ALLOCATE DON'T OPERATE: ONLINE DATA ALLOCATION FOR MULTI-TASK LLM POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training a large language model rarely targets a single capability: a typical run might mix a wide variety of agentic and non-agentic abilities such as coding, tool-calling, mathematical reasoning, and instruction following. Classical multi-task learning studies this setting through the lenses of gradient interference and multi-objective fairness, motivating methods such as gradient surgery and MGDA. These tend to fit post-training poorly, as they are either not scalable or target the wrong horizon. In this work, we propose to set the task composition of each mini-batch as the only control variable, casting multi-task post-training as an online optimization problem that maximizes cumulative policy improvement subject to a "no-task-left-behind" constraint. We propose a concavity assumption on the task-improvement surface, which relaxes the intractable constraint into a linear one that facilitates the use of mirror-descent methods from online convex optimization. This yields **Primal-Dual Curvature Matching** (PDCM), a lightweight allocator driven by first-order estimates of how policy improvement responds to allocation. Unlike prior allocation methods developed for LLM pre-training, it presumes no parametric model of allocation dynamics. By controlling batch composition alone, PDCM leads the uniform baseline on pass@1 by as much as 6.3% and consistently beats gradient-surgery and parametric mixing methods. Additionally, we show that PDCM scales to controlling up to 18 allocation components, achieving an improvement in gained pass@1 of up to 12.5% over the uniform baseline.

1 INTRODUCTION

Post-training has become a central stage in the development of modern large language models, transforming a pre-trained model into one that can reliably follow instructions (Ouyang et al., 2022), reason over complex inputs (Guo et al., 2025), and interact with users across a broad range of applications (Wang et al., 2026; Yue et al., 2026). In practice, this stage is rarely optimized for a single task or capability, which yields a natural multi-task learning problem (NVIDIA et al., 2025). Under a finite training budget, choosing this mixture becomes especially important, as tasks can differ substantially in difficulty and learning dynamics, causing some tasks to improve faster than others under the same allocation (Wu et al., 2026). Furthermore, gradients from different tasks can interfere through shared model updates, suppressing useful task-specific learning signals and reducing overall training efficiency (Yu et al., 2020; Liu et al., 2021).

Existing multi-task learning methods address these issues through two broad pathways: gradient-surgery methods that explicitly compute and manipulate task-wise gradients (Yu et al., 2020; Liu et al., 2021), or fairness and multi-objective-aware methods that reweigh task losses to favor those making the least amount of progress (Liu et al., 2021; Sener & Koltun, 2018). While these approaches provide principled mechanisms for multi-task learning, neither pathway can be relied upon directly for LLM post-training. Gradient-based methods such as PCGrad (Yu et al., 2020) require computing and storing task-wise gradients, introducing substantial additional backward-pass and memory costs for models with billions of parameters. These costs scale further with the number of tasks linearly and are difficult to reconcile with highly optimized LLM training pipelines based on gradient accumulation, parameter sharding, and distributed optimization (Sheng et al., 2024). Fairness- and multi-objective-aware methods such as MGDA instead focus on convergence toward balanced or Pareto-stationary solutions. However, LLM post-training is typically conducted under

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

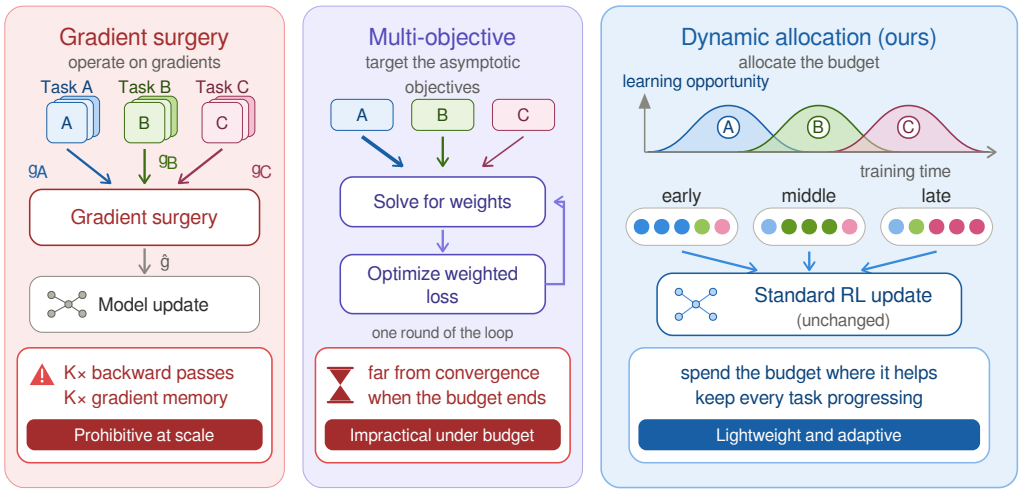


Figure 1: Previous work on multi-task learning either frames the problem as gradient-based optimization or as asymptotic multi-objective optimization. Neither class is well suited to the dynamic nature of multi-task post-training.

a fixed compute budget and stopped well before convergence, making finite-horizon training efficiency more relevant than asymptotic behavior. These limitations make both classes of methods poorly matched to the practical requirements of contemporary LLM post-training.

In this paper, we formulate multi-task post-training as an online convex optimization problem in which the only control variable is the task quota assigned to each mini-batch. Across training steps, the allocation policy seeks to maximize cumulative policy improvement under a set of "*no-task-left-behind*" constraints. These constraints measure, for each task, how much the policy improvement induced by the current allocation can lag behind a counterfactual uniform allocation. To make the constrained problem tractable, we introduce a concavity assumption on the task-improvement surface, which linearizes the constraints. The resulting formulation leads to a primal-dual mirror-descent-like algorithm, which we dub **Primal-Dual Curvature Matching (PDCM)**, that dynamically adjusts task quotas using first-order policy improvement information with allocation.

Empirically, PDCM attains the highest total on every suite–metric combination we report, leading a uniform mixture by as much as 6.3%, without any single task falling behind by more than 1.5 points. The advantage also survives scale, holding a 5.8–12.5% margin in gained performance over an even split as the mixture grows from 9 to 18 components.

Our contributions are summarized as follows:

- We formulate multi-task post-training as a constrained optimization over task allocation: maximize cumulative policy improvement subject to a no-task-left-behind constraint against a uniform-allocation counterfactual, made tractable by linearizing that constraint under a concavity assumption.
- We introduce **Primal-Dual Curvature Matching (PDCM)**, a **lightweight task-allocation layer** that adjusts per-task quotas from observed marginal policy improvement and task-wise constraint signals, with guarantees on both regret and constraint violation.
- We evaluate PDCM on **agentic and reasoning** suites. It attains the best total on all four suite–metric combinations, by up to 6.3% over a uniform mixture. Its largest single-task shortfall against uniform is 1.5 points, versus 6.9 for the parametric allocator of Chen et al. (2025) and 37.5 for gradient surgery. Its margin on gained performance consistently holds at 5.8–12.5% as the mixture grows from 9 to 18 components.

2 Background

In multi-task post-training, we are interested in optimizing a policy π on $K \geq 2$ tasks for a budget of T rounds. The task k has input distribution $\mathcal{X}_k$ and reward r_k, with rewards sharing a common

scale. Let $J_k(\pi)$ denote the expected return of policy π on task k, and let $J(\pi) \coloneqq \sum_{k=1}^{K} J_k(\pi)$. At round t, the learner splits a fixed-size mini-batch according to an *allocation*

$$p_t \in \Delta^K \coloneqq \left\{ p \in \mathbb{R}_+^K : \sum_{k=1}^{K} p_k = 1 \right\},$$

and takes one update of the underlying RL algorithm, which is held fixed. Thus, $p_1, \ldots, p_T$ are the only control variables. Let $\pi_t(p)$ denote the policy obtained by applying the round-t update to π_{t-1} under task allocation p, which we also abbreviate as $\pi_t \coloneqq \pi_t(p_t)$. We define the *task-wise one-step policy improvement*

$$U_{t,k}(p) \coloneqq J_k(\pi_t(p)) - J_k(\pi_{t-1}),$$

and the task-aggregated improvement $U_t(p) \coloneqq \sum_{k=1}^{K} U_{t,k}(p)$. Therefore, total policy improvements across tasks can be telescoped as

$$J(\pi_T) - J(\pi_0) = \sum_{t=1}^{T} U_t(p_t). \tag{1}$$

Maximizing $\sum_{t=1}^{T} U_t(p_t)$ is therefore a finite-budget policy-improvement objective. Two properties of this objective make it difficult to optimize directly. The first is that the shape of $U_{t,k}$ is neither known in advance nor stable across training. All tasks share a single set of parameters, so an update driven by task i also moves the return on task j. *A priori* parametric description of the allocation-to-improvement mapping, such as those in Chen et al. (2025), is unlikely to be reliable over a whole run. Second, the aggregate U_t is indifferent to *which* task supplies the improvement, so its maximizer concentrates the budget on whichever tasks return the largest immediate gain and starves the remainder, which betrays the intent of multi-task post-training.

3 Primal-Dual Curvature Matching

As discussed in Section 2, naively maximizing the total policy improvement in Equation (1) can lead to skewed development over tasks that should otherwise be treated equally (see Section 4.4). We propose the following constraint that prevents any task from falling behind what would be achieved by a uniform allocation $u = (\frac{1}{K}, \ldots, \frac{1}{K})$ by a threshold $\epsilon > 0$:

$$\sum_{t=1}^{T} U_{t,k}(u) - U_{t,k}(p_t) \le \epsilon, \quad \forall k \in [K]. \tag{2}$$

Equation (2) enforces an upper-limit on the cumulative counterfactual lag. In practice, estimating $U_{t,k}(u)$ requires a parallel run with uniform allocation u, which is cost-prohibitive. We relax the parametric assumptions in recent work (Ye et al., 2025; Chen et al., 2025; Kang et al., 2025) to one in which the policy-improvement surface with respect to mixture allocation is concave over the allocation simplex. We validate this assumption empirically in Appendix B.

Assumption 1 (Concave allocation response)**.** *For every task $k \in [K]$ and iteration t, the task-wise utility $U_{t,k}$ is differentiable and concave on Δ_K. It therefore admits the first-order upper bound at p_t, with $g_{t,k} = \nabla U_{t,k}(p_t)$,*

$$U_{t,k}(u) - U_{t,k}(p_t) \le \langle g_{t,k}, u - p_t \rangle. \tag{3}$$

Equation (2) would be satisfied if we instead impose the RHS of Equation (3), albeit being more restrictive. This leads to the following constrained optimization problem:

$$\begin{aligned} \max_{p_1,\ldots,p_T \in \Delta_K} \quad & \sum_{t=1}^{T} U_t(p_t) \\ \text{s.t.} \quad & \sum_{t=1}^{T} \langle g_{t,k}, u - p_t \rangle \le \epsilon, \quad \forall k \in [K]. \end{aligned} \tag{4}$$

Remark. While $g_{t,k}$ in principle needs to be estimated using counterfactual allocations as well, this formulation allows us to assume that $U_{t,k}$ is approximately static around a given t and to estimate $g_{t,k}$ by regressing on the values of $U_{t,k}$ of previous H steps (see Section 3.3).

