# Extended Abstract

**Motivation** Curriculum learning (CL) has a long and storied history. Many of the current CL approaches rely largely on hand-crafted curricula or bespoke approaches. We wanted to create a curriculum learning path that uses the fact that if we know distributions over features of problems we correctly and incorrectly solve, we can construct intermediate distributions of problems that are difficult, but still in some way similar to easier problems. Optimal transport gives the solution to this most probable path problem.

**Method** We propose Optimal Transport (OT) Curriculum Learning, a method to train at the frontier of difficulty. That is, we train on "difficult" problems closest to "easy" problems in a (currently handcrafted) feature space, where the difficulty of a problem is assessed based on empirical solve rates. Although we drift slightly from the original goal of generating synthetic problems (see future work), we adaptively generate the training distribution used by our problem by creating histograms in the feature space we generate of the problems we have mastered and those we have not. We then optimally transport mass from the histogram of correctly answered problems toward the histogram of incorrectly answered problems, while minimizing transport cost ($\ell_2$ distance between problem features). The resulting intermediate distribution between the two is used to train the model on problems close to those already mastered, but closer to those not yet mastered.

**Implementation** To implement we first featurize each problem in the question bank and then run $K$-means in this feature space to assign each problem to one of $K$ bins. We develop the simplest transport cost possible (other possible future work) set to the $\ell_2$ distance between bin centroids. Next, we initialize the curriculum to the empirical bin distribution and train with RLOO. With the current model we can accumulate per-bin correct/incorrect histograms. Every $M$ steps we can solve a partial-Wasserstein coupling from the correct to the incorrect histogram. The marginals of this coupling become the next sampling distribution $q$. To ensure mixing we use what is fundamentally a replay buffer, that is, we sample batches from $q$ with probability $1 - p$ and from the exploration distribution with probability $p$. Repeat and evaluate held-out pass@1.

**Results** We use the standard (cause we had to for the deafult project) Countdown arithmetic-reasoning task with a Qwen2.5-0.5B model. Unforunately, we find the OT curriculum does not improve overall held-out pass@1 over a vanilla RLOO baseline (both $\approx 0.57$) reliably. Looking at some cross-tabs, we see a pattern that the curriculum improves on the *harder* bins (e.g. $+0.15$ on three-number $+, -, \times$ problems, $+0.10$ on four-number $+, -, \times$, $+0.09$ on three-number division problems) while losing a small amount on the easiest bins. The curriculum's target distribution visibly migrates from easy additive problems toward division-heavy problems over training. It would be good to see how we could improve this by actually building a bridge from the original distribution to the target distribution (perhaps by limiting the overall stack of data and still generating more, to show real synthetic data improvements) but that is left for future work.

**Discussion** We see that our separation of correctly answered problems and incorrectly answered problems led us to depart from our correctly answered problems too swiftly. Perhaps if we forced a higher threshold for "mastered problems" it would have improved performance more. The net effect on the (easy-dominated) held-out set is roughly flat. We do not see performance gains from curriculum learning in this case.

**Conclusion** OT-guided curricula could prove promising, although our work showed a negative result. Perhaps allowing the feature space to exist in latents rather than handcrafted for this problem would be better. We would also like to be able to scale the latents to be end to end and be able to generalize beyond countdown.

# Bridging the Gap: Optimal Transport-Guided Curriculum Learning

**Cole Citrenbaum**
Department of Statistics
Stanford University
ccitren@stanford.edu

**Noah Cowan**
Department of Statistics
Stanford University
ncowan@stanford.edu

**Aymen Echarghaoui**
Department of Statistics
Stanford University
aymen20@stanford.edu

## Abstract

Curriculum learning is designed to mimic how humans learn. We want to develop a way to go from learning things that are always within our grasp but challenging. Self-paced CL methods generally just try to measure accuracy but ignore the similarity between the problems that are solved and not. To solve this, we cast curriculum construction as an optimal-transport (OT) problem. We construct, observe, and then compute a partial-Wasserstein coupling between mastered and unmastered problems. In this way, we get an intermediate distribution that lies between the difficult target distribution and our current distribution. On the Countdown task with a Qwen2.5-0.5B policy trained with RLOO, the OT curriculum matches a vanilla RLOO baseline on overall held-out pass@1 ($\approx 0.57$) but consistently improves on the hardest problem types but unforuntaely it regresses on easier problems which offsets the gains in aggregate. We analyze why, and discuss how we could implement this if given more time and compute.

