DATA SCHEDULES AND LEARNING RATE SCHEDULES SHOULD BE DESIGNED TOGETHER

Anonymous authors
Paper under double-blind review

ABSTRACT

Mid-training (MT) is a training phase in LLM training pipelines, which continues pretraining (PT) on smaller datasets of higher value. Taking the PT and MT phases as a whole, training usually starts with only PT data and then transitions completely to MT data. In this work, we view this complete transition from PT to MT data as a special case of a broader class of data mixing schedules that specify how the proportions of PT and MT tokens in each batch change over training. Under fixed total training compute and MT token budgets, we ask: *Is this complete transition to MT data optimal for minimizing final MT loss?* We show that the answer depends on the learning rate (LR) schedule: one data schedule may outperform another under one LR schedule, but changing only the LR schedule can reverse the outcome. Specifically, when the LR decays throughout continued training, introducing MT data earlier and mixing it with PT data lowers the final MT loss compared with placing all MT data at the end of training, but this result is reversed if we use constant LR. We therefore study how to jointly choose when to introduce MT data and when to decay the LR through ablation studies, and find that it is better to decay the LR only after MT data enters training. Finally, we develop a theoretical model that explains why the optimal data schedule of a fixed MT budget depends on the LR schedule.

1 INTRODUCTION

Pretraining (PT) and mid-training (MT) are two consecutive phases in large language model (LLM) training pipelines. PT uses broad web corpora, while MT uses smaller, curated mixtures enriched with mathematics, code, and other targeted data (DeepSeek-AI et al., 2024; Yang et al., 2025; Team OLMo et al., 2025; Abdin et al., 2024; GLM-4.5 Team, 2025; LLM-Core Xiaomi, 2025; K2 Team et al., 2025). In the simplest form of this pipeline, training starts with PT data alone and then transitions completely to MT data.

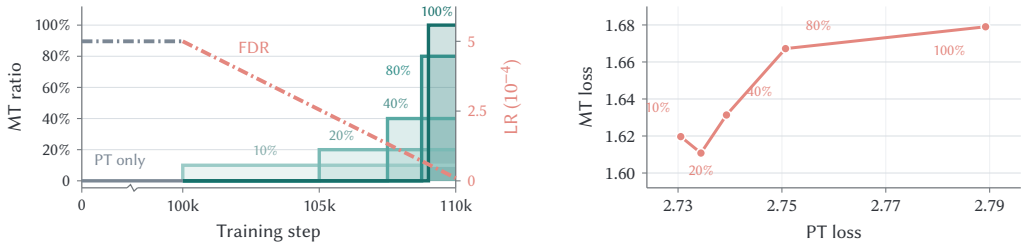
However, reserving all MT tokens for the final phase is not the only way to mix the MT token budget over training. From a data mixing perspective, the two phases described above form a particular *data schedule*. Let $\gamma(t)$ denote the fraction of MT tokens in the batch at step t, which we call the *data schedule function*. The complete transition to MT data as described above corresponds to $\gamma(t) = \mathbf{1}\{t > \tau_0\}$, where τ_0 is the transition time. In general, different allocations of MT tokens over training correspond to different data schedules $\gamma(t)$. This motivates a natural question: *with the total training compute and MT token budget fixed, is this complete transition to MT data optimal for minimizing final MT loss?*

We study this question by comparing two schedule families that introduce MT data either abruptly or gradually. Both schedule families begin with PT data alone and end at a chosen MT ratio $\gamma_{\text{end}} \in (0, 1]$, but they differ in how MT data enters training. The *Two-Stage (TS)* schedule jumps directly to this ratio and holds it fixed until training ends. The *Zero-then-Linear (ZL)* schedule instead increases the ratio gradually:

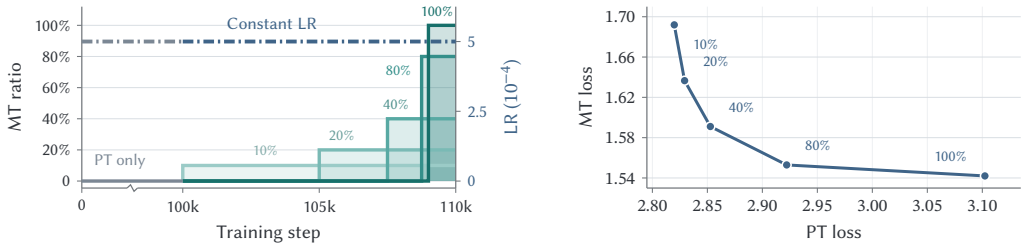
$$\gamma(t) = \begin{cases} 0, & 0 \le t \le \tau_1, \\ \gamma_{\text{end}} \dfrac{t - \tau_1}{T - \tau_1}, & \tau_1 < t \le T, \end{cases} \qquad 0 \le \tau_1 < T,$$

Here, T is the number of steps in the continued training period after the initial PT checkpoint, and τ_1 is the time when MT data first enters training. At a fixed batch size, the area $\int_0^T \gamma(t)\,\mathrm{d}t$ is proportional

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.



(a) **Left:** Linear Decay LR & **T**wo-**S**tage (TS) data schedule. **Right:** PT-MT losses under different **T**wo-**S**tage (TS) data schedules with FDR.



(b) **Left:** Constant LR & **T**wo-**S**tage (TS) data schedule. **Right:** PT-MT losses under different **T**wo-**S**tage (TS) data schedules with constant LR.

Figure 1: **replaying PT data can be both beneficial and harmful in different LR schedules.** Assume the compute and MT data budget is the same (shadow area in the left panels). With **F**ull LR **D**ecay **R**ange throughout continued training (FDR, top), mixing PT and MT data in the second TS stage gives lower final MT loss than using MT data alone. With **constant LR** (Constant LR, bottom), the pure MT stage instead gives the lowest tested MT loss.

to the number of MT tokens used. We hold this area fixed across schedules to compare different ways of allocating the same MT budget over time.

We first revisit the debate whether we should replay PT data during continued training or transition to MT data alone. At first glance, introducing MT data earlier while replaying PT data could produce larger updates while the LR is still high, but it also leaves more subsequent training that may weaken its effect. This tradeoff suggests that whether replaying PT data is beneficial may depend on the LR schedule, and we test this by comparing the same set of data schedules under different LR schedules.

We start by considering LR decay throughout the continued training period. Our results support the findings of Kotha & Liang (2026), who found that mixing PT and MT data in the second stage lowers final MT loss compared with using MT data alone under the same compute budget. In our experiments, we fix both the total training compute and MT token budget. Mixing PT data into the second stage therefore requires introducing MT data earlier and spreading it over more steps. We find that this earlier mixing yields lower final MT loss than placing all MT data in a final pure MT stage, consistently across dense and MoE models, multiple MT budgets, and different lengths of continued training (Figs. 2 and 5).

However, we find that **replaying PT data may not be beneficial if we change the LR schedule.** Specifically, holding the data schedules and all other settings fixed, if we replace LR decay with a constant LR, then the same PT replay now increases the final MT loss. Among TS schedules, placing all MT data in a final stage containing only MT data gives the lowest tested MT loss. Figure 1 illustrates this phenomenon at a 10% MT data budget, showing the data and LR schedules alongside their resulting PT and MT losses. The reversal persists across all our experimental setups (Section 2). These results show that the preferred data schedule cannot be determined independently of the LR schedule. In general, we identify the following *ranking reversal phenomenon* of data schedules: the ranking of data schedules by final MT loss can shift or even reverse under different LR schedules.

The ranking reversal phenomenon above suggests that **the data and LR schedules should be designed together.** We therefore seek joint scheduling of LR and data that minimizes final MT loss under fixed compute and MT token budgets. To explore what the optimal joint schedule of LR and data looks like, we hold each data schedule fixed and vary when LR decay begins relative to the time when MT data enters training, which we call the MT entry time.

We find that **LR should decay after MT data enters training.** Across all 19 sweeps for different LR and data schedules, the best tested data and LR schedule follow the above pattern and achieve lower final MT loss than constant LR or decay beginning at MT entry (Section 2.3). For the dense model, the best tested LR decay covers the last 40% of the MT data window for TS and the last 20% for ZL (Fig. 3). See Appendix D for the complete dense and MoE sweeps.

Finally, **we theoretically explain why the optimal data schedule depends on the LR schedule.** We develop a tractable model of SGD on noisy quadratic objectives and optimize the data schedule under a fixed MT budget. In this model, placing all MT data at the end is optimal under a constant LR, but can lose optimality under sufficiently strong linear decay. Under additional spectral assumptions and for sufficiently long training, we derive a unique optimum that begins with PT data, passes through a mixture of PT and MT data, and ends with MT data (Section 3).

2 Data Schedules and Learning Rate Schedules Should Be Designed Together

In this section, we first reproduce the counterintuitive finding, inspired by Kotha & Liang (2026), that introducing MT data earlier and replaying PT data later during mid-training can improve MT loss. We then diagnose the role of the learning rate (LR) in this phenomenon. In particular, when using a constant LR during mid-training, placing MT data later instead leads to better MT loss. This observation aligns with Luo et al. (2026) and motivates us to design data and LR schedules jointly. We conduct over 250 runs across dense and MoE models. Across our LR decay sweeps, we obtain the lowest final MT loss by keeping the LR high after MT data enters and decaying only near the end of training (Section 2.3).

2.1 Experimental Setup

In our following experiments, we use DCLM-Baseline as the PT data (Li et al., 2024) and the finemath-4plus subset of FineMath as the MT data (Ben Allal et al., 2025) to construct our data schedules. We conduct the data schedule experiments on both dense and MoE models. Our dense model has 630M parameters and mainly follows the Qwen2.5-0.5B architecture, but uses untied input and output embeddings (Qwen et al., 2024). It is first pretrained from scratch on 200B DCLM tokens during the PT phase and then trained on another 20.97B tokens during the MT phase. In the MT phase, MT tokens account for 10%, 25%, or 47.75% of the total tokens (2.10B, 5.24B, or 10.01B MT tokens), while replayed PT tokens make up the remainder. Since FineMath contains approximately 10B tokens in total, the MT data is consumed for at most one epoch. Our MoE model follows the OLMoE design (Muennighoff et al., 2024), with 2.99B total parameters, 64 experts per layer, and top-8 routing. The PT phase consists of 40B PT tokens, while the MT phase contains either 4B or 10B tokens in two experimental settings. More setup details are provided in Appendix B.

Data and LR schedule configurations. To study the interplay between data and LR schedules during MT, we consider two representative data schedules under a fixed MT token budget. Specifically, we denote T as the continual pretraining steps, with $t = 0$ starting from a fixed PT checkpoint. Let $\gamma(t)$ be the fraction of MT tokens in the training batch at training step t. We fix the overall MT token fraction α and the batch size, so the budget constraint of MT tokens is $\int_0^T \gamma(t)\,\mathrm{d}t = \alpha T$.

Two Stage (TS) uses only PT data until τ_{TS}, then switches to a fixed mixture with MT ratio γ_{end}:

$$\gamma_{\mathrm{TS}}(t) = \begin{cases} 0, & 0 \le t < \tau_{\mathrm{TS}}, \\ \gamma_{\mathrm{end}}, & \tau_{\mathrm{TS}} \le t \le T, \end{cases} \qquad \tau_{\mathrm{TS}} = T\left(1 - \frac{\alpha}{\gamma_{\mathrm{end}}}\right).$$

Zero-then-Linear (ZL) uses only PT data until τ_{ZL}, then increases the MT ratio linearly from zero to γ_{end}:

$$\gamma_{\mathrm{ZL}}(t) = \begin{cases} 0, & 0 \le t < \tau_{\mathrm{ZL}}, \\ \gamma_{\mathrm{end}} \dfrac{t - \tau_{\mathrm{ZL}}}{T - \tau_{\mathrm{ZL}}}, & \tau_{\mathrm{ZL}} \le t \le T, \end{cases} \qquad \tau_{\mathrm{ZL}} = T\left(1 - \frac{2\alpha}{\gamma_{\mathrm{end}}}\right).$$

The budget requires $\gamma_{\mathrm{end}} \ge \alpha$ for TS and $\gamma_{\mathrm{end}} \ge 2\alpha$ for ZL.

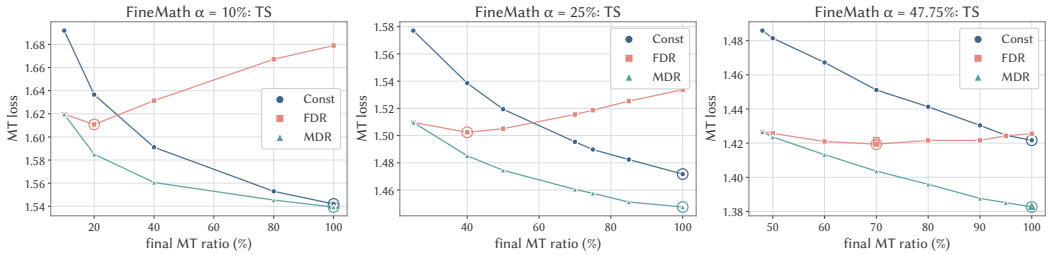


Figure 2: **Across all three MT budgets, replaying PT data improves MT loss under FDR, but not under Const or MDR.** The three panels show final MT loss as the final MT ratio in the second TS stage varies. Under the FDR regime, the lowest tested loss occurs when this stage mixes PT and MT data. Under the Const and MDR regimes, it occurs with pure MT data (100%). Large circles mark these minima, while other hollow markers show repeated runs. See Section 2.2 and Appendix C for the setup and full results.

When comparing different data schedules, we consider 3 training regimes: **Const LR Regime**, where the peak LR remains constant throughout MT; **Full LR Decay Range (FDR) regime**, where the LR decay range is over the entire continual training period regardless of the data schedule; **Matching the Decay Range (MDR) regime**, where the LR decay starts when MT data is introduced and reaches the final LR at the end of MT. Figure 7 illustrates all six combinations of LR and data schedules. Appendix B gives the exact definitions and experimental details. Specifically, in this paper, all the learning rate decays are linear.

2.2 Different LR Schedules Prefer Different Data Schedules

In practice, a TS data schedule typically sets $\gamma_{\text{end}} = 1$, which places all MT data at the end of training. A more general TS, however, allows $\gamma_{\text{end}} < 1$, which replays some PT data in the second stage. In the following paragraphs, we show that replaying PT data can improve MT loss in the FDR regime, but that this advantage disappears in the Const LR regime. The above ranking reversal phenomenon indicates that the timing of LR decay is crucial for MT loss, and that the benefit of a data schedule cannot be assessed independently of the LR schedule.

Replaying PT data can improve MT loss in the FDR regime. Modern training pipelines commonly place most MT data late in training to make better use of it (DeepSeek-AI et al., 2024; Yang et al., 2025; Team OLMo et al., 2025; Abdin et al., 2024). However, some recent work shows that retaining PT data in later stages can improve MT loss (Kotha & Liang, 2026). We observe that this counterintuitive pattern also appears across our experimental setups. As shown in Figure 1a, we adopt a TS data schedule and vary γ_{end}. As γ_{end} increases, more MT data is placed later, and the range over which MT data is used narrows. We find that both PT loss and MT loss generally increase with γ_{end}. A similar trend also holds in Figures 9 to 11, across architectures and MT budgets. These results confirm that the observation in prior work holds across a range of settings (Kotha & Liang, 2026).

Replaying PT data is not beneficial in the Const LR regime. As shown in Figure 1b, the advantage is reversed when we use a constant LR with the same type of TS data schedule. As γ_{end} increases, MT loss gradually decreases while PT loss gradually increases. This observation reveals that small learning rates in the later steps of training lead to underutilization of MT data, consistent with the observation in prior work on data curricula (Luo et al., 2026). These ranking reversal phenomena are also observed in Figures 5, 11 and 12, across architectures and MT budgets as details given in Appendix C. Therefore, different LR schedules favor different data schedules, and the benefit of a data schedule cannot be separated from the LR schedule.

The timing of LR decay is crucial for MT loss. Since small learning rates after LR annealing can lead to underutilization of MT data, the timing of LR decay is crucial for MT loss. For example, matching the LR decay range to the MT data range (MDR) can improve MT data efficiency over a full-range LR decay (FDR). As shown in Figures 2 and 5, MDR can improve MT loss at the same γ_{end} across both dense and MoE models and MT data budgets. As γ_{end} increases, MT data is placed later, and the benefit of matching LR decay also increases. This observation shows that the LR schedule regime is important for optimizing data schedule: even after tuning γ_{end} under FDR, its

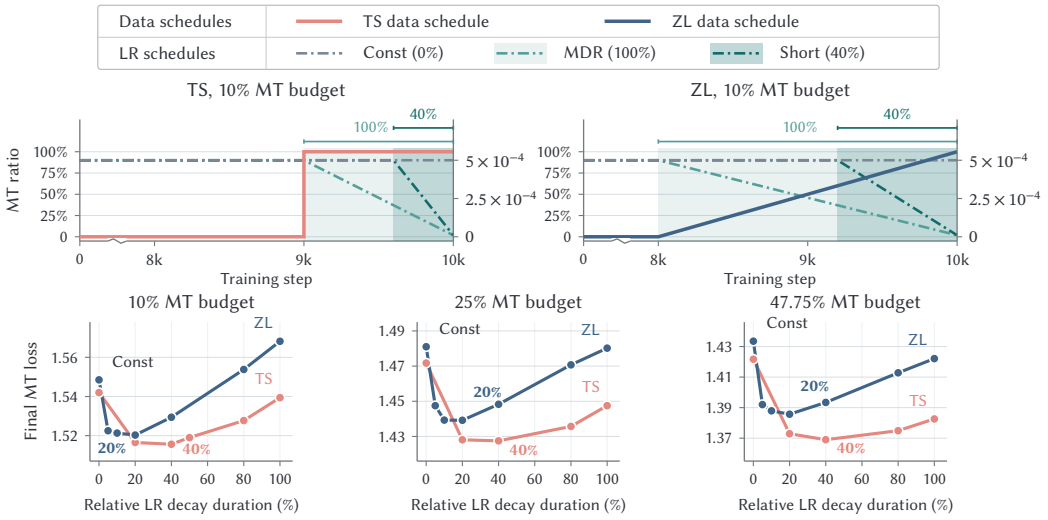


Figure 3: **LR should decay well after MT enters training for TS and ZL data schedules.** The bottom row shows how final MT loss changes with relative LR decay duration for TS and ZL across three MT budgets. ZL reaches its lowest MT loss at 20%, and TS at 40%, improving on both Const (0%) and MDR (100%). The top row illustrates TS (left) and ZL (right) at a 10% MT budget, paired with LR schedules of different *relative LR decay durations*, measured as percentages of the time from MT entry to training completion. See Section 2.3 and Appendix D for the setup and further discussion.

final MT loss is higher than that under MDR at the same γ_{end}. In addition, our results show that MDR lowers final MT loss relative to Const at all three dense budgets but raises it in all three main MoE settings (Figures 2 and 5). This motivates delaying LR decay to maintain a higher average LR over the MT data range. We next test whether keeping the LR high after MT data is introduced, followed by a shorter final cooldown, can improve on both Const and MDR.

2.3 Learning Rate Decay Should Begin Well After MT Data Enters Training

Section 2.2 shows that the data and LR schedules must be chosen together, and matching data schedule and LR decay range can improve MT loss. Moreover, the timing of LR decay affects MT performance, and starting LR decay at MT entry time is not always beneficial. We now ask how long LR decay should last to minimize final MT loss for each fixed data schedule.

Decaying LR later after MT enters training improves model performance in both Const LR and MDR regimes. Keeping the LR high allows larger updates from MT data, while a final LR decay can reduce the effect of gradient noise. To balance these effects, we keep each data schedule fixed and vary when the LR begins to decrease. The LR remains at its peak before decaying linearly to the same final value. We define the *relative LR decay duration* as the number of LR decay steps divided by the number of steps from MT entry time to training completion, expressed as a percentage. It is zero for Const and 100% for MDR. Across all 19 scans, the best tested relative LR decay durations lie between 10% and 40%, giving lower MT loss than both baselines in dense and MoE (Appendix D). Figure 3 shows this pattern for dense models: the lowest losses occur when the LR stays high through most of the MT data window and decreases only near the end. This delayed LR decay design achieves a better balance and outperforms both Const and MDR.

The optimal relative LR decay duration for ZL is about half that for TS. The results above show that a delayed LR decay gives lower MT loss. This holds for both TS and ZL, and the two data schedules favor different relative LR decay durations. Additionally, we find an interesting result that across dense and MoE experiments, the best tested relative LR decay duration for ZL is about half that for TS (Figures 3, 15, 16 and 21). Each schedule uses its own MT data window, from MT entry time to training completion, to measure the relative LR decay duration. TS uses a fixed MT ratio during this interval, while ZL increases it gradually from zero. To use the same MT budget and reach the same final MT ratio, ZL therefore needs an interval twice as long. Its smaller relative LR decay

duration therefore corresponds to approximately the same number of LR decay steps as TS. We also repeat the LR decay sweep for dense TS schedules with different final MT ratios and find that the best tested relative LR decay duration remains unchanged (Figures 11 and 13).

The preferred relative LR decay duration is stable within the model family. We further conduct extensive sweeps on the optimal relative LR decay duration for the MoE model. We find that the above optimal values (10% for ZL and 20% for TS) remain unchanged across different setups. Specifically, in MoE, the best tested relative LR decay durations remain the same across continued training lengths, and TS retains its preference at earlier PT checkpoints (Figures 19 and 24). We also conduct an ablation on the MT dataset. We find that replacing FineMath with StarCoderData (Li et al., 2023) preserves the preferred relative LR decay durations for both TS and ZL (Figure 25). Appendix F describes the dataset and comparison setup. Appendix D summarizes all 19 scans.

3 Theoretical Analysis

In this section, we provide a theoretical analysis to understand how the ranking reversal phenomenon between data schedules for different learning rate schedules might arise. Specifically, we consider a simple setting where SGD optimizes a block diagonal quadratic loss function. Though this model does not fully capture the complexity of data mixing and learning rate schedules, we believe it offers insight into how these two can be coupled together in the optimization landscape. Further, under a certain spectrum assumption in previous scaling law analysis, we could give the closed form of the optimal data schedule under practical learning rate schedules with decay.

3.1 Setup

Given a data distribution $\mathcal{D}$, we consider a quadratic loss $\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}) = \frac{1}{2}\boldsymbol{\theta}^\top \boldsymbol{M}\boldsymbol{\theta}$, where $\boldsymbol{\theta} \in \mathbb{R}^d$ is the trainable parameter vector, and $\boldsymbol{M}$ is the covariance matrix of the distribution $\mathcal{D}$. During training, we consider a mixture of PT and MT data, drawn from the distributions $\mathcal{D}_{\text{PT}}$ and $\mathcal{D}_{\text{MT}}$, respectively. We denote their covariance matrices by $\boldsymbol{G} = \mathbb{E}_{\mathcal{D}_{\text{PT}}}\left[\boldsymbol{x}\boldsymbol{x}^\top\right]$ and $\boldsymbol{H} = \mathbb{E}_{\mathcal{D}_{\text{MT}}}\left[\boldsymbol{x}\boldsymbol{x}^\top\right]$. We evaluate our model using the MT loss $\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\boldsymbol{\theta}^\top \boldsymbol{H}\boldsymbol{\theta}$.

PT and MT data. In practice, PT and MT data share some foundational knowledge, while each also contains knowledge that is specific to its own domain. For example, broad web PT data and mathematical MT data both contain general language patterns, while covering different specialized features. Motivated by this structure, we model shared and domain specific features through block diagonal covariance matrices, with a structure similar to those studied in transfer learning and data mixing theory (Ju et al., 2023; Mallinar et al., 2024; Pan et al., 2026; Dai & Zheng, 2026) as $\boldsymbol{H} = \operatorname{diag}\big(\boldsymbol{H}_0, \boldsymbol{H}_1, \boldsymbol{0}_{d_2\times d_2}\big), \boldsymbol{G} = \operatorname{diag}\big(\boldsymbol{H}_0, \boldsymbol{0}_{d_1\times d_1}, \boldsymbol{H}_2\big)$. Here $\boldsymbol{H}_0$, $\boldsymbol{H}_1$, and $\boldsymbol{H}_2$ represent the covariance blocks for shared, MT-specific, and PT-specific features, respectively, similar to Dai & Zheng (2026). For $l \in \{0, 1, 2\}$, we assume $\boldsymbol{H}_l = \operatorname{diag}(\lambda_1^{(l)}, \ldots, \lambda_{d_l}^{(l)})$ with $\lambda_i^{(l)} > 0$ and $d = d_0 + d_1 + d_2$.

Data mixing and SGD training dynamics. We analyze SGD on mixtures of PT and MT data using a continuous-time SDE approximation (Jastrzębski et al., 2017; Li et al., 2017; 2021; Malladi et al., 2022). Let T denote the training horizon in continuous time, and let $\gamma(t) \in [0, 1]$ denote the probability of sampling MT data at time $t \in [0, T]$. At step t, the training objective is given by $\mathcal{L}_t(\boldsymbol{\theta}) = \frac{1}{2}\boldsymbol{\theta}^\top\big(\gamma(t)\boldsymbol{H} + (1-\gamma(t))\boldsymbol{G}\big)\boldsymbol{\theta}$. We model each stochastic gradient as $\nabla\mathcal{L}_t(\boldsymbol{\theta})$ plus independent additive noise with mean zero and covariance $\boldsymbol{\Sigma}$. The continuous-time approximation describes the average gradient update through a deterministic term and the accumulated gradient noise through Brownian motion. For a learning rate schedule $\eta(t)$, the training dynamics of $\boldsymbol{\theta}_t$ is given by the following SDE:

$$\mathrm{d}\boldsymbol{\theta}_t = -\eta(t)\nabla\mathcal{L}_t(\boldsymbol{\theta}_t)\,\mathrm{d}t + \eta(t)\sqrt{h\boldsymbol{\Sigma}}\,\mathrm{d}\boldsymbol{B}_t. \tag{1}$$

Here $\boldsymbol{B}_t$ is a standard d-dimensional Brownian motion, and $h > 0$ is the interval in continuous time between consecutive SGD updates. $\boldsymbol{\Sigma}$ is the covariance matrix of the additive gradient noise. See Appendix H for the discrete SGD update and the derivation of its SDE approximation. Appendix M.2

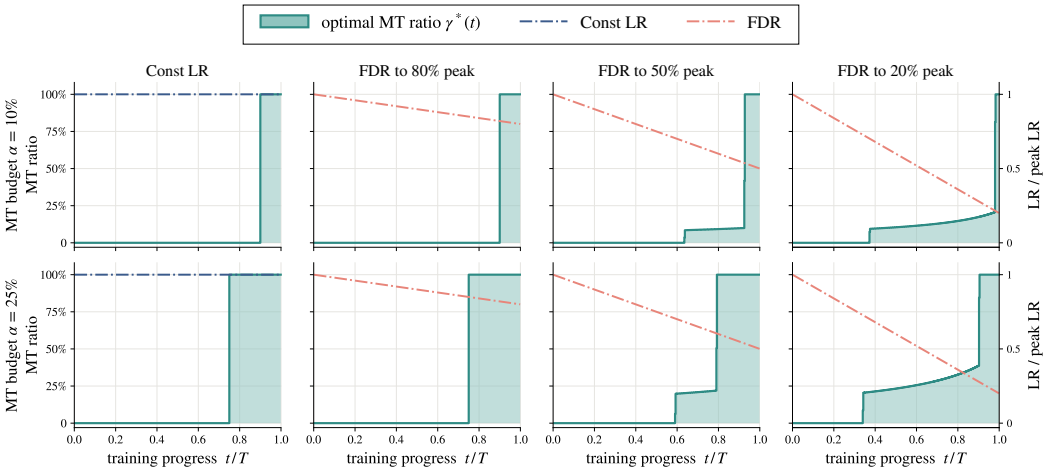


Figure 4: **Optimal data schedule can have three stages under LR decay.** The first two columns show an abrupt switch from pure PT to pure MT. With stronger LR decay in the last two columns, the teal curves instead include a mixing stage in which the MT ratio increases continuously. This change occurs at both MT budgets shown. See Theorem 3.1 for the theoretical result and Appendix M for the numerical setup.

compares the final MT loss predicted by the SDE with exact calculations and Monte Carlo simulations of discrete SGD.

In our analysis, we focus on a regime where the quantity of MT data is constrained:

Condition 3.1. *For some fixed $\alpha \in (0, 1]$, MT data accounts for at most an α fraction of the total training data.*

Under this budget constraint, we seek a data schedule that minimizes the expected final MT loss for a fixed training horizon T and LR schedule η:

$$\min_{\gamma:[0,T]\to[0,1]} \mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_T)] \qquad \text{s.t.} \qquad \frac{1}{T}\int_0^T \gamma(t)\,\mathrm{d}t \le \alpha. \tag{2}$$

The solution is unique up to changes on a set of measure zero, as established in the last paragraph of Appendix J.1. We therefore refer to this solution as *the optimal data schedule*. For additional theoretical settings, see Appendix H for more details.

3.2 Optimal Data Schedules Depend on the LR Schedule

In Section 2.2, placing all MT data in a final stage where only MT data is used gives the lowest MT loss among TS schedules under Const, whereas FDR favors earlier MT entry time with PT replay. We now recover this LR dependence in theory. Furthermore, under some power-law spectral assumptions (Lin et al., 2024; Luo et al., 2025; Yan et al., 2026), we give a closed form of the optimal data schedule under FDR regime.

Theorem 3.1 (Informal)**.** *Under Condition 3.1, let $\eta_{\max} > 0$, the following statements hold:*

(i) ***Const.*** *If $\eta(t) \equiv \eta_{\max}$, the optimal data schedule γ^* is given by $\gamma^*(t) = \bar{\gamma}(t) := \mathbb{1}_{\{t \ge (1-\alpha)T\}}$ namely the TS with $\gamma_{\text{end}} = 1$.*

(ii) ***Linear decay (FDR).*** *If $\eta(t) = \eta_{\max} - ct/T$, where $(1 - 4^{-1/3})\eta_{\max} < c < \eta_{\max}, 0 < \alpha < 1 - (1 - 4^{-1/3})\eta_{\max}/c$, then $\bar{\gamma}$ is not optimal.*

(iii) ***Three stage optimum under FDR.*** *In the FDR regime, under an additional power-law spectral assumption and for sufficiently large T, the optimal data schedule γ^* is given by:*

$$\gamma^*(t) = \begin{cases} 0, & 0 \le t < t_1, \\ C\,\eta(t)^{-\frac{2(b-1)}{b+1}}, & t_1 \le t < t_2, \\ 1, & t_2 \le t \le T, \end{cases} \tag{3}$$

where $0 < t_1 < t_2 < T$, $C > 0$ is constant in time, and $b > 1$ is determined by the power-law spectral assumption.

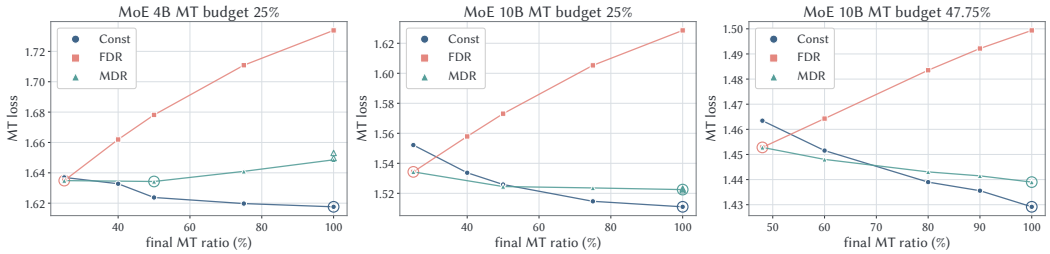


Figure 5: **Different LR regimes show patterns similar to those in dense cases (Figure 2) across all three MT data budgets for MoE model**. In MoE cases, the Constant-LR regime can outperform the MDR regime when MT ratio is 100%. See Section 2.2 and Appendix F for the setup and full results.

Theorem 3.1 shows that different learning rate schedules induce different optimal data schedules, and the optimal data schedule under the FDR regime has a time range when both PT and MT data are used in training under the power-law spectral assumption. We then give some intuition on how this theorem works. The formal statements of our theoretical results are given in Appendix I, and proofs are given in Appendices J to L.