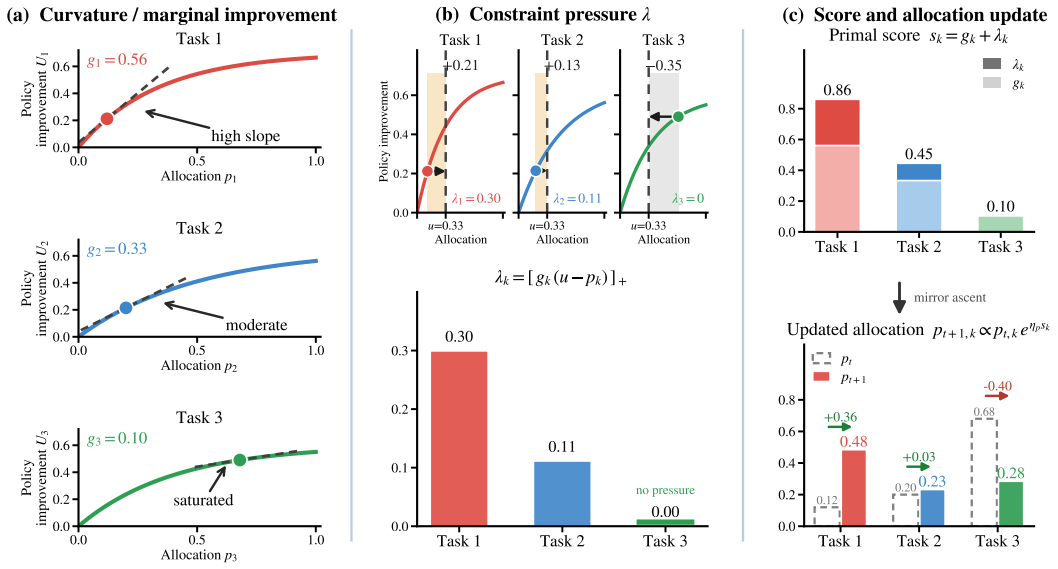


Figure 2: **Mechanistic illustration of PDCM. (a)** Concave policy improvement per task; the slope $g_k = \partial U_k / \partial p_k$ at the current allocation gives the marginal return (Task 1 steep, Task 3 saturated). **(b)** Constraint pressure as an alignment of the uniform gap with curvature, $\lambda_k = [g_k(u - p_k)]_+$, $u = 1/K$: only tasks that are both under-allocated and still improving accumulate pressure. **(c)** The primal score $s_k = g_k + g_k \lambda_k$ drives the mirror-ascent update $p_{t+1,k} \propto p_{t,k} e^{\eta_p s_k}$, moving budget from the saturated task to the steep, under-served one.

3.1 Primal-Dual Curvature Matching

Equation (4) is an online optimization problem with linear constraints, which allows us to employ Primal-Dual Mirror Descent (PDMD) (Beck & Teboulle, 2003; Mahdavi et al., 2012; Jenatton et al., 2016). Generally, PDMD guarantees sublinear regret under convex constraints by controlling the allocation via an exponential update rule

$$p_{t+1,j} = \frac{p_{t,j} \exp(\eta_p s_{t,j})}{\sum_{\ell=1}^{K} p_{t,\ell} \exp(\eta_p s_{t,\ell})}, \tag{5}$$

where $s_{t,j}$ incorporates both the first-order information G_t of the primal objective Equation (1) and a dual variable λ_t that accumulates historical lag.

$$G_t = [g_{t,1}, \ldots, g_{t,K}]^T, \qquad \text{where } g_{t,k} = \nabla U_{t,k}(p_t) \quad \forall k \in [K] \tag{6}$$

$$\lambda_{t+1,k} = [\lambda_{t,k} + \eta_\lambda (\langle g_{t,k}, u - p_t \rangle - \epsilon/T)]_+, \qquad \text{where } [x]_+ := \max\{x, 0\} \tag{7}$$

G_t contains local information on how changes in the allocation p_t affect task-wise policy improvement $U_{t,k}$ and λ_t encodes how much Equation (3) has been violated. The exponential logits s_t are then combined in the following vectorized form

$$s_t = G_t^\top \mathbf{1} + G_t^\top \lambda_t \tag{8}$$

where the term $G_t^\top \mathbf{1}$ captures the first-order marginal utility of each allocation coordinate, while $G_t^\top \lambda_t$ corrects for lag using a dual-weighted marginal utility. We summarize the algorithm in Algorithm 1.

Remark. We note that as we chose Equation (3) to be the constraint, G_t becomes the only quantity we need to estimate throughout the algorithm, a crucial design for the algorithm's tractability and simplicity. Because of this reliance on G_t, we name this algorithm **Primal-Dual Curvature Matching** (PDCM)[1]. The adaptive allocation in PDCM is intended to exploit temporal variation in task-wise marginal utility. The following result shows that this adaptivity remains asymptotically competitive with any fixed feasible allocation, including the uniform mixture, while simultaneously controlling cumulative task-wise lag.

[1]We recognize that it could be confusing to name the algorithm "curvature matching" when G_t is the Jacobian of $U_{t,k}$. Meanwhile, curvature conventionally refers to the Hessian. We argue that G_t contains second-order information of the total improvement $J(\pi_T) - J(\pi_0)$.

Algorithm 1 Primal–Dual Mirror Descent with Full Jacobian

Require: Number of tasks K; horizon T; primal step size $\eta_p > 0$; dual step size $\eta_\lambda > 0$; constraint budget $\epsilon \geq 0$; Dithering strength schedule $\{\nu_t\}$, where $\nu_t \in [0, 1]$.

Initialize $p_1 \leftarrow u$, $\lambda_1 \leftarrow \mathbf{0}$ and initial policy π_0.
Maintain a history of size H of task-wise improvement vectors $\{\widehat{U}_\tau\}_{\tau=t-H}^{t-1}$ and allocation-response Jacobians $\{\widehat{G}_\tau\}_{\tau=t-H}^{t-1}$ and allocations $\{p_\tau\}_{\tau=t-H}^{t-1}$.
for $t = 1, \ldots, T$ **do**
  Add Dirichlet noise $\tilde{p}_t = (1 - \nu_t)p_t + \nu_t\alpha, \quad$ where $\alpha \sim \mathrm{Dirichlet}(\mathbf{1})$
  Obtain π_t through policy update under allocation $\tilde{p}_t$. ▷ *RL step.*
  Compute the task-wise improvement vector by Equation (11).

$$\widehat{U}_t = \left(\widehat{U}_{t,1}, \ldots, \widehat{U}_{t,K}\right)^\top.$$

  ▷ *Extra forward pass with the updated policy.*
  Estimate the Jacobian $\widehat{G}_t$ by Equation (12). ▷ *Applies Theil-Sen regression to estimate first-order sensitivity.*
  Compute the primal scores s_t by Equation (8). ▷ *Combine primal and dual weights.*
  Update the primal allocation p_{t+1} by entropic mirror ascent using Equation (5). ▷ *Exponential weight update.*
  Update the dual variables λ_{t+1} by projected gradient ascent using Equation (7).
end for

Theorem 1 (Constrained no-regret of PDCM)**.** *Suppose Assumption 1 holds, together with the boundedness and dual-stability conditions stated in Appendix E. Let each task-wise utility $U_{t,k}$ be twice-differentiable with bounded Jacobian, and let the cumulative dual lag λ_t be bounded. Let*

$$p_T^\star \in \arg\max_{p \in \mathcal{P}_T} \sum_{t=1}^{T} U_t(p)$$

denote the best fixed feasible allocation in hindsight, where $\mathcal{P}_T := \{p \in \Delta_K : h_t(p) \leq 0$ coordinatewise for every $t \leq T\}$ is the first-order-safe comparator set of Equation (27). If $\eta_p, \eta_\lambda = \Theta(T^{-1/2})$, then update rule Equation (5) satisfies

$$\sum_{t=1}^{T} \left[\sum_{k=1}^{K} U_{t,k}(p_T^\star) - \sum_{k=1}^{K} U_{t,k}(p_t)\right] \leq O(\sqrt{T}), \tag{9}$$

while, for every task k,

$$\left[\sum_{t=1}^{T} \left(U_{t,k}(u) - U_{t,k}(p_t) - \epsilon/T\right)\right]_+ \leq O(\sqrt{T}). \tag{10}$$

Consequently, both the average regret and the average excess counterfactual lag vanish at rate $O(T^{-1/2})$. We provide a proof in Appendix E.

What is left is the estimation of the policy improvement gradient $\hat{G}_t$. For that, we need a fast and tractable way of estimating the policy improvement $U_{t,k}$ and its gradient with respect to p_t.

3.2 Policy Improvement Estimation

The performance difference lemma (Kakade & Langford, 2002) expresses the task-wise policy improvement $U_{t,k}$ as an expectation, over rollouts drawn from π_{t-1}, of the advantage $A_{\pi_{t-1}}$ reweighted by the importance ratio π_t/π_{t-1}. This makes $U_{t,k}$ a convenient target for off-policy evaluation (Precup et al., 2000; Gu et al., 2026): we reuse the rollouts already generated for the training batch and estimate the task-wise policy improvement as

$$\widehat{U}_{t,k} = \frac{1}{N_k} \sum_{i=1}^{N_k} \frac{\pi_t(y_i \mid x_i)}{\pi_{t-1}(y_i \mid x_i)} A_{\pi_{t-1}}(y_i \mid x_i), \tag{11}$$

Table 1: **Multi-task post-training results.** Per-task accuracy (%) on the Nemo-Gym (four tasks) and ACZ (three tasks) suites, at pass@1 and pass@4. **PDCM** is our method; **Uniform** holds the mixture fixed at $u = 1/K$, **Single** trains one specialist per task, **PCGrad** (Yu et al., 2020) resolves gradient conflicts, and **Aioli** (Chen et al., 2025) allocates under a parametric mixing law. Per row, the best result is in **bold** and the second best is underlined; *Total* is the sum over a suite's tasks.

	pass@1					pass@4				
Task	PDCM	Uniform	Single	PCGrad	Aioli	PDCM	Uniform	Single	PCGrad	Aioli
Nemo-Gym										
`code_gen`	$32.61_{\pm 2.2}$	$34.08_{\pm 2.1}$	$31.29_{\pm 1.1}$	$\underline{34.39}_{\pm 2.4}$	$\mathbf{35.40}_{\pm 2.2}$	$40.55_{\pm 2.6}$	$41.41_{\pm 2.3}$	$37.73_{\pm 1.3}$	$\underline{42.43}_{\pm 3.4}$	$\mathbf{43.29}_{\pm 2.7}$
`IF`	$\mathbf{53.33}_{\pm 0.5}$	$47.58_{\pm 1.2}$	$36.63_{\pm 0.3}$	$\underline{51.00}_{\pm 1.5}$	$47.79_{\pm 2.5}$	$\mathbf{61.21}_{\pm 0.7}$	$54.57_{\pm 1.2}$	$45.76_{\pm 0.4}$	$\underline{58.81}_{\pm 1.4}$	$52.09_{\pm 2.1}$
`workplace`	$\mathbf{82.89}_{\pm 2.7}$	$81.74_{\pm 6.6}$	$\underline{82.61}_{\pm 3.0}$	$74.72_{\pm 4.8}$	$82.11_{\pm 1.5}$	$\mathbf{88.15}_{\pm 4.3}$	$84.00_{\pm 8.0}$	$86.46_{\pm 2.8}$	$75.15_{\pm 4.2}$	$\underline{87.97}_{\pm 3.0}$
`xlam`	$\mathbf{84.35}_{\pm 0.4}$	$74.87_{\pm 2.4}$	$\underline{83.57}_{\pm 1.4}$	$82.92_{\pm 1.5}$	$82.32_{\pm 2.2}$	$\mathbf{84.66}_{\pm 0.34}$	$80.70_{\pm 2.0}$	$83.91_{\pm 1.8}$	$83.19_{\pm 1.6}$	$\underline{84.10}_{\pm 2.1}$
Total	**253.18**	238.27	234.10	243.03	<u>247.62</u>	**274.57**	260.68	253.86	259.58	<u>267.45</u>
ACZ										
ARC-1D	$\mathbf{66.33}_{\pm 1.6}$	$\underline{61.28}_{\pm 5.5}$	$50.08_{\pm 1.0}$	$29.29_{\pm 2.6}$	$58.17_{\pm 1.2}$	$\mathbf{80.37}_{\pm 1.5}$	$\underline{77.39}_{\pm 4.7}$	$66.60_{\pm 1.1}$	$39.88_{\pm 2.6}$	$70.53_{\pm 1.3}$
Countdown	$\underline{91.83}_{\pm 0.9}$	$91.06_{\pm 0.8}$	$88.09_{\pm 1.0}$	$89.92_{\pm 0.7}$	$\mathbf{92.33}_{\pm 2.5}$	$\underline{94.48}_{\pm 1.2}$	$94.43_{\pm 0.8}$	$91.80_{\pm 0.7}$	$92.69_{\pm 2.8}$	$\mathbf{94.56}_{\pm 1.4}$
Zebra	$\underline{91.08}_{\pm 1.0}$	$\mathbf{91.12}_{\pm 1.0}$	$87.79_{\pm 1.9}$	$79.42_{\pm 2.1}$	$90.54_{\pm 1.4}$	$\mathbf{97.21}_{\pm 1.4}$	$96.68_{\pm 1.0}$	$95.47_{\pm 0.7}$	$89.25_{\pm 1.8}$	$\underline{96.91}_{\pm 1.5}$
Total	**249.24**	<u>243.46</u>	225.96	198.63	241.04	**272.06**	<u>268.50</u>	253.87	221.82	262.00

