the coefficients decouple across sources and/or across domains, reducing the problem to estimating many independent parameters from bounded linear measurements.

Theorem 8.2 (No structure $\Rightarrow$ linear regression lower bound)**.** *Assume only that Θ is an arbitrary matrix (no rank constraint) and that $\|x(C)\|_2 \le B$ for queried allocations. Then:*

1. *For $D = 1$, there exists a family of instances such that any algorithm requires $n = \Omega(K/\varepsilon^2)$ runs to guarantee $\sup_{C \in \Delta(C_{\text{tot}})} |\log \widehat{L}(C) - \log L(C)| \le \varepsilon$ with constant probability.*
2. *For general D, there exists a family of instances such that guaranteeing simultaneous prediction for all domains requires $n = \Omega(KD/\varepsilon^2)$ runs in the worst case.*

The proof is a reduction to classical minimax lower bounds for linear regression. For (1), take $D = 1$ and choose a hypothesis class where the regression vector $\theta \in \mathbb{R}^K$ has independent coordinates $\theta_k \in \{\pm\alpha\}$; the oracle returns $y^{(i)} = \langle \theta, x^{(i)} \rangle + \xi^{(i)}$. Any estimator that achieves uniform prediction over a bounded set must, in particular, recover the signs of θ_k well enough to predict on feature vectors aligned with coordinate axes (which correspond to allocations concentrating compute on one source, up to the imposed floor). Standard Assouad or Fano arguments then give an $\Omega(K/\varepsilon^2)$ sample requirement. For (2), we apply the same construction independently across domains, i.e. take Θ with rows that vary independently, yielding $\Theta(KD)$ independent signs; each run supplies D noisy scalar observations, so nD observations must cover KD unknown degrees of freedom, giving $n = \Omega(KD/\varepsilon^2)$.

Finally, we remark on a distinct impossibility phenomenon tied to uniform control over mixtures: if one removes the floor $C_{\min}$ and insists on the full simplex boundary, then $x(C) = \log(C)$ is unbounded and no finite-sample procedure can guarantee $\sup_C$ prediction error, even in one dimension. Indeed, for any finite set of queried allocations, one may construct two coefficient vectors that agree (up to the noise level) on those points yet diverge arbitrarily on allocations approaching the boundary, simply by exploiting the unbounded leverage of $\log(C_k)$ as $C_k \to 0$. This justifies the explicit bounded-feature assumption that underlies both our upper and lower bounds.

Taken together, Theorems 8.1 and 8.2 show that our upper bound scaling in (17) is essentially tight: the improvement over naive KD-parameter regression arises precisely from the low-rank coupling across domains and sources, and the multiplicative $1/D$ factor in the required number of runs reflects the fact that each converged run yields D domain measurements.

9 Theory III (Composition to downstream prediction and mixture optimization)

9.1 Error propagation through loss→performance maps

Our stage-1 estimator yields, with high probability, a uniform bound of the form

$$\sup_{C\in\Delta(C_{\text{tot}})} \big\| \log \widehat{L}(C) - \log L(C) \big\|_\infty \;\le\; \varepsilon, \tag{18}$$

where $\log \widehat{L}(C) = \widehat{a} + \widehat{\Theta}\log(C)$. Downstream model selection and mixture choice typically proceed by applying a deterministic map from (predicted) domain losses to a scalar performance metric. We therefore study composition with a function

$$g : \mathbb{R}^D \to \mathbb{R}, \qquad P(C) := g(\log L(C)), \quad \widehat{P}(C) := g(\log \widehat{L}(C)).$$

The simplest sufficient regularity condition is Lipschitz continuity of g on the relevant range of log-loss vectors. Let $\mathcal{Z} \subset \mathbb{R}^D$ denote a compact set containing $\{\log L(C) : C \in \Delta(C_{\text{tot}})\}$ and $\{\log \widehat{L}(C) : C \in \Delta(C_{\text{tot}})\}$ on the event (18). Assume that g is L_g-Lipschitz on $\mathcal{Z}$ with respect to a norm $\|\cdot\|$:

$$|g(z) - g(z')| \le L_g \|z - z'\| \qquad \forall z, z' \in \mathcal{Z}.$$

Then composition immediately transfers uniform prediction error in log-loss space to uniform prediction error in performance:

$$\sup_{C\in\Delta(C_{\text{tot}})} |\widehat{P}(C) - P(C)| \;\le\; L_g \sup_{C\in\Delta(C_{\text{tot}})} \|\log \widehat{L}(C) - \log L(C)\|. \tag{19}$$