## 1 Introduction

Reinforcement learning from verifiable rewards is responsible for some of the most impressive improvements in RL today. Unfortunately, if the model is not strong enough to ever get rewards, it will not learn even in the presence of verification. Similarly, if the problems are trivial there is no gain above baseline. Curriculum learning seeks to improve the efficiency of learning by presenting problems that are properly hard for the model to learn from.

A current data-driven approach, self-paced learning, only selects problems where the current solve rate is in some target range, which both requires re-learning on the same problems and ranks problems by difficulty alone and not how similar two problems might be. Two problem types with the same low solve rate are treated identically, even if one is a small step from something the model already does well and the other is far from anything it knows.

This project is predicated on the notion that we know problems that are very easy and we know the target distribution we wish to solve. We can, using optimal transport (OT), find the most probable path of distributions going from mastered to target problems. Training along this path can most effectively improve the model along this frontier of difficulty. OT prescribes how to move mass from one distribution to another while minimizing some cost function. Previous attempts to do this required bespoke cost functions, but developing a featurization enables us to use $\ell_2$ loss.

We develop this idea for the Countdown task. We note that features here are easy to describe. Our contributions are as follows:

- We develop the language to talk of this online curriculum construction problem as partial-Wasserstein transport between empirical correct/incorrect histograms. We explain why a

budgeted coupling, rather than a full Sinkhorn coupling, is the right object for advancing the frontier without over-committing to the hardest problems.

- We give a fully realized implementation that makes the transport problem trivially cheap and refreshable online from RLOO rollouts (Algorithm 1).
- We evaluate against a vanilla RLOO baseline on Countdown. The curriculum matches the baseline on overall pass@1 but produces a clear, repeatable improvement on the hardest problem types although regression on easy problems hurts performance overall. We analyze this trade-off and outline how to fix it and some other next steps.

## 2 Related Work

Curriculum learning has a long history. Early work proposed hand-designed curricula that expose a model to easier examples before gradually introducing more difficult ones where easier and harder were developed by the intuition of the researcher. More recent approaches have attempted to automate this process. Self-paced learning uses the model's own performance to determine which examples should be emphasized during training. Generally this means favoring examples whose loss lies within a target range.

We avoid the limitation of specifying a target range by reasoning about the geometry of the task space. We want to get a sense of what problems are similar to each other. We develop this idea in the realm of optimal transport. The closest work to this work is CURROT. CURROT uses optimal transport to gradually move a training distribution toward a desired target distribution accoridng to a performance constraint using a specified task specific cost function. The key insight is that transport provides a principled way to define intermediate training distributions rather than relying on manually specified schedules. Unfortunately CURROT uses a cost that cannot expand beyond the problems they designed it for.

Our work (ends up, though not permanently if given more compute) does not do full optimal transport between the starting distribution and the target. Instead we online update the problems that we are doing well on and doing poorly on. We then simply interpolate between these using a partial, budgeted coupling and then use this induced distribution. Instead of transporting toward a fixed target distribution, we continuously estimate distributions of mastered and unmastered problem types from rollouts and try to get a balanced distribution using the intermediate here. As the model improves, the target distribution moves automatically. Our curriculum thus tracks the frontier of difficulty.

Methodologically we rely on many tools from statistics. Classical Wasserstein transport computes the minimum-cost (randomized) plan for moving one distribution into another, solving the Kantorovich problem. Entropic regularization and Sinkhorn iterations have made these computations practical at large scale, while partial-Wasserstein formulations allow only a prescribed fraction of mass to be transported according to some budget. We use the latter because curriculum construction is fundamentally an interpolation problem of going from easy to difficult problems.