where (x_i, y_i) are the i-th state-action pair for task k in the training batch on step t, N_k is the number of task-k samples in the current training batch. We estimate $A_{\pi_{t-1}}$ using the z-score estimator used by group-based RL algorithms such as GRPO (Shao et al., 2024) and GSPO (Zheng et al., 2025).

Compared to the standard RL post-training loop, this estimation only requires an additional forward pass of the updated policy π_t to compute the importance sampling ratio $\frac{\pi_t(y_i|x_i)}{\pi_{t-1}(y_i|x_i)}$. This is a negligible overhead compared to the cost of rollout generation or backpropagation.

3.3 Curvature Estimation

What is left is to estimate the allocation-response Jacobian $g_{t,k}$. Estimating this quantity also requires a counterfactual evaluation of the policy improvement around p_t. However, by maintaining a history of task-wise improvement vectors $\{\widehat{U}_\tau\}_{\tau=1}^{t-1}$ and allocations $\{p_\tau\}_{\tau=1}^{t-1}$, we can approximate the allocation-response Jacobian $g_{t,k}$ by local regression. We opt for the Theil-Sen estimator (Theil, 1992) for its robustness to noisy $\hat{U}_{t,k}$ when the allocation $p_{t,k}$ is low.

$$\widehat{g}_{t,kj} = \operatorname*{median}_{t-H \le a < b \le t-1} \frac{\widehat{U}_{b,k} - \widehat{U}_{a,k}}{p_{b,j} - p_{a,j}}. \qquad \forall k \in [K], j \in [K]. \tag{12}$$

Additionally, to prevent stale p_t, we apply dithering with decaying strength to p_t before sampling (see Line 4, Algorithm 1). Here, p_t denotes the controller's nominal allocation, while $\tilde{p}_t$ denotes the dithered allocation actually used to sample the training batch.

4 Experiments

We evaluate PDCM on two mixtures from the **Nemo-Gym** post-training suite (NVIDIA, 2025b) and **Reasoning-Gym** (Stojanovski et al., 2025).

Nemo-Gym This mixture focuses on fundamental agentic abilities, code generation, instruction following and tool calling. We incorporate four datasets. **`code_gen`** contains competitive coding challenges from OpenCodeReasoning (Ahmad et al., 2025b) dataset. **`IF`** is an instruction following dataset based on the WildChat-1M dataset (Zhao et al., 2024) with instructions from the Open-Instruct code base (Ahmad et al., 2025a). **`xlam_fc`** (Zhang et al., 2025) contains single-turn tool-calling data in xLAM format. **`workplace`** (NVIDIA, 2025a) tests multi-step tool-calling abilities in a harness that simulates real-life logistics tasks in an office.

Reasoning-Gym This mixture evaluates reasoning capabilities through script-generated games. Our experiments include six varieties. **`ARC-1D`** is an abstract pattern-induction puzzle where the agent infers the grid transformation rule from examples. **`countdown`** is a numbers game in

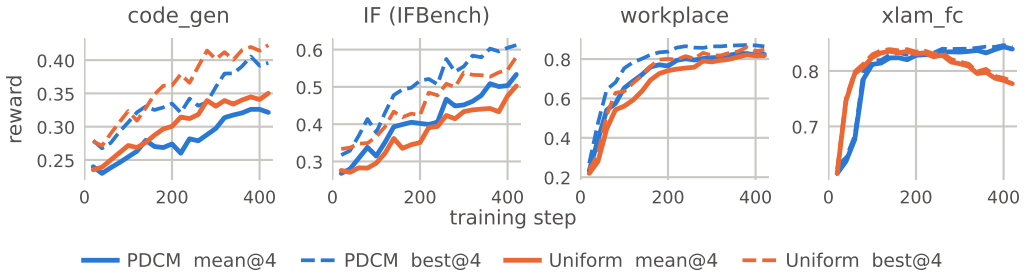


Figure 3: **Per-task validation reward on the Nemo-Gym mixture.** Qwen3-4B, PDCM (blue) against the uniform mixture (orange); solid is `mean@4`, dashed `best@4`. PDCM trails the uniform mixture early on the tasks that saturate fastest, `xlam_fc` above all, and closes the gap as training proceeds: by the end of the run it leads on `IF` and `xlam_fc` and is level on `workplace`, at a small cost on `code_gen`.

which the agent combines given numbers and a fixed set of operations to hit a target. **`zebra`** is a logic game where you use clues and deduction to match attributes across different colored houses. **`word_sorting`** is an alphabetical word-sorting task. In **`puzzle24`**, one needs to combine four numbers and basic arithmetic operations to make 24. In **`manipulate_matrix`**, one must apply given grid operations to reach the final state.

We include more details on the datasets in Appendix A.

Experimental Setup. The experiments are run using Qwen3-4B. All experiments are run in `/nothink` mode and with 4 random seeds. The main experiments are run on two H200s with a training budget of 48 hours, unless otherwise noted. All experiments are implemented with `VeRL`. For the Nemo-Gym mixture, PDCM and Aioli control the allocation quota of each dataset. On Reasoning-Gym, we find that averaging $\widehat{U}_{t,k}$ over difficulty levels erases the policy improvement signal PDCM relies on. We therefore let the allocation methods control each difficulty level directly, so a K-task Reasoning-Gym mixture is effectively a $3K$-unit allocation problem. We report the hyperparameters in Appendix C.

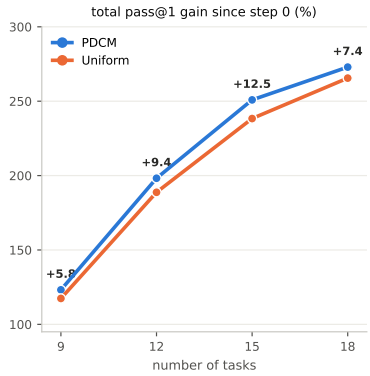
Baselines. All methods share the model, RL algorithm, data pools, and training budget, and differ only in how the per-step mixture is chosen or how the resulting updates are combined.

Uniform is a multi-task baseline that fixes the mixture at $u = 1/K$ for the whole run.

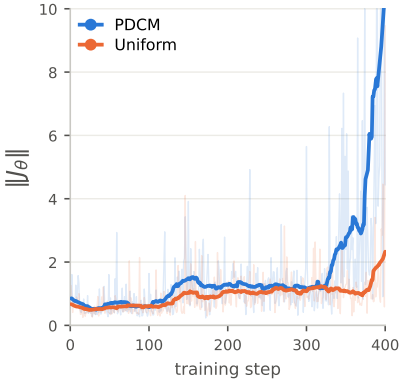
Single trains one specialist per task: K separate runs, each with $1/K$ of multi-task data budget.

PCGrad (Yu et al., 2020) holds the mixture uniform and intervenes on the update instead, projecting conflicting task gradients onto one another's normal plane. It represents the gradient-surgery family.

Aioli (Chen et al., 2025) is the closest prior allocation method. Like PDCM it adjusts the mixture online, but it does so through the parametric mixing law.



(a) Total performance gain percentage difference grows as the number of PDCM-controlled components grows from 9 to 18.



(b) Gradient norm under PDCM and the uniform mixture on the Nemo-Gym mixture.

Figure 4: Scaling and gradient behavior of PDCM.

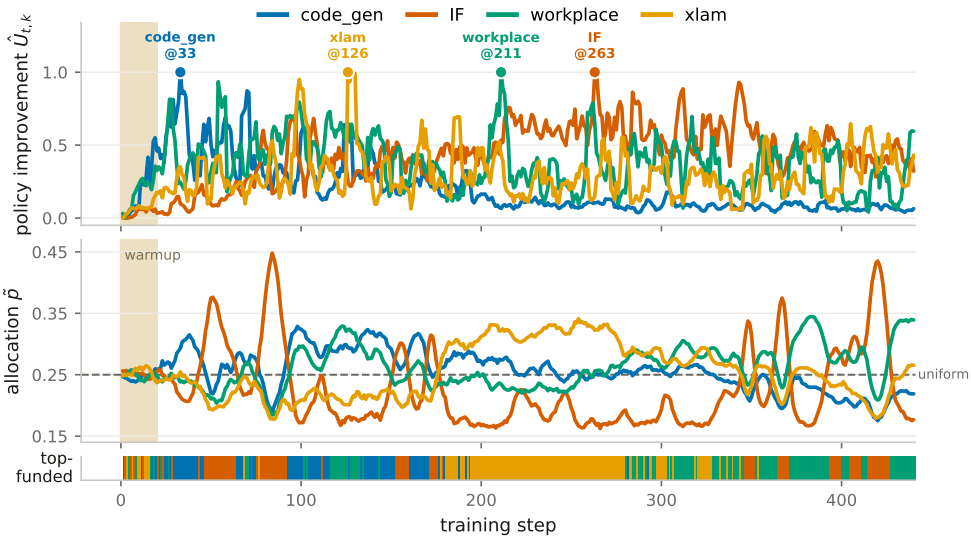


Figure 5: **PDCM allocation vs. estimated per-task improvement. (a)** The landscape of normalized per-task one-step improvement $\hat{U}_{t,k}$ demonstrates task-specific phases. **(b)** PDCM's applied allocation $\tilde{p}_t$ oscillates about the uniform $1/K$. **(c)** The top-funded task over time. Leadership rotates about every 100 steps with coarse structure that tracks (a) for `xlam` (mid), `workplace` (late), and `code_gen`.