In particular, if (18) holds then for $\|\cdot\| = \|\cdot\|_2$ we obtain

$$\sup_C |\widehat{P}(C) - P(C)| \le L_g \sqrt{D}\,\varepsilon,$$

and for $\|\cdot\| = \|\cdot\|_\infty$ we obtain the sharper

$$\sup_C |\widehat{P}(C) - P(C)| \le L_g\,\varepsilon.$$

This covers linear and generalized-linear stage-2 maps, as well as small neural networks with bounded weights on bounded input ranges; in practice, if g is implemented by a trained stage-2 model (e.g. an FLP-M predictor taking domain log-losses as input), one can bound or estimate an empirical Lipschitz constant on $\mathcal{Z}$ via spectral norm products or Jacobian norms.

A further implication is an *optimization regret* guarantee for selecting a mixture by maximizing predicted performance. Let $C^\star \in \arg\max_{C\in\Delta(C_{\text{tot}})} P(C)$ be an optimal (unknown) allocation and let $\widehat{C} \in \arg\max_{C\in\Delta(C_{\text{tot}})} \widehat{P}(C)$ be the recommended allocation. Then deterministically

$$P(C^\star)-P(\widehat{C}) \;\le\; \big(P(C^\star)-\widehat{P}(C^\star)\big)+\big(\widehat{P}(\widehat{C})-P(\widehat{C})\big) \;\le\; 2\sup_C |\widehat{P}(C)-P(C)|. \tag{20}$$

Combining (20) with (19) shows that uniform stage-1 prediction bounds translate to end-to-end mixture-selection guarantees at essentially no additional statistical cost.

9.2 Mixture optimization formulations and convex regimes

We now consider the computational problem of selecting C once $\widehat{L}$ (and optionally g) is fixed. We emphasize two points. First, the stage-1 predictor is cheap to evaluate: computing $\log \widehat{L}(C) = \widehat{a} + \widehat{\Theta}\log(C)$ costs $O(DK)$ arithmetic (or $O(r(D+K))$ if we store a factorization $\widehat{\Theta} = \widehat{U}\widehat{V}$). Second, tractability of mixture optimization depends almost entirely on the structure of the composed objective $\widehat{P}(C) = g(\widehat{a} + \widehat{\Theta}\log(C))$.

To avoid dealing with the equality constraint $\sum_k C_k = C_{\text{tot}}$ directly, it is convenient to work in log-coordinates $x = \log(C)$. If we enforce a floor $C_k \ge C_{\min}$, then $x_k \ge \log C_{\min}$ and the feasible region may be written as the convex set

$$\mathcal{X} \;:=\; \Big\{x \in \mathbb{R}^K : \sum_{k=1}^K e^{x_k} \le C_{\text{tot}}, \quad x_k \ge \log C_{\min}\ \forall k\Big\}. \tag{21}$$

If the objective is monotone in the sense that increasing any C_k cannot decrease performance (equivalently, performance is nondecreasing as losses decrease and losses are nonincreasing in compute), then the constraint in (21) is tight at the maximizer, but using $\le$ is computationally convenient and does not change the optimizer under such monotonicity.