Interpretation of Theorem 3.1. For a fixed LR schedule, only the contribution from features specific to MT depends on the data schedule, since PT and MT train the shared features in the same way. For these MT features, training may leave a nonzero loss even without gradient noise; we call this the *bias term*. Gradient noise introduces an additional contribution to the expected final loss, which we call the *variance term*. The linear SDE and quadratic loss allow us to separate these contributions exactly. Writing $R(s) = \int_s^T \eta(t)\gamma(t)\,\mathrm{d}t$ for the cumulative LR of MT updates after time s, Equation (13) gives

$$\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_T)] = A + \underbrace{\frac{1}{2}\sum_{i=1}^{d_1} \lambda_i^{(1)} \theta_i^{(1)}(0)^2 e^{-2\lambda_i^{(1)} R(0)}}_{\text{bias term}} + \underbrace{\frac{h}{2}\sum_{i=1}^{d_1} \lambda_i^{(1)} \nu_i^{(1)} \int_0^T \eta(s)^2 e^{-2\lambda_i^{(1)} R(s)}\,\mathrm{d}s}_{\text{variance term}},$$

where A is the expected final loss from shared features, independent of the data schedule, and $\nu_i^{(1)}$ is the gradient noise variance for the corresponding MT feature. The bias term decreases with $R(0)$. For noise introduced at time s, the factor $e^{-2\lambda_i^{(1)} R(s)}$ describes how subsequent MT updates reduce its contribution to the final loss.

With a constant LR, using the full MT budget fixes $R(0)$, so all such schedules have the same bias term. Placing all MT data in a final pure MT stage maximizes $R(s)$ at every time s and therefore minimizes the variance term. This explains the optimality of the abrupt switch in part (i).

Under LR decay, consider moving a small amount of MT data from a later time to an earlier one while keeping the MT budget fixed. The larger LR at the earlier time increases $R(0)$, reducing the bias term, and more strongly suppresses noise introduced before that time. However, removing the corresponding later MT updates leaves less MT training to suppress noise introduced between the two times, increasing its contribution to the variance term. Under the conditions of part (ii), moving some MT data earlier lowers the final loss, so the constant LR optimum is no longer optimal.

To determine the optimal data schedule, we compare the benefit of using the same amount of MT data at different times. If moving a small amount of MT data earlier or later lowers the final loss, the original allocation cannot be optimal. The optimal schedule must therefore balance the benefits of earlier and later MT updates so that no such reallocation can further reduce the loss. Under the assumptions of part (iii), solving for this balance gives the three-stage schedule in the theorem. During the middle stage, the optimal MT ratio satisfies $\gamma^*(t) = C\eta(t)^{-2(b-1)/(b+1)}$. Since $b > 1$, this ratio increases as the LR decreases: the optimal mixture contains progressively more MT data before the final pure MT stage. See Appendix L for the derivation.

4 Additional Experiments

In this section, we present additional experiments following the discussion in Section 2.

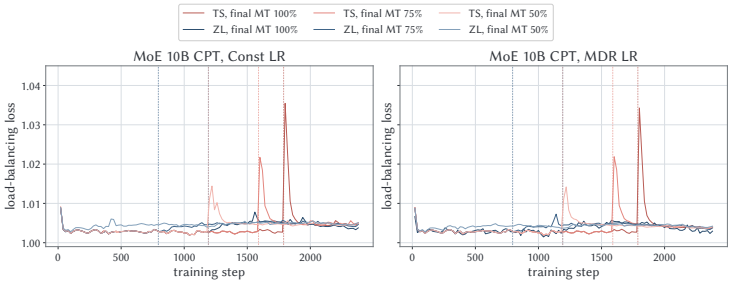


Figure 6: **Gradual MT entry reduces router imbalance under data distribution shift.** TS produces sharp spikes when the MT ratio jumps, while ZL avoids comparable spikes. Under Const (left), both schedules use the same LR trajectory, so this difference comes from the data schedule. MDR (right) shows a similar pattern, although LR decay starts at different times for TS and ZL. These MoE 10B runs use a fixed 25% MT budget. Dotted lines mark MT entry. See Appendix G for setup and diagnostic details.

4.1 Joint scheduling also shapes the PT–MT loss frontier

In addition to MT loss, PT loss also matters in practice, and we usually need to balance the two. As discussed in Section 2.2 and shown in Figures 11 and 12, with FDR, both MT and PT loss decrease as the ending MT ratio decreases under both TS and ZL data schedules. However, in most cases where LR decay does not suppress learning from MT data, including Constant LR and MDR, PT loss usually increases as MT loss decreases. Therefore, we need to strike a balance between the PT and MT loss in these cases. From this perspective, we find that **the ZL data schedule usually yields a narrower range of MT and PT losses** across different ending MT ratios, **whereas the TS data schedule exhibits much larger variations, especially in PT loss.** Thus, in practice, using TS requires more careful tuning. In addition, given a TS or ZL data schedule, we find that the MDR regime dominates both Const and FDR regimes for dense models (Figures 8 and 9). Moreover, under MDR, we find that TS dominates ZL in terms of both PT and MT loss (Figures 8, 9, 17 and 18). But under Constant LR, ZL exhibits advantages over TS for MoE model, which may be attributed to the load-balancing benefits of ZL in the MoE setting discussed in the following section.

4.2 Continuous data schedules reduce load imbalance by distribution shifts

Continuous data schedules suppress the load balancing loss spike. One additional training requirement for MoE models compared with dense models is that MoE must also balance the training tokens assigned to different experts. A common approach is to add an auxiliary load balancing term to the training loss, discouraging the router from sending most tokens to a few experts while leaving others underused (Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2022). In our experiments, the abrupt data distribution shift under TS produces a sharp spike in the load balancing loss, indicating routing instability during data shift. However, introducing MT data gradually through a continuous schedule such as ZL largely avoids this spike (Figure 6). For the 10B runs with a final MT ratio of 100%, ZL also ends with a slightly lower load balancing loss than TS under both Const and MDR. We also provide further ablations across training horizons and LR schedules in Appendix G (Figure 26).

ZL also better preserves the rankings of expert loading across the data shift. Beyond the loss, we further rank experts by how many MT tokens they receive before MT entry time and at the end of training. Under ZL, experts that receive more tokens before the shift tend to remain more heavily used afterwards. Under TS, expert loads become more balanced after the switch, but some experts that received more tokens before the switch drop in the load ranking, while previously less used experts move up. Figure 30 compares both the final load balance and these ranking changes under TS+Const and ZL+MDR. This substantial reshuffling of expert loads may make subsequent training less stable. Appendix G gives the full comparison, with the ranking statistics reported in Table 14.

5 Conclusion

In this work, we show that data schedules and LR schedules should be designed together by combining experiments on dense and MoE models with a quadratic theoretical model. There are several promising directions in the future: 1) Our results provide an initial understanding of this coupling, but determining the form of jointly optimal schedules under fixed compute and MT token budgets remains theoretically challenging. 2) We also focus primarily on minimizing final MT loss. Future work should consider PT and MT losses jointly and/or even incorporate downstream task metrics when evaluating and choosing schedules.

AI USE STATEMENT

In this work, we used generative AI tools to assist with checking mathematical claims, theoretical derivations, and proofs, and with analyzing and interpreting experimental results. We did not use generative AI tools to design experiments or implement methods. Additionally, we used these tools for manuscript drafting and editing, figure preparation, literature searches, and LaTeX formatting. We reviewed all AI-assisted work, checking the text for accuracy, comparing experimental analyses with the original results, and examining theoretical derivations and proofs for correctness. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sebastien Bubeck, Ronen Eldan, et al. Phi-4 technical report. *CoRR*, abs/2412.08905, 2024. URL https://arxiv.org/abs/2412.08905.

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

Christina Baek, Ricardo Pio Monti, David Schwab, Amro Abbas, Rishabh Adiga, Cody Blakeney, Maximilian Böther, Paul Burstein, Aldo Gael Carranza, Alvin Deng, et al. The finetuner's fallacy: When to pretrain with your finetuning data. *arXiv preprint arXiv:2603.16177*, 2026. URL https://arxiv.org/abs/2603.16177.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009. doi: 10.1137/080716542. URL https://doi.org/10.1137/080716542.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, et al. SmolLM2: When smol goes big – data-centric training of a small language model. *CoRR*, abs/2502.02737, 2025. URL https://arxiv.org/abs/2502.02737.

Mayee F. Chen, Michael Y. Hu, Nicholas Lourie, Kyunghyun Cho, and Christopher Ré. Aioli: A unified optimization framework for language model data mixing. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/71cc3acb2578bdd1e52bf9f85bbb1af0-Abstract-Conference.html.

Mayee F. Chen, Tyler Murray, David Heineman, Matt Jordan, Hannaneh Hajishirzi, Christopher Ré, Luca Soldaini, and Kyle Lo. Olmix: A framework for data mixing throughout LM development. *CoRR*, abs/2602.12237, 2026. URL https://arxiv.org/abs/2602.12237.

Rui Dai and Shuran Zheng. Explaining data mixing scaling laws, 2026. URL https://arxiv.org/abs/2606.08167.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, et al. Deepseek-v3 technical report. *CoRR*, abs/2412.19437, 2024. URL https://arxiv.org/abs/2412.19437.

Meng Ding, Kaiyi Ji, Di Wang, and Jinhui Xu. Understanding forgetting in continual learning with linear regression. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10978–11001. PMLR, 2024. URL https://proceedings.mlr.press/v235/ding24c.html.

Aleksandr Dremov, Alexander Hägele, Atli Kosson, and Martin Jaggi. Training dynamics of the cooldown stage in warmup-stable-decay learning rate scheduler. *Transactions on Machine Learning Research*, 2025. URL https://arxiv.org/abs/2508.01483.

Itay Evron, Edward Moroshko, Rachel Ward, Nathan Srebro, and Daniel Soudry. How catastrophic can catastrophic forgetting be in linear regression? In *Proceedings of Thirty Fifth Conference on Learning Theory*, volume 178 of *Proceedings of Machine Learning Research*, pp. 4028–4079. PMLR, 2022. URL https://proceedings.mlr.press/v178/evron22a.html.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL https://jmlr.org/papers/v23/21-0998.html.

Ce Ge, Zhijian Ma, Daoyuan Chen, Yaliang Li, and Bolin Ding. BiMix: A bivariate data mixing law for language model pretraining. *CoRR*, abs/2405.14908, 2024. URL https://arxiv.org/abs/2405.14908.

GLM-4.5 Team. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *CoRR*, abs/2508.06471, 2025. URL https://arxiv.org/abs/2508.06471.

Daniel Goldfarb and Paul Hand. Analysis of catastrophic forgetting for random orthogonal transformation tasks in the overparameterized regime. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 2975–2993. PMLR, 2023. URL https://proceedings.mlr.press/v206/goldfarb23a.html.

Daniel Goldfarb, Itay Evron, Nir Weinberger, Daniel Soudry, and Paul Hand. The joint effect of task similarity and overparameterization on catastrophic forgetting — an analytical model. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/6cb00ce1a21a090a3dae04cebebd8341-Abstract-Conference.html.

Jacopo Graldi, Alessandro Breccia, Giulia Lanzillotta, Thomas Hofmann, and Lorenzo Noci. The importance of being lazy: Scaling limits of continual learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 20326–20367. PMLR, 2025. URL https://proceedings.mlr.press/v267/graldi25a.html.

Jiawei Gu, Zacc Yang, Chuanghao Ding, Rui Zhao, and Fei Tan. CMR scaling law: Predicting critical mixture ratios for continual pre-training of language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 16143–16162. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.903. URL https://aclanthology.org/2024.emnlp-main.903/.

Alexander Hägele, Elie Bakouch, Atli Kosson, Loubna Ben Allal, Leandro von Werra, and Martin Jaggi. Scaling laws and compute-optimal training beyond fixed training durations. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-2427. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/8b970e15a89bf5d12542810df8eae8fc-Abstract-Conference.html.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, Xinrong Zhang, Zheng Leng Thai, Kaihuo Zhang, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun. MiniCPM: Unveiling the potential of small language models with scalable training strategies. In *First Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2404.06395.

Adam Ibrahim, Benjamin Thérien, Kshitij Gupta, Mats L. Richter, Quentin Anthony, Timothée Lesort, Eugene Belilovsky, and Irina Rish. Simple and scalable strategies to continually pre-train large language models. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2403.08763.

Stanisław Jastrzębski, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. Three factors influencing minima in sgd. *arXiv preprint arXiv:1711.04623*, 2017.

Peizhong Ju, Sen Lin, Mark S. Squillante, Yingbin Liang, and Ness B. Shroff. Generalization performance of transfer learning: Overparameterized and underparameterized regimes, 2023. URL https://arxiv.org/abs/2306.04901.

Hyunji Jung, Hanseul Cho, and Chulhee Yun. Convergence and implicit bias of gradient descent on continual linear classification. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/94ab02a30b0e4a692a42ccd0b4c55399-Abstract-Conference.html.

K2 Team, Zhengzhong Liu, Liping Tang, Linghao Jin, Haonan Li, Nikhil Ranjan, Desai Fan, Shaurya Rohatgi, Richard Fan, Omkar Pangarkar, Huijuan Wang, et al. K2-V2: A 360-open, reasoning-enhanced LLM. *arXiv preprint arXiv:2512.06201*, 2025. URL https://arxiv.org/abs/2512.06201.

Suhas Kotha and Percy Liang. Replaying pre-training data improves fine-tuning. *CoRR*, abs/2603.04964, 2026. URL https://arxiv.org/abs/2603.04964.

Sebastian Lee, Sebastian Goldt, and Andrew Saxe. Continual learning in the teacher-student setup: Impact of task similarity. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6109–6119. PMLR, 2021. URL https://proceedings.mlr.press/v139/lee21e.html.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. GShard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=qrwe7XHTmYb.

Binghui Li, Fengling Chen, Zixun Huang, Lean Wang, and Lei Wu. Functional scaling laws in kernel regression: Loss dynamics and learning rate schedules. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2509.19189.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, et al. DataComp-LM: In search of the next generation of training sets for language models. *CoRR*, abs/2406.11794, 2024. URL https://arxiv.org/abs/2406.11794.

Qianxiao Li, Cheng Tai, and Weinan E. Stochastic modified equations and adaptive stochastic gradient algorithms. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2101–2110. PMLR, 2017. URL https://proceedings.mlr.press/v70/li17f.html.

Qianxiao Li, Cheng Tai, and Weinan E. Stochastic modified equations and dynamics of stochastic gradient algorithms I: Mathematical foundations. *Journal of Machine Learning Research*, 20(40): 1–47, 2019. URL https://jmlr.org/papers/v20/17-526.html.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, et al. StarCoder: May the source be with you! *CoRR*, abs/2305.06161, 2023. URL https://arxiv.org/abs/2305.06161.

Zhiyuan Li, Kaifeng Lyu, and Sanjeev Arora. Reconciling modern deep learning with traditional optimization analyses: The intrinsic learning rate. In *Advances in Neural Information Processing Systems*, volume 33, pp. 14544–14555, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/a7453a5f026fb6831d68bdc9cb0edcae-Abstract.html.

Zhiyuan Li, Sadhika Malladi, and Sanjeev Arora. On the validity of modeling sgd with stochastic differential equations (sdes). *Advances in Neural Information Processing Systems*, 34:12712–12725, 2021.

Zhiyuan Li, Tianhao Wang, and Sanjeev Arora. What happens after SGD reaches zero loss? a mathematical framework. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2110.06914.

Licong Lin, Jingfeng Wu, Sham M Kakade, Peter L Bartlett, and Jason D Lee. Scaling laws in linear regression: Compute, parameters, and data. *Advances in Neural Information Processing Systems*, 37:60556–60606, 2024.

Emmy Liu, Graham Neubig, and Chenyan Xiong. Midtraining bridges pretraining and posttraining distributions. *CoRR*, abs/2510.14865, 2025a. URL https://arxiv.org/abs/2510.14865.

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 *International Conference on Learning Representations*, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5f67d864aae6115374fed7beddd119e0-Abstract-Conference.html.

LLM-Core Xiaomi. Mimo: Unlocking the reasoning potential of language model – from pretraining to posttraining. *CoRR*, abs/2505.07608, 2025. URL https://arxiv.org/abs/2505.07608.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1608.03983.

Kairong Luo, Haodong Wen, Shengding Hu, Zhenbo Sun, Zhiyuan Liu, Maosong Sun, Kaifeng Lyu, and Wenguang Chen. A multi-power law for loss curve prediction across learning rate schedules. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/63e88a6ebfb1f68808a058a32b947e71-Abstract-Conference.html.

Kairong Luo, Zhenbo Sun, Haodong Wen, Xinyu Shi, Jiarui Cui, Chenyi Dang, Kaifeng Lyu, and Wenguang Chen. How learning rate decay wastes your best data in curriculum-based LLM pretraining. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2511.18903.

Sadhika Malladi, Kaifeng Lyu, Abhishek Panigrahi, and Sanjeev Arora. On the sdes and scaling rules for adaptive gradient algorithms. *Advances in Neural Information Processing Systems*, 35: 7697–7711, 2022.

Neil Mallinar, Austin Zane, Spencer Frei, and Bin Yu. Minimum-norm interpolation under covariate shift. *arXiv preprint arXiv:2404.00522*, 2024.

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Nouamane Tazi, Aleksandra Piktus, Sampo Pyysalo, Thomas Wolf, and Colin A. Raffel. Scaling data-constrained language models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 50358–50376, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/9d89448b63ce1e2e8dc7af72c984c196-Abstract-Conference.html.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, et al. OLMoE: Open mixture-of-experts language models. *CoRR*, abs/2409.02060, 2024. URL https://arxiv.org/abs/2409.02060.

Brendan O'Donoghue and Emmanuel Candès. Adaptive restart for accelerated gradient schemes. *Foundations of Computational Mathematics*, 15:715–732, 2015. doi: 10.1007/s10208-013-9150-3.

Antonio Orvieto and Aurelien Lucchi. Continuous-time models for stochastic optimization algorithms. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://papers.nips.cc/paper_files/paper/2019/hash/9cd78264cf2cd821ba651485c111a29a-Abstract.html.

Zhixuan Pan, Shaowen Wang, Liao Pengfei, and Jian Li. Understanding llm behaviors via compression: Data generation, knowledge acquisition and scaling laws. *Advances in Neural Information Processing Systems*, 38:168404–168454, 2026.

Scott Pesme, Loucas Pillaud-Vivien, and Nicolas Flammarion. Implicit bias of SGD for diagonal linear networks: A provable benefit of stochasticity. In *Advances in Neural Information Processing Systems*, volume 34, pp. 29218–29230, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/f4661398cb1a3abd3ffe58600bf11322-Abstract.html.

Haoran Que, Jiaheng Liu, Ge Zhang, Chenchen Zhang, Xingwei Qu, Yinghao Ma, Feiyu Duan, Zhiqi Bai, Jiakai Wang, Yuanxing Zhang, Xu Tan, Jie Fu, Jiamang Wang, Lin Qu, Wenbo Su, and Bo Zheng. D-cpt law: Domain-specific continual pre-training scaling law for large language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/a4628e9fbd3002a554923642f74d5d6b-Abstract-Conference.html.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, et al. Qwen2.5 technical report. *CoRR*, abs/2412.15115, 2024. URL https://arxiv.org/abs/2412.15115.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1701.06538.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, et al. Megatron-LM: Training multi-billion parameter language models using model parallelism. *CoRR*, abs/1909.08053, 2019. URL https://arxiv.org/abs/1909.08053.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, et al. 2 olmo 2 furious. *CoRR*, abs/2501.00656, 2025. URL https://arxiv.org/abs/2501.00656.

Howe Tissue, Venus Wang, and Lu Wang. Scaling law with learning rate annealing. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-3044. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/830b1abc6d2da85f23d41169fa44d185-Abstract-Conference.html.

Jiapeng Wang, Changxin Tian, Kunlong Chen, Ziqi Liu, Jiaxin Mao, Wayne Xin Zhao, Zhiqiang Zhang, and Jun Zhou. MergeMix: Optimizing mid-training data mixtures via learnable model merging. *CoRR*, abs/2601.17858, 2026. URL https://arxiv.org/abs/2601.17858.

Xingjin Wang, Howe Tissue, Lu Wang, Linjing Li, and Daniel Dajun Zeng. Learning dynamics in continual pre-training for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 64525–64548. PMLR, 2025. URL https://proceedings.mlr.press/v267/wang25cx.html.

Kaiyue Wen, Zhiyuan Li, Jason Wang, David Hall, Percy Liang, and Tengyu Ma. Understanding warmup-stable-decay learning rates: A river valley loss landscape view. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/6a1fe80a9e2dcda0b3e5fd0fd87eb097-Abstract-Conference.html.

Jingfeng Wu, Difan Zou, Vladimir Braverman, Quanquan Gu, and Sham M. Kakade. The power and limitation of pretraining-finetuning for linear regression under covariate shift. In *Advances in Neural Information Processing Systems*, volume 35, pp. 33041–33053, 2022. doi: 10.52202/068431-2394. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/d5c04aa72b92c53bda5b525b60958295-Abstract-Conference.html.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. DoReMi: Optimizing data mixtures speeds up language model pretraining. In *Advances in Neural Information Processing Systems*, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/dcba6be91359358c2355cd920da3fcbd-Abstract-Conference.html.

Fuzhao Xue, Yao Fu, Wangchunshu Zhou, Zangwei Zheng, and Yang You. To repeat or not to repeat: Insights from scaling LLM under token-crisis. In *Advances in Neural Information Processing Systems*, volume 36, pp. 59304–59322, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/b9e472cd579c83e2f6aa3459f46aac28-Abstract-Conference.html.

Tingkai Yan, Haodong Wen, Binghui Li, Kairong Luo, Wenguang Chen, and Kaifeng Lyu. Larger datasets can be repeated more: A theoretical analysis of multi-epoch scaling in linear regression. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2511.13421.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, et al. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025. URL https://arxiv.org/abs/2505.09388.

Kailai Yang, Xiao Liu, Lei Ji, Hao Li, Xiao Liang, Zhiwei Liu, Yeyun Gong, Peng Cheng, and Mao Yang. Data mixing agent: Learning to re-weight domains for continual pre-training. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 9447–9473. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.427. URL https://aclanthology.org/2026.acl-long.427/.

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 *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/cc84bfabe6389d8883fc2071c848f62a-Abstract-Conference.html.

CONTENTS

1 Introduction **1**

2 Data Schedules and Learning Rate Schedules Should Be Designed Together **3**

2.1 Experimental Setup 3

2.2 Different LR Schedules Prefer Different Data Schedules 4

2.3 Learning Rate Decay Should Begin Well After MT Data Enters Training 5

3 Theoretical Analysis **6**

3.1 Setup 6

3.2 Optimal Data Schedules Depend on the LR Schedule 7

4 Additional Experiments **8**

4.1 Joint scheduling also shapes the PT–MT loss frontier 9

4.2 Continuous data schedules reduce load imbalance by distribution shifts 9

5 Conclusion **9**

A Related Works **18**

B Training Setup and Schedule Definitions **19**

C Complete Data Schedule Results for the Dense Model **22**

D Learning Rate Decay Range Scans **27**

E Data Stream Replicas and Noise Estimates **35**

F MoE Results and Additional Ablations **35**

G Router Dynamics and Interpretation **47**

H Additional Theoretical Setup **54**

I Formal Statements of Theoretical Results **54**

J Proof of Theorem I.1 **56**

J.1 Derivation of the Terminal Loss 56

J.2 KKT Condition for Equation (2) 57

J.3 Proof of Theorem I.1 58

K Proof of Theorem I.2 **59**

L Proof of Theorem I.3 **60**

L.1 Derivations of the Minimization Objective 60
L.2 KKT Condition 61
L.3 The Shape of the Data Schedule when $M_\gamma(t)$ is a Constant 62
L.4 Proof of Theorem I.3 63

M Numerical Validation of the Theoretical Model **66**
M.1 Additional Details on the Simulation Experiments 67
M.2 Checks of the SDE Approximation 68
M.3 Numerical Validation of Theoretical Predictions 68
M.4 How Optimal Data Schedule Changes with Learning Rate Schedule 68
M.5 Verification of the KKT Condition of the Optimal Data Schedule 69

A Related Works

Midtraining and continual pretraining. Modern language model recipes increasingly include a later training phase that shifts the data distribution from broad pretraining corpora toward more curated or capability-focused data. Recent technical reports describe this pattern through late stage training on curated mixtures and multistage data curricula (Team OLMo et al., 2025; Abdin et al., 2024). In Qwen3, the reasoning stage both raises the proportions of STEM, coding, reasoning, and synthetic data and accelerates learning rate decay (Yang et al., 2025). This practice couples the data and learning rate schedules but does not establish whether that coupling is optimal. A closely related literature studies continual pretraining, where a model adapts to a new distribution while retaining performance on its source distribution. Existing work examines source data replay, learning rate rewarming and redecay, and mixture ratios for balancing adaptation and retention (Ibrahim et al., 2024; Que et al., 2024; Wang et al., 2025). Recent controlled studies make the timing question more explicit. Liu et al. (2025a) find that the introduction time and mixture weight of specialized data interact strongly in midtraining. Kotha & Liang (2026) show that source data replay can improve target data efficiency and that its benefit depends on how much target data is already present during pretraining. They also study LR cooldown and report that cosine decay can make uniform mixing outperform placing target data last. Baek et al. (2026) find that repeating domain data during pretraining can improve domain performance and preserve general capabilities after finetuning. A separate line of work studies how loss changes when a finite corpus is reused over multiple epochs (Muennighoff et al., 2023; Xue et al., 2023; Yan et al., 2026). These studies vary the amount of reuse rather than the temporal allocation of a fixed target data budget. Together, prior results establish that replay, timing, and reuse all matter. We build on these observations by comparing the same data schedules across LR schedules while fixing total training compute and the MT token budget.

Learning rate scheduling in LLM training. Cosine decay is widely used in language model pretraining, but it ties the decay horizon to a predetermined training length (Loshchilov & Hutter, 2017; Luo et al., 2025). Warmup Stable Decay separates a long stable phase from a final cooldown, which supports continued training and reusable decay branches (Hu et al., 2024; Hägele et al., 2024). Several studies model how the full learning rate trajectory affects loss rather than treating the schedule as a fixed experimental detail. Schedule aware scaling laws quantify the effects of accumulated learning rate and annealing on loss curves (Tissue et al., 2025; Luo et al., 2025). Complementary analyses study the optimization dynamics of the stable and cooldown phases directly (Wen et al., 2025; Dremov et al., 2025). In continual pretraining, learning rate rewarming and redecay are often coupled with source data replay (Ibrahim et al., 2024), while Wang et al. (2025) jointly model distribution shift and learning rate annealing in the loss curve. Most directly related to our motivation, Luo et al. (2026) show that strong decay can reduce the benefit of high quality examples placed late in a curriculum ordered by data quality. Their remedies use a milder final decay or model averaging. We instead vary the timing of domain mixtures under a fixed target data budget and ask how the preferred data schedule changes across learning rate schedules.

Data mixture ratios and temporal scheduling. Prior work on data mixing primarily optimizes how much data to sample from each domain, whereas our question concerns how a fixed target data budget should be allocated over training time. Proxy based methods learn mixture weights through robust optimization or regression over smaller training runs (Xie et al., 2023; Liu et al., 2025b). Scaling law methods predict performance at unseen mixture ratios and data scales (Ge et al., 2024; Ye et al., 2025; Que et al., 2024; Gu et al., 2024). In particular, Gu et al. (2024) characterize the tradeoff between general and domain performance during continual pretraining, but treat the mixture ratio as a variable for the full training run rather than as a temporal schedule. Other methods allow the proportions to change during training. Albalak et al. (2023) use a bandit algorithm to adapt domain sampling probabilities online, while learned and adaptive frameworks optimize the ratios directly or use a learned policy (Chen et al., 2025; Yang et al., 2026). Recent frameworks address efficient mixture search and mixture reuse as the available domains change (Wang et al., 2026; Chen et al., 2026). These methods broaden how mixture proportions can be selected and adapted. Our comparisons keep both total training compute and the cumulative MT token budget fixed, then vary the LR schedule to test whether the preferred temporal allocation changes.

Theory of data scheduling under distribution shift. A complementary theoretical literature studies how prescribed task sequences affect transfer and forgetting. Analyses of overparameterized linear regression relate forgetting to task geometry, task order, and repeated exposure (Evron et al., 2022; Goldfarb & Hand, 2023; Goldfarb et al., 2024). Teacher student and feature learning models show that task similarity and representation dynamics shape interference (Lee et al., 2021; Graldi et al., 2025). Repeated gradient descent on linear classification tasks can converge to a joint solution under suitable conditions (Jung et al., 2025). Closer to our setting, Wu et al. (2022) derive target risk bounds when online SGD trains first on a source distribution and then on a target distribution under covariate shift. Ding et al. (2024) analyze noisy linear regression across multiple tasks and show that covariance spectra, task order, and step size jointly shape forgetting. These works prescribe the task sequence and then evaluate transfer, forgetting, or target risk. Our analysis instead treats the within training mixture as the decision variable, fixes its cumulative target data budget, and characterizes how the optimal temporal allocation changes with the learning rate schedule.

Comparison with Kotha & Liang (2026). A closely related work is Kotha & Liang (2026), which demonstrates the benefits of replaying pretraining data during mid-training and fine-tuning. They also investigate how pretraining data and target data should be scheduled. However, they focus on a two-stage training scenario consisting of pretraining followed by mid-training or fine-tuning, with a fixed mixing ratio between the two data sources within each stage. In contrast, we use a stepwise data schedule within MT, which allows the proportions of PT and MT data to vary over the training process. More importantly, while Kotha & Liang (2026) observe that the learning rate schedule can affect the effectiveness of data ordering, we systematically study the interaction between the learning rate schedule and the data schedule. Across a wide range of settings, we show that changing only the LR schedule can reverse which data schedule achieves lower final MT loss. Our results therefore show that the effectiveness of a data schedule cannot be considered independently of the LR schedule. Furthermore, we provide a simple guideline for jointly designing the LR schedule and the data schedule: LR decay should begin only after MT data enters training.

B Training Setup and Schedule Definitions

This section describes the training configuration and defines the data and LR schedules used in Section 2.

Figure configuration. The imported figures retain the label *CPT Loss* for FineMath validation loss; this is the *MT loss* used throughout the paper. PT loss denotes DCLM validation loss. Solid markers denote the original runs used to draw curves and identify the lowest tested losses. Hollow markers denote repeated runs using different subsets of training data or a different order of the training examples, as detailed in Appendix E.

Data and preprocessing. PT uses DCLM-Baseline (Li et al., 2024), and MT uses the `finemath-4plus` subset of FineMath released with SmolLM2 (Ben Allal et al., 2025). We tokenize the same source corpora for the dense model and the MoE model using their respective tokenizers. An EOS token is appended to each document, and best-fit packing produces fixed-length rows of 4,097 tokens, padded when necessary and shuffled within each shard. The first 4,096 tokens are inputs and the last 4,096 are prediction targets. Attention is causal across the packed row, without a document-level attention mask. The loss mask excludes positions whose *input* token equals EOS. The tokenized pools contain approximately 206B DCLM and 10.0B FineMath tokens for the dense model, and 212.6B DCLM and 9.75B FineMath tokens for the MoE model. DCLM data used for continued training comes from shards not consumed during PT, and both validation sets are held out from training.

Models and optimization. Tables 1 and 2 give the full configurations of the dense model and MoE model used in this paper. Both models are pretrained on DCLM before any continued training comparison. The dense model uses the layer dimensions of Qwen2.5-0.5B (Qwen et al., 2024), with 630M parameters under the specified untied embedding configuration. The MoE model scales an OLMoE configuration (Muennighoff et al., 2024) to 2.99B total parameters. Within a model family, all continued training runs start from the same final PT weights, with a freshly initialized optimizer and no LR warmup. The implementation for the dense model follows the original MindSpore

optimizer convention of omitting Adam bias correction and applying weight decay to every parameter. MoE training uses Megatron-LM v0.15.3 (Shoeybi et al., 2019), whose default decay exclusions leave normalization weights unregularized. These conventions are held fixed within each family.

Configuration for the dense model	
Architecture	Qwen2.5-0.5B shape; 24 layers; hidden size 896
Attention	14 query heads, 2 KV heads, head dimension 64; Q/K/V biases
FFN and normalization	SwiGLU width 4,864; RMSNorm $\epsilon = 10^{-6}$
Position and vocabulary	RoPE base 10,000; vocabulary 151,936; untied embeddings
Parameters and initialization	630M; weights $\mathcal{N}(0, 0.01^2)$, biases zero, norm weights one
Tokenizer	Qwen2.5; EOS 151643; pad 151656
Optimizer	AdamW, $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$; no bias correction
Regularization	Weight decay 0.1 on all parameters; gradient-norm clipping at 1.0
Precision	BF16 autocast; FP32 master weights and optimizer states
Sequence and global batch	4,096 tokens; 512 sequences (2,097,152 tokens/step)
PT duration	95,368 steps; approximately 200B DCLM tokens
PT learning rate	5,000-step linear warmup to 5×10^{-4}, then constant
PT final validation	DCLM 2.8147; FineMath 2.1472
continued training duration	10,000 steps; 20.97152B tokens
MT fractions and amounts	10%: 2.10B; 25%: 5.24B; 47.75%: 10.01B tokens
continued training LR	Peak 5×10^{-4}; decayed endpoint 10^{-5}

Table 1: Configuration for the dense model.