4.1 Main Results

Table 1 reports the performance of PDCM and the baselines on the Nemo-Gym mix and a three-task mixture comprising `ARC-1D`, `countdown` and `zebra`, abbreviated as `ACZ`. PDCM attains the highest total on all four suite-metric combinations, leading the uniform mixture by 14.91 and 13.89 points on Nemo-Gym in pass@1 and pass@4, respectively, and by 5.78 and 3.56 on ACZ. Aioli is the runner-up on Nemo-Gym and the uniform mixture on ACZ. Scoring each method task by task against the uniform allocation, PDCM's largest single-task shortfall across the 14 task-metric cells is 1.5 points. Crucially, PDCM exhibits lower variance against most multi-task baselines, demonstrating that optimized batch composition induces more stable learning dynamics. In comparison, the Uniform baseline has significantly higher variance than other methods on at least one task.

We report the learning curves of the Nemo-Gym mixture in Fig. 3. Interestingly, with even allocation, the Uniform baseline's performance on `xlam_fc` is hurt by progress in other tasks, whereas PDCM's allocation allows it to recover and eventually lead on that task.

4.2 Allocation Dynamics

We study the allocation dynamics of PDCM on the 4-task Nemo-Gym mixture. As shown in Fig. 5, the allocation can be roughly broken down into three phases. Through the first 200 steps `code_gen` is most often top-funded, induced by its large improvements early in training. Between steps 200 and 300 leadership passes to `xlam`, whose measured improvement degrades when it is under-allocated, which caused PDCM to continue allocating additional budget to it. Once `xlam`'s policy improvement was able to grow without further allocation, `workplace` becomes the best-funded task for the remainder of the run around step 300.

Three of the four tasks receive a dominant phase. The exception is `IF`: its allocation proportion oscillates between 0.15 and 0.45, and these spurts of budget are enough for it to maintain a lead against the mixed baseline. Across all four, the *time-averaged* allocation remains nearly uniform. The near-uniform mean indirectly supports that the no-task-left-behind constraint of Equation (2) takes effect. A remarkable side effect of such allocation is that PDCM induces a larger learning gradient from each batch, as seen in Figure 4b.

Table 2: We compare full PDCM, PDCM with the dual term removed (PDCM$_{-\lambda}$), the Uniform baseline, and OSMD applied directly. Best per task in **bold**, second best underlined (OSMD is excluded from ranking).

	pass@1				pass@4			
Task	PDCM	PDCM$_{-\lambda}$	Uniform	OSMD	PDCM	PDCM$_{-\lambda}$	Uniform	OSMD
code_gen	32.61	32.14	**34.08**	38.97	40.55	39.37	**41.41**	46.57
IF	**53.33**	48.33	47.58	43.08	**61.21**	59.74	54.57	48.79
workplace	**82.89**	81.65	81.74	80.41	**88.15**	84.51	84.00	83.30
xlam	**84.35**	82.45	74.87	75.19	**84.66**	83.68	80.70	73.98
Total	**253.18**	244.57	238.27	237.65	**274.57**	267.30	260.68	252.64

4.3 Scaling to More Tasks

We scale PDCM control from the 9 allocation components in the ACZ mixture to 18 with the additional `word_sorting`, `puzzle24` and `manipulate_matrix`, in such order. Figure 4a shows PDCM consistently improves baseline performance with increasing components.

4.4 Ablation

We ablate the effect of Equation (2) in two ways. First, we remove the constraint entirely, reducing it to unconstrained utility maximization, efficiently solved by online stochastic mirror descent (OSMD) (Audibert et al., 2013). Second, as a soft ablation, we remove the dual control λ, producing $\mathrm{PDCM}_{-\lambda}$, where the task-wise score $s_t = G_t^T \mathbf{1}$ is driven only by the curvature. Table 2 compares both settings with the full PDCM and the uniform baseline; both ablations perform poorly.

5 Related Work

Multi-task optimization via gradient and loss manipulation. A large body of work modifies the joint update to mitigate task conflict, either by combining per-task gradients, PCGrad (Yu et al., 2020), CAGrad (Liu et al., 2021), MGDA (Sener & Koltun, 2018), and the Nash bargaining solution of Nash-MTL (Navon et al., 2022), or by adaptively reweighting losses, as in GradNorm (Chen et al., 2018), uncertainty weighting (Kendall et al., 2018), and FAMO (Liu et al., 2023), which balances progress in $O(1)$ space and time rather than storing all per-task gradients. Simpler alternatives such as random loss weighting (Lin et al., 2022) and unitary scalarization (Kurin et al., 2022) match these methods; Kurin et al. (2022) report no clear advantage over PCGrad, so we do not expect further variants to change our conclusions.

Data mixture and domain reweighting. For pre-training, domain weights are set by proxy-model or parametric methods: DoReMi (Xie et al., 2023), DoGE (Fan et al., 2024), Skill-it (Chen et al., 2023), data-mixing laws (Li et al., 2025), RegMix (Liu et al., 2025), and AutoScale (Kang et al., 2025), with online variants ODM (Albalak et al., 2023), ADO (Jiang et al., 2025), and Aioli (Chen et al., 2025). Fine-tuning studies examine SFT composition and interference (Dong et al., 2024). These fit a mixing law or bandit weights against a proxy loss. PDCM instead optimizes cumulative *policy improvement* online under a no-task-left-behind constraint. The closest prior formulation, Aioli (Chen et al., 2025), casts pre-training data mixing as the same kind of optimization problem but closes it with a power mixing law tailored to pre-training dynamics. PDCM assumes only concavity of the task-improvement response and uses local first-order information without over-prescribing the general shape of the learning curve.

6 Conclusion

We cast multi-task post-training as an online allocation problem and solve it with PDCM, a primal–dual method that turns measured per-task improvement into a mixture without assuming a parametric mixing law. On Nemo-Gym and Reasoning-Gym, PDCM leads on both suites tested while never trailing the uniform mixture by more than 1.5 points on any individual task, a combination that none of the gradient-surgery, specialist, and mixing-law baselines achieve. It also scales to 18 allocation components with a consistent margin. PDCM's main limitation is the curvature estimate, which is noisy over short horizons; a more robust estimator is left to future work.

AI USE STATEMENT

In this work, we used generative AI tools to assist with writing, specifically to draft and revise prose, to help with LaTeX formatting, and to proof-read the statements and proofs in the appendix. The research idea, the problem formulation, the algorithm, the theoretical results and their proofs, the experimental design, and the experiments themselves are the authors' own. All AI-assisted text was reviewed and edited by the authors, the proofs were verified by the authors independently of the AI proof-reading pass, and every quantitative claim was checked by an author against the underlying experimental results. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Wasi Uddin Ahmad, Aleksander Ficek, Mehrzad Samadi, Jocelyn Huang, Vahid Noroozi, Somshubra Majumdar, and Boris Ginsburg. Opencodeinstruct: A large-scale instruction tuning dataset for code llms, 2025a. URL https://arxiv.org/abs/2504.04030.

Wasi Uddin Ahmad, Sean Narenthiran, Somshubra Majumdar, Aleksander Ficek, Siddhartha Jain, Jocelyn Huang, Vahid Noroozi, and Boris Ginsburg. Opencodereasoning: Advancing data distillation for competitive coding, 2025b. URL https://arxiv.org/abs/2504.01943.

Alon Albalak, Liangming Pan, Colin Raffel, and William Yang Wang. Efficient online data mixing for language model pre-training, 2023. URL https://arxiv.org/abs/2312.02406.

Jean-Yves Audibert, Sébastien Bubeck, and Gábor Lugosi. Regret in online combinatorial optimization, 2013. URL https://arxiv.org/abs/1204.4710.

Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Oper. Res. Lett.*, 31:167–175, 2003. URL https://api.semanticscholar.org/CorpusID:7036108.

Mayee Chen, Nicholas Roberts, Kush Bhatia, Jue WANG, Ce Zhang, Frederic Sala, and Christopher Ré. Skill-it! a data-driven skills framework for understanding and training language models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 36000–36040. Curran Associates, Inc., 2023. doi: 10.52202/075280-1562. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/70b8505ac79e3e131756f793cd80eb8d-Paper-Conference.pdf.

Mayee F Chen, Michael Y. Hu, Nicholas Lourie, Kyunghyun Cho, and Christopher Re. Aioli: A unified optimization framework for language model data mixing. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=sZGZJhaNSe.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 794–803. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/chen18a.html.

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. How abilities in large language models are affected by supervised fine-tuning data composition. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 177–198, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.12. URL https://aclanthology.org/2024.acl-long.12/.

Simin Fan, Matteo Pagliardini, and Martin Jaggi. DOGE: Domain reweighting with generalization estimation. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International*

Conference on Machine Learning, volume 235 of *Proceedings of Machine Learning Research*, pp. 12895–12915. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/fan24e.html.

Zhengyao Gu, Jonathan Light, Raul Astudillo, Ziyu Ye, Langzhou He, Henry Peng Zou, Wei Cheng, Santiago Paternain, Philip S. Yu, and Yisong Yue. Actor-curator: Co-adaptive curriculum learning via policy-improvement bandits for rl post-training, 2026. URL https://arxiv.org/abs/2602.20532.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Rodolphe Jenatton, Jim Huang, and Cedric Archambeau. Adaptive algorithms for online convex optimization with long-term constraints. In Maria Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 402–411, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/jenatton16.html.

Yiding Jiang, Allan Zhou, Zhili Feng, Sadhika Malladi, and Zico Kolter. Adaptive data optimization: Dynamic sample selection with scaling laws. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 58014–58037, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/923285deb805c3e14e1aeebc9854d644-Paper-Conference.pdf.

Sham M. Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, 2002. URL https://api.semanticscholar.org/CorpusID:31442909.

Feiyang Kang, Yifan Sun, Bingbing Wen, Si Chen, Dawn Song, Rafid Mahmood, and Ruoxi Jia. Autoscale: Scale-aware data mixing for pre-training LLMs. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=rujwIvjooA.

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Vitaly Kurin, Alessandro De Palma, Ilya Kostrikov, Shimon Whiteson, and Pawan K Mudigonda. In defense of the unitary scalarization for deep multi-task learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 12169–12183. Curran Associates, Inc., 2022. doi: 10.52202/068431-0884. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/4f301ae934f396086bfefd1139039dbd-Paper-Conference.pdf.

Yuan Li, Zhengzhong Liu, and Eric Xing. Data mixing optimization for supervised fine-tuning of large language models. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 35419–35437. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/li25bh.html.

Baijiong Lin, Feiyang Ye, Yu Zhang, and Ivor Tsang. Reasonable effectiveness of random weighting: A litmus test for multi-task learning. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=jjtFD8A1Wx.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 18878–18890. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/9d27fdf2477ffbff837d73ef7ae23db9-Paper.pdf.

Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. Famo: Fast adaptive multitask optimization. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 57226–57243. Curran Associates, Inc., 2023. doi: 10.52202/075280-2500. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/b2fe1ee8d936ac08dd26f2ff58986c8f-Paper-Conference.pdf.

Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. Regmix: Data mixture as regression for language model pre-training. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 38305–38339, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/5f67d864aae6115374fed7beddd119e0-Paper-Conference.pdf.

Mehrdad Mahdavi, Rong Jin, and Tianbao Yang. Trading regret for efficiency: Online convex optimization with long term constraints. *Journal of Machine Learning Research*, 13(81):2503–2528, 2012. URL http://jmlr.org/papers/v13/mahdavi12a.html.

Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 16428–16446. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/navon22a.html.

NVIDIA. Nemo gym: An open source library for scaling reinforcement learning environments for llm. https://github.com/NVIDIA-NeMo/Gym, 2025a. GitHub repository.

NVIDIA. Nemotron-post-training-v3. https://huggingface.co/collections/nvidia/nemotron-post-training-v3, 2025b. Hugging Face dataset collection, accessed 2026-09-19.