A clean convex regime arises when g is concave as a function of $z = \log L$ (or, more realistically, concave as a function of $-z$ and then we maximize $-g(z)$). For definiteness, suppose that g is concave on $\mathcal{Z}$. Since $x \mapsto \widehat{a}+\widehat{\Theta}x$ is affine, the composition $x \mapsto g(\widehat{a}+\widehat{\Theta}x)$ is concave. Therefore the optimization problem

$$\max_{x\in\mathcal{X}} \; g(\widehat{a} + \widehat{\Theta}x) \tag{22}$$

is a convex optimization problem (maximization of a concave objective over a convex feasible set) and admits polynomial-time algorithms with global optimality guarantees (interior point, first-order methods with appropriate step sizes, etc.). A particularly important special case is a linear stage-2 map

$$g(z) \;=\; \beta_0 - \sum_{d=1}^{D} w_d z_d, \qquad w_d \geq 0, \tag{23}$$

corresponding to maximizing a weighted negative log-loss. Then (22) reduces to maximizing an affine function of x over $\mathcal{X}$, which can be solved efficiently and is often numerically stable.

Outside such structured cases, mixture optimization becomes non-convex. This includes essentially all multi-layer neural-network stage-2 maps (such as a generic FLP-M stage-2 MLP), as well as objectives designed to approximate thresholded or saturated metrics. In these settings we do not expect polynomial-time global optimality guarantees without further restrictions; the best we can generally ensure is that the *statistical* error in the objective is controlled (via (19)) while the optimization itself is handled by standard non-convex heuristics.

9.3 Practical algorithms on the simplex

When (22) applies, we may use projected gradient ascent in x:

$$x^{t+1} \;=\; \Pi_{\mathcal{X}}\Big(x^t + \eta_t \nabla_x g(\widehat{a} + \widehat{\Theta} x^t)\Big),$$

where $\Pi_{\mathcal{X}}$ denotes Euclidean projection onto $\mathcal{X}$. The projection can be implemented by a scalar dual search because the constraint $\sum_k e^{x_k} \leq C_{\text{tot}}$ is separable after introducing a Lagrange multiplier, and the lower bounds $x_k \geq \log C_{\min}$ can be imposed by clipping. If g is differentiable and implemented by a stage-2 neural model, then $\nabla_x g(\widehat{a} + \widehat{\Theta} x)$ is obtained by backpropagation through the affine map $x \mapsto \widehat{a} + \widehat{\Theta} x$.

A mirror-descent parameterization is also natural. Writing $C = C_{\text{tot}} p$ with p on the probability simplex (and a corresponding floor), we have $\log C = \log C_{\text{tot}} \cdot \mathbf{1} + \log p$. Updates in p using exponentiated gradients respect nonnegativity automatically:

$$p_k^{t+1} \;\propto\; p_k^t \exp\Big(\eta_t \, \partial_{x_k} g(\widehat{a} + \widehat{\Theta} x^t)\Big),$$

followed by renormalization and enforcement of the floor. This is often preferable numerically when K is moderate to large, because it avoids repeated exponentiation inside projections.

In the non-convex regime, we recommend two families of methods. First, multi-start projected gradient (or mirror descent) remains effective because evaluating $\widehat{P}(C)$ is cheap and differentiable in x whenever g is differentiable. One runs several initializations (e.g. near vertices, near the uniform mixture, and near mixtures suggested by simpler baselines such as one-source optima) and keeps the best outcome. Second, if gradients are unavailable or unreliable (e.g. g is a black box trained with non-smooth preprocessing), one may apply derivative-free optimization directly over $\Delta(C_{\mathrm{tot}})$, including Bayesian optimization with a simplex-aware kernel, random search with low-discrepancy sequences, or evolutionary strategies. Because the objective evaluation is inexpensive, these methods can be run with a large number of function evaluations without incurring additional converged-run cost.

Finally, when we have uncertainty estimates for $\log \widehat{L}(C)$ (e.g. from a bootstrap over the stage-1 fit or a debiased low-rank estimator), we may incorporate them into mixture selection via robust or optimistic criteria, such as maximizing a lower confidence bound

$$\widehat{P}(C) \; - \; \beta \, \mathrm{rad}(C),$$

where $\mathrm{rad}(C)$ upper bounds the possible performance error induced by uncertainty in $\log \widehat{L}(C)$ through a Lipschitz factor. This does not change the fundamental sample complexity—which is governed by the stage-1 oracle calls—but can materially improve practical stability of the chosen mixture when n is small or when the design is imperfect.