MoE configuration	
Architecture	OLMoE-based; 12 layers; hidden size 1,536
Attention	12 MHA heads, head dimension 128; no linear biases
Experts	64 per layer; top-8 routing; dropless; no shared experts
Expert FFN	SwiGLU width 768 per expert
Normalization and position	RMSNorm and QK-norm; RoPE base 10,000
Parameters	2.99B total; 454M active non-embedding parameters; 155M combined input/output embedding parameters
Embeddings and initialization	Untied input/output embeddings; weights $\mathcal{N}(0, 0.02^2)$
Tokenizer	OLMoE-1B-7B-0924; padded vocabulary 50,304; EOS 50279; pad 1
Router	Softmax in FP32; balancing coefficient 0.01; z-loss coefficient 0.001
Optimizer	AdamW, $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$; BF16 training
Regularization	Weight decay 0.1, excluding one-dimensional parameters; gradient-norm clipping at 1.0
Sequence and global batch	4,096 tokens; 1,024 sequences (4,194,304 tokens/step)
PT duration	9,536 steps; approximately 40B DCLM tokens
PT learning rate	238-step warmup to 4×10^{-4}, then constant
PT final validation	DCLM 2.6824; FineMath 2.1625
continued training duration	954 steps (4B) or 2,384 steps (10B); MT fraction 25%
continued training LR	Peak 4×10^{-4}; decayed endpoint 8×10^{-6}
Implementation	Megatron-LM v0.15.3; local transformer, FlashAttention, PyTorch norms, sequential experts

Table 2: MoE model configuration. Unlike the original OLMoE recipe, PT retains a constant LR after warmup for the following continued training comparison, and normalization weights have no weight decay.

Fixed data budgets. Let T be the number of continued training steps, α the fraction of its token budget allocated to MT, and γ_{end} the final MT ratio, as in the main text. Write $\gamma(t)$ for the MT fraction, with $\int_0^T \gamma(t)\,\mathrm{d}t = \alpha T$. For Two-Stage (TS),

$$\gamma_{\text{TS}}(t) = \begin{cases} 0, & 0 \le t < t_0, \\ \gamma_{\text{end}}, & t_0 \le t \le T, \end{cases} \qquad \frac{t_0}{T} = 1 - \frac{\alpha}{\gamma_{\text{end}}}, \quad \alpha \le \gamma_{\text{end}} \le 1. \tag{4}$$

Zero-then-Linear (ZL) linearly increases the MT fraction after an initial PT-only interval:

$$\gamma_{\mathrm{ZL}}(t) = \begin{cases} 0, & 0 \le t < t_0, \\ \gamma_{\mathrm{end}} \dfrac{t - t_0}{T - t_0}, & t_0 \le t \le T, \end{cases} \qquad \frac{t_0}{T} = 1 - \frac{2\alpha}{\gamma_{\mathrm{end}}}. \tag{5}$$

ZL is feasible when $\alpha \le \gamma_{\mathrm{end}}/2$, equivalently $2\alpha \le \gamma_{\mathrm{end}} \le 1$. At a shared $(\alpha, \gamma_{\mathrm{end}})$, its MT data window $L := T - t_0$ is twice the TS window. Changing γ_{end} therefore changes both the final mixture and the time when MT data first enter. At the respective minimum feasible final MT ratio, $t_0 = 0$ and the initial PT-only interval disappears.

To determine each batch's composition, we calculate the total number of PT sequences prescribed by the schedule up to that batch and round it up. The increase from the previous batch gives the PT count for the current batch, with the remaining sequences drawn from MT data. Sequences are assigned to workers in a fixed repeating order. This makes batch composition deterministic, avoids random fluctuations in the mixing ratio, and allows the allocation to be reconstructed from the current training step. Reported switch times and LR decay durations use the actual integer step counts.

Learning rate windows. For a positive decay length W, LR stays at $\eta_{\max}$ until $T - W$ and then decreases linearly to $\eta_{\min} = \eta_{\max}/50$:

$$\eta_W(t) = \begin{cases} \eta_{\max}, & 0 \le t < T - W, \\ \eta_{\min} + (\eta_{\max} - \eta_{\min}) \dfrac{T - t}{W}, & T - W \le t \le T. \end{cases} \tag{6}$$

Const keeps $\eta_{\max}$ throughout; it is plotted at zero window length and does not decay to $\eta_{\min}$. **Full-Decay-Range (FDR)** uses $W = T$. **Matched-Decay-Range (MDR)** uses $W = L = T - t_0$, beginning decay when MT data first enter. The scans vary $r := W/L$ with the data schedule fixed. The denominator is the entire interval *from first MT exposure to training end*, including the low-MT-density part of a ZL ramp; it is neither T nor the MT token fraction αT. A 20% window means $W = 0.2L$, and a 120% window begins decay before MT exposure. When $t_0 = 0$, MDR and FDR coincide and use one run.

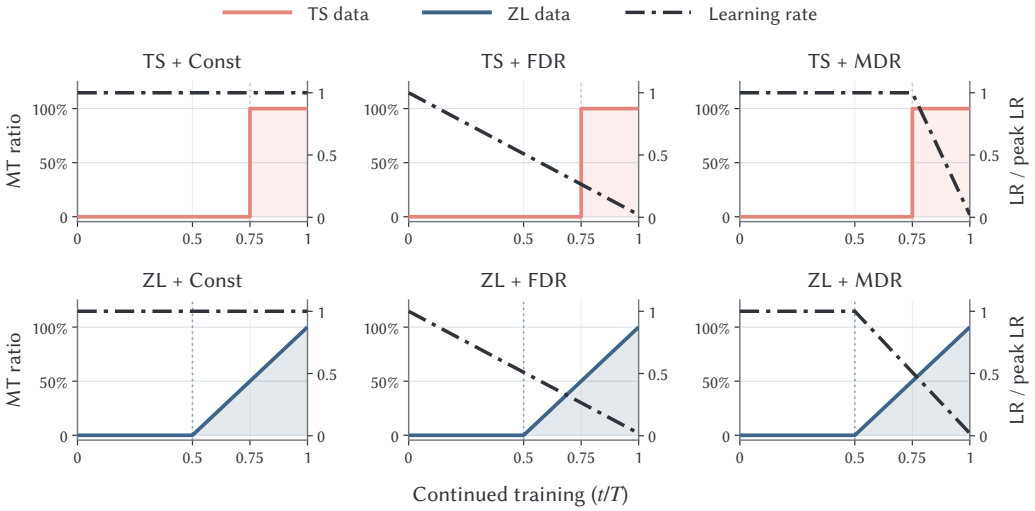


Figure 7: **Illustration of different LR schedules and data schedules.** Rows show TS and ZL, and columns show Const, FDR, and MDR. Solid lines and shading show the MT ratio on the left axes. Dash dotted lines show LR relative to its peak on the right axes. Both data schedules use a 25% MT budget and end at 100% MT, so ZL introduces MT earlier. MDR therefore begins LR decay earlier for ZL than for TS. Time starts at the shared PT checkpoint.

Validation and comparability. All final PT/MT loss comparisons use the average cross entropy per token on the validation set, evaluated at the final checkpoint. Final evaluation for the dense model uses the full held-out sets, approximately 560M DCLM and 160M FineMath tokens; intermediate evaluations use 32M tokens every 500 steps. MoE final evaluation uses a separate skip-training pass on each domain, with $32 \times 1024 \times 4096 = 134{,}217{,}728$ tokens per domain. Because the two families use different tokenizers and training implementations, their absolute loss values are not directly comparable. Cross-family comparisons concern schedule ordering and the shape of the response to LR changes.

C Complete Data Schedule Results for the Dense Model

Table 3 lists all tested final MT ratios, each crossed with Const, FDR, and MDR, with identical boundary schedules run only once. The continued training token budget remains 20.97152B across all three MT fractions. The largest fraction is 47.75%, rather than 50%, to use approximately one pass through FineMath: 2,444,800 of 2,445,076 packed rows. At this fraction, TS requires $\gamma_{\text{end}} \geq 47.75\%$, and ZL requires $\gamma_{\text{end}} \geq 95.5\%$. For ZL, we test five final MT ratios across its narrow feasible range, from 95.5% to 100%.

MT budget	TS final MT ratios (%)	ZL final MT ratios (%)
10%	10, 20, 40, 80, 100	20, 40, 60, 80, 100
25%	25, 40, 50, 70, 75, 85, 100	50, 60, 70, 80, 90, 100
47.75%	48, 50, 60, 70, 80, 90, 95, 100	95.5, 97, 98, 99, 100

Table 3: Complete data schedule runs for the dense model. Total and MT token budgets are fixed within each group.

Figures 8 to 10 show the final PT and MT losses for all tested configurations. Table 4 gives the best MT loss in each sampled data/LR family. For TS, the best tested final MT ratio under Const and MDR is 100% at every budget, whereas FDR selects 20%, 40%, and 70% as the MT budget increases. The FDR loss range contracts from approximately 0.068 to 0.031 to 0.007. Lowering γ_{end} does not always improve the FDR loss: at 10%, loss rises from 1.6108 at $\gamma_{\text{end}} = 20\%$ to 1.6197 at $\gamma_{\text{end}} = 10\%$; at 25%, it rises from 1.5024 at $\gamma_{\text{end}} = 40\%$ to 1.5097 at $\gamma_{\text{end}} = 25\%$. At 47.75%, the FDR minimum at $\gamma_{\text{end}} = 70\%$ is only 0.0016 below $\gamma_{\text{end}} = 60\%$, so the broad flattening is more informative than a precise preferred endpoint.

MT budget	Data schedule	Const	FDR	MDR
10%	TS	1.5420 (100)	1.6108 (20)	1.5394 (100)
	ZL	1.5485 (100)	1.6144 (20)	1.5682 (100)
25%	TS	1.4717 (100)	1.5024 (40)	1.4475 (100)
	ZL	1.4810 (100)	1.5064 (50)	1.4802 (100)
47.75%	TS	1.4217 (100)	1.4194 (70)	1.3826 (100)
	ZL	1.4335 (100)	1.4246 (97)	1.4221 (100)

Table 4: Lowest final MT loss among the tested final MT ratios for each MT budget, data schedule, and LR schedule. Parentheses give the final MT ratio γ_{end} in percent.

TS versus ZL depends on the LR rule and final ratio. Table 5 compares the shapes at the same final MT ratio of 100%. Here ZL gives lower observed MT loss under FDR, while TS gives lower loss under MDR and Const. This differs from selecting each shape's best final ratio: TS then gives lower FDR loss at all three MT budgets for the dense model (Table 4). Selecting each shape's best final ratio also favors TS under FDR in all three main MoE settings (Table 11 and Fig. 20). The points labeled 100% in Figures 8 and 9 show the first comparison, while each curve's minimum gives the second. Under MDR, changing from TS to ZL also starts LR decay earlier, so this comparison does not isolate data shape at a fixed LR trajectory. MT gains can also come at a PT cost. At the 25% budget under Const, TS has lower MT loss than ZL but substantially higher PT loss.

To quantify the LR applied to MT data, define $E = \int_0^T \eta(t)\gamma(t)\,\mathrm{d}t$ for an LR trajectory $\eta(t)$. This quantity weights each MT token by the LR at which it is used. For TS and ZL with the same budget α and final MT ratio γ_{end}, let $u = \alpha/\gamma_{\mathrm{end}}$ and $\rho = \eta_{\min}/\eta_{\max}$. Integrating the schedules in Section 2.1 gives

$$\frac{E_{\mathrm{ZL}}}{E_{\mathrm{TS}}} = \begin{cases} 1, & \text{Const}, \\ \dfrac{\rho + \frac{2}{3}(1-\rho)u}{\rho + \frac{1}{2}(1-\rho)u}, & \text{FDR}, \\ \dfrac{2(1+2\rho)}{3(1+\rho)}, & \text{MDR}. \end{cases}$$

Both models use $\rho = 0.02$, giving an MDR ratio of approximately 0.68 and FDR ratios of approximately 1.24 to 1.31 across settings where TS and ZL have the same MT budget and final MT ratio. These are continuous time calculations, with small differences possible from discrete step rounding. They describe the LR assigned to MT tokens, but omit the effects of later updates and optimization noise and do not by themselves predict final loss. In particular, equal exposure under Const does not imply equal loss, and ZL can have higher FDR loss despite greater exposure, as in the dense model at a 10% MT budget with $\gamma_{\mathrm{end}} = 20\%$. The role of LR in learning from late MT data is consistent with Luo et al. (2026), who find that strong LR decay can reduce the benefit of valuable examples encountered late in training.

MT budget	Const	FDR	MDR
10%	+0.0065	−0.0185	+0.0288
25%	+0.0093	−0.0138	+0.0327
47.75%	+0.0118	−0.0006	+0.0395

Table 5: MT loss difference in the dense model, ZL minus TS, with both schedules ending at 100% MT. Negative values favor ZL. Differences use the reported final losses and are not significance estimates.

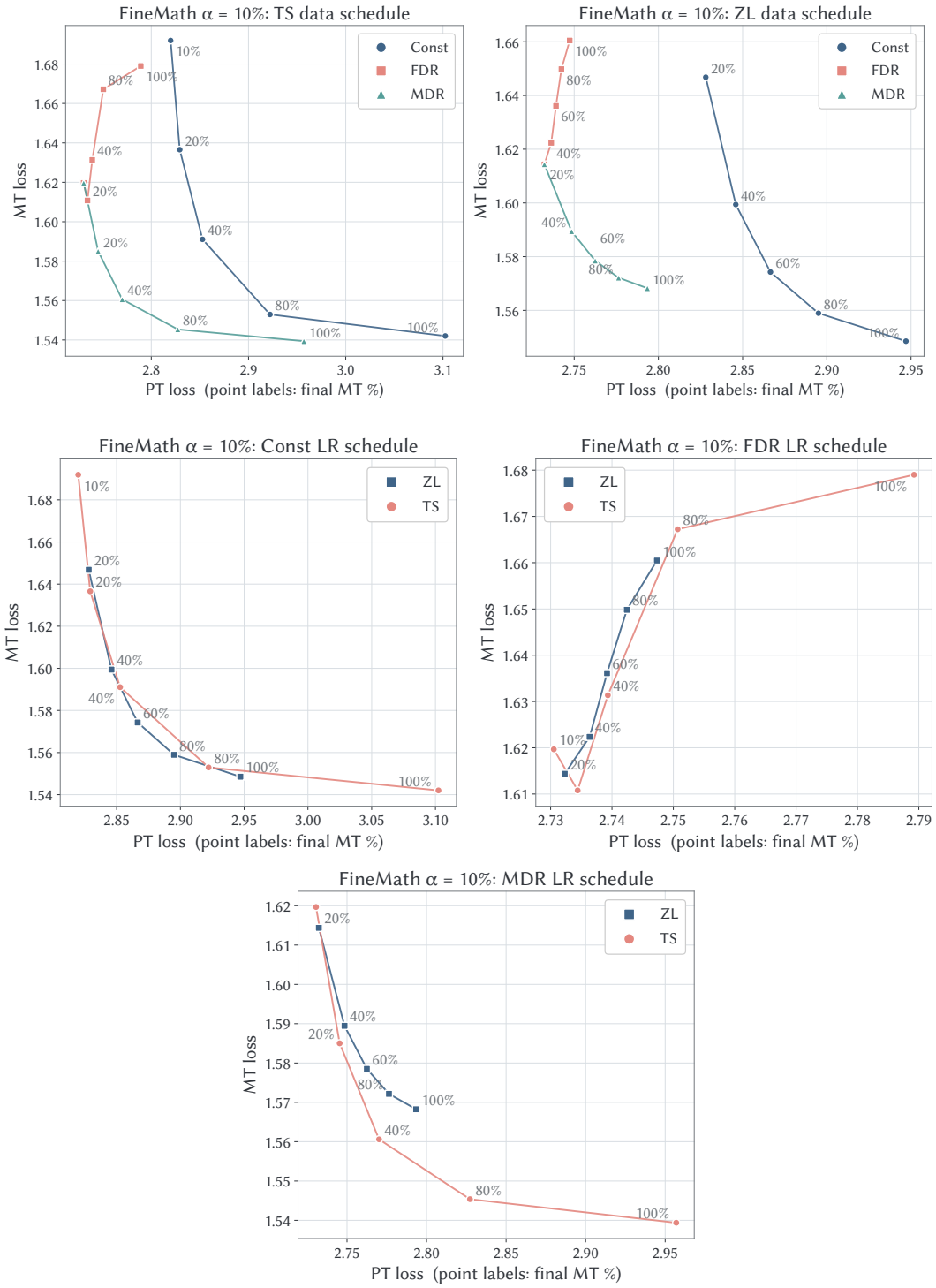


Figure 8: Complete results at a 10% MT budget for the dense model (20.97B continued training tokens; 2.10B FineMath). Top: TS and ZL with all three LR rules. Remaining panels: the same results grouped by LR rule. All panels plot final PT loss horizontally and final MT loss vertically; point labels give the final MT ratio in percent. Const and MDR select 100% MT for MT loss, while FDR selects 20% MT.

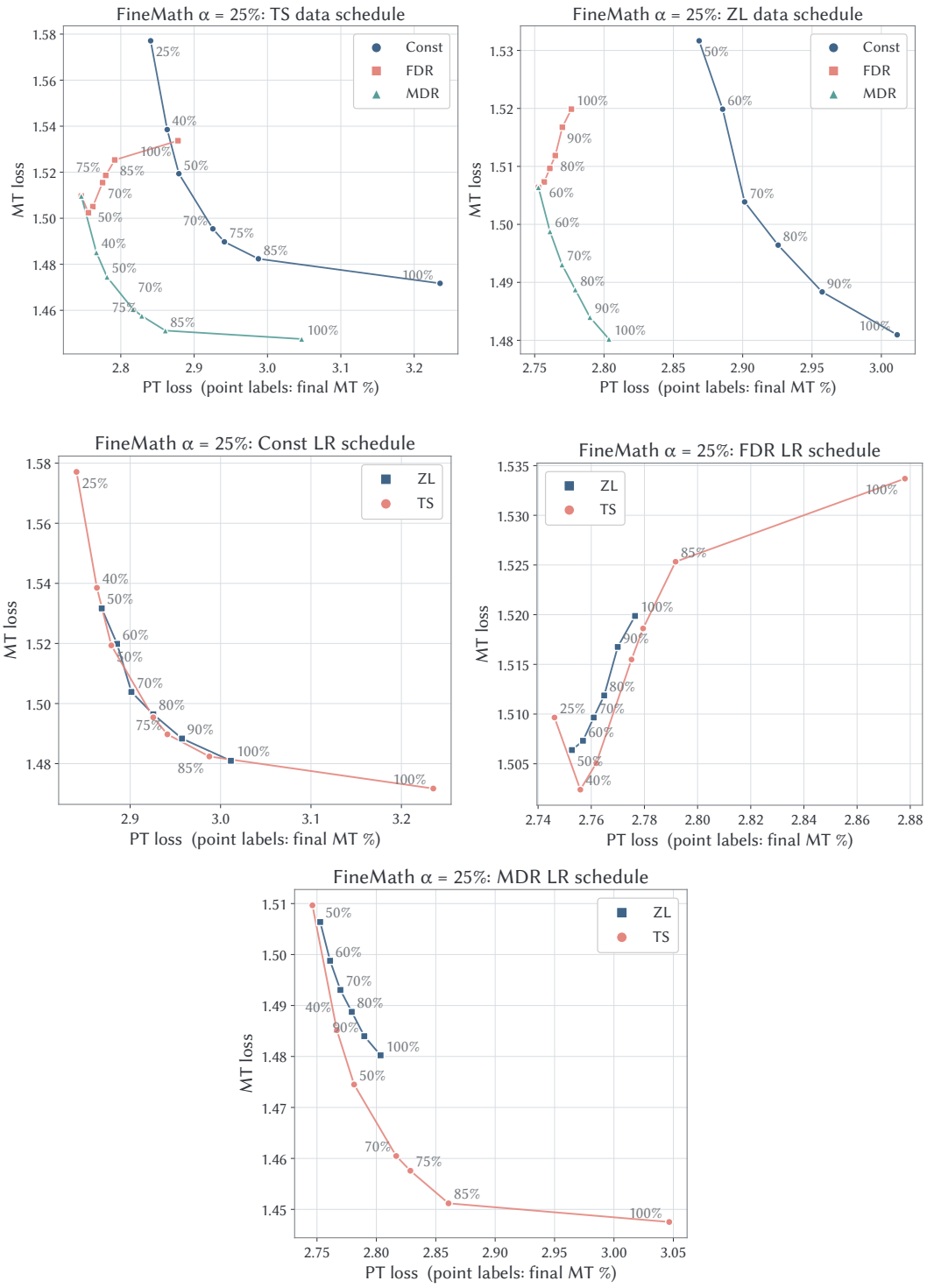


Figure 9: Complete results at a 25% MT budget for the dense model (20.97B continued training tokens; $5.24B$ FineMath). Top: all TS and ZL points under Const, FDR, and MDR. Remaining panels: the same final PT–MT loss coordinates grouped by LR. Point labels give the final MT ratio in percent. FDR's minimum is at 40% MT for TS and 50% MT for ZL; Const and MDR attain their sampled minima at 100% MT.

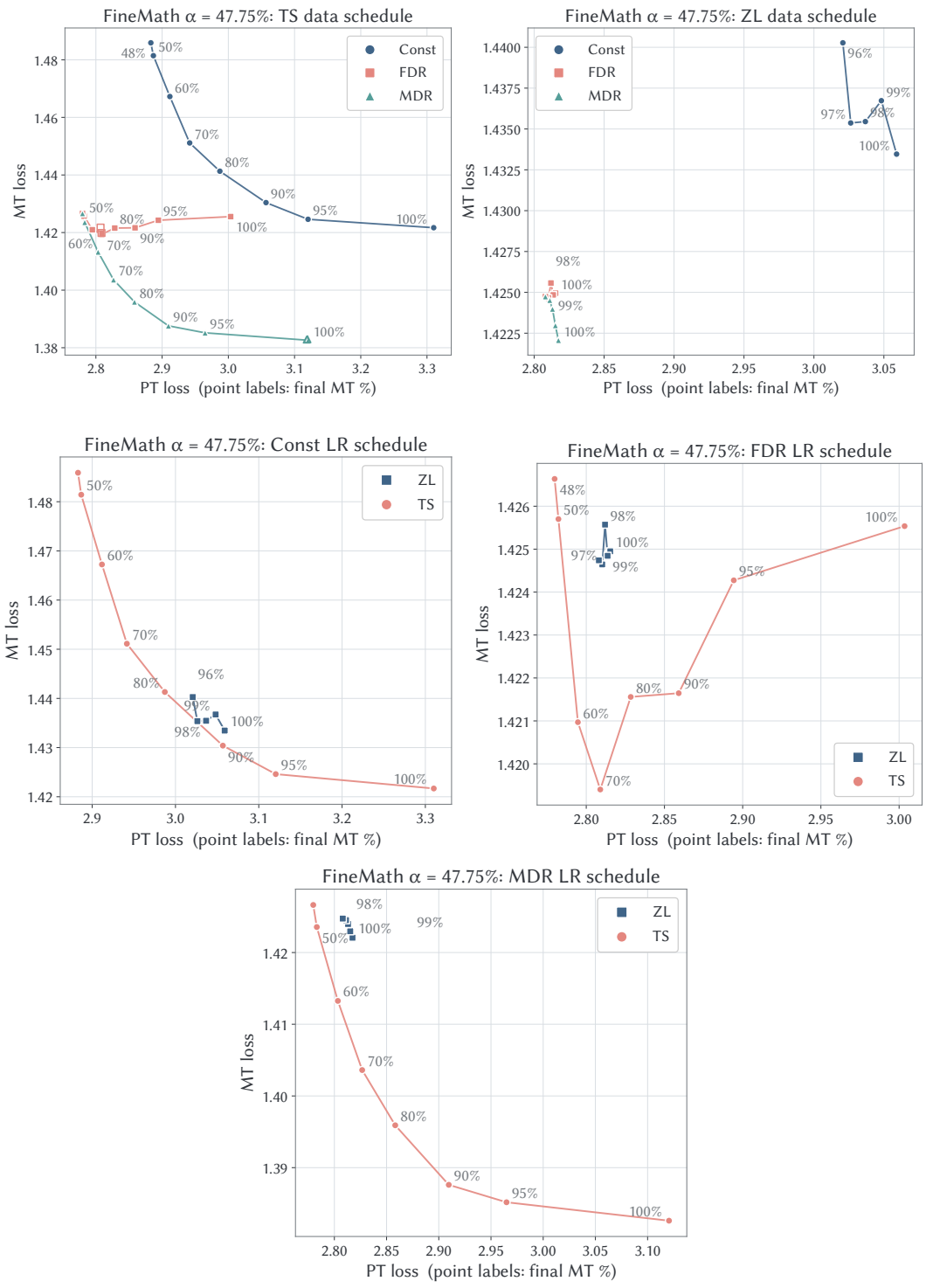


Figure 10: Complete results at a 47.75% MT budget for the dense model (20.97B continued training tokens; $10.01B$ FineMath). Top: TS across eight final MT ratios and ZL across five, from 95.5% to 100%. Remaining panels show the same final PT loss (horizontal) and MT loss (vertical), grouped by LR. For ZL, Const and MDR favor 100% MT, while FDR losses vary by only 0.001 across the tested ratios. Labels give the final MT ratio in percent, with 95.5% rounded to 96%. Hollow markers show selected data sampling replicas.

D LEARNING RATE DECAY RANGE SCANS

Following Section 2.3, we keep each data schedule fixed and vary when LR decay begins. The LR remains at the same peak until decay begins, then decreases linearly to the same final value at training completion. The *LR decay fraction* expresses this decay duration as a percentage of the MT data window, from MT entry time to training completion. For example, a 20% fraction means that LR decay covers the final 20% of this interval. Const corresponds to 0%, and MDR to 100%. Fractions above 100% start LR decay before MT entry time.

Table 6 lists the tested fractions for all 19 scans, and Table 7 summarizes their lowest observed losses. Each scan includes Const and MDR as baselines. We also test FDR, which decays the LR throughout continued training, except for StarCoder ZL.

A delayed LR decay gives the lowest tested loss in all nineteen scans. For the dense model, the best tested fractions are 40% for TS and 20% for ZL. For the MoE model, they are 20% for TS and 10% for ZL. They improve on both Const and MDR, with reductions in MT loss relative to MDR ranging from 0.0094 to 0.0660. These results support keeping the LR high after MT entry time and delaying its decay, as discussed in Section 2.3. The preferred fractions refer to the tested values. In particular, the scans from the two earlier MoE PT checkpoints test only 20% and 40% in addition to Const, MDR, and FDR, so they do not locate the optimum more precisely.

Group	Schedule, γ_{end}	MT data window (steps)	Additional LR decay fractions (%)
10% MT, dense model	TS, 100%	1,000	20, 40, 50, 80, 120
	TS, 80%	1,250	20, 40, 80, 90, 110, 120
	ZL, 100%	2,000	5, 10, 20, 40, 80, 90, 110, 120
25% MT, dense model	TS, 100%	2,500	20, 40, 80, 120
	ZL, 100%	5,000	5, 10, 20, 40, 80
47.75% MT, dense model	TS, 100%	4,775	20, 40, 80, 120
	TS, 80%	5,969	20, 40, 80, 120
	TS, 60%	7,958	20, 40, 80
	ZL, 100%	9,550	5, 10, 20, 40, 80
MoE 4B	TS, 100%	238	5, 10, 20, 40, 80, 120
	ZL, 100%	477	5, 10, 20, 40, 80
MoE 10B	TS, 100%	596	5, 10, 20, 40, 80, 120
	ZL, 100%	1,192	5, 10, 20, 40, 80
MoE 10B, 47.75%	TS, 100%	1,138	5, 10, 20, 40, 80, 120
	ZL, 100%	2,277	5, 10, 20, 40, 80
MoE 4B, PT 20B	TS, 100%	238	20, 40
MoE 4B, PT 30B	TS, 100%	238	20, 40
MoE 10B, StarCoder	TS, 100%	596	5, 10, 20, 40, 80, 120
	ZL, 100%	1,192	10, 20, 40, 80

Table 6: Complete LR decay scans. The MT data window counts training steps from MT entry time to training completion. LR decay fractions are measured relative to this window for each data schedule. Const (0%) and MDR (100%) are included in addition to the listed fractions. FDR decays the LR throughout continued training and is also tested except for StarCoder ZL. MoE groups use a 25% MT budget unless marked 47.75%, and start from 40B PT tokens unless marked PT 20B or PT 30B. MT data comes from FineMath unless marked StarCoder.

Table 8 reports every sampled MT loss for the four further configurations: TS and ZL at the MoE 10B 47.75% MT budget, and TS at the MoE 4B 25% budget starting from 20B or 30B PT tokens. The best tested LR decay fractions remain 20% for TS and 10% for ZL. The two earlier PT checkpoints confirm that delaying LR decay improves on Const and MDR, although testing only two additional fractions leaves the shape of the loss curve near its minimum uncertain.

Figures 11 to 16 show the LR trajectories alongside the final MT loss curves. These figures cover the twelve primary scans, including the complete LR/data and loss panels for TS at a 47.75% MT budget in the dense model, as highlighted in the main text. For the dense model, the additional TS scan at a 10% MT budget and a final MT ratio of 100% appears in Figure 3. Its MT data window lasts 1,000 steps, and the lowest tested loss is 1.5156 at a 40% LR decay fraction. The 120% fraction,

Group	Schedule, γ_{end}	Best LR decay fraction	Best loss	Const	MDR	MDR−best
10% MT, dense model	TS, 100%	40%	1.5156	1.5420	1.5394	0.0238
	TS, 80%	40%	1.5215	1.5529	1.5454	0.0239
	ZL, 100%	20%	1.5203	1.5485	1.5682	0.0479
25% MT, dense model	TS, 100%	40%	1.4275	1.4717	1.4475	0.0200
	ZL, 100%	20%	1.4392	1.4810	1.4802	0.0410
47.75% MT, dense model	TS, 100%	40%	1.3690	1.4217	1.3826	0.0136
	TS, 80%	40%	1.3831	1.4413	1.3959	0.0128
	TS, 60%	40%	1.4039	1.4672	1.4133	0.0094
	ZL, 100%	20%	1.3857	1.4335	1.4221	0.0364
MoE 4B	TS, 100%	20%	1.6080	1.6176	1.6486	0.0406
	ZL, 100%	10%	1.6000	1.6131	1.6586	0.0586
MoE 10B	TS, 100%	20%	1.4853	1.5110	1.5225	0.0372
	ZL, 100%	10%	1.4872	1.5101	1.5508	0.0636
MoE 10B, 47.75%	TS, 100%	20%	1.4007	1.4292	1.4390	0.0383
	ZL, 100%	10%	1.4110	1.4424	1.4770	0.0660
MoE 4B, PT 20B	TS, 100%	20%	1.6562	1.6666	1.6969	0.0407
MoE 4B, PT 30B	TS, 100%	20%	1.6260	1.6409	1.6673	0.0413
MoE 10B, StarCoder	TS, 100%	20%	0.9317	0.9468	0.9594	0.0277
	ZL, 100%	10%	0.9332	0.9476	0.9762	0.0430

Table 7: Final MT losses for all 19 scans. Group labels follow Table 6. Best loss is the lowest MT loss among the tested LR schedules, using the original run for each configuration. The corresponding LR decay fraction is measured relative to that data schedule's MT data window. The last column gives the reduction in loss relative to MDR, calculated from the displayed values. The TS scan at a 47.75% MT budget and a final MT ratio of 100% for the dense model also appears in Figure 3.