NVIDIA, :, Aaron Blakeman, Aaron Grattafiori, Aarti Basant, Abhibha Gupta, Abhinav Khattar, Adi Renduchintala, Aditya Vavre, Akanksha Shukla, Akhiad Bercovich, Aleksander Ficek, Aleksandr Shaposhnikov, Alex Kondratenko, Alexander Bukharin, Alexandre Milesi, Ali Taghibakhshi, Alisa Liu, Amelia Barton, Ameya Sunil Mahabaleshwarkar, Amir Klein, Amit Zuker, Amnon Geifman, Amy Shen, Anahita Bhiwandiwalla, Andrew Tao, Ann Guan, Anubhav Mandarwal, Arham

Mehta, Ashwath Aithal, Ashwin Poojary, Asif Ahamed, Asma Kuriparambil Thekkumpate, Ayush Dattagupta, Banghua Zhu, Bardiya Sadeghi, Barnaby Simkin, Ben Lanir, Benedikt Schifferer, Besmira Nushi, Bilal Kartal, Bita Darvish Rouhani, Boris Ginsburg, Brandon Norick, Brandon Soubasis, Branislav Kisacanin, Brian Yu, Bryan Catanzaro, Carlo del Mundo, Chantal Hwang, Charles Wang, Cheng-Ping Hsieh, Chenghao Zhang, Chenhan Yu, Chetan Mungekar, Chintan Patel, Chris Alexiuk, Christopher Parisien, Collin Neale, Damon Mosk-Aoyama, Dan Su, Dane Corneil, Daniel Afrimi, Daniel Rohrer, Daniel Serebrenik, Daria Gitman, Daria Levy, Darko Stosic, David Mosallanezhad, Deepak Narayanan, Dhruv Nathawani, Dima Rekesh, Dina Yared, Divyanshu Kakwani, Dong Ahn, Duncan Riach, Dusan Stosic, Edgar Minasyan, Edward Lin, Eileen Long, Eileen Peters Long, Elena Lantz, Ellie Evans, Elliott Ning, Eric Chung, Eric Harper, Eric Tramel, Erick Galinkin, Erik Pounds, Evan Briones, Evelina Bakhturina, Faisal Ladhak, Fay Wang, Fei Jia, Felipe Soares, Feng Chen, Ferenc Galko, Frankie Siino, Gal Hubara Agam, Ganesh Ajjanagadde, Gantavya Bhatt, Gargi Prasad, George Armstrong, Gerald Shen, Gorkem Batmaz, Grigor Nalbandyan, Haifeng Qian, Harsh Sharma, Hayley Ross, Helen Ngo, Herman Sahota, Hexin Wang, Himanshu Soni, Hiren Upadhyay, Huizi Mao, Huy C Nguyen, Huy Q Nguyen, Iain Cunningham, Ido Shahaf, Igor Gitman, Ilya Loshchilov, Ivan Moshkov, Izzy Putterman, Jan Kautz, Jane Polak Scowcroft, Jared Casper, Jatin Mitra, Jeffrey Glick, Jenny Chen, Jesse Oliver, Jian Zhang, Jiaqi Zeng, Jie Lou, Jimmy Zhang, Jining Huang, Joey Conway, Joey Guman, John Kamalu, Johnny Greco, Jonathan Cohen, Joseph Jennings, Joyjit Daw, Julien Veron Vialard, Junkeun Yi, Jupinder Parmar, Kai Xu, Kan Zhu, Kari Briski, Katherine Cheung, Katherine Luna, Keshav Santhanam, Kevin Shih, Kezhi Kong, Khushi Bhardwaj, Krishna C. Puvvada, Krzysztof Pawelec, Kumar Anik, Lawrence McAfee, Laya Sleiman, Leon Derczynski, Li Ding, Lucas Liebenwein, Luis Vega, Maanu Grover, Maarten Van Segbroeck, Maer Rodrigues de Melo, Makesh Narsimhan Sreedhar, Manoj Kilaru, Maor Ashkenazi, Marc Romeijn, Mark Cai, Markus Kliegl, Maryam Moosaei, Matvei Novikov, Mehrzad Samadi, Melissa Corpuz, Mengru Wang, Meredith Price, Michael Boone, Michael Evans, Miguel Martinez, Mike Chrzanowski, Mohammad Shoeybi, Mostofa Patwary, Nabin Mulepati, Natalie Hereth, Nave Assaf, Negar Habibi, Neta Zmora, Netanel Haber, Nicola Sessions, Nidhi Bhatia, Nikhil Jukar, Nikki Pope, Nikolai Ludwig, Nima Tajbakhsh, Nirmal Juluru, Oleksii Hrinchuk, Oleksii Kuchaiev, Olivier Delalleau, Oluwatobi Olabiyi, Omer Ullman Argov, Ouye Xie, Parth Chadha, Pasha Shamis, Pavlo Molchanov, Pawel Morkisz, Peter Dykas, Peter Jin, Pinky Xu, Piotr Januszewski, Pranav Prashant Thombre, Prasoon Varshney, Pritam Gundecha, Qing Miao, Rabeeh Karimi Mahabadi, Ran El-Yaniv, Ran Zilberstein, Rasoul Shafipour, Rich Harang, Rick Izzo, Rima Shahbazyan, Rishabh Garg, Ritika Borkar, Ritu Gala, Riyad Islam, Roger Waleffe, Rohit Watve, Roi Koren, Ruoxi Zhang, Russell J. Hewett, Ryan Prenger, Ryan Timbrook, Sadegh Mahdavi, Sahil Modi, Samuel Kriman, Sanjay Kariyappa, Sanjeev Satheesh, Saori Kaji, Satish Pasumarthi, Sean Narentharen, Sean Narenthiran, Seonmyeong Bak, Sergey Kashirsky, Seth Poulos, Shahar Mor, Shanmugam Ramasamy, Shantanu Acharya, Shaona Ghosh, Sharath Turuvekere Sreenivas, Shelby Thomas, Shiqing Fan, Shreya Gopal, Shrimai Prabhumoye, Shubham Pachori, Shubham Toshniwal, Shuoyang Ding, Siddharth Singh, Simeng Sun, Smita Ithape, Somshubra Majumdar, Soumye Singhal, Stefania Alborghetti, Stephen Ge, Sugam Dipak Devare, Sumeet Kumar Barua, Suseella Panguluri, Suyog Gupta, Sweta Priyadarshi, Syeda Nahida Akter, Tan Bui, Teodor-Dumitru Ene, Terry Kong, Thanh Do, Tijmen Blankevoort, Tom Balough, Tomer Asida, Tomer Bar Natan, Tugrul Konuk, Twinkle Vashishth, Udi Karpas, Ushnish De, Vahid Noorozi, Vahid Noroozi, Venkat Srinivasan, Venmugil Elango, Vijay Korthikanti, Vitaly Kurin, Vitaly Lavrukhin, Wanli Jiang, Wasi Uddin Ahmad, Wei Du, Wei Ping, Wenfei Zhou, Will Jennings, William Zhang, Wojciech Prazuch, Xiaowei Ren, Yashaswi Karnati, Yejin Choi, Yev Meyer, Yi-Fu Wu, Yian Zhang, Ying Lin, Yonatan Geifman, Yonggan Fu, Yoshi Subara, Yoshi Suhara, Yubo Gao, Zach Moshe, Zhen Dong, Zihan Liu, Zijia Chen, and Zijie Yan. Nemotron 3 nano: Open, efficient mixture-of-experts hybrid mamba-transformer model for agentic reasoning, 2025. URL https://arxiv.org/abs/2512.20848.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. doi: 10.52202/

068431-2011. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Doina Precup, Richard S. Sutton, and Satinder P. Singh. Eligibility traces for off-policy policy evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pp. 759–766, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1558607072.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hanna Hajishirzi. Generalizing verifiable instruction following. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc., 2025. doi: 10.52202/085713-1645. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/46499a0622ecf568b72d17b61e45dbd5-Paper-Datasets_and_Benchmarks_Track.pdf.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/432aca3a1e345e339f35a30c8f65edce-Paper.pdf.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Zafir Stojanovski, Oliver Stanley, Joe Sharratt, Richard Jones, Abdulhakeem Adefioye, Jean Kaddour, and Andreas Köpf. Reasoning gym: Reasoning environments for reinforcement learning with verifiable rewards. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc., 2025. doi: 10.52202/085713-1913. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/528d56195a2c77c808494c86fa7c77ad-Paper-Datasets_and_Benchmarks_Track.pdf.

Henri Theil. *A Rank-Invariant Method of Linear and Polynomial Regression Analysis*, pp. 345–381. Springer Netherlands, Dordrecht, 1992. ISBN 978-94-011-2546-8. doi: 10.1007/978-94-011-2546-8_20. URL https://doi.org/10.1007/978-94-011-2546-8_20.

Yinjie Wang, Xuyang Chen, Xiaolong Jin, Mengdi Wang, and Ling Yang. Openclaw-rl: Train any agent simply by talking, 2026. URL https://arxiv.org/abs/2603.10165.

Runzhe Wu, Ankur Samanta, Ayush Jain, Scott Fujimoto, Jeongyeol Kwon, Ben Kretzu, Youliang Yu, Kaveh Hassani, Boris Vidolov, and Yonathan Efroni. Imbalanced gradients in RL post-training of multi-task LLMs. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 3137–3150, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-386-9. doi: 10.18653/v1/2026.findings-eacl.164. URL https://aclanthology.org/2026.findings-eacl.164/.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. Doremi: Optimizing data mixtures speeds up language model pretraining. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 69798–69818. Curran Associates, Inc., 2023. doi: 10.52202/

075280-3059. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/dcba6be91359358c2355cd920da3fcbd-Paper-Conference.pdf.

Jiasheng Ye, Peiju Liu, Tianxiang Sun, Jun Zhan, Yunhua Zhou, and Xipeng Qiu. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 82263–82287, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/cc84bfabe6389d8883fc2071c848f62a-Paper-Conference.pdf.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 5824–5836. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/3fe78a8acf5fda99de95303940a2420c-Paper.pdf.

Baoqing Yue, Zihan Zhu, Yutong Han, Brian Fan, Qian Sun, Jichen Feng, Hufei Yang, Yifan Zhang, and Mengdi Wang. Interactive benchmarks, 2026. URL https://arxiv.org/abs/2603.04737.

Jianguo Zhang, Tian Lan, Ming Zhu, Zuxin Liu, Thai Hoang, Shirley Kokane, Weiran Yao, Juntao Tan, Zhiwei Liu, Yihao Feng, Juan Carlos Niebles, Shelby Heinecke, Huan Wang, Silvio Savarese, and Caiming Xiong. xLAM: A family of large action models to empower AI agent systems. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 11583–11597, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.578. URL https://aclanthology.org/2025.naacl-long.578/.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. Wildchat: 1m chatGPT interaction logs in the wild. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Bl8u7ZRlbM.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization, 2025. URL https://arxiv.org/abs/2507.18071.

APPENDIX CONTENTS

A Data **17**
A.1 Reasoning-Gym Data Generation 17

B Concavity Test **18**

C Hyperparameters **19**

D Diagonal-Jacobian PDCM **20**

E Theory **21**
E.1 Problem formulation and idealized full-Jacobian PDCM 21
E.2 Constrained no-regret guarantee 22
E.3 Dual-weighted marginal-utility matching 24

A DATA

Nemo-Gym. (NVIDIA, 2025a) We selected four heterogeneous tasks served from the NeMo-Gym servers, some requiring tool use or multi-turn capabilities and others not. **`IF`** draws its training prompts from the WildChat-1M dataset (Zhao et al., 2024) with instructions from the Open-Instruct code base (Ahmad et al., 2025a), and is evaluated on the held-out IF-Bench (Pyatkin et al., 2025). **`code_gen`** contains competitive coding challenges from NVIDIA's OpenCodeReasoning (Ahmad et al., 2025b) dataset. This dataset is served with Nemo-Gym's single-turn verification engine `code_gen`. **`xlam_fc`** (Zhang et al., 2025) is an API/function calling dataset in xLAM format. **`workplace`** (NVIDIA, 2025a) is a tool-use multi-step agentic environment that tests the agent's ability to execute tasks in a workplace setting such as calendar scheduling or email management. Tab. 3 lists the per-task train and test sizes.