10 Experiments (recommended)

We outline an experimental suite intended to validate (i) the stage-1 low-rank log-linear mixing law, (ii) the sample-efficiency claims implied by the theory, and (iii) the practical usefulness of the resulting predictor for mixture selection. The experiments are structured to be feasible with a small number of fully-converged runs while still enabling informative ablations.

10.1 Real pretraining study with $K \in \{3, 4, 5, 6\}$ sources

Data sources and compute accounting. We select $K \in \{3, 4, 5, 6\}$ pretraining data sources that are sufficiently distinct to plausibly induce domain tradeoffs, e.g. *web*, *code*, *math*, *multilingual*, *instruction-like synthetic*, and *academic*. We fix a total compute budget C_{tot} (token-FLOPs proxy) and represent an allocation by $C \in \Delta(C_{\mathrm{tot}})$, enforcing a floor $C_k \geq C_{\min}$ for numerical stability. For each converged run we record (a) the allocation C, (b) the converged validation losses on D held-out domains, and (c) auxiliary traces (training loss, gradient norm) for sanity checking convergence.

Validation domains. We choose D validation domains aligned with the sources but not identical to them, so that generalization across related distributions is measured rather than memorization of source-specific validation sets. For example, for a *code* source we include multiple code validation sets (different languages or repositories); for *multilingual* we include multiple scripts; for *math* we include both formal and informal math. We report domain losses in log-space, i.e. $y_d = \log L_d(C)$, consistent with the model.

Designs and number of runs. We compare at least three experiment designs for choosing allocations $\{C^{(i)}\}_{i=1}^n$: (i) the proposed sparse randomized design (sampling supports of size s and drawing weights from a well-spread Dirichlet, with floor on the complement); (ii) a dense Dirichlet design (weights over all K sources, no sparsity beyond the floor); (iii) a vertex-plus-uniform design (the K near-vertices and a few near-uniform points), which is a common heuristic but is not tailored to low-rank recovery. For each design we sweep n over a small grid (e.g. $n \in \{K, 2K, 4K, 8K\}$) to expose scaling. The primary evaluation is out-of-design prediction error on a separate test set of allocations $C \sim \text{Dirichlet}(\alpha)$ (with the same floor), reporting

$$\text{Err}_\infty := \max_{d\in[D]} \max_{j\in[m]} \left|\widehat{y}_d(C^{\text{test},(j)}) - y_d(C^{\text{test},(j)})\right|, \qquad \widehat{y}(C) := \widehat{a} + \widehat{\Theta}\log C.$$

We also report ℓ_2 and per-domain errors to diagnose whether a few domains dominate.

10.2 Stage-1 fitting details and diagnostics

Estimators. We fit the nuclear-norm regularized regression model

$$(\widehat{a}, \widehat{\Theta}) \in \arg\min_{a,\Theta} \frac{1}{2n}\|Y - \mathbf{1}a^\top - X\Theta^\top\|_F^2 + \lambda\|\Theta\|_*,$$

and compare it to (i) an explicit rank-r factorization fit $\Theta = UV$ with alternating minimization, and (ii) an unconstrained (full-rank) ridge baseline. We tune λ (or r) by cross-validation over runs, using held-out allocations. To check qualitative model fit we report residual plots versus $\log C_k$ and test for systematic curvature indicating misspecification.

Monotonicity and sign constraints. When the application justifies that more compute on a source should not increase loss, we optionally enforce $\Theta \le 0$ coordinatewise (either as a hard constraint or via a penalty). We then evaluate whether the constraint improves out-of-design accuracy and whether it stabilizes mixture selection. We report the fraction of unconstrained fits that violate $\widehat{\Theta}_{d,k} \le 0$ and the magnitude of violations.

Rank diagnostics. We compute the singular values of $\widehat{\Theta}$ and report the effective rank (e.g. the smallest r capturing 95% of Frobenius energy). We also evaluate prediction error as a function of enforced rank, including under- and over-specified regimes, to empirically test the sensitivity implicit in the low-rank assumption.

10.3 Ablations: rank, design sparsity, and noise

Rank ablation. Holding the design fixed, we sweep r (or the regularization strength) and plot error versus effective rank. The goal is to verify that a low-dimensional latent structure is sufficient and that gains saturate after a small $r \ll \min\{D, K\}$.