Finally, we train using REINFORCE Leave-One-Out (RLOO), a simple policy-gradient algorithm that has become a common baseline in RLVR. Countdown works well as a testbed for this problem because difficulty is relatively easy to estimate and featurize and performance is easy to measure.

## 3 Method

We view the problem of curriculum learning from the perspective of Optimal Transport (OT). OT provides a framework for bridging between two distributions— given an initial marginal $p_0$, how should mass be reallocated to reach target marginal $p_1$, while also minimizing the transport cost? In the setting of curriculum learning, we can think of $p_0$ and $p_1$ as distributions over features of easy and hard problems, respectively, and OT answers how to move mass from the easy problems to the harder problems. OT respects the geometry of the task— with well chosen problem features, $\ell_2$ distance between problems plausibly corresponds to the difficulty in transferring skills from one problem to another. Since OT minimizes transport cost, our intermediate distributions should correspond to hard problems that are similar to the "easy" distribution. For example, if 3-number addition problems are mastered, then 4-number addition problems, or 3-number addition and subtraction problems might be close in feature space.

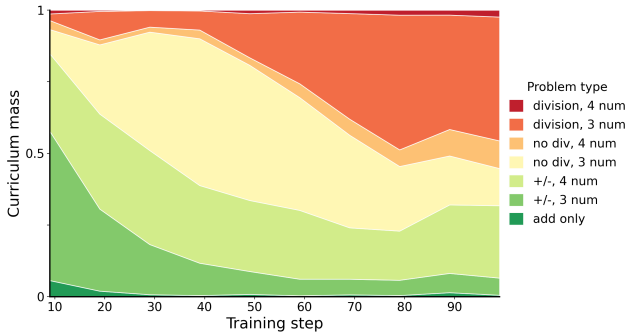


Figure 1: **The OT curriculum reallocates training effort over time.** Each band is the share of the OT-computed target distribution $q$ assigned to a problem type ($K = 100$ bins, before the exploration mixture). Early in training mass sits on easy additive and three-number problems. As those are mastered, the partial-Wasserstein coupling migrates mass toward harder "needs division" and four-number problems. The difficulty frontier advances automatically.

We argue that reasonable choices for $p_0$ and $p_1$ are the distribution of problems the model has correctly and incorrectly solved, respectively, in the last $M$ gradient steps. An alternate proposal would fix $p_1$ as the target distribution. However, by choosing $p_1$ as the distribution of problems incorrectly solved, we focus training effort on problems just beyond the model's current skill level. By bridging these two distributions, we sample classes of problems that were answered incorrectly, but are near the correctly answered problems in feature space.

A natural way to approach this problem would be with a Sinkhorn coupling, where example $i$ from the correctly-answered distribution is coupled with a problem $j$ from the incorrectly-answered distribution, with features $z_i$ and $z_j$ respectively. Then for the next $M$ training steps, we sample $(i, j)$ pairs and compute:

$$z^{(t)} = tz_i + (1 - t)z_j, \tag{1}$$

so that $z^{(t)}$ is a convex combination of the features of the correctly solved and incorrectly solved problems. Then we add a problem with features $z^{(t)}$ to the batch for training. However, this approach is not obviously tractable for Countdown problems— we would need to generate problems with features $z$: how, for example, should one generate a problem with precisely 10 possible solutions? An alternative would be to find a nearest-neighbor problem in the training dataset. However, recalling that we must find a nearest neighbor within the *incorrectly*-solved subset, we face a sharp trade-off. Suppose we score $n$ problems as correct or incorrect— the computational cost of computing the Sinkhorn coupling scales as $\mathcal{O}(n^2)$, so this is only feasible for a few thousand questions at most. However, if we have too few questions available as known incorrect examples, then the candidate nearest-neighbors sparsely cover feature space, and the matching will be very coarse. Finally, it is unclear how to approach interpolated features $z$ when some relevant features of the problems are discrete (e.g. number of operations, number of numbers, etc). For these reasons, we seek an alternate approach.

