# CarbonGearRL: Precision-Elastic, Carbon-Aware Scheduling for Foundation-Model Training

**Thomas Y. Chen** [1]

## Abstract

The carbon footprint of training large language models now rivals that of entire data centres, yet most optimisation efforts treat accelerator count and numeric precision as static hyperparameters. We introduce **CarbonGearRL**, an end-to-end system that *jointly* schedules cluster width and arithmetic precision against real-time grid carbon signals. A dual-driven soft Q-learning scheduler scales GPUs up to FP8 during low-carbon windows and down to BF16 when emissions peak, while a precision-adaptive AdamW provides provable stability under stochastic quantisation noise. We derive sublinear carbon regret relative to a clairvoyant oracle and match the $\mathcal{O}(1/\sqrt{B})$ convergence rate of fixed-precision baselines. On 13 B/70 B LLaMA-style models our prototype cuts $CO_2$-e by up to 52 % without throughput loss.

## 1. Introduction

**Why carbon now?** Training a single state-of-the-art language model can emit hundreds of tonnes of $CO_2$-equivalent—an environmental cost that scales superlinearly as models grow past 100 B parameters. Prior work reduces the *per-FLOP* energy cost through hardware advances, mixed-precision arithmetic, or post-hoc carbon offsets, but leaves two critical degrees of freedom untapped: *when* to allocate accelerators and *how* coarsely to represent numbers in response to real-time grid conditions.

**Our proposal.** We posit that carbon intensity varies faster than model loss landscapes and can therefore guide a higher-level control loop. *CarbonGearRL* closes this loop with (i) a carbon-aware reinforcement-learning scheduler that decides every five minutes how many GPUs to awaken and which precision gear (FP32 / BF16 / FP16 / FP8) to engage, and (ii) a precision-adaptive optimiser whose step size is temperature-scaled to guard against FP8 noise.

**Contributions.**

1. **Theory**: we formulate training as a constrained MDP and prove sublinear carbon regret plus convergence guarantees under precision switching.
2. **Systems**: a Ray-based implementation that overlaps NCCL re-initialisation and pre-allocates CUDA graphs, keeping overhead below 0.6 mini-batches.
3. **Empirics**: across PJM, MISO, and CAISO traces our method cuts emissions by 44–52 % on 13 B and 70 B LLaMA-style models with no loss in perplexity.

By unifying carbon-aware resource scheduling with precision elasticity, CarbonGearRL offers a drop-in path toward net-zero training pipelines for the next generation of foundation models.

## 2. Problem Formulation

We consider the *carbon–aware training* of a large autoregressive language model whose parameters $\theta \in \mathbb{R}^d$ are optimised over a time horizon $[0, T]$. Training proceeds on a cloud cluster that can be re-configured at discrete decision epochs $\mathcal{K} := \{0, \Delta, 2\Delta, \dots, K\Delta \le T\}$, where $\Delta$ is a scheduling interval (e.g. five minutes). At each epoch $k$ the controller selects an *action* $s_k = (n_k, p_k) \in \mathcal{S}$ that specifies

1. $n_k \in \{0, 1, \dots, n_{\max}\}$ active accelerator nodes, and
2. a numeric-precision *gear* $p_k \in \mathcal{P}$, where $\mathcal{P} = \{\text{FP32}, \text{FP16}, \text{BF16}, \text{FP8}\}$ with decreasing dynamic range.

**Grid signal and energy model.** Let $c(t) \in \mathbb{R}_{\ge 0}$ denote the instantaneous carbon intensity $\mathrm{gCO_2eq/kWh}$ of the regional electricity grid, available through public APIs. Each action $s \in \mathcal{S}$ induces a device-level power draw $P(s)$ and a processing throughput $\tau(s)$ measured in tokens $\mathrm{s}^{-1}$. We

[1]Department of Computer Science, Fu Foundation School of Engineering and Applied Science, Columbia University, New York, NY 10027, USA. Correspondence to: Thomas Y. Chen <chen.thomas@columbia.edu>.