Reasoning-Gym. (Stojanovski et al., 2025) Our reasoning mixture is procedurally generated with scripts offered by Reasoning-Gym (Stojanovski et al., 2025). Each task is instantiated at three difficulty levels (easy, medium, and hard) with 30000 training and 600 test problems. We describe the individual tasks below. **`ARC-1D`** is an abstract pattern-induction puzzle where the agent infers the grid transformation rule from examples. **`countdown`** is a numbers game in which the agent combines given numbers and a fixed set of operations to hit a target. **`zebra`** is a logic game where you use clues and deduction to match attributes across different colored houses. **`word_sorting`** is an alphabetical word-sorting task. In **`puzzle24`**, one needs to combine four numbers and basic arithmetic operations to make 24. In **`manipulate_matrix`**, one must apply given grid operations in sequence to reach a given state. App. A.1 reports the complete generation configurations.

Task	Capability	# Train	# Test	Test set
instruction_following	constraint following	45891	300	**IF-Bench** (held out)
code_gen	program synthesis	23971	322	in-distribution
xlam_fc	function / tool calling	59000	1000	in-distribution
workplace_assistant	multi-turn assistant	1255	545	in-distribution

Table 3: Agentic multi-task suite. Instruction following is trained on a constraint-following pool and evaluated on the held-out IF-Bench benchmark; the other three tasks use in-distribution test splits. Sizes are the number of served problems.

A.1 REASONING-GYM DATA GENERATION

All six reasoning tasks are generated procedurally with Reasoning-Gym (Stojanovski et al., 2025) using a single builder that fixes the prompt template, answer schema, difficulty schedule, and random seeding, so every instance is fully reproducible and paired with a deterministic, rule-based verifier.

Difficulty control. Difficulty is set at generation time through each task's generator configuration rather than by post-hoc filtering. All six tasks are produced at three levels—*easy*, *medium*, and *hard*. The controlling knob per task is:

- **`word_sorting`**: the Reasoning-Gym `WordSortingCurriculum` global level ($0/1/2$ for easy/medium/hard) under upper-bound range semantics, which widens the number-of-words and word-length ranges as difficulty increases.
- **`manipulate_matrix`**: a fixed 10×10 integer grid with the number of chained transformations set to $1/3/5$ for easy/medium/hard. We scale difficulty by transformation *depth* rather than grid size, because larger grids inflate the target answer beyond the response-length budget.
- **`puzzle24`**: the generator default. The operand values, $[1, 10]$, are evenly split between three difficulty levels.
- **`arc`**, **`countdown`**, **`zebra`**: These three families are drawn from Reasoning-Gym's procedural generators under their built-in difficulty *curricula*, whose discrete levels the easy/medium/hard tiers index. *Countdown* presents a multiset of source integers to

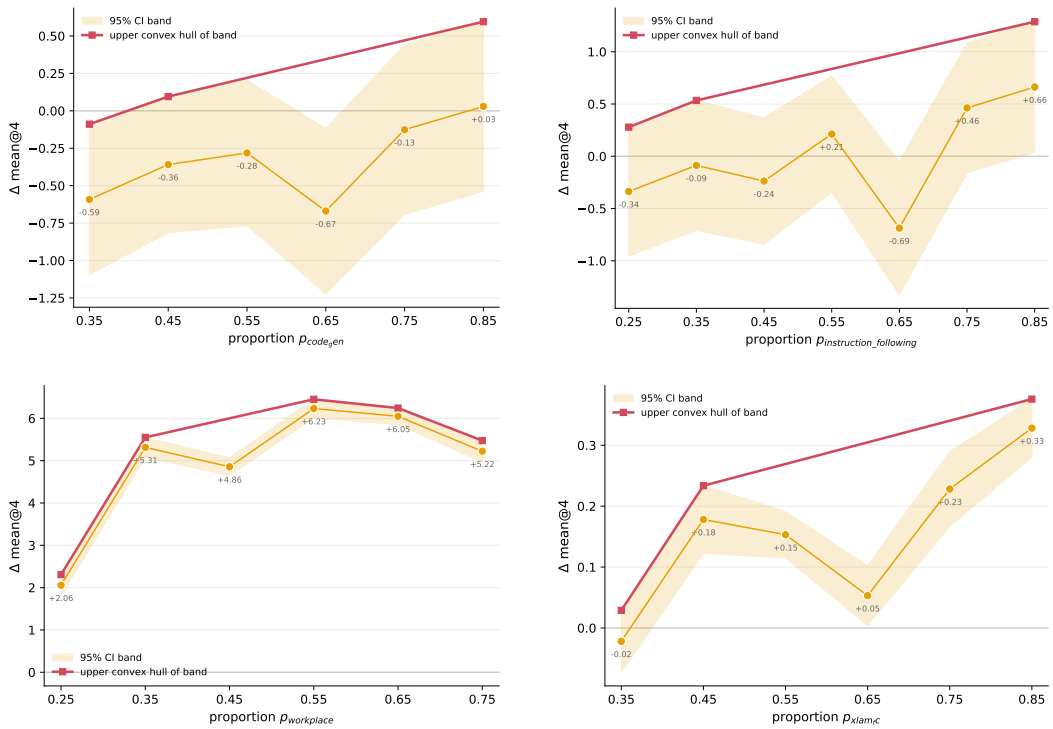


Figure 6: **Allocation response of each Nemo-Gym task.** Change in mean@4 after 10 steps from a common step-80 checkpoint, as the probed task's share of the batch is swept from 0.25 to 0.85 and the remaining mass is split evenly among the other three tasks. Shaded region is the 95% confidence band; the upper curve is its upper convex hull, the tightest concave envelope the measurements admit.

be combined with $+, -, \times, \div$ into a target. Its `CountdownCurriculum` sets difficulty through three range attributes: the operand count (level ladder $\{3, 6, 9, 12, 15\}$), the target magnitude ($\{100, 500, 1000, 5000, 10000\}$), and the source-value magnitude ($\{1, 100, 200, 300\}$). Harder tiers grant more operands and larger targets and values. *Zebra* is an Einstein-style constraint-satisfaction puzzle. Its `ZebraPuzzleCurriculum` scales two attributes, the number of people and the number of characteristics, each over $\{2, \ldots, 7\}$, enlarging the constraint grid as difficulty rises. *ARC* draws ARC-AGI-1 grid-transformation tasks and raises difficulty through two augmentation-weight distributions: a rotation distribution over $\{\text{identity}, 90°, 180°, 270°\}$, shifted from identity-dominant ($[0.3, 0.2, 0.3, 0.2]$) toward rotation-dominant ($[0.0, 0.4, 0.2, 0.4]$), and a mirror distribution over $\{\text{identity}, \text{horizontal}, \text{vertical}, \text{diagonal}, \text{counter-diagonal}\}$, shifted from simple toward diagonal mirrors ($[0.3, 0.3, 0.2, 0.1, 0.1] \rightarrow [0.05, 0.05, 0.1, 0.4, 0.4]$). All three inherit the per-instance seeding, three-tier schedule, and `\boxed{}` answer schema of the other tasks.

B Concavity Test

Assumption 1 asks that task improvement responds concavely to allocation, and we check this directly rather than assume it. Starting from a common step-80 checkpoint of the four-task Nemo-Gym run, we probe one task at a time: its share is held at one of seven values from 0.25 to 0.85 in steps of 0.10, the remaining budget is split evenly among the other three tasks, and training continues for 10 steps. Each probe is scored by the change in mean@4 from the checkpoint, and repeated sampling gives the 95% confidence band shown in Figure 6, against which we run an upper-convex-hull test for concavity. All four tasks display a concave trend over the probed range, which supports Assumption 1 as a working assumption for this setting. The probe covers one checkpoint and one mixture, so it speaks to local rather than global concavity of the task-improvement surface.

Table 4: Hyperparameters for the two PDCM runs: the Nemo-Gym run with the full-Jacobian sampler, and the Reasoning-Gym run with the scalar sampler over difficulty tiers.

Hyperparameter	Nemo-Gym (Jacobian)	Reasoning-Gym
Primal step η_p	0.1	0.3
Dual step η_λ	0.01	0.05
Window H	10	10
Dither ν_t (init/floor)	0.20 / 0.10	0.20 / 0.10
Dither anneal steps	400	400
Lag budget ε	0.1	0.1
Warmup steps	20	40
Optimization		
Learning rate	1e-6	1e-6
LR warmup steps	10	10
Weight decay	0.1	0.1
Grad clip	1.0	1.0
Train batch size	128	256
PPO mini-batch	64	64
Total training steps	600	600
Sequence / rollout		
Max prompt / response len	8192 / 8192	2048 / 4096
vLLM max-model-len	24576	6144

C HYPERPARAMETERS

Table 4 reports the hyperparameters for our two PDCM runs: the Nemo-Gym run with the full-Jacobian sampler and the `ACZPWM` run with the scalar sampler over difficulty tiers. Both use Qwen3-4B with GSPO, learning rate 10^{-6}, and $n{=}8$ rollouts per prompt. They differ mainly in the number of controller arms ($K{=}4$ domains vs. $K{=}18$ difficulty tiers), and the sequence and batch budget of the two task suites. Full settings are given in the table.

Table 5: Full versus diagonal Jacobian in PDCM, on the four Nemo-Gym tasks. PDCM estimates the full allocation-response Jacobian G_t; $\text{PDCM}_{\text{diag}}$ keeps only its diagonal, treating each task's improvement as a univariate function $\widehat{U}_{t,k}(p_{t,k})$ of its own share (Section D); Uniform is the even-split baseline. Best per task in **bold**, second best underlined.

	pass@1			pass@4		
Task	PDCM	$\text{PDCM}_{\text{diag}}$	Uniform	PDCM	$\text{PDCM}_{\text{diag}}$	Uniform
code_gen	32.61	**35.09**	34.08	40.55	**43.50**	41.41
IF	53.33	**53.92**	47.58	**61.21**	60.38	54.57
workplace	**82.89**	79.86	81.74	**88.15**	83.35	84.00
xlam	**84.35**	83.95	74.87	**84.66**	84.28	80.70
Total	**253.18**	252.82	238.27	**274.57**	271.51	260.68

D DIAGONAL-JACOBIAN PDCM

Algorithm 1 estimates the full allocation-response Jacobian G_t, whose entry $g_{t,kj}$ records how task k's improvement responds to task j's share of the batch. A simpler variant keeps only the diagonal. The restriction amounts to assuming that each task's improvement depends on the allocation only through that task's own share,

$$U_{t,k}(p) \;=\; U_{t,k}(p_k), \qquad k \in [K], \tag{13}$$

so the aggregate utility $U_t(p) = \sum_{k=1}^{K} U_{t,k}(p_k)$ is separable and each $U_{t,k}$ is a *univariate* function of p_k alone. This is exactly the object the estimator of Equation (11) measures: $\widehat{U}_{t,k}$ is read off the rollouts that task k received under its own share $p_{t,k}$, so the empirical quantity the controller actually observes is $\widehat{U}_{t,k}(p_{t,k})$, a scalar response to a scalar input. Under Equation (13) the supergradient of Assumption 1 has a single nonzero coordinate, $g_{t,k} = U'_{t,k}(p_{t,k})\, e_k$, and G_t collapses to $\text{diag}(g_{t,11}, \ldots, g_{t,KK})$ with $g_{t,kk} = U'_{t,k}(p_{t,k})$.