We instead discretize the feature space by assigning problems to bins: we pre-compute K-means clustering on the full training dataset, and assign each problem its nearest centroid label. Now, the transport plan must operate only over $K$ clusters, and for $K < 1000$, a Sinkhorn coupling can be computed nearly instantly. Furthermore, we have dramatically simplified the task of finding incorrectly solved problems near correctly solved ones: whereas before, we needed to know if a specific problem was solved (in)correctly so that it could be coupled, we can now determine which clusters are easy/hard empirically on a smaller subset of problems, and easily sample problems from the *full training bank* at any chosen cluster that is at the frontier of difficulty, by minimizing the transport cost of moving mass from one cluster to another based on centroid distance. A remaining issue is that in discretizing, computing an interpolated feature as in 1 ceases to make sense.

**Algorithm 1:** OT-Guided Curriculum Training

**Input:** Training problem bank $\mathcal{D}$ ($n = |\mathcal{D}|$), number of bins $K$, exploration probability $p$, update period $M$, transport budget $m$

**Preprocessing**
Solve bank of problems and compute features
Normalize features
Run $K$-means; assign each problem $x \in \mathcal{D}$ a hard label $\ell(x) \in \{1, \dots, K\}$
$S_\ell \leftarrow \{x \in \mathcal{D} : \ell(x) = \ell\}$; let $x^{(i)} \in \mathbb{R}^d$ be the centroid of cluster $i$
$C_{ij} \leftarrow \|x^{(i)} - x^{(j)}\|_2^2$ for $i, j \in \{1, \dots, K\}$
$q_0(\ell) \leftarrow \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[\, i \in S_\ell \,]$
$q \leftarrow q_0$
$\mathrm{correct}_\ell,\ \mathrm{incorrect}_\ell \leftarrow 0$ for all $\ell$
**Training**
**for** $t = 1, 2, \dots$ **do**
  batch $\leftarrow \{\}$ **for** $b = 1, 2 \dots B$ **do**
    With prob. $1 - p$ sample bin $\ell \sim q$ from the curriculum, else (prob. $p$) sample bin $\ell \sim q_0$
    Draw a random problem $x$ from bin $\ell$
    batch $\leftarrow$ batch $\cup \{x\}$
  Perform RLOO update on the sampled batch
  Update $\mathrm{correct},\ \mathrm{incorrect}$ histograms on $K$ bins from observed rollouts
  **if** $t \bmod M = 0$ **then**
    $\pi \leftarrow \textsc{PartialWasserstein}(\mathrm{incorrect}, \mathrm{correct}, C, m)$
    $q(j) \leftarrow \sum_{i=1}^{K} \pi_{ij}$ (target of the partial Wasserstein)

Hence, instead of interpolating in feature space between coupled pairs, we interpolate directly at the distribution level using the **partial Wasserstein coupling**.

The partial Wasserstein coupling solves the following optimization problem:

$$\arg\min_{\pi} \sum_{i,j} \pi_{ij} C_{ij}$$

subject to:

$$\pi 1 \le a$$
$$\pi^T 1 \le b$$
$$\pi \ge 0$$
$$1^T \pi^T 1 = m.$$

Compare this with the typical Sinkhorn (Wasserstein) coupling, where the first two inequalities are equalities (ie $\pi$ transports the initial marginal *fully* to the target marginal). By contrast, the partial Wasserstein allows us to compute a partial transport to the target (unmastered) distribution; in effect, form a new distribution, with mass in the incorrect problem clusters, subject to an allocation budget.

The partial Wasserstein optimization is a natural way to select an intermediate training distribution between correctly solved and incorrectly solved problems— at training time, we simply sample from this intermediate distribution $\pi$ after solving the above linear program.

Concretely, follows the algorithm block in 1.