	MoE 10B, 47.75% MT		MoE 4B, 25% MT, TS	
LR decay fraction or rule	TS	ZL	PT 20B	PT 30B
Const (0%)	1.4292	1.4424	1.6666	1.6409
5%	1.4091	1.4131	—	—
10%	1.4046	**1.4110**	—	—
20%	**1.4007**	1.4137	**1.6562**	**1.6260**
40%	1.4040	1.4268	1.6578	1.6303
80%	1.4252	1.4609	—	—
MDR (100%)	1.4390	1.4770	1.6969	1.6673
120%	1.4534	—	—	—
FDR	1.4994	1.4806	1.7899	1.7552

Table 8: Final MT losses at every tested LR decay fraction for the four additional scans, including the reference LR schedules. All data schedules end at 100% MT. The MoE 10B runs start from 40B PT tokens, while the last two columns use earlier checkpoints from the same PT trajectory. Bold entries mark the lowest tested loss in each column. Dashes indicate fractions that were not tested.

outside the displayed range, gives 1.5512, while FDR gives 1.6790. Table 8 supplies the complete sampled losses for the four further configurations. Their LR trajectories and loss curves are shown in Figures 21 and 23 in Appendix F. The two StarCoder scans are compared with their FineMath counterparts in Figure 25.

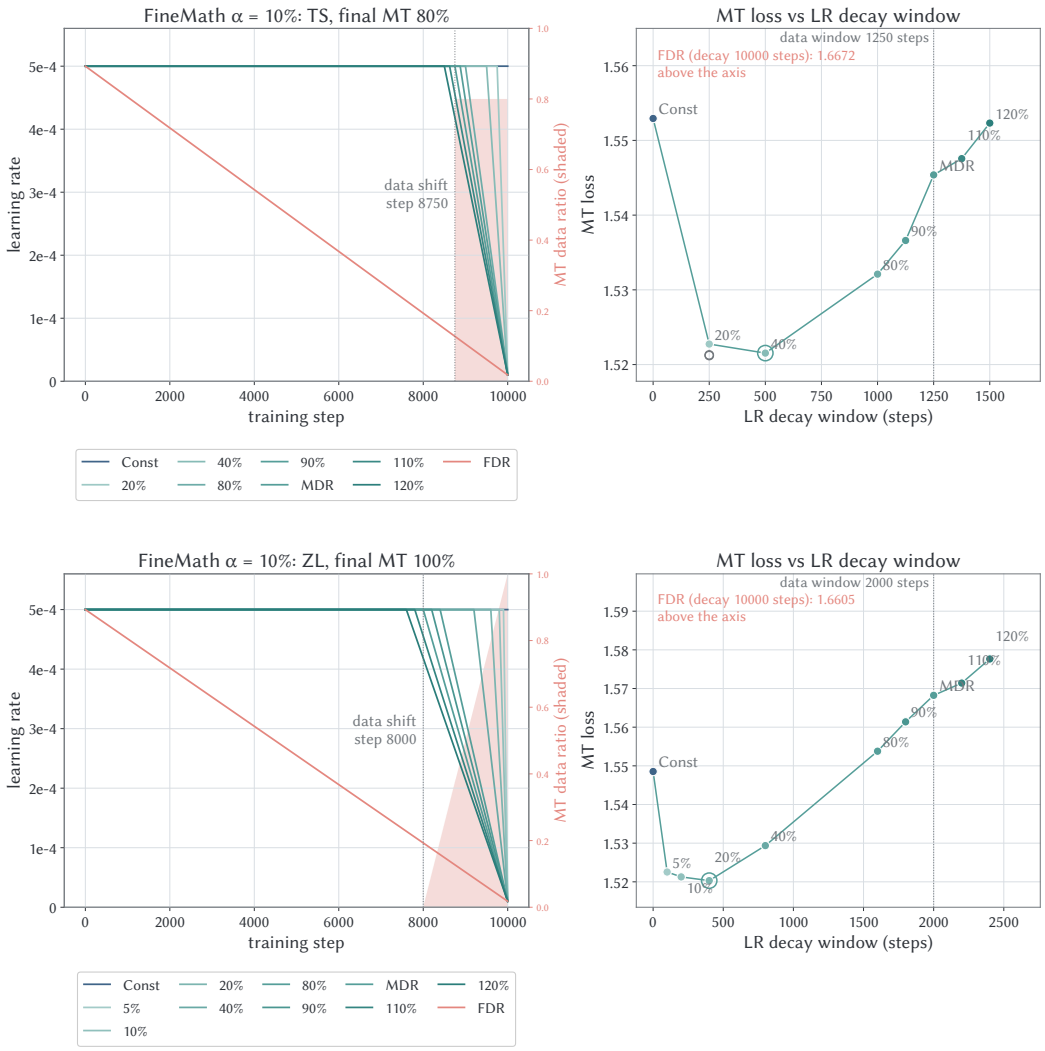


Figure 11: 10% MT-budget decay scans for the dense model. Top: TS ends at 80% MT and introduces MT at step 8,750, giving an MT data window of 1,250 steps. Its lowest tested loss is 1.5215 at a 40% LR decay fraction. Bottom: ZL ends at 100% MT and begins its ramp at step 8,000, giving an MT data window of 2,000 steps. Its lowest tested loss is 1.5203 at 20%. Left panels show the data and LR schedules. Right panels plot final MT loss against LR decay duration in steps, with point labels giving the duration as a percentage of each schedule's MT data window. Hollow TS markers show repeated runs at 20%. No repeated runs were evaluated at the 40% minimum.

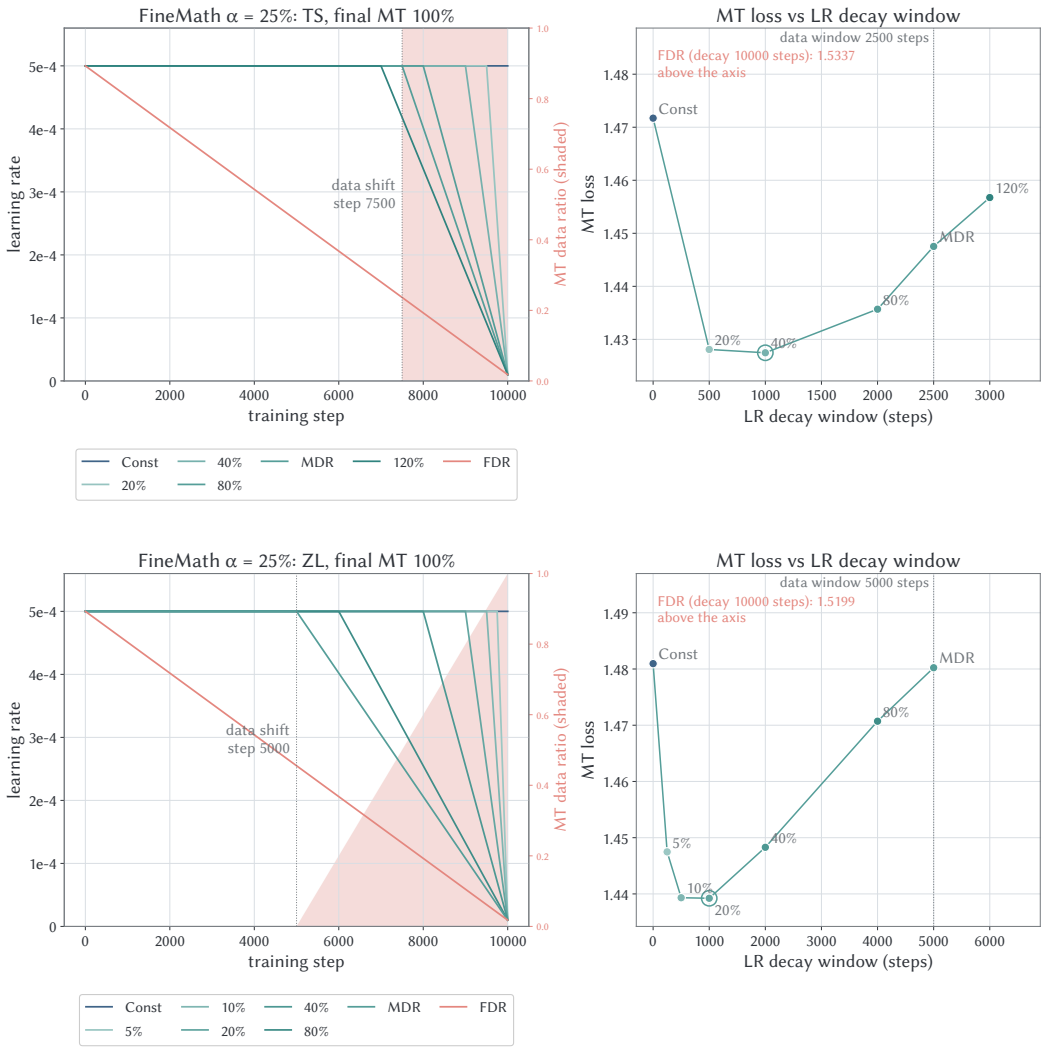


Figure 12: 25% MT-budget scans at a final MT ratio of 100% for the dense model. Top: TS has an MT data window of 2,500 steps and reaches its lowest tested loss of 1.4275 at a 40% LR decay fraction. Bottom: ZL has an MT data window of 5,000 steps and reaches its lowest tested loss of 1.4392 at 20%. Left panels show the data and LR schedules. Right panels plot final MT loss against LR decay duration in steps. Point labels give the duration as a percentage of each schedule's MT data window. Nearby short decay durations give similar losses, and these configurations have no repeated runs.

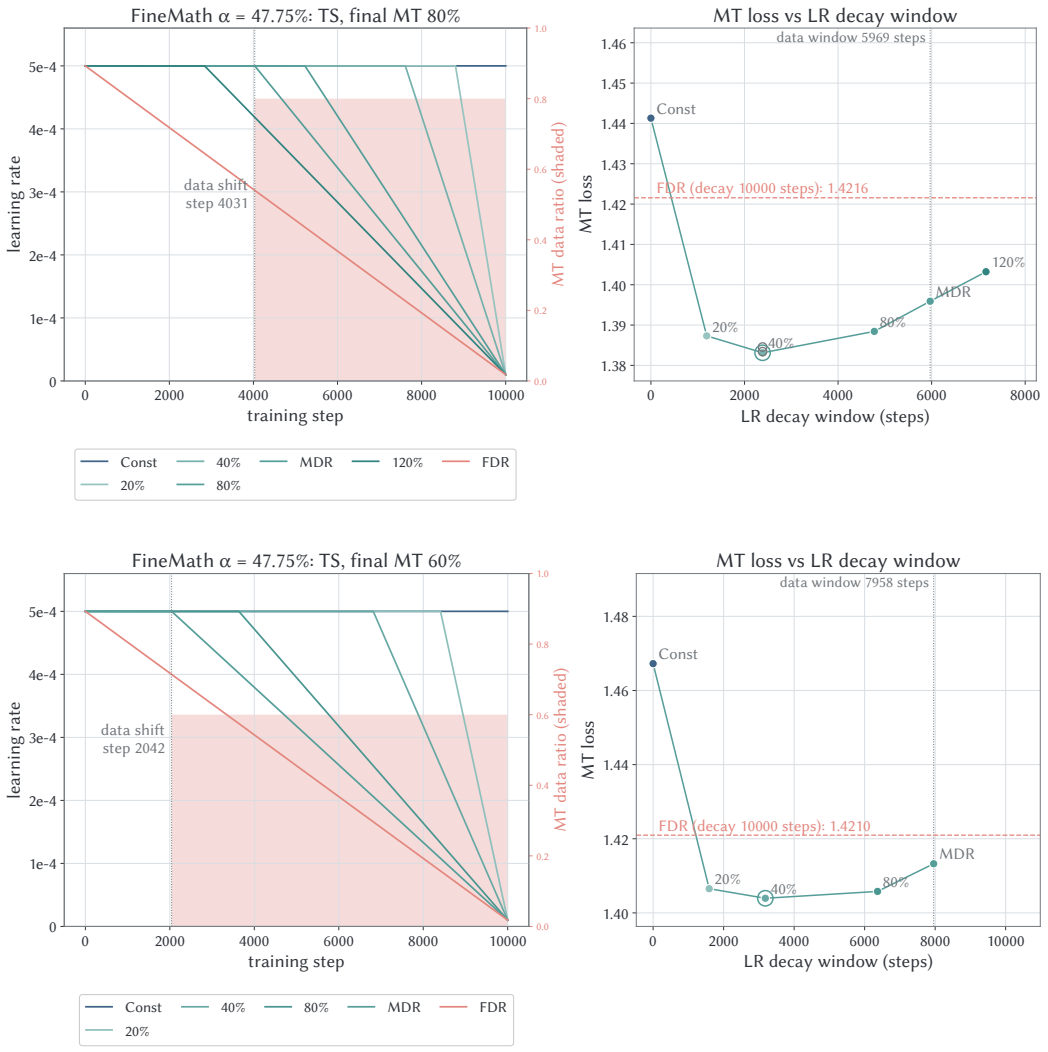


Figure 13: Additional TS scans at a 47.75% MT budget for the dense model. Top: a final MT ratio of 80% gives an MT data window of 5,969 steps. A 40% LR decay fraction yields a loss of 1.3831, with repeated runs at 1.3834 and 1.3846 shown as hollow points. Bottom: a final MT ratio of 60% gives an MT data window of 7,958 steps. A 40% LR decay fraction yields 1.4039, and fractions between 20% and 80% give similar losses. Left panels show the data and LR schedules. Right panels plot final MT loss against LR decay duration in steps. Point labels give the duration as a percentage of each schedule's MT data window. The scan ending at 100% MT also appears in Figure 3.

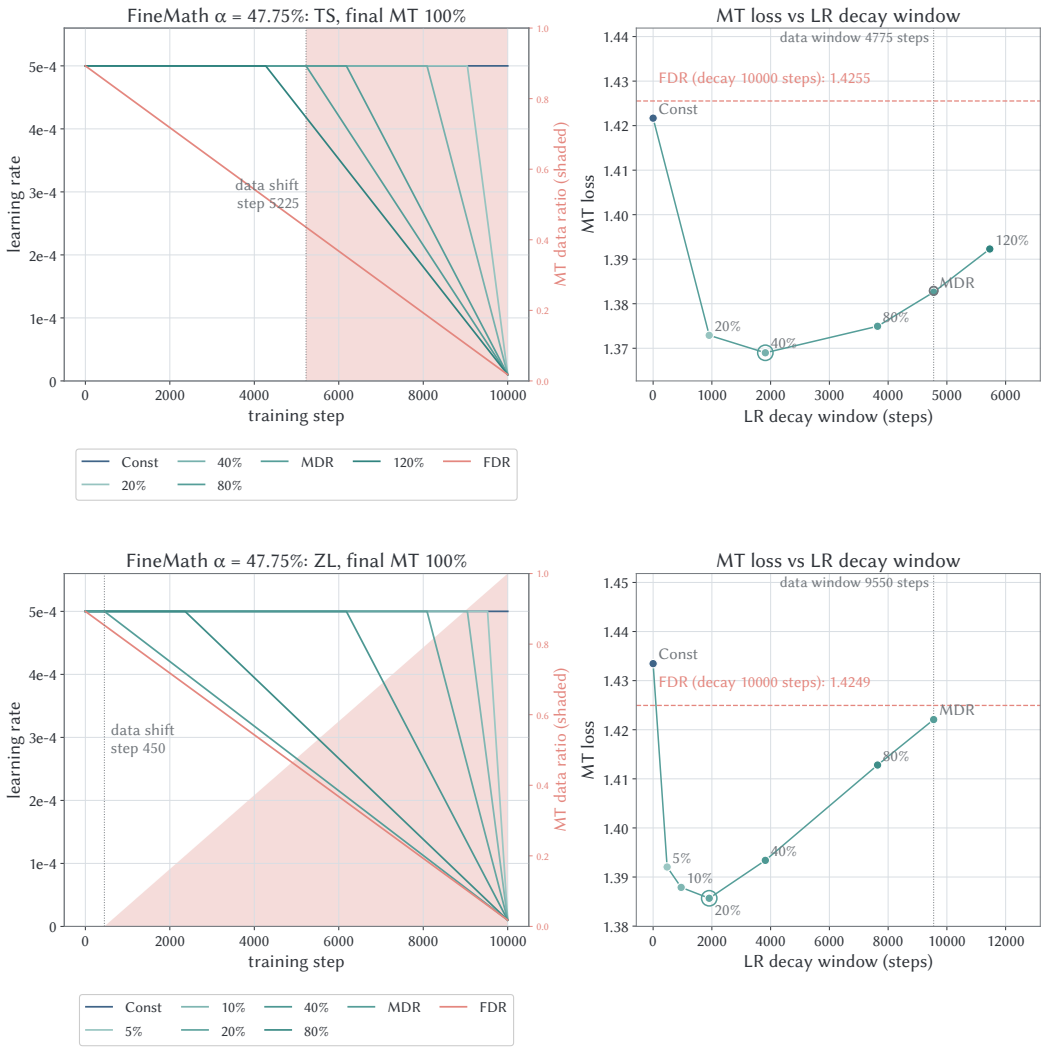


Figure 14: 47.75% MT-budget scans at a final MT ratio of 100% for the dense model. Top: TS introduces MT at step 5,225, giving an MT data window of 4,775 steps. Its lowest tested loss is 1.3690 at a 40% LR decay fraction. Bottom: ZL begins its ramp at step 450, giving an MT data window of 9,550 steps. Its lowest tested loss is 1.3857 at 20%. Left panels show the data and LR schedules. Right panels plot final MT loss against LR decay duration in steps. Point labels give the duration as a percentage of each schedule's MT data window. For ZL, MDR and FDR differ by only 450 decay steps and yield losses of 1.4221 and 1.4249. This ZL configuration has no repeated runs.

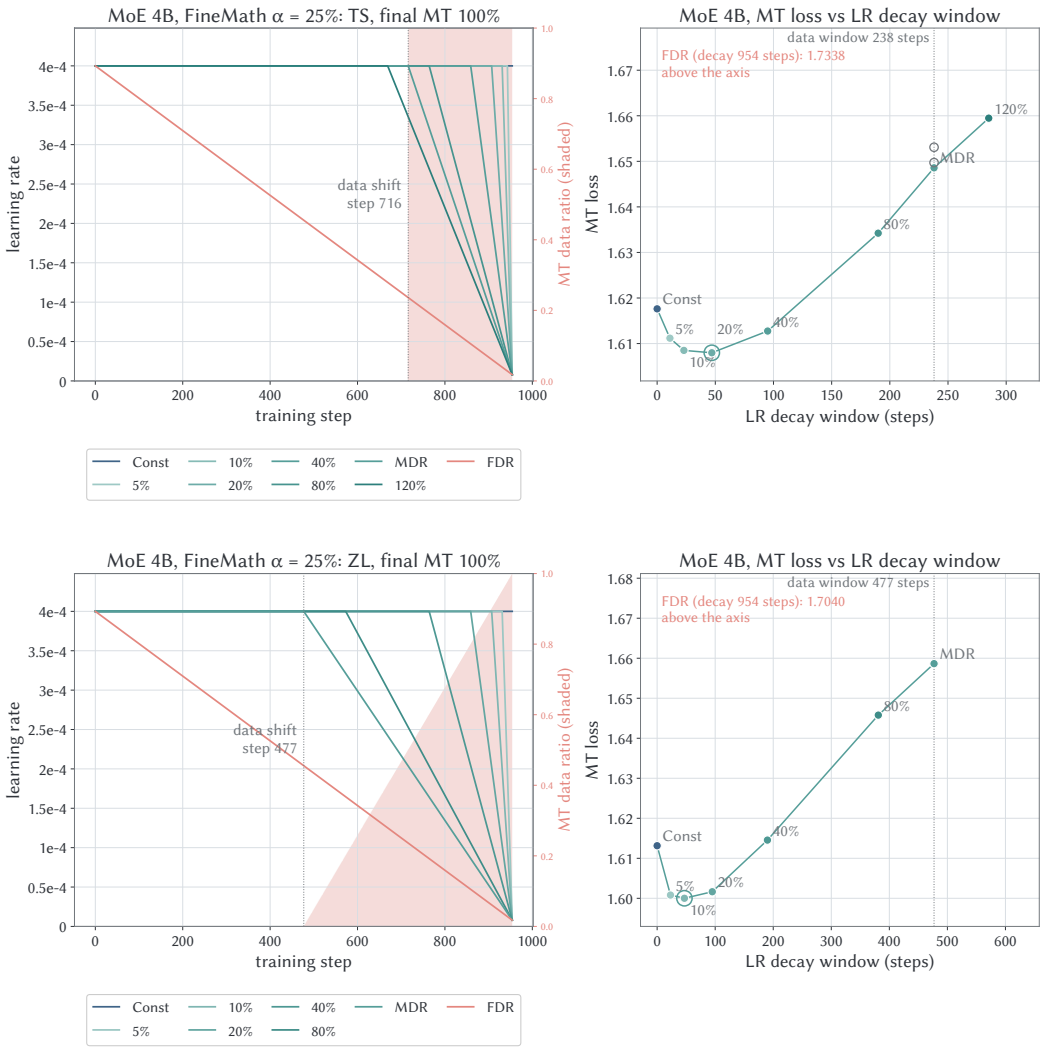


Figure 15: MoE scans over 4B continued training tokens, with a 25% MT budget and a final MT ratio of 100%. Top: TS has an MT data window of 238 steps and reaches its lowest tested loss of 1.6080 at a 20% LR decay fraction (47 decay steps). Bottom: ZL has an MT data window of 477 steps and reaches its lowest tested loss of 1.6000 at 10% (48 decay steps). Left panels show the data and LR schedules. Right panels plot final MT loss against LR decay duration in steps. Point labels give the duration as a percentage of each schedule's MT data window. Hollow TS markers show repeated MDR runs. No repeated runs were evaluated at the best tested fractions.

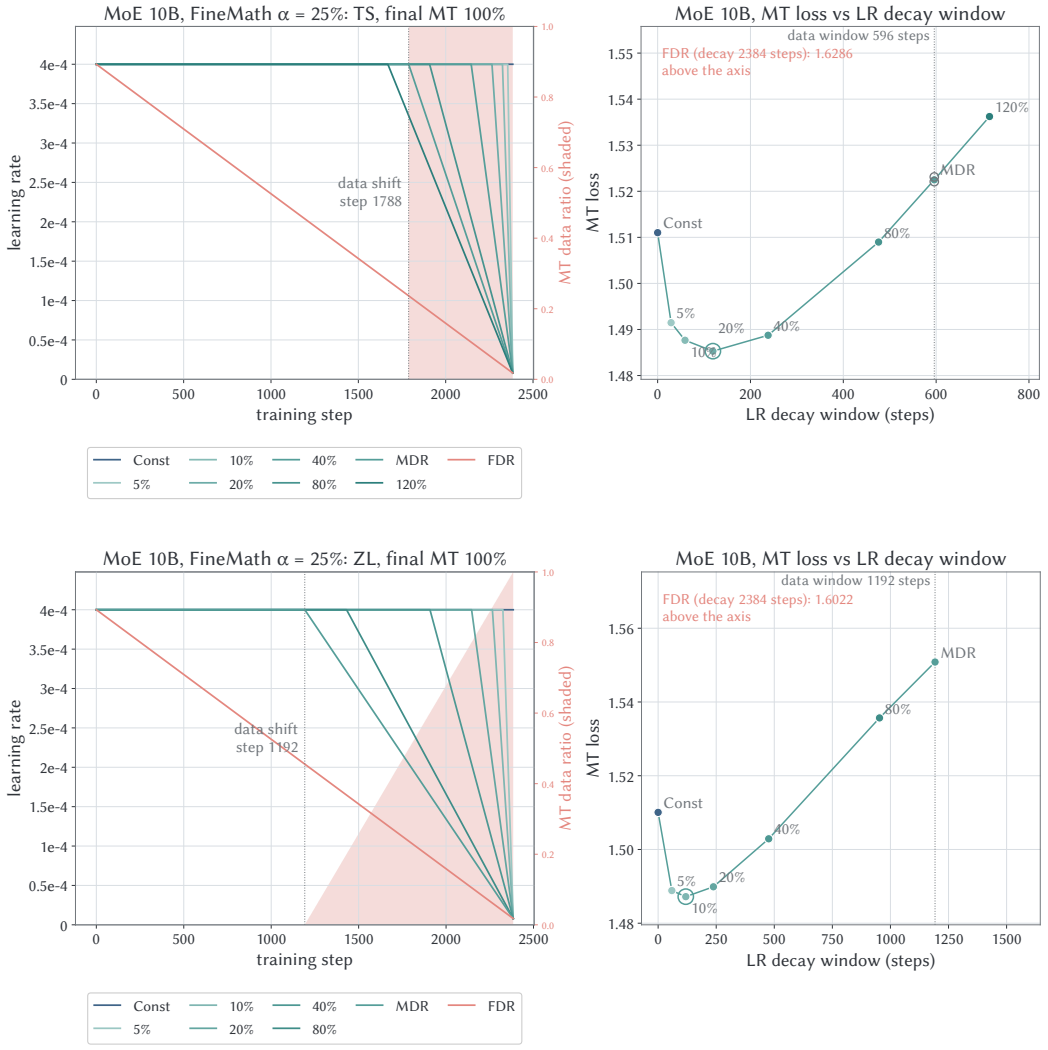


Figure 16: MoE scans over 10B continued training tokens, with a 25% MT budget and a final MT ratio of 100%. Top: TS has an MT data window of 596 steps and reaches its lowest tested loss of 1.4853 at a 20% LR decay fraction. Bottom: ZL has an MT data window of 1,192 steps and reaches its lowest tested loss of 1.4872 at 10%. Both preferred fractions give 119 decay steps. Left panels show the data and LR schedules. Right panels plot final MT loss against LR decay duration in steps. Point labels give the duration as a percentage of each schedule's MT data window. Hollow TS points show repeated MDR runs. The best TS and ZL configurations have not been compared using repeated runs with matched data sampling.

E Data Stream Replicas and Noise Estimates

For six selected configurations, we repeat training twice using different training examples or a different order of the same examples. All three runs start from the same PT checkpoint and use the same training setup. For the dense model, the two repeats begin one third and two thirds of the way through each dataset, respectively, returning to the beginning when they reach the end. At the 47.75% MT budget, each run uses almost the entire FineMath dataset, so the runs mainly differ in the order of MT examples. Each repeat shares about 36% of its DCLM examples with the original run. For the MoE model, the three runs instead use separate subsets of both PT and MT data, with no overlap between runs.

Configuration	Primary	Replica 1	Replica 2	Dispersion s_3
TS for the dense model, 10% MT, $\gamma_{\text{end}} = 80\%$, 20% LR decay fraction	1.5227	1.5213	1.5212	0.000685
TS for the dense model, 47.75% MT, $\gamma_{\text{end}} = 100\%$, MDR	1.3826	1.3828	1.3830	0.000163
TS for the dense model, 47.75% MT, $\gamma_{\text{end}} = 70\%$, FDR	1.4194	1.4198	1.4217	0.001003
TS for the dense model, 47.75% MT, $\gamma_{\text{end}} = 80\%$, 40% LR decay fraction	1.3831	1.3834	1.3846	0.000648
MoE 4B, TS $\gamma_{\text{end}} = 100\%$, MDR	1.6486	1.6497	1.6531	0.001915
MoE 10B, TS $\gamma_{\text{end}} = 100\%$, MDR	1.5225	1.5221	1.5231	0.000411

Table 9: Final MT losses at all six repeated configurations. The descriptive dispersion is $s_3 = \sqrt{\frac{1}{3}\sum_{j=1}^{3}(\ell_j - \bar{\ell})^2}$, calculated from displayed rounded losses. It uses denominator three rather than the denominator-two sample-standard-deviation convention. Replica order is immaterial.

Repeated runs are shown as hollow points. We use only the original runs to fit the curves and identify the best tested configuration. These repeats show how the loss changes when we change the training examples or their order at the six selected configurations. All runs start from the same PT checkpoint, so they do not capture variation across independently pretrained models. The repeats for the dense model also share many training examples. Repeating an MDR run does not tell us how much the loss would vary at other LR decay fractions or with other data schedules. We therefore do not treat the small loss differences between nearby ZL decay fractions at a 25% MT budget for the dense model, or between the best TS and ZL settings for MoE 10B, as evidence that one setting reliably outperforms another.

F MoE Results and Additional Ablations

Our main MoE experiments compare approximately 4B and 10B tokens of continued training, with MT data accounting for 25% in both settings. These runs last 954 and 2,384 steps and use approximately 1.0B and 2.5B MT tokens, respectively, corresponding to 244,224 and 610,304 packed FineMath sequences. The 10B setting uses approximately 25.6% of the FineMath dataset. For the original runs, continued training uses DCLM shards 415–454 in the 4B setting and 415–494 in the 10B setting, while PT uses shards 0–414. Table 10 lists all tested final MT ratios and the training step at which MT data first enters for each schedule. Figure 5 also shows results for 10B tokens of continued training with a 47.75% MT budget.

LR dependent preferences persist, but MDR is not uniformly better than Const. Figures 17 and 18 show the final PT and MT losses for all tested configurations. At both horizons, TS+Const favors $\gamma_{\text{end}} = 100\%$, whereas TS+FDR favors $\gamma_{\text{end}} = 25\%$. At 4B, the best TS+MDR loss is worse than the best TS+Const loss, and MDR's own minimum occurs at $\gamma_{\text{end}} = 50\%$. This does not assert pointwise dominance: at the $\gamma_{\text{end}} = 25\%$ boundary, MDR and FDR coincide. At 10B, MDR is nearly flat for the non-boundary endpoints: its loss is 1.5225 at $\gamma_{\text{end}} = 100\%$, 1.5236 at 75%, and 1.5245 at 50%. The MDR-minus-Const difference at $\gamma_{\text{end}} = 100\%$ shrinks from $+0.0310$ at 4B to $+0.0115$ at 10B; at $\gamma_{\text{end}} = 50\%$ in the 10B experiments, MDR is 0.0014 below Const. The approximately 63%

Data schedule	Final MT ratio	4B first MT step	10B first MT step
TS	100%	716	1,788
	75%	636	1,589
	50%	477	1,192
	40%	358	894
	25%	0	0
ZL	100%	477	1,192
	75%	318	795
	50%	0	0

Table 10: Tested data schedules for the MoE model at a 25% MT budget. The first MT step denotes the TS switch or ZL ramp start. The TS $\gamma_{\text{end}} = 100\%$ data windows are 238 and 596 steps.

Horizon	Data schedule	Const	FDR	MDR
4B	TS	1.6176 (100)	1.6349 (25)	1.6343 (50)
	ZL	1.6131 (100)	1.6548 (50)	1.6548 (50)
10B	TS	1.5110 (100)	1.5343 (25)	1.5225 (100)
	ZL	1.5101 (100)	1.5548 (50)	1.5508 (100)

Table 11: Lowest final MT loss among the tested final MT ratios for each training duration, data schedule, and LR schedule in the MoE model at a 25% MT budget. Parentheses give the final MT ratio γ_{end} in percent. Both 4B MDR minima occur at $\gamma_{\text{end}} = 50\%$, so MDR does not select $\gamma_{\text{end}} = 100\%$ in all model/horizon groups.

reduction of the gap at 100% MT is consistent with sensitivity to adaptation time, but extending the horizon increases total PT and MT tokens together and does not isolate a single mechanism.

TS versus ZL. At matched endpoints, ZL improves MT loss under FDR and worsens it under MDR in both horizons (Table 12). Const gives smaller differences, with the 4B advantage of ZL largely disappearing at 10B. After separately selecting each shape's best scanned window, ZL gives 1.6000 versus TS's 1.6080 at 4B, whereas TS gives 1.4853 versus ZL's 1.4872 at 10B. This ordering change prevents a general claim that either shape is preferable for MoE training. The scans appear in Appendix D; Figure 19 compares the TS horizon response directly.

Horizon and LR	$\gamma_{\text{end}} = 100\%$	$\gamma_{\text{end}} = 75\%$	$\gamma_{\text{end}} = 50\%$
4B Const	-0.0045	-0.0029	-0.0028
4B FDR	-0.030	-0.028	-0.023
4B MDR	$+0.010$	$+0.016$	$+0.021$
10B Const	-0.0010	$+0.0010$	$+0.0015$
10B FDR	-0.026	-0.025	-0.018
10B MDR	$+0.028$	$+0.028$	$+0.030$

Table 12: MoE MT loss difference, ZL minus TS, at matched final MT ratios, with the precision reported in the source comparison table. Negative values favor ZL. The 10B Const difference at 100% MT is approximately -0.001 (the individually rounded losses differ by -0.0009). These are primary-run comparisons, not significance tests.