Three parts of the algorithm simplify accordingly. The score of Equation (8) loses its cross terms and becomes coordinatewise,

$$s_{t,k} \;=\; \big(1 + \lambda_{t,k}\big)\, g_{t,kk}, \tag{14}$$

so each task's score is its own marginal utility amplified by its own lag pressure, and budget reaches a lagging task only by being withdrawn from the others through the simplex rather than by exploiting cross-task transfer. The lag certificate reduces from an inner product to a product of scalars, $\langle g_{t,k}, u - p_t \rangle = g_{t,kk}\,(u - p_{t,k})$, and the dual update of Equation (7) follows unchanged. Finally, curvature estimation (Section 3.3) reduces from K^2 entries to K: the Theil–Sen regression of Equation (12) is run once per task over the pairs $\{(p_{\tau,k}, \widehat{U}_{\tau,k})\}_{\tau=t-H}^{t-1}$, and needs only variation in a task's own share rather than independent variation in every coordinate of p.

The trade-off is that the diagonal variant cannot represent transfer: if training on task j raises task k's return through shared parameters, that effect is invisible to Equation (14) and is absorbed into the drift of $U_{t,k}$ over rounds. What it buys is conditioning. Identifying the off-diagonal entries requires the allocation to vary independently along every coordinate, which the controller does not deliberately produce, and the number of entries grows as K^2 while each task's share shrinks as $1/K$ — the same pressure discussed in Section 4.3. The diagonal entries, by contrast, are identified by precisely the variation the primal update generates on its own. The probe of Section B measures this univariate response directly, sweeping $p_{t,k}$ with the remaining mass held even, and is therefore a test of Assumption 1 in exactly the diagonal form used by Theorem 1.

Table 5 compares the two variants on the four Nemo-Gym tasks. Both clear the uniform baseline comfortably, and their totals are close: the diagonal variant trails the full Jacobian by 0.36 points at pass@1 and 3.06 at pass@4. The per-task pattern is more informative than the totals. $\texttt{PDCM}_{\text{diag}}$ is the stronger of the two on `code_gen` at both metrics and edges ahead on `IF` at pass@1, but it gives up 3.03 points on `workplace` at pass@1 and 4.80 at pass@4, the task whose gains the full Jacobian attributes most strongly to allocation elsewhere in the mixture. Discarding the off-diagonal entries therefore costs little in aggregate while removing the mechanism by which one task's budget is credited with another's improvement.

E THEORY

This appendix gives a self-contained analysis of the full-Jacobian version of PDCM used in Algorithm 1. The analysis isolates the online allocation mechanism. At each round it treats the allocation-response Jacobian as available exactly.

E.1 PROBLEM FORMULATION AND IDEALIZED FULL-JACOBIAN PDCM

We consider $K \geq 2$ tasks and the probability simplex

$$\Delta_K \coloneqq \left\{p \in \mathbb{R}_+^K : \mathbf{1}^\top p = 1\right\}, \qquad u \coloneqq \frac{1}{K}\mathbf{1}. \tag{15}$$

At round t, let

$$U_t(p) \coloneqq [U_{t,1}(p) \quad \cdots \quad U_{t,K}(p)]^\top \in \mathbb{R}^K \tag{16}$$

denote the vector of expected one-step policy improvements that would result from applying the round-t update under allocation $p \in \Delta_K$. The aggregate improvement is

$$F_t(p) \coloneqq \mathbf{1}^\top U_t(p) = \sum_{k=1}^{K} U_{t,k}(p). \tag{17}$$

Because the policy state changes with training, the functions U_t may be nonstationary in t.

We restate here the structural condition assumed in the main text. Concavity on all of Δ_K, rather than along a single reference direction, is what allows PDCM to be compared with an arbitrary fixed allocation in the regret statement.

Assumption 2 (Concave allocation response; restatement of Assumption 1)**.** *For every round t and task k, $U_{t,k}$ is differentiable and concave on Δ_K. Equivalently, for every $p, q \in \Delta_K$ and $\alpha \in [0,1]$,*

$$U_{t,k}((1-\alpha)p + \alpha q) \geq (1-\alpha)U_{t,k}(p) + \alpha U_{t,k}(q). \tag{18}$$

Assumption 2 implies the usual first-order upper bound for a concave function.

Lemma 1 (Tangent upper bound)**.** *Under Assumption 2, for every t, k and $p, q \in \Delta_K$,*

$$U_{t,k}(q) - U_{t,k}(p) \leq \langle \nabla U_{t,k}(p), q - p \rangle. \tag{19}$$

Proof. For $\alpha \in (0,1]$, concavity gives

$$U_{t,k}(p + \alpha(q-p)) - U_{t,k}(p) \geq \alpha\big(U_{t,k}(q) - U_{t,k}(p)\big). \tag{20}$$

Divide by α and let $\alpha \downarrow 0$. Differentiability yields equation 19. □

At the played allocation p_t, define the allocation-response Jacobian

$$G_t \coloneqq \nabla_p U_t(p_t) = \begin{bmatrix} g_{t,1}^\top \\ \vdots \\ g_{t,K}^\top \end{bmatrix} \in \mathbb{R}^{K \times K}, \qquad g_{t,k} \coloneqq \nabla_p U_{t,k}(p_t). \tag{21}$$

Thus $[G_t]_{kj} = \partial U_{t,k}(p_t)/\partial p_j$ measures the local effect of allocating additional mass to training task j on the improvement of evaluation task k. In particular,

$$\nabla F_t(p_t) = G_t^\top \mathbf{1}. \tag{22}$$

For task k, let $\epsilon_{t,k} \geq 0$ denote the lag budget assigned to round t. For a total cumulative budget ϵ_k, one may choose any schedule satisfying $\sum_{t=1}^{T} \epsilon_{t,k} = \epsilon_k$, for example $\epsilon_{t,k} = \epsilon_k/T$. Define the true one-step counterfactual lag residual

$$c_{t,k}(p_t) \coloneqq U_{t,k}(u) - U_{t,k}(p_t) - \epsilon_{t,k}. \tag{23}$$

Applying Lemma 1 with $p = p_t$ and $q = u$ gives

$$c_{t,k}(p_t) \leq \langle g_{t,k}, u - p_t \rangle - \epsilon_{t,k}. \tag{24}$$

Therefore the first-order quantity on the right is a conservative certificate of the unobserved uniform-allocation lag.

For the online analysis, extend this certificate affinely to an arbitrary comparator $p \in \Delta_K$:

$$h_t(p) \coloneqq G_t(u-p) - \epsilon_t, \qquad \epsilon_t \coloneqq (\epsilon_{t,1}, \dots, \epsilon_{t,K})^\top. \tag{25}$$

equation 24 is equivalently

$$c_t(p_t) \le h_t(p_t) \qquad \text{coordinatewise.} \tag{26}$$

For horizon T, define the fixed first-order-safe comparator set

$$\mathcal{P}_T \coloneqq \{p \in \Delta_K : h_t(p) \le 0 \text{ coordinatewise for every } t \le T\}. \tag{27}$$

This set is always nonempty because $h_t(u) = -\epsilon_t \le 0$.

Let $\lambda_t \in \mathbb{R}_+^K$ be the task-wise dual variables. Dropping terms independent of p from the linearized round-t Lagrangian gives the primal score

$$s_t \coloneqq G_t^\top(\mathbf{1} + \lambda_t). \tag{28}$$

The idealized full-Jacobian PDCM recursion is

$$p_{t+1} \coloneqq \arg\max_{p \in \Delta_K} \{\eta_p \langle s_t, p\rangle - D_{\mathrm{KL}}(p \| p_t)\}, \tag{29}$$

$$\lambda_{t+1} \coloneqq [\lambda_t + \eta_\lambda h_t(p_t)]_+, \tag{30}$$

with $p_1 = u$ and $\lambda_1 = 0$. Here $[\cdot]_+$ denotes coordinatewise projection onto $\mathbb{R}_+^K$. The primal step has the closed form

$$p_{t+1,j} = \frac{p_{t,j}\exp(\eta_p s_{t,j})}{\sum_{\ell=1}^K p_{t,\ell}\exp(\eta_p s_{t,\ell})}. \tag{31}$$

We impose only boundedness conditions needed by the finite-horizon analysis.

Assumption 3 (Bounded Jacobian and lag budget)**.** *There exist finite constants L, E such that for every round t,*

$$\|G_t\|_1 \coloneqq \max_{j \in [K]} \sum_{k=1}^K |[G_t]_{kj}| \le L, \qquad 0 \le \epsilon_{t,k} \le E. \tag{32}$$

For a horizon T, define the realized dual magnitude

$$\Lambda_T \coloneqq \max_{1 \le t \le T+1} \|\lambda_t\|_\infty, \qquad H \coloneqq 2L + E. \tag{33}$$

Then Assumption 3 implies

$$\|s_t\|_\infty \le L(1 + \Lambda_T), \qquad \|h_t(p_t)\|_\infty \le H. \tag{34}$$

The second inequality follows from $\|u - p_t\|_1 \le 2$ and $|[G_t]_{kj}| \le L$.

E.2 Constrained no-regret guarantee

For a fixed comparator $p \in \Delta_K$, define cumulative utility regret

$$R_T(p) \coloneqq \sum_{t=1}^T \big(F_t(p) - F_t(p_t)\big). \tag{35}$$

The next theorem gives the finite-horizon bound from which the $O(\sqrt{T})$ statement in the main text follows.

Theorem 2 (Constrained no-regret of full-Jacobian PDCM)**.** *Suppose Assumptions 2–3 hold and (p_t, λ_t) is generated by equation 29–equation 30. Then for every horizon $T \ge 1$, every $p \in \mathcal{P}_T$, and every $\eta_p, \eta_\lambda > 0$,*

$$R_T(p) \le \frac{\log K}{\eta_p} + \frac{\eta_p T}{2} L^2 (1 + \Lambda_T)^2 + \frac{\eta_\lambda T}{2} K H^2. \tag{36}$$

Moreover, for every task k,

$$\left[\sum_{t=1}^{T} c_{t,k}(p_t)\right]_+ \le \left[\sum_{t=1}^{T} h_{t,k}(p_t)\right]_+ \le \frac{\Lambda_T}{\eta_\lambda}. \tag{37}$$

If the dual sequence is stable, i.e., $\Lambda_T \le \Lambda < \infty$ for all T, and $\eta_p, \eta_\lambda = \Theta(T^{-1/2})$, then

$$R_T(p) = O(\sqrt{T}), \qquad \left[\sum_{t=1}^{T} c_{t,k}(p_t)\right]_+ = O(\sqrt{T}), \tag{38}$$

so both average regret and average excess counterfactual lag vanish as $O(T^{-1/2})$.