## 4 Experimental Setup

Our experiments were solely from the Countdown dataset. In countdown we are given a set of numbers and a target. One must then produce an arithmetic expression using each number that evaluates

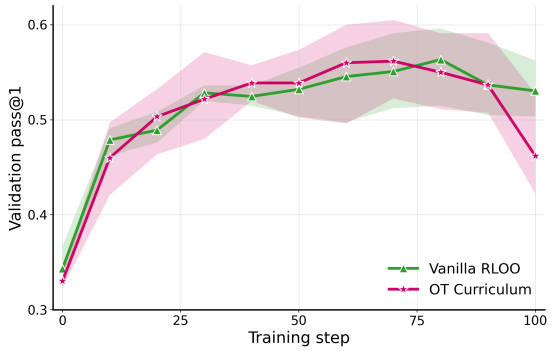


Figure 2: **Overall held-out pass@1 over training** (mean $\pm$ s.e. over 3 seeds). The OT curriculum tracks the vanilla RLOO baseline throughout and ends at essentially the same overall accuracy.

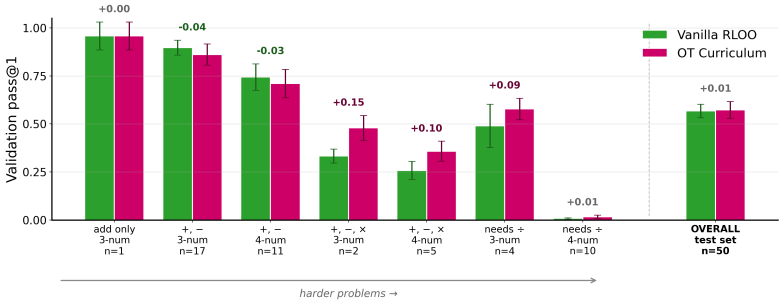


Figure 3: **Per-difficulty-bin held-out pass@1** (final checkpoints, mean $\pm$ s.e. over 3 seeds, numbers above bars are OT $-$ vanilla). The OT curriculum improves on the harder bins, $+0.15$ on three-number $+, -, \times$, $+0.10$ on four-number $+, -, \times$, $+0.09$ on three-number division, while losing a little on the easier $+, -$ bins. The two effects roughly cancel on the easy-dominated overall test set ($+0.01$).

to the target. For example, `<answer>(8 - 4) * 6</answer>` for target 24 from $\{4, 6, 8\}$. The reward is 1.0 for a valid expression that hits the target, 0.1 for a well-formed but incorrect/invalid expression, and 0.0 when no answer is produced. We use the `asingh15/countdown_tasks_3to4` bank (three- and four-number problems). For our task we needed to generate features for each problem. These features are `difficulty`, `min_depth` (minimum expression depth to a solution), `solution_count`, and statistics of the number set (`std`, `min`, `max`, `n_numbers`, `exists_divisor`). The features are standardied before clustering and then we run $K$-means with $K = 100$ and use the resulting labels as bins where the centroid of the bin is used as the label for the OT cost matrix, using $\ell_2$ distance. The base model is Qwen2.5-0.5B. We train with RLOO for 100 gradient steps and mix exploration with our curriculum and random sampling at a fifty percent mixture. The vanilla baseline is identical RLOO with curriculum sampling disabled. To evaluate, we report pass@1 on 50 held-out problems, estimated from 16 samples per problem. We run 3 seeds each for vanilla RLOO and the OT curriculum and report means with standard-error bands. The held-out set is stratified into seven difficulty bins (from addition-only three-number up to division-requiring four-number problems) for us to evaluate the model on different difficulty of problem.

## 5 Results

Figure 2 compares overall held-out pass@1 over training. Unfortunately, the OT curriculum and vanilla RLOO are statistically indistinguishable throughout and finish at the same overall accuracy. On the headline metric, the curriculum yields *no* overall improvement. But, the aggregate number obscures the fact that we see improvements in harder problems while worse performance on easier ones. Perhaps if we had done an on-policy self-distillation step to reverse the forgetting issue, it would have been possible to get improvement. Figure 3 breaks final accuracy down by difficulty bin.