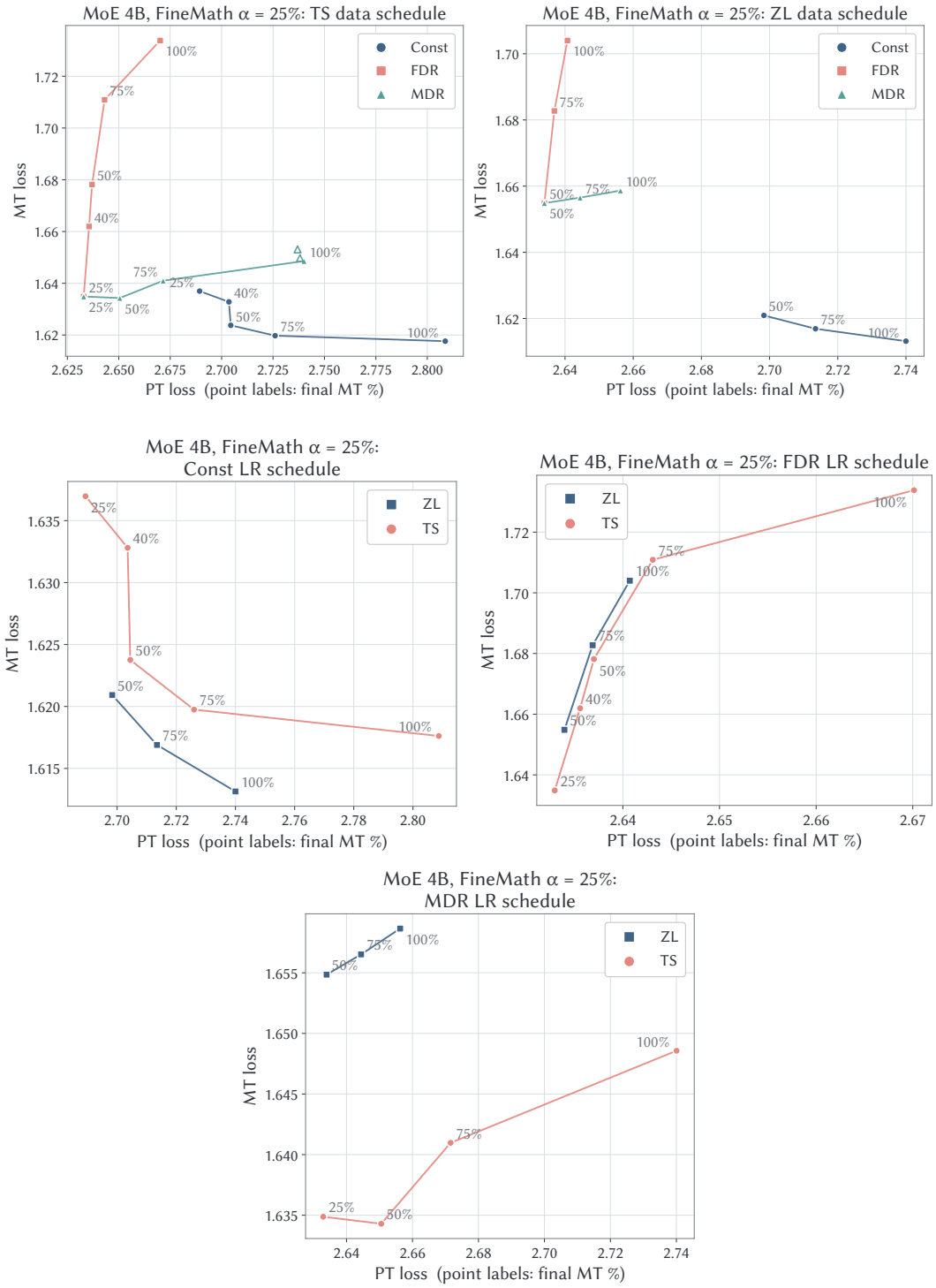


Figure 17: Complete MoE results for 4B tokens of continued training with a 25% MT budget. Top: TS and ZL under all three LR rules. Remaining panels: the same final PT loss (horizontal) and MT loss (vertical) regrouped by LR. Labels give the final MT ratio in percent; hollow points repeat TS+MDR at 100% MT. Const gives the lowest basic-rule MT losses, while MDR selects 50% MT for both data shapes.

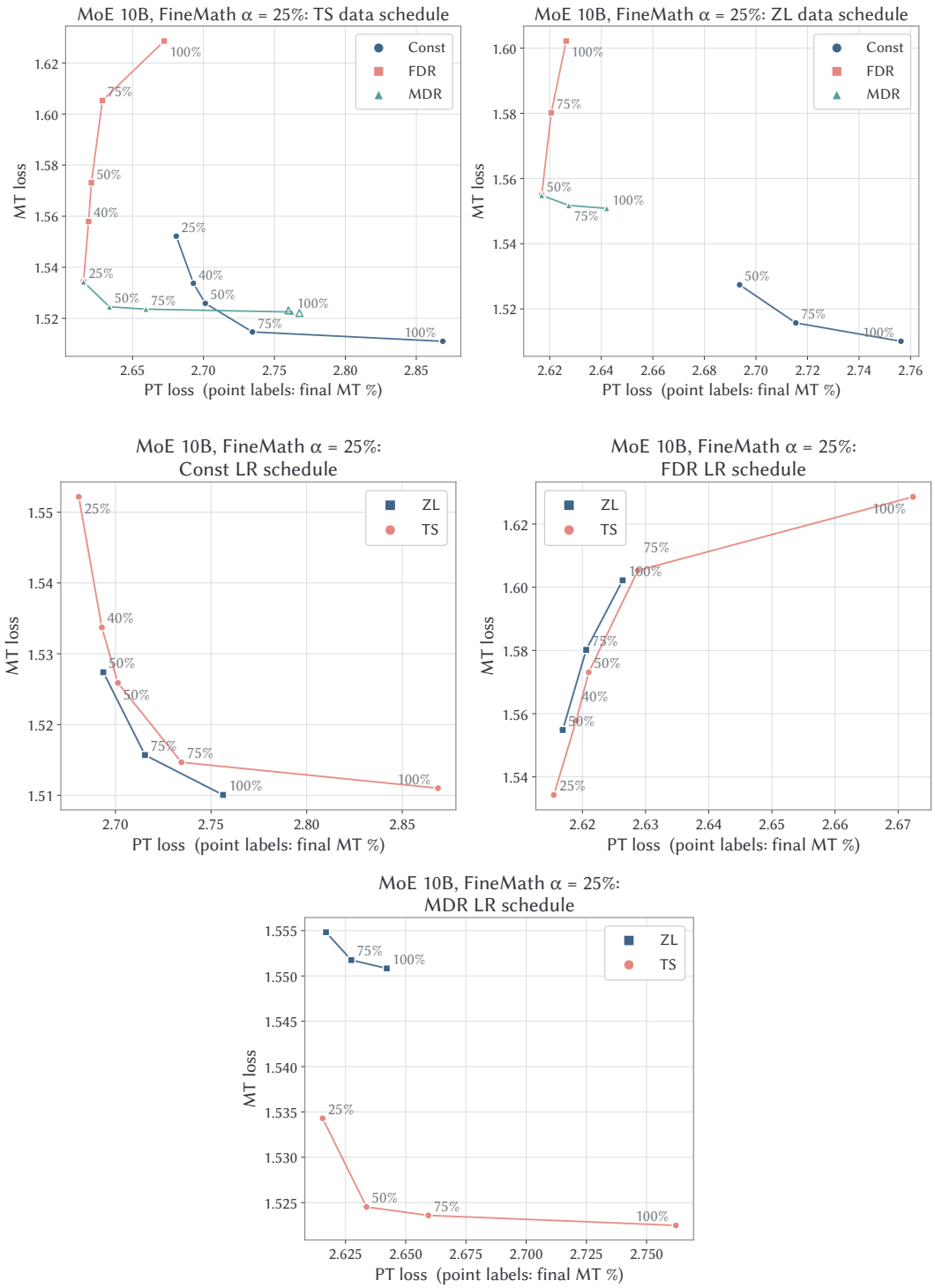


Figure 18: Complete MoE results for 10B tokens of continued training with a 25% MT budget. Top: TS and ZL with all three LR rules. Remaining panels: the same final PT–MT loss coordinates grouped by LR. Labels give the final MT ratio in percent; hollow points are TS+MDR replicas at 100% MT. Relative to 4B, the gap between MDR and Const at 100% MT is smaller for TS, while ZL+MDR remains worse than ZL+Const.

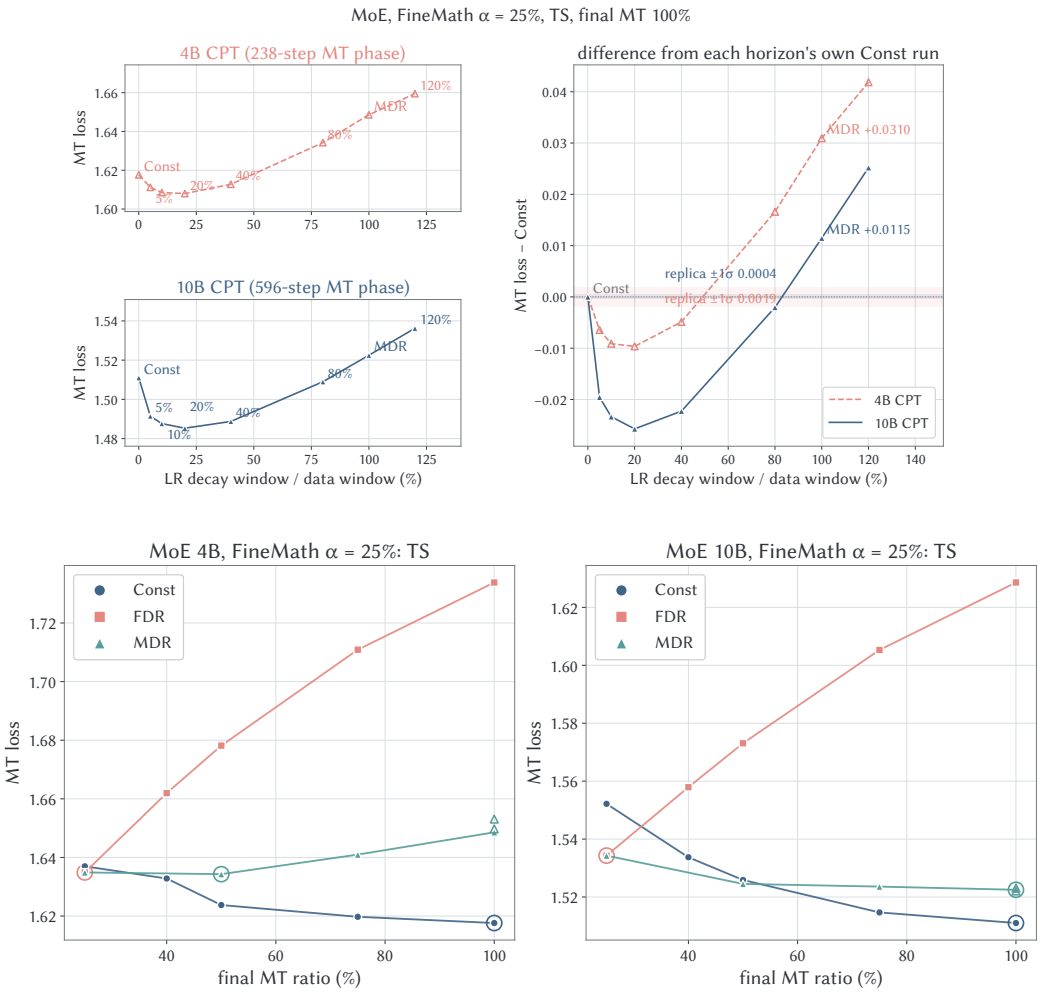


Figure 19: **MoE horizon comparison for TS at 25% MT budget.** Top: scans at a final MT ratio of 100%, showing absolute final MT loss and the change from each horizon's own Const baseline against learning rate decay range fraction. Shaded bands reflect variation of selected MDR replicas, without estimating uncertainty for every window. Bottom: final MT loss against final MT percentage at both horizons. The best scanned window stays at 20%, while the MDR–Const gap at a final MT ratio of 100% decreases from 0.0310 to 0.0115.

A larger MT budget at the 10B horizon. We increase the MT budget from 25% to 47.75% while retaining the 40B PT checkpoint and the 10B continuation horizon of 2,384 steps. This group, named `fm48` in the experiment files, uses approximately 4.77B FineMath tokens. The optimizer, data sources, batch size, and peak and final LRs are unchanged. TS is tested at final MT ratios of 100%, 90%, 80%, 60%, and 47.75%. ZL is tested at $\gamma_{\mathrm{end}} = 100\%$. The group contains 28 runs across the basic LR rules and the additional decay windows. At $\gamma_{\mathrm{end}} = 100\%$, the TS switch occurs at step 1,246 and the ZL ramp starts at step 107, giving MT data windows of 1,138 and 2,277 steps, respectively.

The ranking reversal phenomenon between Const and FDR persists at this larger budget (Figure 20). Const favors a final MT ratio of 100%, with MT loss 1.4292, while FDR favors the lowest tested final MT ratio, 47.75%, with loss 1.4528. The range of FDR losses across TS endpoints decreases from 0.094 at the 25% budget to 0.047 at the 47.75% budget. At $\gamma_{\mathrm{end}} = 100\%$, MDR gives 1.4390, remaining 0.0098 above Const despite the longer MT data window. At this endpoint, ZL improves on TS under FDR by 0.0188, but has higher MT loss under Const and MDR by 0.0132 and 0.0380, respectively. Thus, increasing the MT budget preserves the dependence on the LR rule without making either MDR or a smooth data transition uniformly preferable.

A short final decay improves both data shapes (Figure 21). The best tested TS window covers 20% of its MT data window and reaches 1.4007, while the best tested ZL window covers 10% and reaches 1.4110. Both selected windows span 228 steps because ZL has an MT data window approximately twice as long. Relative to their own Const baselines, these losses are lower by 0.0285 and 0.0314, respectively. Table 8 reports all sampled values, and Table 7 places these minima alongside the other sweeps. No data stream replicas were run for this budget, so the comparisons describe the observed runs rather than uncertainty across repetitions.

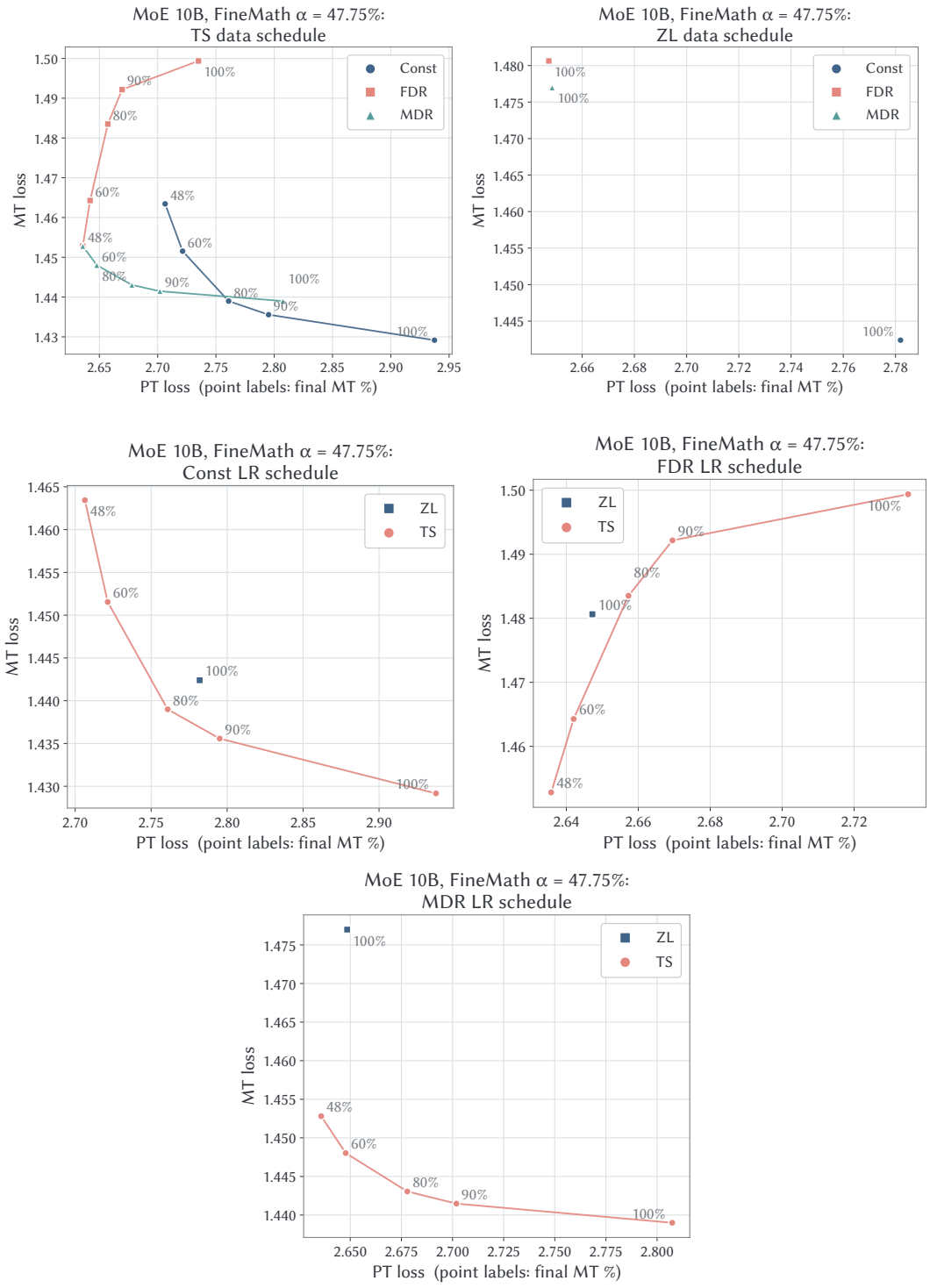


Figure 20: MoE 10B results at a 47.75% MT budget, starting from the 40B PT checkpoint. Top panels show TS and ZL under Const, FDR, and MDR. Remaining panels regroup the same final PT and MT losses by LR rule. Point labels give the final MT ratio in percent, with 47.75% rounded to 48%. ZL is tested only at 100% MT. Const and FDR favor opposite TS endpoints, while ZL improves on TS under FDR and has higher MT loss under Const and MDR at their shared endpoint. Each point is one run.

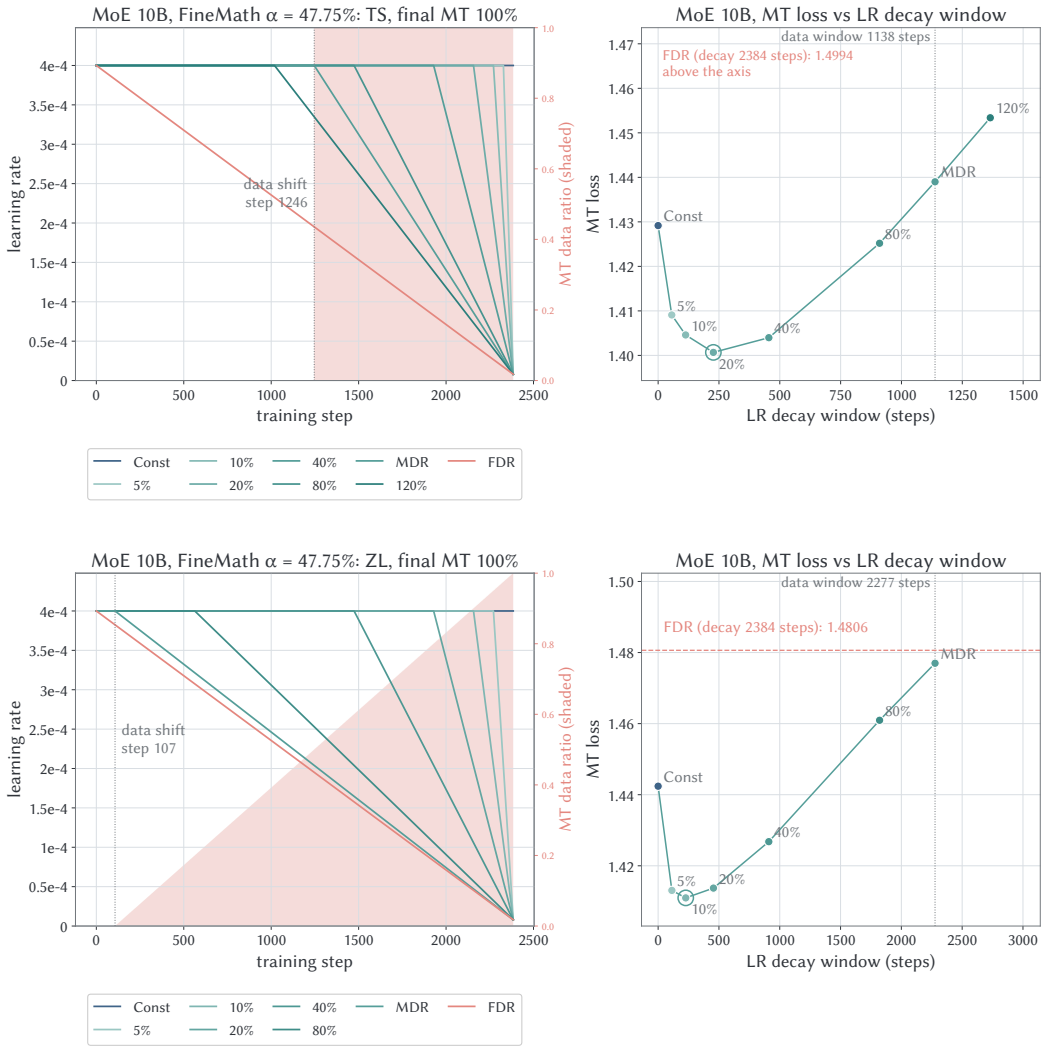


Figure 21: Decay scans for MoE 10B with a 47.75% MT budget and final MT ratio 100%. Top: TS, with a 1,138-step MT data window and its lowest tested MT loss at 20%. Bottom: ZL, with a 2,277-step MT data window and its lowest tested loss at 10%. Both selected decays last 228 steps. Left panels show the data and LR schedules. Right panels plot final MT loss against decay duration in steps, with labels expressing duration as a percentage of each schedule's MT data window. No configurations in these scans have data stream replicas.

Earlier checkpoints from the same PT trajectory. We repeat a subset of the MoE 4B experiments from checkpoints saved after 4,800 and 7,200 PT steps, corresponding to approximately 20.1B and 30.2B tokens. We call these the 20B and 30B starting points and compare them with the main 40B checkpoint at step 9,536. All three checkpoints come from the same PT run. Each continuation uses 954 steps, a fixed 25% MT budget, and the same training configuration as the main MoE 4B experiments, including optimizer reinitialization and no LR warmup. At each earlier checkpoint, we test nine TS runs: Const and FDR at final MT ratios of 100%, 50%, and 25%, plus MDR and 20% and 40% decay windows at a final MT ratio of 100%. At a final MT ratio of 25%, mixing begins at step zero, so MDR and FDR share the same run. These groups do not include ZL runs or data stream replicas.

When we compare the same final MT ratios, changing the PT checkpoint preserves the ranking reversal phenomenon (Figure 22). At all three checkpoints, Const favors a final MT ratio of 100%, while FDR favors 25%. Longer PT lowers the final MT loss at each shared configuration. For example, Const at a final MT ratio of 100% gives 1.6666, 1.6409, and 1.6176 after 20B, 30B, and 40B PT tokens, respectively. The corresponding FDR losses exceed Const by 0.1233, 0.1143, and 0.1162. At a final MT ratio of 25%, FDR instead improves on Const by 0.0072, 0.0049, and 0.0021. These comparisons show that the ranking reversal phenomenon is present across the sampled checkpoints, without requiring that the model first completes the full 40B PT run.

The preference for a short final decay also persists (Figures 23 and 24). At a final MT ratio of 100%, MT data enters at step 716 and occupies the last 238 steps. The lowest tested MT loss occurs at a 20% window for all three checkpoints, improving on Const by 0.0104, 0.0149, and 0.0096 for the 20B, 30B, and 40B starting points, respectively. MDR instead increases MT loss by 0.0303, 0.0264, and 0.0310. The earlier checkpoints test only the 20% and 40% windows in addition to Const, MDR, and FDR. They therefore support the same ordering among these choices, but do not locate a continuous optimum or establish the shape of the curve near its minimum. Full values for the earlier checkpoints appear in Table 8.

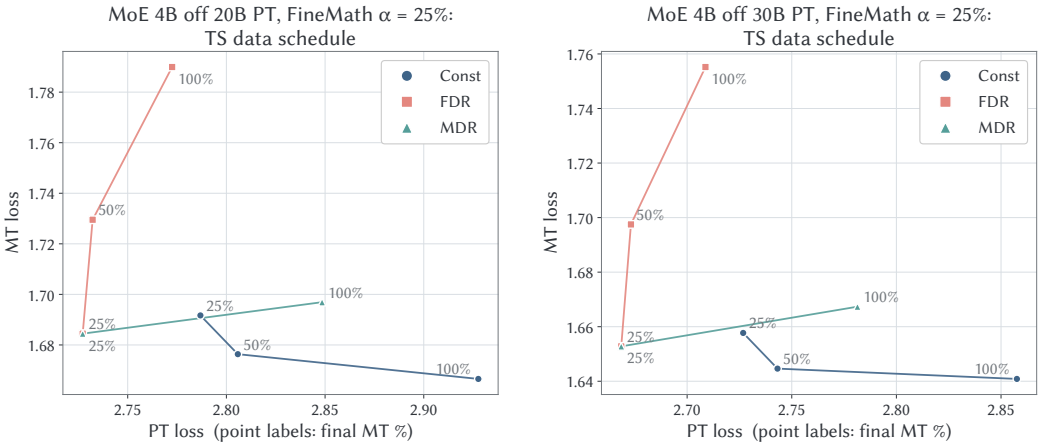


Figure 22: TS continual training results from earlier checkpoints of the same MoE PT run. Left: the 20B starting point. Right: the 30B starting point. Both use a 4B continuation horizon and a 25% MT budget. Const and FDR each have three tested endpoints, with an additional MDR run at 100% MT and the FDR run reused at 25% MT, where the two LR rules coincide. Axes show final PT and MT losses, and point labels give the final MT ratio in percent. Const favors the final pure MT block and FDR favors mixing from the start at both checkpoints. No ZL or replica runs are included.

Changing the MT dataset. StarCoderData is a collection of source code from GitHub released for training StarCoder (Li et al., 2023). It is derived from The Stack, with additional cleaning, removal of near duplicates, and filtering of personal information. We use its Python subset as MT data to test whether the preferred LR decay fractions persist when the target domain changes from mathematics to code.

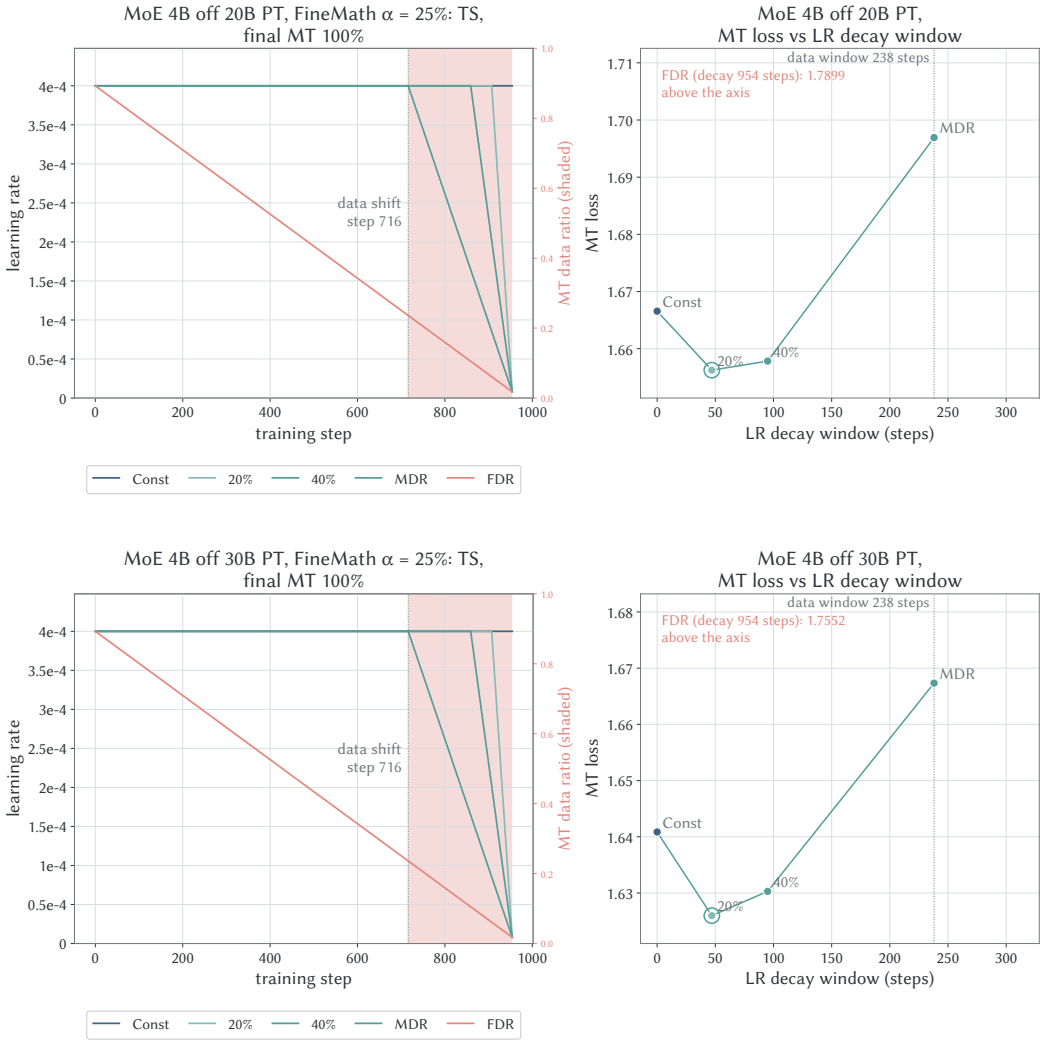


Figure 23: TS decay scans after 20B PT tokens (top) and 30B PT tokens (bottom), with a 4B continuation horizon, 25% MT budget, and final MT ratio 100%. Both configurations have a 238-step MT data window. Only 20% and 40% windows are tested in addition to Const, MDR, and FDR. The lowest observed losses are 1.6562 and 1.6260 at 20% (47 decay steps). Left panels show schedules and right panels show final MT loss. Lines connect tested values and do not imply that intervening durations were evaluated.

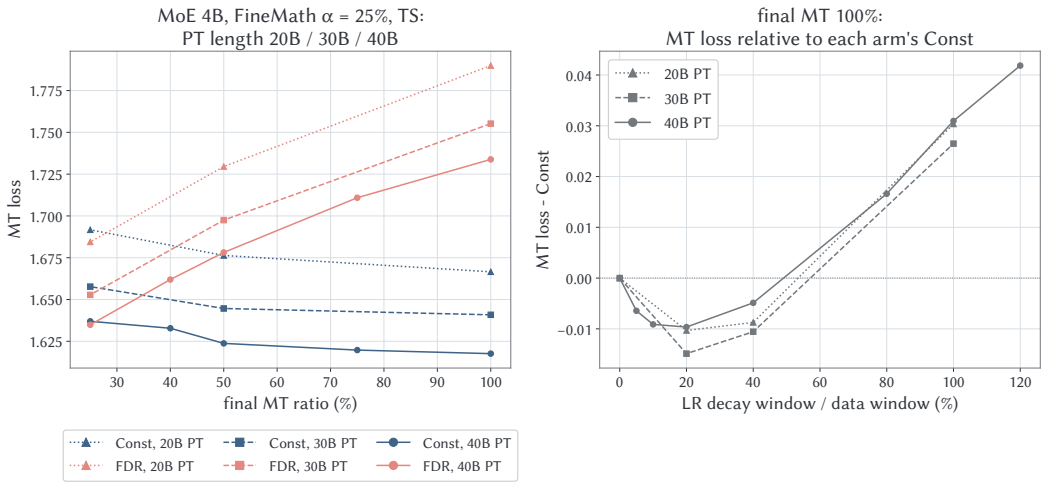


Figure 24: MoE TS results from 20B, 30B, and 40B checkpoints of the same PT trajectory, with a fixed 4B continuation horizon and 25% MT budget. Left: final MT loss against final MT ratio under Const and FDR. Right: the change in final MT loss relative to each checkpoint's own Const baseline at a final MT ratio of 100%, plotted against decay duration as a percentage of the MT data window. The 40B checkpoint has additional sampled durations. Across the configurations tested at all three checkpoints, Const and FDR prefer opposite data schedules, 20% decay gives the lowest observed loss, and MDR remains worse than Const.