Design sparsity and conditioning. In the sparse design we sweep the support size s (e.g. $s \in \{2, 3, 4, K\}$ for $K \leq 6$) and record both predictive performance and numerical conditioning of X (restricted eigenvalues or empirical condition number on the sampled subspace). This ablation distinguishes whether improvements arise from sparsity per se or from improved spread in log-coordinates.

Noise robustness. We separately consider (i) measurement noise from finite validation tokens and (ii) optimization noise from imperfect convergence. To probe robustness we repeat evaluation with reduced validation-set size (increasing measurement variance) and with early-stopped checkpoints (introducing bias). We then check whether the error behaves approximately as $O(n^{-1/2})$ when noise dominates, and we report failure modes when bias dominates.

10.4 Baselines: pairwise/binary mixing laws and classical heuristics

To justify the multivariate low-rank law, we compare against two families of alternatives.

Pairwise laws. A common approach is to fit pairwise tradeoff curves between sources, e.g. modeling loss as a function of the mixture weight between two sources while holding others at a floor. We implement a pairwise surrogate by fitting, for each pair (k, k'), a univariate model on the line segment between near-vertices and then combining pairwise preferences to propose a global mixture (e.g. via tournament or aggregation). We evaluate both predictive accuracy on random mixtures and the quality of the proposed mixture under a fixed downstream objective.

Binary/one-vs-rest laws. Another heuristic is to treat each source as either "on" or "off" and fit a regression on the indicator vector (or on a coarse grid of weights). We fit such models at equal n and compare sample efficiency. Since these models discard the log-scale structure, we expect degraded extrapolation; the experiment quantifies the extent of degradation.

10.5 Mixture optimization case studies

We recommend at least two case studies that use the fitted stage-1 predictor as the sole expensive component and then perform mixture selection by optimizing a downstream map (trained cheaply, or specified analytically).

Case study A: linear downstream objective. We define a linear proxy

$$g(z) = \beta_0 - \sum_{d=1}^{D} w_d z_d, \qquad w_d \geq 0,$$

representing a weighted preference over validation domains. We then solve $\max_{C \in \Delta(C_{\mathrm{tot}})} g(\widehat{a} + \widehat{\Theta} \log C)$ and report (i) the recommended C, (ii) the predicted improvement relative to uniform and best single-source baselines, and (iii) the realized improvement when we actually run a small number of confirmatory trainings near the recommended mixture. This directly tests whether prediction accuracy is sufficient for decision quality.

Case study B: learned stage-2 predictor. We train a small stage-2 model (e.g. a shallow MLP) mapping predicted domain log-losses to a measured downstream metric (such as accuracy on a held-out evaluation suite). The training labels for stage-2 are obtained from intermediate checkpoints or from a small pool of already-trained models, making stage-2 inexpensive relative to converged runs. We then optimize the composed objective and evaluate regret by comparing the chosen mixture to the best mixture found by an expensive grid search over a low-dimensional simplex (possible when $K \leq 6$).

10.6 Scaling simulations to $K \in [10, 20]$

To test regimes where real converged runs are infeasible, we recommend a controlled simulator generated from the assumed model: draw a ground-truth $\Theta_\star$ of rank r (e.g. $\Theta_\star = U_\star V_\star$ with bounded entries), draw intercepts $a_\star$, and generate noisy observations

$$Y = \mathbf{1} a_\star^\top + X \Theta_\star^\top + E,$$

with E sub-Gaussian. We then repeat the full pipeline (design, fit, prediction, mixture optimization) for $K \in \{10, 15, 20\}$, varying n and comparing sparse versus dense designs. The main outputs are (i) empirical sample complexity curves for Err_∞, (ii) sensitivity to rank mis-specification, and (iii) decision regret under a Lipschitz downstream map. This simulation isolates statistical scaling from systems artifacts and provides evidence that the proposed approach continues to behave favorably as K grows beyond what is practical for exhaustive mixture sweeps.