And highlights the result. The OT curriculum trained model is consistently better on harder bins but slightly worse on easier ones. Because the held-out set is dominated by easier problem types, the gains and losses roughly cancel, giving the flat overall result. Just for kicks we also look at how the curriculum changes over time. We note that in figure 1 it appears the mechanism we choose is working as intended. The target $q$ starts concentrated on easy additive and three-number clusters and migrates toward division-requiring and four-number clusters as the easy clusters are mastered. The curriculum is automatically advancing the difficulty frontier rather than dwelling on already-solved problems.

## 6 Discussion

At the level of the curriculum, the method is what we thought it was. It identifies which problem clusters are at the frontier of the model's ability and reallocates mass toward them in a fun optimal transport way along low-cost paths. The model spends budget on harder problems and the experiments confirm that these results pay off on these harder problems. The reason this does not translate into an overall gain is worsening performance on easier problems. As the curriculum stops sampling the easiest clusters (their mass in $q$ decays to near zero in Figure 1), the policy slowly deteriorates. This produces the small negative $\Delta$ on the easy $+, -$ bins. Because those bins make up most of the held-out distribution, their regression cancels the frontier gains. We note that our exploration mixture (sampling from $q_0$ with probability $p = 0.5$) is meant to rehearse mastered material, but this level of replay is perhaps not enough to fully correct the errors on easier problems. Unfortunately we also see that the model simply might be too small to learn effectively for some of these problem types. We also note that this implementation is not largely extendable to general problems as our features are simple and hand-coded. It works for Countdown precisely because difficulty there is well captured by a few arithmetic features, but we note that there is a clear path here for a much more applicable and possibly more useful method. We see that we could build some encoder and decoder to autoencode countdown problems (or, perhaps, the reasoning traces of countdown problems since the problems themselves are very low dimensional) and then look at the distribution of the current learned problem traces and the distribution of the traces for the target distribution. We could then diffuse between these two and sample from the intermediate distributions along the way. That way we could avoid the creation of online "difficult" distributions and instead be able to avoid the idea of "frontier of difficulty" at all. It would be a different project but one more theoretically justified.

## 7 Conclusion

In this work we presnted an optimal transport inspired method for automatic curriculum learning. This method builds an online training distribution by partially transporting mass from the problems a model solves toward the problems it fails. On Countdown, the method learns but unfortunately we do not see large gains due to lowered performance on easier problems. We hope in the future to build a latent Schrodiner bridge model that would be able to actually create the curriculum simply from sampling target and source, but that is left for when we get more Modal credits.

## 8 Team Contributions

- **Cole Citrenbaum:** Solved the SFT homework (implementation+results), developed the optimal-transport implementation, implemented the OT curriculum sampler, and contributed to writing (algorithm).
- **Noah Cowan:** Came up with idea for project, did the early literature review proposal, poster, contributed to writing (all sections except algorithm and method).
- **Aymen Echarghaoui:** Solved the RLOO/IPO homework (implementation+results), contributed to project idea discussion, contributed to writing (all sections except algorithm and method).

**Changes from Proposal** Our proposal focused on the synthetic data generation abilities of the Schrodginer bridge way of interpolating between these two distirbutions. We wound up without the time needed to train the model to use all the data we had anyway and found the creation of latents to be more difficult than the more straightforward path we took here.

# A AI Disclosure

## A.1 Code and Methods

We engaged in extensive back-and-forth discussion about the approach. For example, rather than a *full* Sinkhorn coupling with negatively weighted costly paths, such as $q(j) = \sum_i \pi_{ij} \exp(-C_{ij})$, Claude suggested the partial-Wasserstein coupling as a simpler object solving the same problem. Claude was most useful at the planning stage and in making code production-ready: we first implemented a notebook version of the method ourselves, then used Claude to plan and adapt it into an object-oriented form for full training-loop integration. Claude Code also helped monitor Modal experiments and produce result figures. Copilot was used for in-line completion as we wrote code.