Proof. We first prove a one-step entropic mirror-ascent inequality. By optimality of equation 29 and the three-point identity for the negative-entropy Bregman divergence, for every $p \in \Delta_K$,

$$\eta_p \langle s_t, p - p_{t+1} \rangle \le D_{\mathrm{KL}}(p\|p_t) - D_{\mathrm{KL}}(p\|p_{t+1}) - D_{\mathrm{KL}}(p_{t+1}\|p_t). \tag{39}$$

Adding $\eta_p \langle s_t, p_{t+1} - p_t \rangle$ to both sides and using Hölder's inequality together with Pinsker's inequality,

$$\langle s_t, p_{t+1} - p_t \rangle \le \|s_t\|_\infty \|p_{t+1} - p_t\|_1, \qquad D_{\mathrm{KL}}(p_{t+1}\|p_t) \ge \frac{1}{2}\|p_{t+1} - p_t\|_1^2, \tag{40}$$

we obtain

$$\langle s_t, p - p_t \rangle \le \frac{D_{\mathrm{KL}}(p\|p_t) - D_{\mathrm{KL}}(p\|p_{t+1})}{\eta_p} + \frac{\eta_p}{2}\|s_t\|_\infty^2. \tag{41}$$

Summing over t, using $p_1 = u$, $D_{\mathrm{KL}}(p\|u) \le \log K$, and equation 34, gives

$$\sum_{t=1}^{T} \langle s_t, p - p_t \rangle \le \frac{\log K}{\eta_p} + \frac{\eta_p T}{2} L^2 (1 + \Lambda_T)^2. \tag{42}$$

We now relate the score to utility regret and the safety residuals. From equation 25,

$$h_t(p_t) - h_t(p) = G_t(p - p_t). \tag{43}$$

Therefore, using equation 28,

$$\langle s_t, p - p_t \rangle = \left\langle G_t^\top \mathbf{1}, p - p_t \right\rangle + \langle \lambda_t, G_t(p - p_t) \rangle \tag{44}$$

$$= \left\langle G_t^\top \mathbf{1}, p - p_t \right\rangle + \langle \lambda_t, h_t(p_t) - h_t(p) \rangle . \tag{45}$$

For $p \in \mathcal{P}_T$, we have $h_t(p) \le 0$ coordinatewise and $\lambda_t \ge 0$, so

$$\left\langle G_t^\top \mathbf{1}, p - p_t \right\rangle + \langle \lambda_t, h_t(p_t) \rangle \le \langle s_t, p - p_t \rangle. \tag{46}$$

Since each $U_{t,k}$ is concave, F_t is concave, and Lemma 1 implies

$$F_t(p) - F_t(p_t) \le \langle \nabla F_t(p_t), p - p_t \rangle = \left\langle G_t^\top \mathbf{1}, p - p_t \right\rangle . \tag{47}$$

Combining equation 42, equation 46, and equation 47 yields

$$R_T(p) + \sum_{t=1}^{T} \langle \lambda_t, h_t(p_t) \rangle \le \frac{\log K}{\eta_p} + \frac{\eta_p T}{2} L^2 (1 + \Lambda_T)^2. \tag{48}$$

It remains to control the dual term. Projection onto the nonnegative orthant is nonexpansive and fixes the origin, hence equation 30 gives

$$\|\lambda_{t+1}\|_2^2 \le \|\lambda_t + \eta_\lambda h_t(p_t)\|_2^2 \tag{49}$$

$$= \|\lambda_t\|_2^2 + 2\eta_\lambda \langle \lambda_t, h_t(p_t) \rangle + \eta_\lambda^2 \|h_t(p_t)\|_2^2. \tag{50}$$

Rearranging and summing from $t = 1$ to T, using $\lambda_1 = 0$ and $\|h_t(p_t)\|_2^2 \le KH^2$, gives

$$-\sum_{t=1}^{T} \langle \lambda_t, h_t(p_t) \rangle \le \frac{\eta_\lambda T}{2} KH^2. \tag{51}$$

Substituting equation 51 into equation 48 proves equation 36.

For the constraint bound, the coordinatewise projection satisfies

$$\lambda_{t+1,k} = \max\{0, \lambda_{t,k} + \eta_\lambda h_{t,k}(p_t)\} \geq \lambda_{t,k} + \eta_\lambda h_{t,k}(p_t). \tag{52}$$

Telescoping and using $\lambda_{1,k} = 0$ gives

$$\eta_\lambda \sum_{t=1}^{T} h_{t,k}(p_t) \leq \lambda_{T+1,k} \leq \Lambda_T. \tag{53}$$

Taking positive parts proves the second inequality in equation 37. The first follows from equation 26 after summing over t. Finally, if $\Lambda_T \leq \Lambda$ and both step sizes are of order $T^{-1/2}$, every term in equation 36 and equation 37 is $O(\sqrt{T})$. □

The preceding theorem immediately yields the comparator used in the main text.

Corollary 1 (Best fixed feasible allocation)**.** *Let*

$$p_T^\star \in \arg\max_{p \in \mathcal{P}_T} \sum_{t=1}^{T} F_t(p). \tag{54}$$

Under the dual-stability condition $\Lambda_T \leq \Lambda$ and step sizes $\eta_p, \eta_\lambda = \Theta(T^{-1/2})$,

$$\sum_{t=1}^{T} [F_t(p_T^\star) - F_t(p_t)] = O(\sqrt{T}), \tag{55}$$

and for every task k,

$$\left[\sum_{t=1}^{T} (U_{t,k}(u) - U_{t,k}(p_t) - \epsilon_{t,k})\right]_+ = O(\sqrt{T}). \tag{56}$$

Corollary 2 (Uniform-mixture safeguard)**.** *Because $u \in \mathcal{P}_T$, Theorem 2 applies in particular to the uniform mixture. Under dual stability and $\eta_p, \eta_\lambda = \Theta(T^{-1/2})$,*

$$\sum_{t=1}^{T} F_t(u) - \sum_{t=1}^{T} F_t(p_t) = O(\sqrt{T}). \tag{57}$$

Hence the average aggregate-utility gap between PDCM and the static uniform mixture vanishes as $O(T^{-1/2})$. In addition,

$$\frac{1}{T} \sum_{t=1}^{T} (U_{t,k}(u) - U_{t,k}(p_t)) \leq \frac{1}{T} \sum_{t=1}^{T} \epsilon_{t,k} + O(T^{-1/2}) \tag{58}$$

for every task k.

Remark 1 (Dual stability)**.** *The finite-horizon bounds in equation 36–equation 37 hold for the realized quantity Λ_T and do not assume that the dual sequence is bounded in advance. The $O(\sqrt{T})$ rates require $\Lambda_T = O(1)$. This is a stability condition on the coupled allocation and training dynamics; it may be checked empirically, imposed as an assumption, or enforced by an upper projection on the dual variables when such a projection is inactive relative to the desired optimum.*

E.3 Dual-weighted marginal-utility matching

The no-regret result describes finite-horizon behavior. A complementary stationary characterization explains the allocation condition targeted by the primal-dual coupling. Let $U_k : \Delta_K \to \mathbb{R}$ be differentiable and concave, define

$$U(p) \coloneqq [U_1(p) \quad \cdots \quad U_K(p)]^\top, \qquad G(p) \coloneqq \nabla_p U(p), \qquad F(p) \coloneqq \mathbf{1}^\top U(p), \tag{59}$$

and consider the static constrained allocation problem

$$\begin{aligned} \max_{p \in \Delta_K} \quad & F(p) \\ \text{subject to} \quad & U_k(u) - U_k(p) - \epsilon_k \leq 0, \qquad k \in [K]. \end{aligned} \tag{60}$$

Since F is concave and each constraint function $U_k(u) - U_k(p) - \epsilon_k$ is convex, equation 60 is a convex optimization problem after negating the objective.

Theorem 3 (Dual-weighted marginal-utility matching)**.** *Assume equation 60 satisfies Slater's condition, and let $(p^\star, \lambda^\star)$ be a primal-dual optimum. Define*

$$s^\star := G(p^\star)^\top(\mathbf{1} + \lambda^\star). \tag{61}$$

Then there exists $\mu^\star \in \mathbb{R}$ such that

$$s_j^\star = \mu^\star \quad \textit{for every } j \textit{ with } p_j^\star > 0, \qquad s_j^\star \le \mu^\star \quad \textit{for every } j \textit{ with } p_j^\star = 0. \tag{62}$$

The multipliers additionally satisfy

$$\lambda_k^\star \ge 0, \qquad U_k(u) - U_k(p^\star) - \epsilon_k \le 0, \qquad \lambda_k^\star\big(U_k(u) - U_k(p^\star) - \epsilon_k\big) = 0. \tag{63}$$

If $p_j^\star > 0$ for every j, then $p^\star$ is a fixed point of the exponentiated-gradient primal map with $\lambda^\star$ if and only if $s^\star = \mu^\star \mathbf{1}$.

Proof. Rewrite equation 60 as the convex minimization problem

$$\begin{aligned} \min_{p \in \Delta_K} \quad & -F(p) \\ \text{subject to} \quad & c_k(p) := U_k(u) - U_k(p) - \epsilon_k \le 0, \qquad k \in [K]. \end{aligned} \tag{64}$$

Under Slater's condition, the Karush–Kuhn–Tucker conditions are necessary and sufficient. Introduce multipliers $\lambda_k \ge 0$ for $c_k(p) \le 0$, $\mu \in \mathbb{R}$ for $\mathbf{1}^\top p = 1$, and $\alpha_j \ge 0$ for $-p_j \le 0$. The KKT Lagrangian is

$$\mathcal{L}_{\mathrm{KKT}}(p, \lambda, \mu, \alpha) = -F(p) + \sum_{k=1}^{K} \lambda_k\big(U_k(u) - U_k(p) - \epsilon_k\big) + \mu(\mathbf{1}^\top p - 1) - \alpha^\top p. \tag{65}$$

Differentiating with respect to p gives the stationarity condition

$$-G(p^\star)^\top(\mathbf{1} + \lambda^\star) + \mu^\star \mathbf{1} - \alpha^\star = 0. \tag{66}$$

Hence $s_j^\star = \mu^\star - \alpha_j^\star$. If $p_j^\star > 0$, complementary slackness for $-p_j \le 0$ gives $\alpha_j^\star = 0$, so $s_j^\star = \mu^\star$. If $p_j^\star = 0$, then $\alpha_j^\star \ge 0$, so $s_j^\star \le \mu^\star$. The remaining KKT conditions give equation 63.

For the fixed-point statement, suppose first that $p_j^\star > 0$ for every j and $s^\star = \mu^\star \mathbf{1}$. Then the exponentiated-gradient map gives

$$\frac{p_j^\star \exp(\eta_p s_j^\star)}{\sum_{\ell=1}^K p_\ell^\star \exp(\eta_p s_\ell^\star)} = \frac{p_j^\star \exp(\eta_p \mu^\star)}{\exp(\eta_p \mu^\star) \sum_{\ell=1}^K p_\ell^\star} = p_j^\star. \tag{67}$$

Conversely, if the map fixes an interior $p^\star$, division by $p_j^\star > 0$ shows that $\exp(\eta_p s_j^\star)$ equals the same normalization constant for every j. Taking logarithms gives $s_j^\star = s_\ell^\star$ for all j, ℓ, which is exactly $s^\star = \mu^\star \mathbf{1}$ for some scalar $\mu^\star$. □

Theorem 3 exposes the role of the full Jacobian. Expanding the stationary score coordinatewise,

$$s_j^\star = \sum_{k=1}^{K} (1 + \lambda_k^\star) \frac{\partial U_k(p^\star)}{\partial p_j}. \tag{68}$$

Thus the dual variable of task k amplifies the entire kth row of the allocation-response Jacobian, not only its diagonal entry. A training coordinate j is valuable when increasing its allocation improves any evaluation task with high marginal utility, and its score receives additional weight when those tasks are under active lag pressure. At an interior stationary optimum, these dual-weighted cross-task marginal effects are equalized across all training coordinates.