11 Discussion and limitations

Misspecification of the log-linear law. Our guarantees are conditional on the approximation

$$\log L_d(C) \approx a_d + \theta_d^\top \log C,$$

together with low rank of the coefficient matrix. In practice, several deviations are plausible. First, losses may exhibit curvature in $\log C$ due to threshold effects (e.g. a source only becomes useful beyond some minimum exposure) or diminishing returns that are not well captured by a single affine function on the entire simplex. Second, interactions between sources may be non-additive in log-space: a domain may benefit from *joint* exposure to two sources more than the sum of their individual contributions (synergy), or conversely suffer from interference. Third, the noise may be heteroskedastic across allocations, e.g. if certain mixtures make optimization less stable and yield higher run-to-run variance, violating the simplest sub-Gaussian homoskedastic abstraction.

From a methodological standpoint, we view the low-rank log-linear model as a first-order local approximation in the coordinate system induced by $\log C$. Accordingly, we recommend explicit misspecification diagnostics in any deployment: (i) residual-versus-$\log C_k$ plots; (ii) held-out prediction on allocations sampled from a distribution different from the training design; and (iii) sensitivity of recommendations to mild perturbations of $\widehat{\Theta}$ (a stability proxy). When misspecification is detected, natural extensions preserve the same experimental philosophy while relaxing the model: add a small set of nonlinear features in $\log C$ (quadratic terms, splines, or random features) with a low-rank constraint on the resulting coefficient tensor; or adopt a piecewise-affine model over a partition of the simplex (at the cost of more runs). These extensions reduce to structured regression problems in which the stage-1 objective remains convex (or nearly so) and the primary bottleneck remains oracle calls, not computation.

Rank as an approximation and identifiability under limited designs. Even when the log-linear form holds approximately, the effective rank of Θ may not be small globally. Empirically, we expect approximate low rank when domains share latent "skills" (e.g. reasoning, multilinguality) that are elicited by multiple sources, but the number of such factors can grow with the diversity of D and with the granularity at which sources are defined. When r is misspecified upward, nuclear-norm regularization typically yields graceful degradation but may produce overly smooth predictions; when misspecified downward, bias can be substantial and can manifest as systematic errors localized to particular corners of the simplex.

A separate limitation concerns identifiability induced by the design distribution for C. The theory presumes that the induced design matrix X

satisfies an RSC-type condition on low-rank directions, which in turn requires sufficient variation in $\log C$ along informative coordinates. Designs that are overly concentrated near the uniform mixture, or that vary only a few sources in a correlated manner, can make Θ effectively unidentifiable even if low rank holds. This is not merely a proof artifact: if two sources are always co-allocated, their effects cannot be disentangled from loss data alone. In such cases, we should either (i) redesign experiments to decorrelate allocations, or (ii) accept that only a lower-dimensional combination of sources is estimable and explicitly merge the confounded sources into a single composite coordinate.

Nonstationary data and shifting objectives. The formulation treats sources as stationary distributions and assumes that the loss mapping $C \mapsto L(C)$ is stable across time. Modern pretraining pipelines violate this in at least three ways: (i) data refresh and deduplication alter the marginal distribution of each source; (ii) model architecture, tokenizer changes, or optimizer schedules change the mapping from exposure to loss; and (iii) the downstream objective that ultimately matters may drift as benchmarks evolve. Under such nonstationarity, a single Θ estimated from historical runs can become stale, and mixture recommendations can be systematically suboptimal.

A principled extension is to treat Θ (and possibly a) as time-indexed, Θ_t, and impose a smoothness or bounded-drift prior, e.g. $\|\Theta_t - \Theta_{t-1}\|_F \leq \eta$. One can then fit Θ_t via windowed nuclear-norm regression or via an online proximal method with a forgetting factor. The experimental implication is that a small number of periodic re-estimation runs can maintain calibration, analogous to standard monitoring in production ML systems. We emphasize that this requires committing to a reproducible compute accounting scheme so that "one unit of compute" is comparable across time; otherwise, even the coordinate system $x(C) = \log C$ becomes ill-defined.