We replace FineMath with StarCoderData Python, starting from the same 40B PT checkpoint and continuing training for 10B tokens with a 25% MT budget. Both TS and ZL end at a final MT ratio of 100%. PT data, tokenizer, batch size, and LR settings remain unchanged. These two scans add 15 runs. Both include Const and MDR, while FDR is tested only for TS. ZL does not include a 5% window, and neither scan includes data sampling replicas.

Changing MT data from mathematics to code preserves the best tested LR decay fraction. Figure 25 compares each scan with its FineMath counterpart after subtracting its own Const loss. For both datasets, the lowest tested loss occurs at 20% for TS and 10% for ZL, corresponding to the same 119 steps of LR decay. The improvement over Const differs between datasets, but the best fractions among those tested remain unchanged. MT loss is evaluated on each dataset separately, so absolute losses should not be compared across the two datasets.

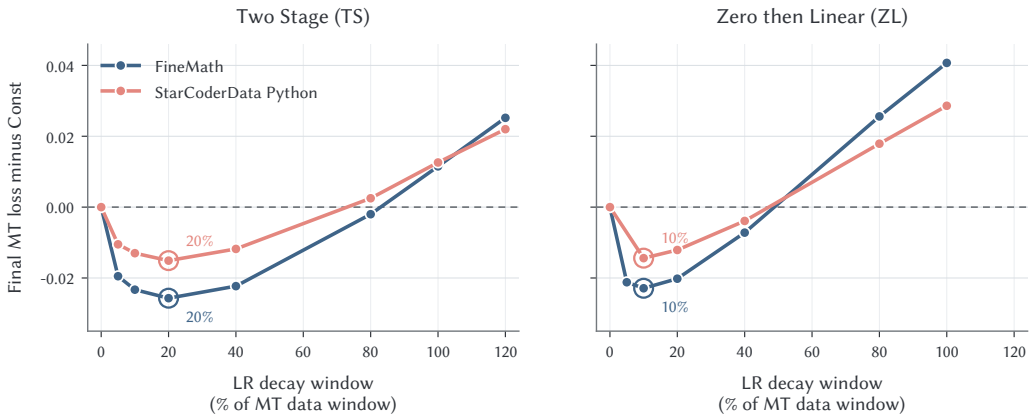


Figure 25: **Changing MT data from mathematics to code preserves the best tested LR decay fraction** for both TS (left) and ZL (right). Both datasets use the same MoE 10B setup with a 25% MT budget and final MT ratio 100%. The horizontal axis gives decay duration as a percentage of each schedule's MT data window, with zero denoting Const and 100% denoting MDR. The vertical axis gives final MT loss minus the corresponding Const loss for that dataset and data schedule. Points show measured values, and rings mark the lowest tested losses at 20% for TS and 10% for ZL. Lines connect tested values.

G Router Dynamics and Interpretation

Measurement. Training logs record an aggregated load balancing statistic every 20 steps. We also evaluate routing separately on held-out DCLM and FineMath, using approximately 34M tokens per domain at each checkpoint. For each of the 12 layers, let $p_{\ell j}$ be the fraction of routing assignments sent to expert j, normalized to sum to one across the 64 experts. We measure **expert load imbalance** using

$$I_\ell = 64 \sum_{j=1}^{64} p_{\ell j}^2. \tag{7}$$

Uniform routing gives $I_\ell = 1$, and larger values indicate greater expert load imbalance. The heatmaps and regressions express expert shares relative to uniform routing as $64p_{\ell j}$, so a value of one denotes the uniform share. These checkpoint diagnostics describe routing on each domain and differ from the aggregate statistic logged on the changing training mixture.

Abrupt data distribution shifts under TS produce transient imbalance. At both 4B and 10B continuation horizons, the TS data switch produces a sharp load balancing spike that subsides over approximately 80 steps (Figures 27 and 28). Larger jumps in the MT ratio produce larger spikes, whereas ZL avoids a comparable peak as MT data enters gradually. The effect is therefore concentrated around the change in training distribution.

ZL reduces load balancing spikes under the same LR schedule. To separate the effect of the data transition from that of LR decay, we compare TS and ZL under Const. Both schedules use the same LR at every training step, yet TS produces a sharp load balancing spike while ZL avoids a comparable spike (Figures 6 and 26). This difference persists across both training lengths and all three tested final MT ratios, supporting the benefit of gradually introducing MT data.

Expert loads change as the router adapts to MT data. To understand the load balancing spike under TS, we examine checkpoints before the data switch, during MT training, and at training completion in the MoE 10B TS+Const run ending at 100% MT. At each checkpoint, we separately evaluate how the router distributes DCLM (PT) and FineMath (MT) tokens across experts. Before the switch, expert loads are nearly balanced on DCLM. However, evaluating the same checkpoint on FineMath shows that a few experts receive many more tokens than the others, even though MT data has not yet entered training. This suggests that the spike occurs because the routing learned on PT data initially assigns MT tokens unevenly across experts. As MT training proceeds, the router distributes FineMath tokens more evenly, and the measured imbalance falls substantially (Table 13). The FineMath heatmaps in Figure 29 show this change directly: experts that receive a large share of tokens before the switch become less dominant by training completion. Meanwhile, expert loads on DCLM become somewhat less balanced, showing that adaptation to MT data also changes how the router distributes PT tokens.

10B TS+Const checkpoint	DCLM imbalance	FineMath imbalance
1,750 (before switch)	1.005	1.216
2,250 (462 steps after switch)	1.029	1.006
2,384 (final)	1.043	1.015

Table 13: Expert load imbalance on DCLM and FineMath for the MoE 10B TS+Const run ending at 100% MT. Each value is the average of I_ℓ across the 12 layers, as defined in Equation (7). The first checkpoint is 38 steps before the data switch. These measurements evaluate each domain separately on validation data, whereas the training logs summarize routing on the changing mixture of PT and MT data.

How expert loads change under TS and ZL. The preceding analysis shows that expert loads become more balanced after the TS switch. We next examine which experts receive more or fewer FineMath assignments after adaptation and compare these changes under TS and ZL. In panel (b) of Figure 30, the horizontal axis shows how much each expert's initial assignment share differs from the uniform share, and the vertical axis shows how its share changes by training completion. The

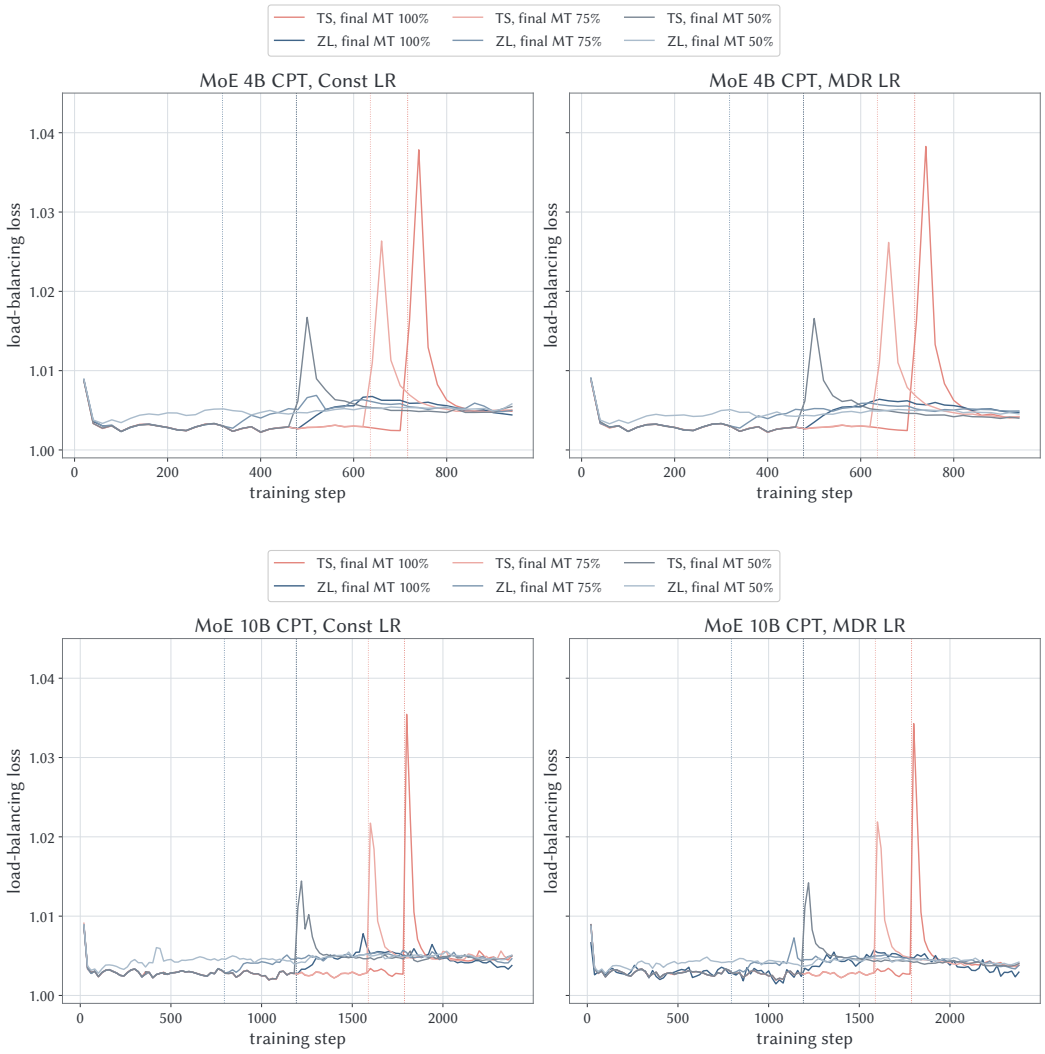


Figure 26: **Gradual MT entry reduces load balancing spikes.** Under Const (left), TS and ZL use the same LR at every step, allowing us to compare the data schedules without also changing the LR. TS produces a sharp spike when the MT ratio suddenly increases, while ZL avoids a comparable spike by introducing MT data gradually. MDR (right) shows a similar pattern, although LR decay begins at different times for TS and ZL. Rows show MoE experiments with 4B (top) and 10B (bottom) tokens of continued training. Curves compare final MT ratios of 100%, 75%, and 50%. Dotted lines mark when MT data first enters. The 10B results also appear in Figure 6.

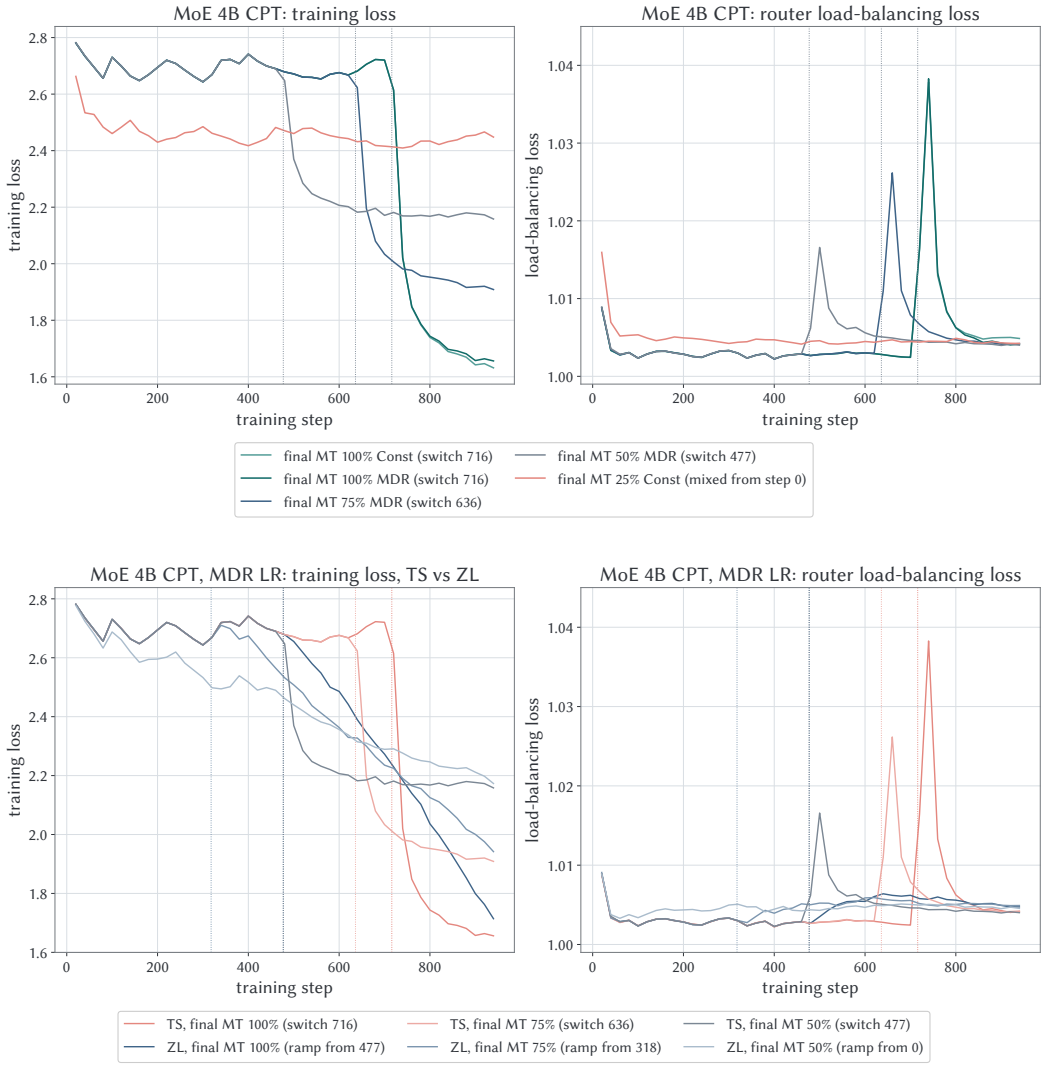


Figure 27: Router switching dynamics during MoE 4B continued training. Top: TS runs at several final MT ratios, including Const and MDR at 100% MT. Bottom: TS and ZL under MDR at final MT ratios of 100%, 75%, and 50%. Left panels show training loss; right panels show the logged load-balancing statistic against training step, with one logged point every 20 steps. These are time-series measurements, not final validation-loss coordinates or replica error bars.

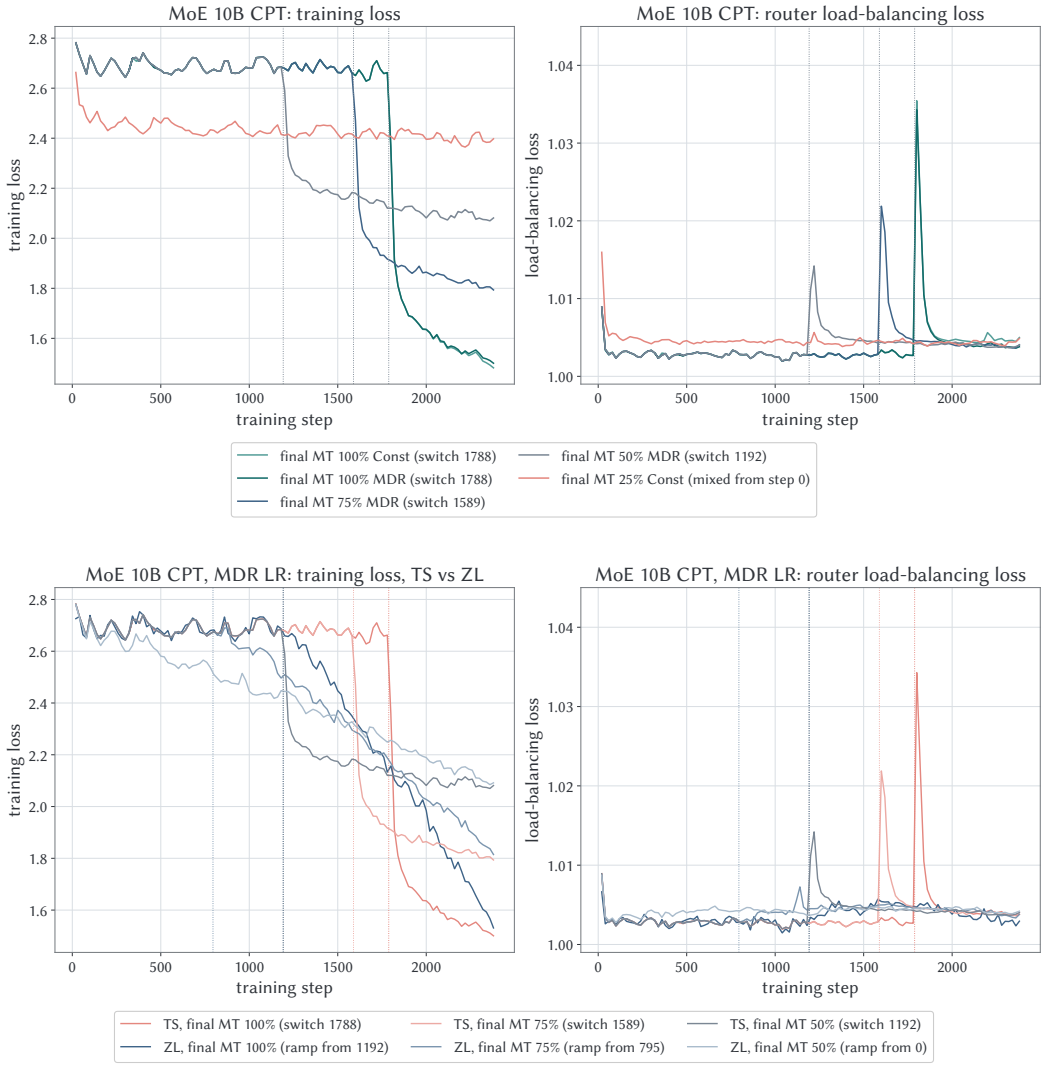


Figure 28: Router switching dynamics during MoE 10B continued training. Top: TS comparisons, with switches at steps 1,788, 1,589, and 1,192 for final MT ratios of 100%, 75%, and 50%. Bottom: TS and ZL under MDR at the same endpoints. Training step is horizontal; training loss and logged load balancing appear at left and right. Peaks are smaller for milder jumps and absent under ZL; final validation losses are reported separately in Tables 11 and 12.

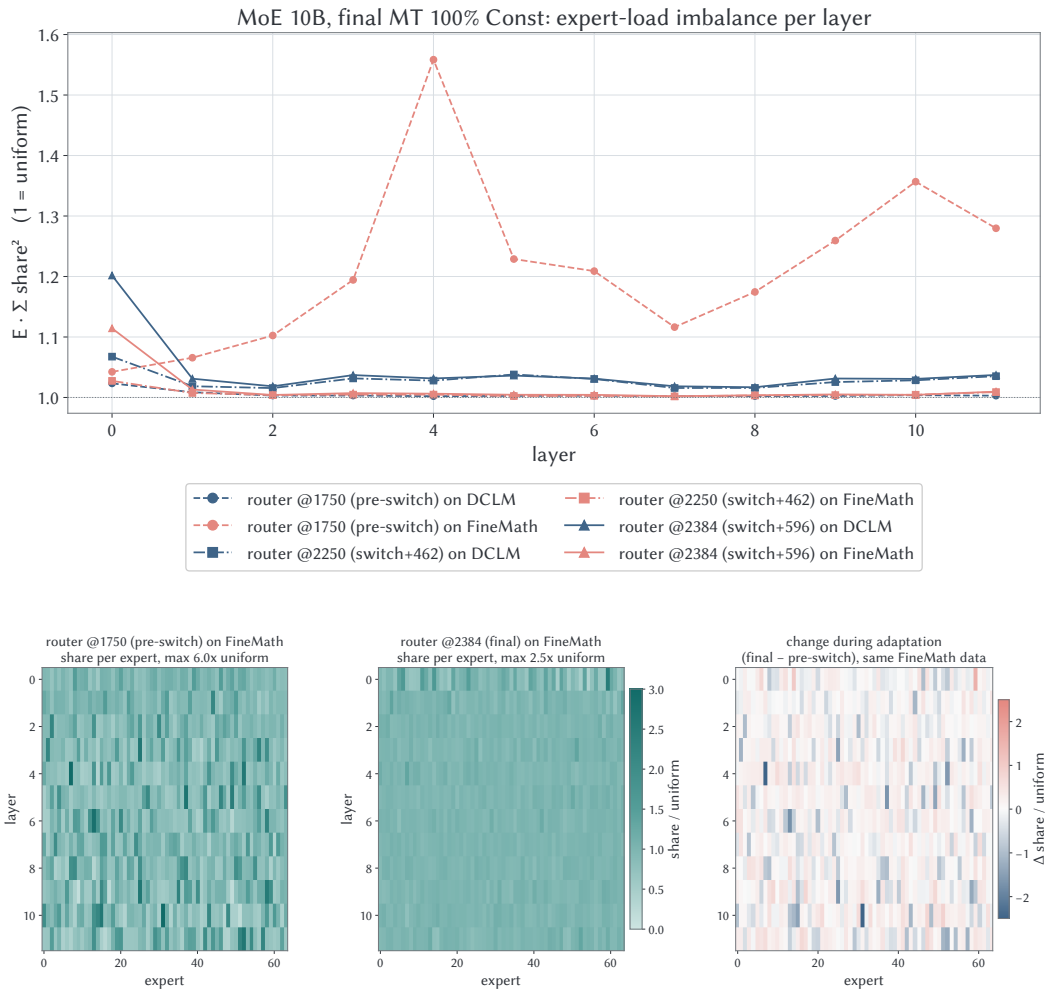


Figure 29: **The router initially assigns MT tokens unevenly across experts and becomes more balanced during MT training.** Checkpoint routing diagnostics for the MoE 10B TS+Const run ending at 100% MT. Top: imbalance by layer on DCLM and FineMath at steps 1,750, 2,250, and 2,384. Bottom: FineMath expert shares before the switch, at training end, and their difference, with expert and layer indices as coordinates. Shares are relative to uniform routing. These diagnostics describe one run, without replica averaging.

negative trend means that experts receiving more than the uniform share tend to receive less after adaptation, while those receiving less tend to receive more. To compare TS and ZL, we examine both the size of these changes and whether experts that were initially more heavily used remain so afterwards.

We use three statistics to compare the above changes (Table 14). For routing shares p before MT entry time and q at training completion, **total variation (TV)** is

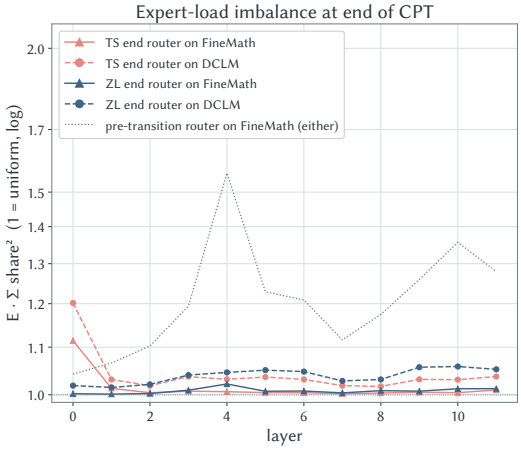
$$\mathrm{TV}(p, q) = \frac{1}{12} \sum_{\ell=1}^{12} \frac{1}{2} \sum_{j=1}^{64} |p_{\ell j} - q_{\ell j}|. \tag{8}$$

It measures the net change in the routing distribution, averaged across layers. For example, TV of 0.15 means that matching the final shares requires redistributing at least 15% of the assignment mass, on average across layers. It does not measure cumulative movement during training. The regression residual standard deviation measures how far each expert's share change deviates from the fitted relationship with its initial excess. Spearman correlation compares the expert rankings by assignment share before and after adaptation within each layer, then averages across layers.

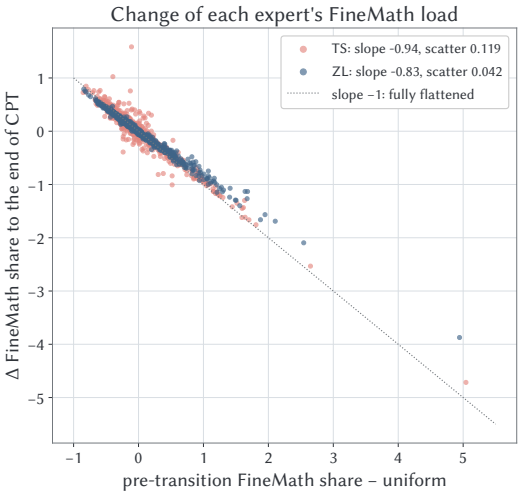
TS+Const and ZL+MDR show comparable net redistribution, but ZL+MDR has substantially less scatter around the fitted trend and preserves more of the initial expert ordering (Table 14 and Fig. 30). In other words, ZL+MDR shows a more consistent contraction of the initial load differences, while TS+Const shows more reshuffling of which experts receive the most assignments. These checkpoint statistics compare different LR schedules and similar but nonidentical starting routers, so they describe the redistribution pattern rather than isolating the effect of data shape alone.

Schedule	Final imbalance FineMath / DCLM	Net routing change FineMath TV	Regression residual std.	Rank retention Spearman
TS+Const	1.015 / 1.043	0.156	0.119	0.30
ZL+MDR	1.008 / 1.039	0.140	0.042	0.74

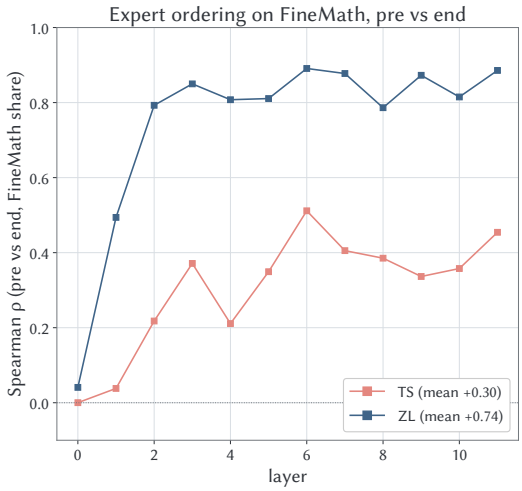
Table 14: Routing comparison for MoE 10B with a 25% MT budget and final MT ratio 100%. TS+Const compares steps 1,750 and 2,384, and ZL+MDR compares steps 1,150 and 2,384, from before each schedule's MT entry time to training completion. TV reports the net change in FineMath assignment shares. Regression residuals use shares relative to uniform routing, and Spearman correlations are averaged across layers. Figure 6 provides the separate control with identical LR trajectories.



(a) Final expert load imbalance.



(b) Changes in the fraction of FineMath assignments sent to each expert.



(c) Preservation of expert load rankings.

Figure 30: **ZL+MDR preserves expert load rankings better than TS+Const, although both reach nearly balanced MT loads.** (a) Both schedules bring FineMath (MT) loads close to uniform in most layers; one denotes equal assignment shares across experts. (b) Initially more heavily used experts lose share, while less used experts gain it. Each point represents one expert in one layer. The horizontal axis shows its share before MT entry time minus the uniform share, and the vertical axis shows its final minus initial share, both in units of $1/64$. Points on the dotted line end at the uniform share. (c) Despite similar final balance, the ordering of experts changes more under TS+Const. The higher rank correlations under ZL+MDR mean that experts receiving relatively more FineMath assignments initially tend to remain more heavily used at the end. All checkpoint measurements use held-out data. These MoE 10B runs use different LR schedules, so the comparison does not isolate the effect of the data schedule. See Table 14 for the setup.

H Additional Theoretical Setup

Original SGD training algorithm with data mixing. We denote K as the total training steps, and at each training step $k \in [0, K-1]$, we sample MT data with probability $\gamma(k)$ and sample PT data with probability $1-\gamma(k)$. Thus, for the kth data point $\boldsymbol{x}_k$, its covariance matrix is $\mathbb{E}\boldsymbol{x}_k\boldsymbol{x}_k^\top = \gamma(k)\boldsymbol{H} + (1-\gamma(k))\,\boldsymbol{G}$. At the kth step, we therefore have $\mathcal{L}_k(\boldsymbol{\theta}) = \frac{1}{2}\boldsymbol{\theta}^\top\left(\gamma(k)\boldsymbol{H} + (1-\gamma(k))\,\boldsymbol{G}\right)\boldsymbol{\theta}$. We train our model using online stochastic gradient descent (SGD) with learning rate schedule $E = \{\eta_0, \cdots, \eta_{K-1}\}$. The update rule can be written as

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta_k\left(\nabla\mathcal{L}_k(\boldsymbol{\theta}_k) + \xi_k\right). \tag{9}$$

Here $\xi_k \sim \mathcal{N}(0, \boldsymbol{\Sigma})$ is the additional noise independent of the signal. We let $\boldsymbol{\Sigma} = \operatorname{diag}(\boldsymbol{\Sigma}_0, \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2)$, where $\boldsymbol{\Sigma}_l = \operatorname{diag}(\nu_1^{(l)}, \ldots, \nu_{d_l}^{(l)}) \in \mathbb{R}^{d_l \times d_l}$ for $l = 0, 1, 2$. We assume that $\nu_i^{(l)} > 0$ for all $0 < i \le d_l$ and $l = 0, 1, 2$. Our goal is to minimize the expected validation loss $\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_K)]$, where the expectation is taken over the randomness in the gradient noise and the data sampling.

SDE approximation. Motivated by Li et al. (2017; 2019; 2020; 2021); Pesme et al. (2021); Li et al. (2022; 2025), we analyze the continuous time limit of Equation (9), which makes the analysis more tractable. We fix a discretization step size $h > 0$, set $t_k = kh$ for $k = 0, \ldots, K$, and let $T = Kh$. We also define $\eta(t_k) = \eta_k/h$ and $\gamma(t_k) = \gamma(k)$ as the continuous-time learning rate schedule and data schedule, respectively. The corresponding continuous-time loss function is defined as

$$\mathcal{L}_t(\boldsymbol{\theta}) := \frac{1}{2}\boldsymbol{\theta}^\top\left(\gamma(t)\boldsymbol{H} + (1-\gamma(t))\,\boldsymbol{G}\right)\boldsymbol{\theta}.$$

Using $\eta_k = h\eta(t_k)$, Equation (9) becomes

$$\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k = -\eta(t_k)\nabla\mathcal{L}_{t_k}(\boldsymbol{\theta}_k)h - \eta(t_k)h\xi_k.$$

Following Li et al. (2019); Orvieto & Lucchi (2019), we approximate the iteration above with an Itô SDE given in Equation (1). The minimization of the expected validation loss $\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_K)]$ is then approximated by the minimization of $\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_T)]$, where the expectation is taken over the randomness in the Brownian motion and the data sampling.

I Formal Statements of Theoretical Results

In this section, we provide the formal statement of Theorem 3.1 and its related assumptions. We rewrite Theorem 3.1 into Theorems I.1 to I.3, and they correspond to (i), (ii) and (iii) respectively. The proofs of Theorems I.1 to I.3 are given in Appendices J to L respectively.

We assume that $\sum_{i=1}^{d_1}\theta_i^{(1)}(0)^2 > 0$ in subesquent results. We first show the optimal data schedule under a constant learning rate.

Theorem I.1 (Optimal Data Schedule under a Constant Learning Rate)**.** *Suppose that $\eta(t) \equiv \eta_{\max} > 0$. Under Condition 3.1, the optimal data schedule is given by the following schedule:*

$$\gamma^*(t) = \begin{cases} 0, & t \in [0, (1-\alpha)T), \\ 1, & t \in [(1-\alpha)T, T], \end{cases}$$

which we refer to as the 0-1 data schedule.

We next replace the constant learning rate with a learning rate schedule with linear decay, while keeping all other settings fixed. The following theorem gives sufficient conditions on the decay strength of the learning rate and MT budget, under which the 0-1 data schedule in Theorem I.1 is no longer optimal.

Theorem I.2 (Optimal Schedule under Linear Learning Rate Decay)**.** *Suppose that the learning rate decays linearly according to*

$$\eta(t) = \eta_{\max} - \frac{c}{T}t, \qquad (1 - 4^{-\frac{1}{3}})\eta_{\max} < c < \eta_{\max}.$$

Then, under Condition 3.1, when $0 < \alpha < 1 - \frac{(1-4^{-\frac{1}{3}})\eta_{\max}}{c}$, $\gamma^*(t)$ in Theorem I.1 is not the optimal schedule.

Theorem I.2 does not characterize the exact form of the optimal data schedule under a linear decaying learning rate schedule. However, it proves that changing only the learning rate schedule from constant to linear decay can change the corresponding optimal data schedule with everything else fixed.