Overlap, contamination, and the meaning of "sources" and "domains". Our abstraction treats sources as distinct and validation domains as fixed held-out distributions. In reality, sources can overlap substantially (e.g. web data contains code and math; multilingual corpora contain English), and validation domains can be contaminated by training data or by near-duplicates. Overlap has two distinct consequences. First, it reduces the effective dimension of the problem: if two sources are nearly identical in distribution, then varying their relative compute has little effect, and the corresponding columns of Θ become nearly collinear. Second, contamination breaks the semantic link between loss and generalization, potentially yielding misleadingly optimistic losses in domains that are actually present in training.

Mitigations are partly procedural: perform deduplication between sources and between training and validation; quantify overlap via hashing or embedding-based near-neighbor estimates; and report overlap statistics alongside fitted coefficients. Conceptually, one can incorporate overlap by modeling each observed source as a mixture of latent "pure" sources and estimating in that latent basis; however, this introduces additional non-identifiability unless one has external measurements of overlap. For the purpose of mixture selection, a conservative stance is to treat any domain with suspected contamination as a weak label for generalization and to downweight it in the downstream map g (or to exclude it entirely), rather than attempting to correct post hoc via the stage-1 fit.

Extension to continual pretraining and curriculum-like schedules. Our allocation variable C describes a *static* distribution of compute across sources. Many training runs use curricula: allocations change over steps, sources are introduced or removed, and the effective learning dynamics depend on order. A static mixture can approximate such schedules only insofar as the optimization is path-independent with respect to the empirical distribution of tokens, which is not generally true.

To extend the framework, we can parameterize a schedule by a small set of interpretable variables (e.g. two-phase mixtures with a switch time, or a low-dimensional basis of time-varying weights) and treat those variables as the decision vector in place of C. The same measurement model then applies with $x(\cdot)$ replaced by a feature map of the schedule. The limitation is that schedule spaces are larger and can quickly invalidate the sample-efficiency advantages unless we impose strong structure (low-dimensional parameterizations, low rank, monotonicity) and correspondingly design experiments that excite the schedule degrees of freedom.

Interaction with post-training (SFT/RLHF) and evaluation pipelines. The stage-1 predictor targets pretraining losses, whereas many downstream metrics are realized after substantial post-training. The composition theorem with a Lipschitz map g provides a clean abstraction, but it may fail if post-training introduces sharp regime changes (e.g. instruction tuning that disproportionately benefits certain domains once a capability threshold is crossed). Moreover, post-training may depend on *representations* and not merely on pretraining losses, so two mixtures with similar $L(C)$ can yield different post-training outcomes.

In such settings, we see two practical directions. First, treat post-training as inducing additional "domains" by measuring losses (or proxy scores) on post-training-relevant validation sets at intermediate checkpoints; this increases D but keeps the stage-1 fit conceptually intact. Second, learn g on a dataset that spans mixtures, so that the composed objective implicitly captures post-training sensitivity; this shifts burden to stage-2 generalization and demands careful control of covariate shift between the mixtures used to train g and those explored during optimization.

Reproducibility artifacts and what should be released. Because oracle calls are expensive, the credibility of any empirical claim depends on eliminating avoidable sources of variance. We therefore regard reproducibility as part of the technical contribution. At minimum, an artifact should include: (i) the allocation design generator (including random seeds, Dirichlet parameters, and the floor $C_{\min}$); (ii) the exact compute accounting conventions (tokenization, sequence packing, and any per-source weighting that affects token-FLOPs); (iii) the fitting code for the nuclear-norm estimator (solver choice, stopping criteria, regularization path, and cross-validation splits); and (iv) the evaluation protocol for out-of-design allocations. If mixture optimization is performed, the optimizer (initialization, constraints, and termination) should be fixed and logged, since many objectives are non-convex in practice and can be sensitive to these choices.

Finally, we emphasize that the experimental efficiency of the approach can create a false sense of certainty: producing a recommendation after relatively few converged runs is only useful if uncertainty is quantified and the recommendation is robust. Reporting confidence intervals for predicted losses (e.g. via bootstrap over runs, or via asymptotic variance estimates under a fixed design) and performing a small number of confirmatory runs near the proposed optimum are therefore not optional if the method is used to make high-stakes training decisions.