Theorem I.1 and Theorem I.2 show that different learning rate schedules induce different optimal data schedules. Nevertheless, the optimal data schedule under a linear decay learning rate is not explicitly solvable. Motivated by the multi-power law in Luo et al. (2025), we introduce a power-law spectral model that allows us to derive the optimal data schedule under a linear decay learning rate in closed form.

Assumption I.1 (Power-Law Spectral Model). *For each block $l \in \{0, 1, 2\}$, assume that the pairs $\{(\lambda_i^{(l)}, \nu_i^{(l)})\}_{i=1}^{d_l}$ are independent and identically distributed across i, with*

$$\lambda_i^{(l)} \sim p_l(\lambda) = D_l \lambda^{\alpha_l} \exp(-K_l \lambda), \qquad \lambda \in [0, \infty), \qquad i = 1, \dots, d_l,$$
$$\nu_i^{(l)} \mid \lambda_i^{(l)} \sim q_l(\nu \mid \lambda_i^{(l)}), \qquad \nu \in [0, \infty), \qquad i = 1, \dots, d_l,$$

where $-1 < \alpha_l < 0$, $K_l > 0$, and

$$D_l = \frac{K_l^{\alpha_l + 1}}{\Gamma(\alpha_l + 1)}.$$

Moreover, for some $\rho_l > 0$,

$$\mathbb{E}_{q_l(\cdot|\lambda)}[\nu \mid \lambda] = F_l \lambda^{\rho_l} \exp(-G_l \lambda),$$

where $F_l, G_l > 0$ are constants independent of the learning rate schedule.

The spectral assumption describes a collection of directions that are learned at different speeds. A smaller eigenvalue λ means that the loss changes more weakly along the corresponding direction, so the error in that direction is corrected more slowly. The density p_l describes how common different eigenvalues are. Since $-1 < \alpha_l < 0$, the factor λ^{α_l} places more mass near zero, representing many slowly learned directions, while $e^{-K_l \lambda}$ makes very large eigenvalues increasingly rare. Together, these factors give a Gamma density with shape $\alpha_l + 1$ and rate K_l. In Luo et al. (2025), they use an eigenvalue density with a power-law expression to explain how Multi-Power Law arises. The two models differ in how they describe large eigenvalues: Luo et al. (2025) imposes a finite upper bound on the eigenvalues, and our model uses a density on $[0, \infty)$ with an exponentially decaying tail.

The noise assumption describes how the strength of the additional gradient noise varies with eigenvalues. Here, $\nu_i^{(l)}$ is the variance of noise in direction i of block l: a larger value means stronger random perturbations to the gradient along that direction. Thus, $\mathbb{E}[\nu \mid \lambda] = F_l \lambda^{\rho_l} e^{-G_l \lambda}$ specifies the average noise variance among directions with eigenvalue λ. The parameter F_l sets the overall noise scale. At those directions with small eigenvalues, where the exponential factor $e^{-G_l \lambda}$ is close to one, $\rho_l > 0$ describes weaker noise in those directions. The factor $e^{-G_l \lambda}$ makes the average noise variance decay at the directions with very large eigenvalues. This functional form follows Luo et al. (2025), who use such a relationship between eigenvalues and noise variance to derive the additional loss reduction caused by learning rate decay in a quadratic model.

Under Assumption I.1, we consider minimizing the expected validation loss, where the expectation is taken with respect to the training process and the distribution of the eigenvalues. Under this setup, we show the shape of the optimal data schedule under a linear decaying learning rate in closed form.

Theorem I.3 (Optimal Data Schedule under Power-Law Spectral Model). *Suppose that $0 < \rho_1 < 1$ and that $\eta(t) = \eta_{\max} - ct/T$, where $\eta_{\max}$ and c are fixed constants satisfying $\left(1 - 4^{-1/3}\right) \eta_{\max} < c < \eta_{\max}$. Define $x_\star := c^{-1} \left(4^{-1/3} \eta_{\max} - (\eta_{\max} - c)\right) \in (0, 1)$, and suppose that the data fraction $0 < \alpha < x_\star$ is fixed. Then, when T is sufficiently large, the optimal data schedule under Condition 3.1 and Assumption I.1 can be given by*

$$\gamma^*(t) = \begin{cases} 0, & t \in [0, t_1), \\ \left(G_{\text{eff}} + \displaystyle\int_{t_2}^{T} \eta(\tau)\, \mathrm{d}\tau\right) \dfrac{4c}{T(b+1)} \left(\eta(t_2)\right)^{-\frac{4}{b+1}} \left(\eta(t)\right)^{-\frac{2(b-1)}{b+1}}, & t \in [t_1, t_2), \\ 1, & t \in [t_2, T], \end{cases}$$

where $b := \alpha_1 + \rho_1 + 2$, $G_{\text{eff}} := \frac{K_1 + G_1}{2}$, and t_1, t_2 are two time points that satisfy $0 < t_1 < t_2 < T$, $t_1 = \Theta(T)$ and $t_2 = \Theta(T)$.

J Proof of Theorem I.1

In this section, we derive the evaluation loss from the SDE in Equation (1) and then prove the optimality of the 0–1 data schedule in Theorem I.1.

J.1 Derivation of the Terminal Loss

Before we begin our proof, we first define some additional notations. We denote

$$a_i^{(l)}(t) := \begin{cases} \lambda_i^{(0)}, & l = 0, \\ \gamma(t)\lambda_i^{(1)}, & l = 1, \\ \big(1-\gamma(t)\big)\lambda_i^{(2)}, & l = 2, \end{cases} \qquad i = 1, \dots, d_l. \tag{10}$$

For $0 \le s \le t \le T$, we define $R_i^{(l)}(s,t) := \exp\Big(-\int_s^t \eta(u) a_i^{(l)}(u)\,\mathrm{d}u\Big)$, and $E_i^{(l)}(s,t) := R_i^{(l)}(s,t)^2 = \exp\Big(-2\int_s^t \eta(u) a_i^{(l)}(u)\,\mathrm{d}u\Big)$. Because $\boldsymbol{H}$, $\boldsymbol{G}$, and $\boldsymbol{\Sigma}$ are diagonal, the ith coordinate of Equation (1) satisfies

$$\mathrm{d}\theta_i^{(l)}(t) = -\eta(t) a_i^{(l)}(t)\theta_i^{(l)}(t)\,\mathrm{d}t + \eta(t)\sqrt{h\nu_i^{(l)}}\,\mathrm{d}B_i^{(l)}(t). \tag{11}$$

Solving Equation (11) gives

$$\theta_i^{(l)}(T) = R_i^{(l)}(0,T)\theta_i^{(l)}(0) + \sqrt{h\nu_i^{(l)}}\int_0^T \eta(s) R_i^{(l)}(s,T)\,\mathrm{d}B_i^{(l)}(s).$$

The stochastic integral has zero mean. Since $R_i^{(l)}(0,T)\theta_i^{(l)}(0)$ is deterministic, its cross term with the stochastic integral vanishes after taking expectations. The Itô isometry then gives

$$\mathbb{E}\Big[\theta_i^{(l)}(T)^2\Big] = \theta_i^{(l)}(0)^2 E_i^{(l)}(0,T) + h\nu_i^{(l)}\int_0^T \eta(s)^2 E_i^{(l)}(s,T)\,\mathrm{d}s. \tag{12}$$

We denote $\mathcal{L}_l(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{d_l} \lambda_i^{(l)}\big(\theta_i^{(l)}\big)^2$. Since the model is evaluated on the CPT loss $\mathcal{L}(\boldsymbol{\theta}) = \sum_{l=0}^{1}\mathcal{L}_l(\boldsymbol{\theta}) = \frac{1}{2}\sum_{l=0}^{1}\sum_{i=1}^{d_l}\lambda_i^{(l)}\big(\theta_i^{(l)}\big)^2$, substituting Equation (12) yields

$$\begin{aligned}\mathcal{J}(\gamma,\eta) := \mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_T)] &= \frac{1}{2}\sum_{l=0}^{1}\sum_{i=1}^{d_l}\lambda_i^{(l)}\left[\theta_i^{(l)}(0)^2 E_i^{(l)}(0,T) + h\nu_i^{(l)}\int_0^T \eta(s)^2 E_i^{(l)}(s,T)\,\mathrm{d}s\right] \\ &= \frac{1}{2}\sum_{l=0}^{1}\sum_{i=1}^{d_l}\lambda_i^{(l)}\theta_i^{(l)}(0)^2 E_i^{(l)}(0,T) + \frac{1}{2}\sum_{l=0}^{1}\sum_{i=1}^{d_l}\lambda_i^{(l)} h\nu_i^{(l)}\int_0^T \eta(s)^2 E_i^{(l)}(s,T)\,\mathrm{d}s. \end{aligned} \tag{13}$$

We call the first term bias term, which arises because the initialization of the parameters is different from the global optimum. We call the second term the variance term, which is induced by the accumulation of gradient noise. Note that for all i, l, and $s < t$, $E_i^{(l)}(s,t)$ is convex up to a set of measure zero. Thus $\mathcal{J}(\gamma,\eta)$ is a convex functional of the data schedule $\gamma(t)$ up to a set of measure zero. Furthermore, the feasible set $\{\gamma(t) : [0,T] \to [0,1], \frac{1}{T}\int_0^T \gamma(t)\,\mathrm{d}t \le \alpha\}$ is convex. Thus the optimization problem in Equation (2) is a convex optimization problem. The feasible set also satisfies the Slater condition, so the KKT conditions are necessary and sufficient for optimality.

Next we prove that if the optimal data schedule exists, then the optimal data schedule is unique up to a set of measure zero. We prove by contradiction. We assume that there exist two different optimal data schedules γ_1 and γ_2, and we define $M = \mathcal{J}(\gamma_1,\eta)$. By the convexity of $\mathcal{J}(\gamma,\eta)$, we have

$$\mathcal{J}(x\gamma_1 + (1-x)\gamma_2, \eta) \le x\mathcal{J}(\gamma_1,\eta) + (1-x)\mathcal{J}(\gamma_2,\eta) = M.$$

The equality must hold, otherwise $x\gamma_1 + (1-x)\gamma_2$ is the optimal data schedule, but γ_1 and γ_2 are not. Notice that the summands for $l = 0$ are independent of the data schedule. Thus by the strict convexity of the exponential function, the equality holds if and only if for all i and for all $0 \le s \le T$,

$$-2\int_s^t \eta(u)\gamma_1(u)\lambda_i^{(1)}\,\mathrm{d}u = -2\int_s^t \eta(u)\gamma_2(u)\lambda_i^{(1)}\,\mathrm{d}u.$$

Taking the derivative, we have $\gamma_1(u) = \gamma_2(u), a.e.$, which is a contradiction.

J.2 KKT Condition for Equation (2)

We denote the bracketed term in Equation (13) by

$$Q_i^{(l)}(t) := \theta_i^{(l)}(0)^2 E_i^{(l)}(0,T) + h\nu_i^{(l)} \int_0^t \eta(s)^2 E_i^{(l)}(s,T)\,\mathrm{d}s.$$

Note that $\gamma(t)$ does not affect $Q_i^{(0)}(t)$, so we only need to minimize $\frac{1}{2}\sum_{i=1}^{d_1} \lambda_i^{(1)} Q_i^{(1)}(T)$. From now on, we omit the superscript (1) for simplicity.

We rewrite the CPT-data constraint as

$$\int_0^T \gamma(t)\,\mathrm{d}t - \alpha T \le 0,$$

and define the Lagrangian functional

$$\mathcal{F}(\gamma,\kappa;\eta) := \mathcal{J}(\gamma,\eta) + \kappa\left(\int_0^T \gamma(t)\,\mathrm{d}t - \alpha T\right).$$

Now we derive the first variation of $\mathcal{F}(\gamma,\kappa;\eta)$. For a bounded perturbation $v(t)$, let $\gamma_\varepsilon = \gamma + \varepsilon v$. We use a subscript ε to denote quantities induced by the perturbed schedule and define

$$\begin{aligned} a_{i,\varepsilon}(u) &:= \lambda_i \gamma_\varepsilon(u) = a_i(u) + \varepsilon\lambda_i v(u), \\ E_{i,\varepsilon}(s,T) &:= \exp\left(-2\int_s^T \eta(t) a_{i,\varepsilon}(t)\,\mathrm{d}t\right) \\ &= E_i(s,T)\exp\left(-2\varepsilon\lambda_i \int_s^T \eta(t)v(t)\,\mathrm{d}t\right). \end{aligned}$$

In particular, $a_{i,0} = a_i$ and $E_{i,0} = E_i$, so the notation without an ε subscript always refers to the original schedule γ. Differentiating the second identity at $\varepsilon = 0$ gives

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon} E_{i,\varepsilon}(s,T)\Big|_{\varepsilon=0} = -2\lambda_i E_i(s,T)\int_s^T \eta(t)v(t)\,\mathrm{d}t. \tag{14}$$

To differentiate the terminal loss, we substitute Equation (14) into Equation (13), and get that

$$\begin{aligned} D\mathcal{J}(\gamma)[v] = &-\sum_{i=1}^{d_1} \lambda_i^2 \theta_i(0)^2 E_i(0,T)\int_0^T \eta(t)v(t)\,\mathrm{d}t \\ &- h\sum_{i=1}^{d_1} \lambda_i^2 \nu_i \int_0^T \eta(s)^2 E_i(s,T)\left[\int_s^T \eta(t)v(t)\,\mathrm{d}t\right]\mathrm{d}s. \end{aligned}$$

The second term integrates over the triangular region $\{(s,t) : 0 \le s \le t \le T\}$. By Fubini's theorem,

$$\begin{aligned} &\int_0^T \eta(s)^2 E_i(s,T)\left[\int_s^T \eta(t)v(t)\,\mathrm{d}t\right]\mathrm{d}s \\ &\quad = \int_0^T \eta(t)v(t)\left[\int_0^t \eta(s)^2 E_i(s,T)\,\mathrm{d}s\right]\mathrm{d}t. \end{aligned}$$

Both parts of the objective variation can therefore be collected into a single integral. Including the variation of the constraint gives

$$\begin{aligned} D_\gamma \mathcal{F}(\gamma,\kappa;\eta)[v] &= D\mathcal{J}(\gamma)[v] + \kappa\int_0^T v(t)\,\mathrm{d}t \\ &= \int_0^T \big(\kappa - M_\gamma(t)\big)v(t)\,\mathrm{d}t. \end{aligned}$$

where the marginal value of CPT data at time t is

$$M_\gamma(t) := \eta(t) \sum_{i=1}^{d_1} \lambda_i^2 \left[\theta_i(0)^2 E_i(0,T) + h\nu_i \int_0^t \eta(s)^2 E_i(s,T)\, \mathrm{d}s \right]. \tag{15}$$

Consequently, the pointwise KKT conditions for the box constraint $0 \le \gamma(t) \le 1$ are

$$\begin{cases} M_{\gamma^\star}(t) \le \kappa, & \gamma^\star(t) = 0, \\ M_{\gamma^\star}(t) = \kappa, & 0 < \gamma^\star(t) < 1, \\ M_{\gamma^\star}(t) \ge \kappa, & \gamma^\star(t) = 1, \end{cases} \tag{16}$$

together with primal feasibility, $\kappa \ge 0$, and complementary slackness

$$\kappa \left(\int_0^T \gamma^\star(t)\, \mathrm{d}t - \alpha T \right) = 0.$$

J.3 Proof of Theorem I.1

Proof. Recall that

$$Q_i(t) := \theta_i(0)^2 E_i(0,T) + h\nu_i \int_0^t \eta(s)^2 E_i(s,T)\, \mathrm{d}s.$$

The fundamental theorem of calculus gives $Q_i'(t) = h\nu_i \eta(t)^2 E_i(t,T)$. Hence, for a differentiable learning-rate schedule,

$$M_\gamma'(t) = \eta'(t) \sum_{i=1}^{d_1} \lambda_i^2 Q_i(t) + h\eta(t)^3 \sum_{i=1}^{d_1} \lambda_i^2 \nu_i E_i(t,T).$$

Under a constant learning rate $\eta_{\max}$, we have $\eta'(t) = 0$, and therefore

$$M_\gamma'(t) = h\eta_{\max}^3 \sum_{i=1}^{d_1} \lambda_i^2 \nu_i E_i(t,T) > 0.$$

The strict inequality follows from the nondegeneracy condition in Theorem I.1.

We now give a global comparison, which also proves uniqueness. For a feasible schedule γ, define its remaining CPT mass after time s by

$$C_\gamma(s) := \int_s^T \gamma(u)\, \mathrm{d}u.$$

Because $M_\gamma(t) > 0$ for almost every $t > 0$, adding CPT mass on any positive-measure set on which $\gamma < 1$ strictly decreases the objective. Thus, except for the immediate boundary cases $\alpha = 0$ and $\alpha = 1$, every optimal schedule exhausts the budget:

$$\int_0^T \gamma(t)\, \mathrm{d}t = \alpha T. \tag{17}$$

For any schedule satisfying (17), the box constraint gives

$$C_\gamma(s) \le \min\{\alpha T, T - s\}, \qquad 0 \le s \le T.$$

The schedule $\bar\gamma$ in Theorem I.1 attains this upper bound for every s. Moreover, under the constant learning rate,

$$E_i^\gamma(s,T) = \exp[-2\eta_{\max}\lambda_i C_\gamma(s)]\,.$$

Consequently, $E_i^{\bar\gamma}(s,T) \le E_i^\gamma(s,T)$ for every i and s. The initialization term in Equation (13) is the same for all schedules satisfying (17), while the variance term of $\bar\gamma$ is no larger coordinate-wise. It follows that

$$\mathcal{J}(\bar\gamma, \eta_{\max}) \le \mathcal{J}(\gamma, \eta_{\max}).$$

If equality holds, the nondegeneracy condition forces $C_\gamma(s) = C_{\bar\gamma}(s)$ for almost every s. Taking weak derivatives then gives $\gamma = \bar\gamma$ almost everywhere. The cases $\alpha = 0$ and $\alpha = 1$ follow directly from the box constraint and the strict decrease of the objective when CPT mass is added. □

K Proof of Theorem I.2

Proof. We prove the theorem by contradiction. Let

$$t_\alpha := (1-\alpha)T, \qquad \bar{\gamma}(t) := \mathbf{1}\{t \geq t_\alpha\}$$

denote the transition time and the 0-1 schedule from Theorem I.1 respectively. Suppose that $\bar{\gamma}$ is optimal under the linear decay learning rate schedule. The KKT conditions derived above imply

$$M_{\bar{\gamma}}(t) \leq \kappa \quad (t < t_\alpha), \qquad M_{\bar{\gamma}}(t) \geq \kappa \quad (t > t_\alpha).$$

Because $M_{\bar{\gamma}}$ is continuously differentiable, it follows that $M_{\bar{\gamma}}(t_\alpha) = \kappa$ and

$$M'_{\bar{\gamma}}(t_\alpha) \geq 0. \tag{18}$$

We now calculate this derivative explicitly. Define the integrated learning rate over an interval by

$$\Phi(a,b) := \int_a^b \eta(u)\,\mathrm{d}u, \qquad 0 \leq a \leq b \leq T.$$

The two intervals before and after the transition occur repeatedly below, so we introduce the mnemonic abbreviations

$$\Phi^-_\alpha(s) := \Phi(s, t_\alpha), \qquad \Phi^+_\alpha := \Phi(t_\alpha, T),$$

where the superscripts $-$ and $+$ indicate accumulation before and after t_α, respectively. For $\eta(t) = \eta_{\max} - ct/T$, these quantities are

$$\begin{aligned}
\eta_\alpha := \eta(t_\alpha) &= \eta_{\max} - c(1-\alpha), \\
\Phi^+_\alpha &= \eta_{\max}(T - t_\alpha) - \frac{c}{2T}\big(T^2 - t_\alpha^2\big) \\
&= \alpha T\left[\eta_{\max} - c\left(1 - \frac{\alpha}{2}\right)\right], \\
\Phi^-_\alpha(s) &= \eta_{\max}(t_\alpha - s) - \frac{c}{2T}\big(t_\alpha^2 - s^2\big), \qquad 0 \leq s \leq t_\alpha.
\end{aligned}$$

Under $\bar{\gamma}$, coordinate i has zero curvature before t_α and curvature λ_i afterwards. Let $\bar{E}_i$ denote the squared propagator induced by this candidate schedule. Consequently,

$$\bar{E}_i(s,T) = \exp\big(-2\lambda_i \Phi^+_\alpha\big), \qquad 0 \leq s \leq t_\alpha. \tag{19}$$

In particular, the quantity in Equation (15) at the transition point is

$$\begin{aligned}
\bar{Q}_i(t_\alpha) = \theta_i(0)^2 \exp\big(-2\lambda_i \Phi^+_\alpha\big) \\
+ h\nu_i \exp\big(-2\lambda_i \Phi^+_\alpha\big) \int_0^{t_\alpha} \eta(s)^2\,\mathrm{d}s.
\end{aligned}$$

Since $\eta'(t) = -c/T$, substituting this identity and Equation (19) into the general expression for $M'_\gamma(t)$ gives

$$\begin{aligned}
M'_{\bar{\gamma}}(t_\alpha) = &-\frac{c}{T}\sum_{i=1}^{d_1} \lambda_i^2 \theta_i(0)^2 \exp\big(-2\lambda_i \Phi^+_\alpha\big) \\
&+ h\sum_{i=1}^{d_1} \lambda_i^2 \nu_i \exp\big(-2\lambda_i \Phi^+_\alpha\big)\left[\eta_\alpha^3 - \frac{c}{T}\int_0^{t_\alpha} \eta(s)^2\,\mathrm{d}s\right] \\
= &-\frac{c}{T}\sum_{i=1}^{d_1} \lambda_i^2 \theta_i(0)^2 \exp\big(-2\lambda_i \Phi^+_\alpha\big) \\
&+ h\sum_{i=1}^{d_1} \lambda_i^2 \nu_i \exp\big(-2\lambda_i \Phi^+_\alpha\big)\left[\frac{\eta_\alpha}{(1-\alpha)T}\int_0^{t_\alpha} \eta_\alpha^2\,\mathrm{d}s - \frac{c}{T}\int_0^{t_\alpha} \eta(s)^2\,\mathrm{d}s\right].
\end{aligned} \tag{20}$$

Because η is decreasing, $\eta(s) \geq \eta_\alpha$ for $0 \leq s \leq t_\alpha$. Therefore, the bracket in Equation (20) satisfies

$$\eta_\alpha^3 - \frac{c}{T}\int_0^{t_\alpha} \eta(s)^2\,\mathrm{d}s \leq \eta_\alpha^3 - \frac{1}{3}\left(\eta_{\max}^3 - \eta_\alpha^3\right) = \frac{4}{3}\left(\eta_{\max} - c(1-\alpha)\right)^3 - \frac{1}{3}\eta_{\max}^3 < 0,$$

where the last inequality uses $\alpha < 1 - \frac{(1-4^{-\frac{1}{3}})\eta_{\max}}{c}$. The first term of Equation (20) is nonpositive, and the nondegeneracy condition in Theorem I.2 makes at least one of the two contributions strictly negative. Hence $M'_{\bar{\gamma}}(t_\alpha) < 0$, contradicting Equation (18). □

L Proof of Theorem I.3

L.1 Derivations of the Minimization Objective

We begin from the final loss expression in Equation (13). Recall that only the shared block $l = 0$ and the CPT-specific block $l = 1$ contribute to the final loss. Since the $l = 0$ term is independent of γ, we only need to minimize

$$\mathcal{L}_1(\boldsymbol{\theta}) = \underbrace{\frac{1}{2}\sum_{i=1}^{d_1} \lambda_i^{(1)} \theta_i^{(1)}(0)^2 \exp\left(-2\lambda_i^{(1)} \int_0^T \eta(t)\gamma(t)\,\mathrm{d}t\right)}_{\mathcal{L}_1^{\mathrm{bias}}(\boldsymbol{\theta})} + \underbrace{\frac{1}{2}\sum_{i=1}^{d_1} \lambda_i^{(1)} h\nu_i^{(1)} \int_0^T \eta(s)^2 \exp\left(-2\lambda_i^{(1)} \int_s^T \eta(t)\gamma(t)\,\mathrm{d}t\right)\mathrm{d}s}_{\mathcal{L}_1^{\mathrm{var}}(\boldsymbol{\theta})}.$$

We now average $\mathcal{L}_1(\boldsymbol{\theta})$ over the i.i.d. spectral pairs $\{(\lambda_i^{(1)}, \nu_i^{(1)})\}_{i=1}^{d_1}$ specified in Assumption I.1. In subsequent analysis, we will repeatedly use the Gamma-integral identity

$$\int_0^\infty \lambda^{r-1} e^{-x\lambda}\,\mathrm{d}\lambda = \frac{\Gamma(r)}{x^r}, \qquad r, x > 0.$$

Because the $\lambda_i^{(1)}$ are identically distributed, the same spectral expectation multiplies every initialization coefficient. Thus

$$\begin{aligned}\mathbb{E}\mathcal{L}_1^{\mathrm{bias}}(\boldsymbol{\theta}) &= \frac{D_1}{2}\left(\sum_{i=1}^{d_1} \theta_i^{(1)}(0)^2\right)\int_0^\infty \lambda^{\alpha_1+1} \exp\left(-\left(K_1 + 2\int_0^T \eta(t)\gamma(t)\,\mathrm{d}t\right)\lambda\right)\mathrm{d}\lambda \\ &= \frac{D_1}{2}\left(\sum_{i=1}^{d_1} \theta_i^{(1)}(0)^2\right)\frac{\Gamma(\alpha_1+2)}{\left(K_1 + 2\int_0^T \eta(t)\gamma(t)\,\mathrm{d}t\right)^{\alpha_1+2}}. \end{aligned} \tag{21}$$

We next turn to the variance term. We define the remaining accumulated learning rate after time s by

$$R_\gamma(s) := \int_s^T \eta(t)\gamma(t)\,\mathrm{d}t.$$

For a representative spectral pair (λ, ν), we first condition on λ. Then Assumption I.1 gives

$$\mathbb{E}_{\nu|\lambda}\left[\lambda\nu e^{-2\lambda R_\gamma(s)} \,\middle|\, \lambda\right] = \lambda e^{-2\lambda R_\gamma(s)} \mathbb{E}_{\nu|\lambda}[\nu \mid \lambda] = F_1 \lambda^{\rho_1+1} \exp\left[-\left(2R_\gamma(s) + G_1\right)\lambda\right].$$

We then average over λ. Using $p_1(\lambda) = D_1\lambda^{\alpha_1} e^{-K_1\lambda}$, the law of total expectation yields

$$\begin{aligned}\mathbb{E}_{\lambda,\nu}\left[\lambda\nu e^{-2\lambda R_\gamma(s)}\right] &= \mathbb{E}_\lambda\left[\mathbb{E}_{\nu|\lambda}\left[\lambda\nu e^{-2\lambda R_\gamma(s)} \,\middle|\, \lambda\right]\right] \\ &= D_1 F_1 \int_0^\infty \lambda^{\alpha_1+\rho_1+1} \exp\left[-\left(2R_\gamma(s) + G_1 + K_1\right)\lambda\right]\mathrm{d}\lambda.\end{aligned}$$

Tonelli's theorem permits exchanging the expectation with respect to $\{(\lambda_i^{(1)}, \nu_i^{(1)})\}_{i=1}^{d_1}$ and the time integral, so

$$\begin{aligned}\mathbb{E}[\mathcal{L}_1^{\text{var}}(\boldsymbol{\theta})] &= \frac{hd_1}{2}\int_0^T \eta(s)^2 \mathbb{E}_{\lambda,\nu}\left[\lambda\nu e^{-2\lambda R_\gamma(s)}\right] \mathrm{d}s \\ &= \frac{hd_1 D_1 F_1 \Gamma(\alpha_1+\rho_1+2)}{2^{\alpha_1+\rho_1+3}} \int_0^T \frac{\eta(s)^2}{(R_\gamma(s)+(K_1+G_1)/2)^{\alpha_1+\rho_1+2}}\,\mathrm{d}s. \end{aligned} \tag{22}$$

Here we require $\alpha_1+\rho_1+2>0$ so that the spectral integral is finite at the origin. We define $C_0 := \mathbb{E}\mathcal{L}_0(\boldsymbol{\theta})$,

$$C := D_1\left(\sum_{i=1}^{d_1}\theta_i^{(1)}(0)^2\right)\Gamma(\alpha_1+2)2^{-\alpha_1-3}, \qquad C' := hd_1D_1F_1\Gamma(\alpha_1+\rho_1+2)2^{-\alpha_1-\rho_1-3}.$$

Combining Equations (21) and (22) gives the following minimization objective:

$$\begin{aligned}\mathbb{E}[\mathcal{J}(\gamma,\eta)] = C_0 &+ \frac{C}{\left(K_1/2+\int_0^T \eta(t)\gamma(t)\,\mathrm{d}t\right)^{\alpha_1+2}} \\ &+ C'\int_0^T \frac{\eta(s)^2}{\left((K_1+G_1)/2+\int_s^T \eta(t)\gamma(t)\,\mathrm{d}t\right)^{\alpha_1+\rho_1+2}}\,\mathrm{d}s.\end{aligned} \tag{23}$$

Both terms in Equation (23) are nonincreasing when CPT data is added. Therefore, for $0<\alpha<1$, an optimal schedule exhausts the CPT-data budget:

$$\int_0^T \gamma(t)\,\mathrm{d}t = \alpha T.$$

Equation (23), together with this active constraint and $0\le\gamma(t)\le 1$, is the functional optimized in the next subsection.

By a similar argument to that in Appendix J.1, KKT condition is necessary and sufficient for optimality, and the optimal data schedule is unique if it exists.

L.2 KKT Condition

For compactness, define

$$a := \alpha_1+2, \qquad b := \alpha_1+\rho_1+2, \qquad G_{\text{eff}} := \frac{K_1+G_1}{2},$$

and introduce the two schedule-dependent accumulated learning rates

$$A_\gamma := \frac{K_1}{2}+\int_0^T \eta(t)\gamma(t)\,\mathrm{d}t, \qquad B_\gamma(s) := G_{\text{eff}}+\int_s^T \eta(t)\gamma(t)\,\mathrm{d}t.$$

Thus, after dropping terms independent of γ, the objective in Equation (23) is

$$J(\gamma) = CA_\gamma^{-a} + C'\int_0^T \eta(s)^2 B_\gamma(s)^{-b}\,\mathrm{d}s. \tag{24}$$

Let $v\in L^\infty([0,T])$ be an admissible perturbation. Differentiating Equation (24) in the direction v gives

$$\begin{aligned}DJ(\gamma)[v] = &-\frac{aC}{A_\gamma^{a+1}}\int_0^T \eta(t)v(t)\,\mathrm{d}t \\ &- bC'\int_0^T \frac{\eta(s)^2}{B_\gamma(s)^{b+1}}\left(\int_s^T \eta(t)v(t)\,\mathrm{d}t\right)\mathrm{d}s.\end{aligned}$$

The second term is integrated over $\{(s,t) : 0 \le s \le t \le T\}$. Applying Fubini's theorem therefore yields

$$DJ(\gamma)[v] = -\int_0^T M_\gamma(t) v(t)\,\mathrm{d}t,$$

where the marginal value of CPT data at time t is

$$M_\gamma(t) := \eta(t)\left[\frac{aC}{A_\gamma^{a+1}} + bC'\int_0^t \frac{\eta(s)^2}{B_\gamma(s)^{b+1}}\,\mathrm{d}s\right]. \tag{25}$$

We now solve the optimization problem. Let $\kappa \ge 0$ be the multiplier of $\int_0^T \gamma(t)\,\mathrm{d}t - \alpha T \le 0$, and let $u_0(t), u_1(t) \ge 0$ be the pointwise multipliers of $-\gamma(t) \le 0$ and $\gamma(t) - 1 \le 0$, respectively. The Lagrangian is

$$\begin{aligned}\mathfrak{L}(\gamma, \kappa, u_0, u_1) := J(\gamma) &+ \kappa\left(\int_0^T \gamma(t)\,\mathrm{d}t - \alpha T\right) \\ &- \int_0^T u_0(t)\gamma(t)\,\mathrm{d}t + \int_0^T u_1(t)\big(\gamma(t) - 1\big)\,\mathrm{d}t.\end{aligned}$$

The KKT conditions are, almost everywhere on $[0,T]$,

$$\begin{aligned}&0 \le \gamma(t) \le 1, \qquad \int_0^T \gamma(t)\,\mathrm{d}t \le \alpha T,\\ &\kappa \ge 0, \qquad u_0(t) \ge 0, \qquad u_1(t) \ge 0,\\ &-M_\gamma(t) + \kappa - u_0(t) + u_1(t) = 0, \qquad (26)\\ &\kappa\left(\int_0^T \gamma(t)\,\mathrm{d}t - \alpha T\right) = 0, \quad u_0(t)\gamma(t) = 0, \quad u_1(t)\big(1 - \gamma(t)\big) = 0. \qquad (27)\end{aligned}$$

Because $M_\gamma(t) > 0$ whenever $\eta(t) > 0$, adding CPT data strictly decreases the objective. Hence, for $0 < \alpha < 1$, the optimal data schedule must satisfy

$$\int_0^T \gamma^\star(t)\,\mathrm{d}t = \alpha T.$$

Eliminating $u_0(t)$ and $u_1(t)$ from Equations (26) and (27) gives the equivalent conditions

$$\begin{cases} M_{\gamma^\star}(t) \le \kappa, & \gamma^\star(t) = 0,\\ M_{\gamma^\star}(t) = \kappa, & 0 < \gamma^\star(t) < 1,\\ M_{\gamma^\star}(t) \ge \kappa, & \gamma^\star(t) = 1. \end{cases}$$

Since $x \mapsto x^{-a}$ and $x \mapsto x^{-b}$ are convex on $(0,\infty)$, J is convex in γ. The feasible set is also convex, so these KKT conditions are sufficient as well as necessary.

L.3 The Shape of the Data Schedule when $M_\gamma(t)$ is a Constant

In this section, we assume that $\eta'(t) < 0$ for $t \in [0,T]$, and $M_\gamma(t)$ is constant for $t \in [t_1, t_2]$. Differentiation of Equation (25) gives

$$M_\gamma'(t) = \frac{\eta'(t)}{\eta(t)} M_\gamma(t) + \frac{bC'\eta(t)^3}{B_\gamma(t)^{b+1}}. \tag{28}$$

Thus when $M_\gamma(t) = \kappa$ is constant,

$$\kappa\eta'(t) + \frac{bC'\eta(t)^4}{B_\gamma(t)^{b+1}} = 0,$$

which implies

$$B_\gamma(t) = \left(-\frac{bC'\eta(t)^4}{\kappa\eta'(t)}\right)^{1/(b+1)}.$$

When $\eta(t) = \eta_{\max} - \frac{c}{T}t$, we have $\eta'(t) = -c/T$ is constant. Thus for a fixed time t_2, we have

$$\frac{B_\gamma(t)}{B_\gamma(t_2)} = \left(\frac{\eta(t)^4}{\eta(t_2)^4}\right)^{1/(b+1)},$$

and thus

$$B_\gamma(t) = B_\gamma(t_2)\left(\frac{\eta(t)^4}{\eta(t_2)^4}\right)^{1/(b+1)}. \tag{29}$$

We define

$$B_2 := B_\gamma(t_2) = G_{\text{eff}} + \int_{t_2}^{T} \eta(\tau)\gamma(\tau)\,\mathrm{d}\tau, \qquad K_2 := B_2\eta(t_2)^{-4/(b+1)}, \qquad \kappa_2 := \frac{bC'T}{cK_2^{b+1}}.$$

A direct corollary is that

$$B_{\gamma_{t_1,t_2}}(t) = K_2\eta(t)^{4/(b+1)}.$$

Then we take the derivative of Equation (29) and substitute $B'_\gamma(t) = -\eta(t)\gamma(t)$. We get

$$\gamma(t) = \frac{4cK_2}{T(b+1)}\eta(t)^{-\frac{2(b-1)}{b+1}}. \tag{30}$$

L.4 Proof of Theorem I.3

Proof. We define a family of data schedules $\gamma_{t_1,t_2}(t)$ parameterized by two time steps $0 < t_1 < t_2 < T$. The expression of $\gamma_{t_1,t_2}(t)$ is given by

$$\gamma_{t_1,t_2}(t) := \begin{cases} 0, & 0 \le t < t_1, \\ \dfrac{4cK_2}{T(b+1)}\eta(t)^{-\frac{2(b-1)}{b+1}}, & t_1 \le t < t_2, \\ 1, & t_2 \le t \le T. \end{cases}$$

We first need to verify that $\gamma_{t_1,t_2}(t)$ is a valid data schedule, i.e. $0 \le \gamma_{t_1,t_2}(t) \le 1$.

Lemma L.1. *There exists a constant $T_0 > 0$ such that for any $T > T_0$, $\gamma_{t_1,t_2}(t)$ is a valid data schedule for any $0 < t_1 < t_2 < T$.*

Proof. The assumptions $-1 < \alpha_1 < 0$ and $0 < \rho_1 < 1$ imply $1 < b < 3$. Therefore, $\gamma_{t_1,t_2}(t)$ is monotonically increasing in $t \in (t_1, t_2)$. So we only need to verify that $\gamma_{t_1,t_2}(t_2) \le 1$. By definition, we have

$$\gamma_{t_1,t_2}(t_2) = \frac{4cB_2}{T(b+1)\eta(t_2)^2}.$$

Lemma L.2. *When $t \ge t_2$, we have*

$$B_{\gamma_{t_1,t_2}}(t) \le G_{\text{eff}} + \frac{T\eta(t)^2}{2c}.$$

Proof. We conclude the proof through a direct calculation:

$$\begin{aligned} B_{\gamma_{t_1,t_2}}(t) &= G_{\text{eff}} + \int_t^T \eta(\tau)\,\mathrm{d}\tau = G_{\text{eff}} + \int_t^T \left(\eta_{\max} - \frac{c}{T}\tau\right)\mathrm{d}\tau \\ &\le G_{\text{eff}} + \int_t^{c^{-1}T\eta_{\max}} \left(\eta_{\max} - \frac{c}{T}\tau\right)\mathrm{d}\tau \\ &\le G_{\text{eff}} + \frac{T\eta(t)^2}{2c}. \end{aligned}$$

□

By Lemma L.2, we have

$$B_2 = B_{\gamma_{t_1,t_2}}(t_2) \le G_{\text{eff}} + \frac{T\eta(t_2)^2}{2c}.$$

Substituting the upper bound of B_2 into the expression of $\gamma_{t_1,t_2}(t_2)$ gives an upper bound of $\gamma_{t_1,t_2}(t_2)$:

$$\gamma_{t_1,t_2}(t_2) \le \frac{4c}{T(b+1)\eta(t_2)^2} \cdot \left(\frac{T\eta(t_2)^2}{2c} + G_{\text{eff}}\right) \le \frac{2}{b+1} + \frac{4cG_{\text{eff}}}{T(b+1)\eta(T)^2}.$$

Thus a sufficient condition for $\gamma_{t_1,t_2}(t_2) \le 1$ is

$$\frac{2}{b+1} + \frac{4G_{\text{eff}}}{T(b+1)\eta(T)^2} \le 1,$$

which is equivalent to

$$T \ge \frac{4G_{\text{eff}}}{(b+1)\left(1-\frac{2}{b+1}\right)\eta(T)^2} = \frac{4G_{\text{eff}}}{(b-1)\eta(T)^2} =: T_0.$$

□

Lemma L.3. *There exists a constant $T_1 > 0$ such that for any $T > T_1$, $M''_{\gamma_{t_1,t_2}}(t) < 0$ for all $0 < t < t_1$ and $M'_{\gamma_{t_1,t_2}}(t) > 0$ for all $t_2 < t < T$.*

Proof. Equation (28) gives

$$\begin{aligned} M'_{\gamma_{t_1,t_2}}(t) &= \frac{\eta'(t)}{\eta(t)} M_{\gamma_{t_1,t_2}}(t) + \frac{bC'\eta(t)^3}{B_{\gamma_{t_1,t_2}}(t)^{b+1}} \\ &= \eta'(t)\left[\frac{aC}{A_\gamma^{a+1}} + bC'\int_0^t \frac{\eta(s)^2}{B_{\gamma_{t_1,t_2}}(t)^{b+1}}\,\mathrm{d}s\right] + \frac{bC'\eta(t)^3}{B_{\gamma_{t_1,t_2}}(t)^{b+1}} \\ &= -\frac{c}{T}\frac{aC}{A_\gamma^{a+1}} + bC'\left[-\frac{c}{T}\int_0^t \frac{\eta(s)^2}{B_{\gamma_{t_1,t_2}}(s)^{b+1}}\,\mathrm{d}s + \frac{\eta(t)^3}{B_{\gamma_{t_1,t_2}}(t)^{b+1}}\right]. \end{aligned} \tag{31}$$

Notice that $B_{\gamma_{t_1,t_2}}(t) \equiv B_{\gamma_{t_1,t_2}}(t_1)$ when $t \in [0, t_1]$, and we denote this constant by B_1. We write it explicitly as

$$B_1 := B_{\gamma_{t_1,t_2}}(t_1) = G_{\text{eff}} + \int_{t_1}^T \eta(\tau)\gamma_{t_1,t_2}(\tau)\,\mathrm{d}\tau.$$

Then we have

$$M''_{\gamma_{t_1,t_2}}(t) = -\frac{4bcC'}{T}\frac{\eta(t)^2}{B_1^{b+1}} < 0, \qquad 0 < t < t_1.$$

When $t_2 < t < T$, $B_\gamma(t) = G_{\text{eff}} + \int_t^T \eta(\tau)\,\mathrm{d}\tau$ and $\gamma(t) = 1$. Direct differentiation gives

$$M''_\gamma(t) = \frac{bC'\eta(t)^2}{B_\gamma(t)^{b+2}}\left[-\frac{4c}{T}B_\gamma(t) + (b+1)\eta(t)^2\right].$$

By Lemma L.2, we have

$$M''_\gamma(t) \ge \frac{bC'\eta(t)^2}{B_\gamma(t)^{b+2}}\left[-\frac{4c}{T}G_{\text{eff}} + (b-1)\eta(T)^2\right] > 0$$

when $T > \frac{4cG_{\text{eff}}}{(b-1)\eta(T)^2}$. □

Theorem I.3 holds if and only if there exists (t_1, t_2) such that $\gamma_{t_1,t_2}(t)$ satisfies the KKT conditions and the constraints for γ. For this candidate, the pointwise KKT conditions and the active budget

require

$$\begin{aligned} M_{\gamma_{t_1,t_2}}(t) &\le \kappa_2, && 0 \le t < t_1, \\ M_{\gamma_{t_1,t_2}}(t) &= \kappa_2, && t_1 \le t < t_2, \\ M_{\gamma_{t_1,t_2}}(t) &\ge \kappa_2, && t_2 \le t \le T, \\ 0 \le \gamma_{t_1,t_2}(t) &\le 1, && 0 \le t \le T, \\ \int_0^T \gamma_{t_1,t_2}(t)\,\mathrm{d}t &= \alpha T. \end{aligned}$$

By Lemma L.3 and Lemma L.1, a sufficient condition for the above condition is

$$\begin{aligned} M'_{\gamma_{t_1,t_2}}(t_1) &= 0, \\ \int_0^T \gamma_{t_1,t_2}(t)\,\mathrm{d}t &= \alpha T. \end{aligned}$$

Before we proceed, we define some additional notations. Let $x = \frac{T-t}{T}$ to be the reversed-and-normalized time variable. Then $\eta(t) = \eta_{\max} - cx$. We define

$$\epsilon := \eta(T) = \eta_{\max} - c > 0, \quad P(x) := \eta(t) = \epsilon + cx, \quad L(x) := \frac{1}{T}\int_t^T \eta(s)\,\mathrm{d}s = \epsilon x + \frac{c}{2}x^2,$$

and write

$$x_j := \frac{T - t_j}{T}, \qquad j \in \{1, 2\}.$$

We first rewrite Equation (31) in terms of the new variables:

$$M'_{\gamma_{t_1,t_2}}(t) = \frac{bC'}{B_1^{b+1}}\left[P(x)^3 - c\int_x^1 P(u)^2\,\mathrm{d}u - \frac{caC}{bC'T}\frac{B_1^{b+1}}{A_{\gamma_{t_1,t_2}}^{a+1}}\right].$$

Define

$$F(x) := P(x)^3 - c\int_x^1 P(u)^2\,\mathrm{d}u.$$

Then $M'_{\gamma_{t_1,t_2}}(t_1) = 0$ is equivalent to

$$F(x_1) = \frac{caC}{bC'T}\frac{B_1^{b+1}}{A_{\gamma_{t_1,t_2}}^{a+1}}. \tag{32}$$

It remains to solve for x_2 using the budget. From Equation (30), we have

$$\gamma_{t_1,t_2}\big(T(1-x)\big) = \frac{4cB_2}{T(b+1)}P(x_2)^{-4/(b+1)}P(x)^{-2(b-1)/(b+1)}.$$

Recall that $B_2 = B_{\gamma_{t_1,t_2}}(t_2) = G_{\text{eff}} + \int_{t_2}^T \eta(\tau)\,\mathrm{d}\tau = G_{\text{eff}} + TL(x_2)$. Then we have

$$\begin{aligned} \alpha &= \frac{1}{T}\int_0^T \gamma_{t_1,t_2}(t)\,\mathrm{d}t \\ &= \frac{1}{T}\left(\int_{t_1}^{t_2} \gamma_{t_1,t_2}(t)\,\mathrm{d}t + (T - t_2)\right) \\ &= x_2 + \frac{4\big(L(x_2) + T^{-1}G_{\text{eff}}\big)}{3-b}P(x_2)^{-4/(b+1)}\left(P(x_1)^{(3-b)/(b+1)} - P(x_2)^{(3-b)/(b+1)}\right). \end{aligned} \tag{33}$$

We denote $f_{\epsilon,b}(x_1, x_2)$ as the right-hand side of the above equation. Thus the finite-T budget equation differs from $\alpha = f_{\epsilon,b}(x_1, x_2)$ by an additive term of order T^{-1}, uniformly on compact subsets of $0 < x_2 < x_1 < 1$.

Lemma L.4. *Under the assumptions of Theorem I.3, the equation*

$$F(x_1) = 0,$$

has a unique solution $x_1 = x_\star$ satisfying $0 < x_\star < 1$. This solution is independent of T.

Proof. Since $P'(x) = c$ and $P(1) = \eta_{\max}$, direct integration gives

$$F(x) = P(x)^3 - c\int_x^1 P(u)^2\,\mathrm{d}u = \frac{4P(x)^3 - \eta_{\max}^3}{3}.$$

Consequently, $F'(x) = 4cP(x)^2 > 0$, so F has at most one zero. Solving $F(x) = 0$ gives

$$x = x_\star = \frac{4^{-1/3}\eta_{\max} - \epsilon}{c}.$$

The assumption $c > (1 - 4^{-1/3})\eta_{\max}$ implies $x_\star > 0$, while $4^{-1/3}\eta_{\max} - \epsilon < c$ implies $x_\star < 1$. Thus the first equation has exactly one solution in $(0, 1)$.

□

For some fixed $\frac{\alpha+x^*}{2} \le x_1 \le 1$, we have $f_{\epsilon,b}(x_1, \alpha) > \alpha$, and

$$\begin{aligned} f_{\epsilon,b}(x_1, x_2) &\le x_2 + \frac{4\big(\epsilon x_2 + \frac{c}{2}x_2^2 + T^{-1}G_{\text{eff}}\big)}{3-b} P(x_2)^{-4/(b+1)} P(x_1)^{(3-b)/(b+1)} \\ &\le x_2 + \frac{4\big(\epsilon x_2 + \frac{c}{2}x_2 + T^{-1}G_{\text{eff}}\big)}{3-b} P(0)^{-4/(b+1)} P(1)^{(3-b)/(b+1)} \\ &\le \left(1 + \frac{4\big(\epsilon + \frac{c}{2}\big)\eta_{\max}^{(3-b)/(b+1)}}{(3-b)\epsilon^{4/(b+1)}}\right) x_2 + \frac{4G_{\text{eff}}\eta_{\max}^{(3-b)/(b+1)}}{T(3-b)\epsilon^{4/(b+1)}}. \end{aligned}$$

for all $0 < x_2 < x_\star$. We denote $D = \left(1 + \frac{4\left(\epsilon+\frac{c}{2}\right)\eta_{\max}^{(3-b)/(b+1)}}{(3-b)\epsilon^{4/(b+1)}}\right) \ge 1$, $E = \frac{4G_{\text{eff}}\eta_{\max}^{(3-b)/(b+1)}}{(3-b)\epsilon^{4/(b+1)}}$, and $T_2 = \frac{3E}{\alpha}$. Then when $T > T_2$, we have

$$f_{\epsilon,b}(x_1, \frac{\alpha}{3D}) \le D \times \frac{\alpha}{3D} + \frac{E}{T_2} \le \frac{2\alpha}{3} < \alpha.$$

Hence, for sufficiently large T and any fixed $x_1 \in \left[\frac{\alpha+x^*}{2}, 1\right]$, there exists a unique solution $x_2(x_1) = \Theta(1)$.

We now return to Equation (32). As $x_2(x_1) = \Theta(1)$ for all $x_1 \in \left[\frac{\alpha+x^*}{2}, 1\right]$, we have

$$\int_0^T \eta(s)\gamma(s)\,\mathrm{d}s \ge \frac{1}{2}\eta(t_2)(T - t_2) \ge \frac{1}{2} \times x_2\eta_0 \times x_2 T = \Theta(T).$$

Therefore, $A_{\gamma_{t_1,t_2}} = \Theta(T)$ and $B_1 = \Theta(T)$. Since $b - a = \rho_1 < 1$, the right-hand side of the smooth-pasting equation is $O(T^{\rho_1 - 1}) = o(1)$, uniformly on $x_1 \in \left[\frac{\alpha+x^*}{2}, 1\right]$. So there exists $T_3 > 0$ such that for all $T > T_3$, the right hand side of Equation (32) satisfies

$$0 = F(x^*) < \frac{caC}{bC'T}\frac{B_1^{b+1}}{A_{\gamma_{t_1,t_2}}^{a+1}} < F(1).$$

As $F(x)$ is increasing, when $T > \max\{T_0, T_1, T_2, T_3\}$, there exists a unique solution $x_1(T) \in (x_\star, 1)$ to Equation (32). This concludes our proof of the existence of a solution $(x_1(T), x_2(T))$ that satisfies the KKT conditions and the constraints.

□

M Numerical Validation of the Theoretical Model

This section provides the details of the simulation and additional experiments supporting the theoretical predictions in Theorem 3.1. In Appendix M.1, we provide additional details on the simulation setup. In Appendix M.2, we provide a check on the validity of the SDE approximation. In Appendix M.3, we numerically solve the same scheduling problem analyzed in Section 3, thereby checking the closed form solution given in Theorem I.3. In Appendix M.4, we intuitively explain how the optimal data schedule changes with the learning rate schedule as shown in Theorems I.1 and I.3. In Appendix M.5, we verify that the data schedule solved by numerical methods satisfies the KKT condition.

M.1 Additional Details on the Simulation Experiments

Simulation setup. We use the same quadratic model, structure of PT and MT data, power-law spectrum, linearly decaying learning rate and constrained MT data budget as in the theorem. We use the closed form of SDE approximation to calculate the expected validation loss. Regarding why this approximation is reasonable, see Appendix M.2 for more details.

Calculation of the validation loss. In our simulation, we use two methods to numerically calculate the expected validation loss of the SDE approximation: one is to sample all the eigenvalues of $\boldsymbol{G}$ and $\boldsymbol{H}$ according to their corresponding distributions, and the other one is to calculate the expected validation loss with respect to the distributions of eigenvalues directly. For each numerical setup and MT data budget, we first discretize the interval $[0, T]$ into $M = 400$ identical time bins $\{[\tau_{i-1}, \tau_i]\}_{i=1}^{M}$ and approximate $\gamma(t)$ with $\gamma(\tau_j)$ for $\tau_j \le t \le \tau_{j+1}$.

Hyperparameter selection. For the hyperparameters of the MPL spectral model, we take $\alpha_l = -0.5$, $K_l = 1$, $\rho_l = 0.5$, $F_l = 1$, $G_l = 1$. The total training steps is set to $T = 200$. We take $\eta_{\max} = 1$ as the initial learning rate and initialize $\boldsymbol{\theta}_i^l = 1$ for all $l = 0, 1, 2$ and $1 \le i \le d_l$. We also set the discretization step size $h = 0.1$. Note that by the derivations in Appendix J.1, the minimization objective is determined only by the contribution of knowledge specific to MT domain, so the minimization objective depends on d_1 and is independent of d_0 and d_2. In our simulation experiments, we set $d_1 = 4000$.

Numerical optimization. We discretize $[0, T]$ into $K = 400$ equal-width bins and minimize the resulting objective $J(y)$ over the feasible set

$$\mathcal{Y} = \left\{ y \in [0,1]^K : \sum_{k=1}^{K} y_k = \alpha K \right\}.$$

We minimize J over $\mathcal{Y}$ using an accelerated projected-gradient method based on FISTA (Beck & Teboulle, 2009). Starting from a feasible point $y^{(0)} \in \mathcal{Y}$, we initialize an auxiliary point $z^{(0)} = y^{(0)}$. At iteration n, we take a gradient step from $z^{(n)}$ and project the result onto $\mathcal{Y}$:

$$v^{(n)} = z^{(n)} - \eta_n \nabla J\left(z^{(n)}\right), \qquad y_k^{(n+1)} = \mathrm{clip}\left(v_k^{(n)} - \tau_n,\, 0,\, 1\right).$$

The scalar threshold τ_n is determined by the bisection method to enforce $\sum_k y_k^{(n+1)} = \alpha K$. We choose the step size η_n by backtracking. For each trial step size, we compute $y^{(n+1)}$ using the gradient and projection steps above and check whether

$$J\left(y^{(n+1)}\right) \le J\left(z^{(n)}\right) + \left\langle \nabla J\left(z^{(n)}\right), y^{(n+1)} - z^{(n)} \right\rangle + \frac{\left\| y^{(n+1)} - z^{(n)} \right\|_2^2}{2\eta_n}.$$

If the inequality fails, we halve η_n and recompute $y^{(n+1)}$, repeating until the condition is satisfied. After an accepted step, the next line search is initialized with $1.2\eta_n$.

The next auxiliary point incorporates momentum along the latest update direction:

$$z^{(n+1)} = y^{(n+1)} + \beta_n \left(y^{(n+1)} - y^{(n)} \right),$$

where β_n is the standard FISTA momentum coefficient. Whenever $J(y^{(n+1)}) > J(y^{(n)})$, we restart the momentum and set $z^{(n+1)} = y^{(n+1)}$ (O'Donoghue & Candès, 2015). Iterations terminate when the absolute objective change is below $10^{-15} \max\{1, |J(y^{(n)})|\}$, or after 20,000 iterations.

Calculation of the closed form schedule predicted by Theorem I.3. For a fixed parameter setting, the schedule in Theorem I.3 is determined by its two transition times t_1 and t_2. We compute them from the two scalar conditions used in the proof. The first one, given in Equation (32), is a sufficient condition of the KKT condition. The second is the active budget condition in Equation (33), which states that the total amount of MT data should equal the MT data budget. After solving these two equations, we calculate the interior part of the schedule using the expression in Theorem I.3, and set the schedule to zero before t_1 and to one after t_2.

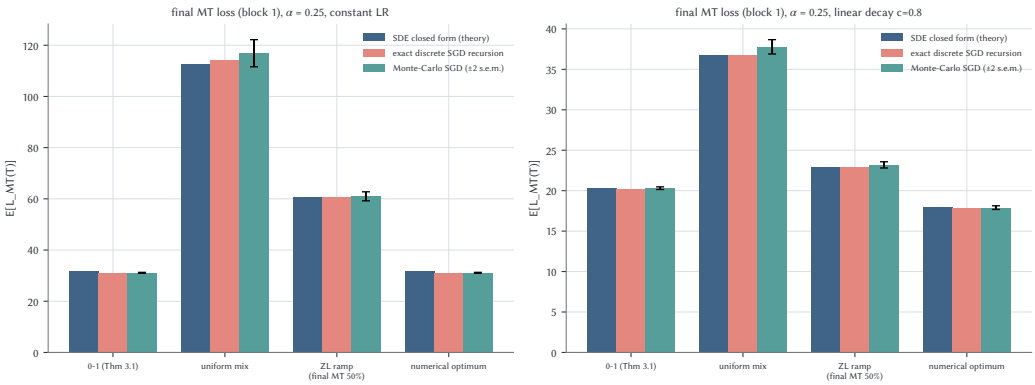


Figure 31: Consistency of the SDE calculation, the exact discrete SGD second moment recursion, and Monte Carlo simulation for $\alpha = 0.25$. Results are shown for a constant learning rate (left) and linear decay with $c = 0.8$ (right), across four representative data schedules. Error bars give two standard errors of the Monte Carlo mean.

M.2 Checks of the SDE Approximation

To show the accuracy of the SDE approximation, for each coordinate l, we denote $m_k^{(l)} := \mathbb{E}\left[\left(\theta_k^{(l)}\right)^2\right]$. The discrete SGD update in Equation (9) gives

$$m_{k+1}^{(l)} = \left(1 - \eta_k a_k^{(l)}\right)^2 m_k^{(l)} + \eta_k^2 \nu_i \tag{34}$$

through a direct calculation, where $a_i^{(l)}$ is defined in Equation (10). Iterating (34) yields the exact expected terminal loss of the discrete model. As a further check, we simulate the stochastic recursion itself and average the terminal loss across all the trajectories. We plot the results of the three methods in Figure 31. We also report the uncertainty of the Monte Carlo estimates using error bars, which indicate ± 2 standard errors of the mean across simulated trajectories. Figure 31 shows that the results of the three methods are almost identical, which supports the accuracy of SDE approximation.

M.3 Numerical Validation of Theoretical Predictions

In this section, we verify numerically that the closed form solution in Theorem I.3 agrees with the optimum of Equation (2).

Figure 32 shows that the solutions of the two numerical methods closely agree with the closed form solution given in Theorem I.3 under different MT data budgets. In particular, the numerical solutions recover the three stage structure predicted by the theorem: an initial PT only phase, an intermediate region in which the MT ratio increases continuously, and a final MT only phase. The agreement also extends to the locations of the two transition points. These results confirm that the schedule structure derived in the theorem is also observed when the expected validation loss is optimized directly.

M.4 How Optimal Data Schedule Changes with Learning Rate Schedule

Theorems I.1 and I.3 together show that the shape of the optimal data schedule differs when the learning rate changes from a constant learning rate to a linearly decaying learning rate schedule. To test the predictions of Theorem I.1 and Theorem I.3 regarding how the optimal data schedule changes with the learning rate, we numerically optimize the expected validation loss over the data schedule. We use the same setting as in Theorem I.3, and the expectation is taken with respect to the gradient noise and the eigenvalues of $\boldsymbol{G}$ and $\boldsymbol{H}$. We take two data budgets $\alpha \in \{0.1, 0.25\}$ and vary the learning rate decay strength c from a constant learning rate to a learning rate schedule with a strong linear decay.

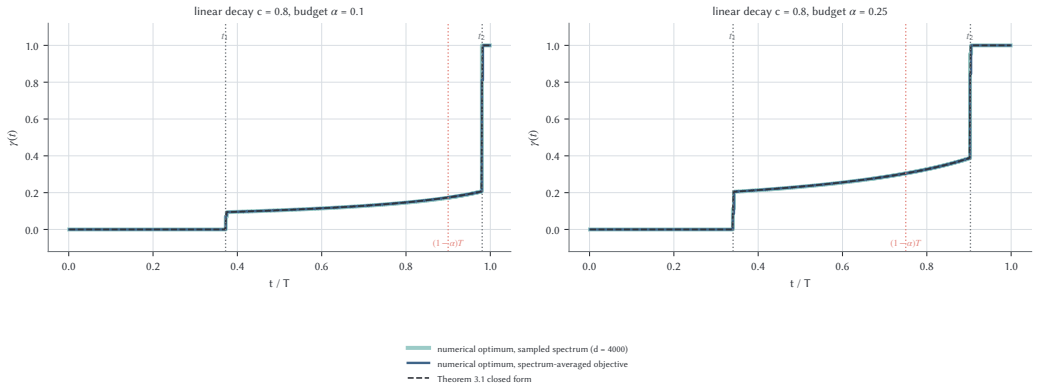


Figure 32: Numerical validation of Theorem I.3. Solid curves show the schedules predicted by the closed form solution, while markers show the optima obtained by directly optimizing the expected validation loss using different numerical methods. The two solutions agree for both MT data budgets considered.

The numerical results are shown in Figure 4. With a constant learning rate, the optimum is the two stage schedule from Theorem I.1: the first stage is to use only PT data, and the second stage is to use only MT data. The conclusion also holds for a learning rate schedule with a weak decay, namely $c = 0.2$, in these instances. Once the decay becomes sufficiently strong, namely $c = 0.5$ and $c = 0.8$, the optimal data schedule changes to a three stage schedule. In particular, the optimal schedule begins with PT data only, then transitions to a mixture of PT and MT data, and ends with MT data only. During the second stage, the proportion of MT data gradually increases. These observations are consistent with the predictions of Theorem I.3.

M.5 Verification of the KKT Condition of the Optimal Data Schedule

To further demonstrate the optimality of the data schedule shown in Theorems I.1 to I.3, we additionally evaluate the KKT conditions given by Equation (15) and Equation (16).

In the left panel of Figure 33, we fix the data schedule to the two-stage schedule given by Theorem I.1, and select two learning rate schedules: a constant learning rate and a learning rate schedule with linear decay. Using Equation (15), we plot the ratio of $M_\gamma(t)$ and $M_\gamma((1-\alpha)T)$ under each of the two learning rate schedules. In the right panel of Figure 33, we select a learning rate schedule with linear decay and its corresponding optimal data schedule. For easier visual comparison, we rescale $M_\gamma(t)$ by a positive constant. This rescaling only changes the overall scale of $M_\gamma(t)$, but does not change its shape over time. We then plot $M_\gamma(t)$ divided by this rescaling factor.

If the 0-1 data schedule given by Theorem I.1 is optimal, then by Equation (16), we have $M_\gamma(t) \leq \kappa$ for $0 \leq t \leq (1-\alpha)T$, and $M_\gamma(t) \geq \kappa$ for $(1-\alpha)T \leq t \leq T$. The left panel of Figure 33 shows that under a constant learning rate, the 0-1 data schedule satisfies this condition, but under a learning rate schedule with linear decay, the 0-1 data schedule violates this condition. This demonstrates that 0-1 data schedule is optimal for a constant learning rate, but is not optimal for a linear decay learning rate schedule, which aligns with the predictions of Theorems I.1 and I.2.

The right panel of Figure 33 shows that $M_\gamma(t)$ also has a three-stage structure and is divided into the same three stages as the data schedule. Specifically, in the first stage where only PT data is used, $M_\gamma(t)$ is monotonically increasing; in the second stage where both PT and MT data are used, $M_\gamma(t)$ is constant; and in the third stage where only MT data is used, $M_\gamma(t)$ also monotonically increases. We take the Lagrange multiplier κ and the rescaling factor as the value of $M_\gamma(t)$ in the second stage. Then by Equation (16), the optimal data schedule given by Theorem I.3 satisfies the KKT conditions, which further confirms the optimality of the data schedule.

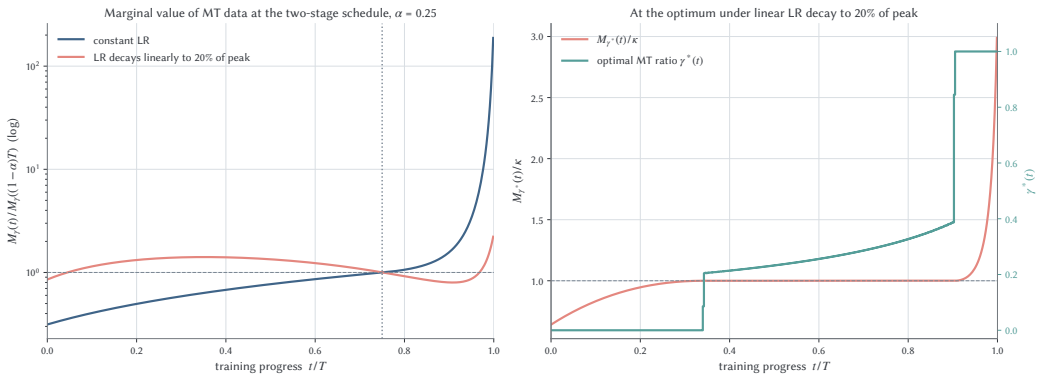


Figure 33: Numerical illustration of the KKT mechanism. *Left:* At the two stage schedule with $\alpha = 0.25$, the marginal value of MT data is increasing under a constant learning rate, whereas linear decay with $c = 0.8$ violates the switching condition around $(1 - \alpha)T$. *Right:* At the numerical optimum under linear decay, the normalized marginal value equals the KKT multiplier throughout the interior mixing phase and satisfies the appropriate inequalities on the two boundary phases.