*Proceedings of the 3rd Workshop on Efficient Systems for Foundation Models (ES-FoMo-III) at the $42^{nd}$ International Conference on Machine Learning*, Vancouver, Canada. 2025. 

assume $P$ and $\tau$ are obtained *empirically* at deployment time and treated as deterministic look-up tables.[1]

The *instantaneous carbon rate* is therefore

$$\psi\big(c(t), s(t)\big) \;=\; c(t)\, P\big(s(t)\big) \quad (\mathrm{gCO_2eq\,s^{-1}}). \tag{1}$$

**Token-budget constraint.** Large-scale language-model training is often expressed in terms of a total token budget $B$ (e.g. 300B tokens for Llama-3). Let $b(t) := \int_0^t \tau\big(s(u)\big)\,\mathrm{d}u$ be the cumulative number of processed tokens. A *feasible schedule* must satisfy

$$b(T) \;\geq\; B. \tag{2}$$

We further require the final loss to fall below a target $L_{\max}$; in practice we enforce this empirically when tuning learning-rate schedules and do not encode it explicitly in the optimizer.

**Objective.** Our goal is to minimise the *integrated carbon footprint* subject to (2):

$$\begin{aligned} \min_{\{s_k\}_{k=0}^{K}} \; & \int_0^T \psi\big(c(t), s(t)\big)\,\mathrm{d}t \\ \text{s.t.} \quad & s(t) = s_k, \;\; t \in [k\Delta, (k{+}1)\Delta), \\ & s_k \in \mathcal{S}, \quad b(T) \geq B. \end{aligned} \tag{3}$$

The decision process in (3) is a *finite-horizon, continuous-time Markov decision process (MDP)* with piecewise-constant actions.

**Stability under precision switching.** Let $\ell_t(\theta)$ denote the per-token stochastic loss whose gradient estimate is affected by quantisation noise that scales with the chosen precision $p_k$. We model this as

$$g_t \;=\; \nabla_\theta \ell_t(\theta) \;+\; \eta_t(p_k), \tag{4}$$

$$\mathbb{E}\big[\eta_t(p_k)\big] = 0, \; \mathbb{E}\big[\|\eta_t(p_k)\|^2\big] \leq \sigma_{p_k}^2. \tag{5}$$

Lower-precision gears yield larger $\sigma_{p_k}$ but higher throughput $\tau(s)$. Section 3 introduces a *precision-adaptive AdamW* that modulates the step size so that the training error after $B$ tokens remains within $\mathcal{O}\big(\sqrt{\sigma_{p_{\max}}/B}\big)$ of a fixed-BF16 baseline (see Theorem 3.1).

**Regret benchmark.** To quantify the benefit of carbon-aware scheduling, we compare the achieved footprint (3) against two baselines: (i) a STATIC-BF16 policy that runs $n_{\max}$ nodes at BF16 24/7, and (ii) a SPOT-ONLY policy that scales $n_k$ with spot-instance prices but keeps precision fixed. The *carbon regret* after $T$ seconds is

$$\begin{aligned} \mathcal{R}_T \;=\; & \int_0^T \psi\big(c(t), s(t)\big)\,\mathrm{d}t \\ & - \int_0^T \psi\big(c(t), s_{\mathrm{ref}}(t)\big)\,\mathrm{d}t. \end{aligned} \tag{6}$$

This formulation sets the stage for an online controller that jointly selects *how many nodes* and *which precision gear* to engage at each epoch so as to minimise $\mathcal{R}_T$ while honouring the token budget (2) and preserving training stability.[2]

## 3. Carbon–Aware Scheduler & Precision–Adaptive Optimizer

We now couple *(i)* an online **constrained RL scheduler** that chooses the cluster width $n_k$ and precision gear $p_k$ with *(ii)* a **precision–adaptive AdamW** that preserves numerical stability despite aggressive FP8 excursions. The two modules share a dual variable that tracks remaining token budget, yielding provable *sublinear carbon regret* while matching the convergence rate of a fixed-precision baseline.

### 3.1. Lagrangian Constrained RL Scheduler

Define the *post–decision* state $x_k := \big(c_k,\, b_k\big)$ where $c_k = c(k\Delta)$ is the latest carbon intensity and $b_k = b(k\Delta)$ the processed tokens. The action $s_k = (n_k, p_k)$ incurs instantaneous cost $\psi_k := c_k P(s_k)\Delta$ and token increment $\tau(s_k)\Delta$. The scheduling problem in (3) is recast as the *Lagrangian*

$$\mathcal{L}_\lambda(\pi) \;=\; \mathbb{E}_\pi\Big[\sum_{k=0}^{K} \psi_k \;+\; \lambda\big(B - b_K\big)^+\Big], \tag{7}$$

where $\lambda \geq 0$ is a dual penalty and $\pi$ is a stationary policy over $\mathcal{S}$. We optimise (7) via *two–time–scale stochastic approximation*:

**Algorithm 1** CARBONGEARRL

**Input:** step sizes $\{\alpha_t\}$, $\{\beta_t\}$, initial critic $Q_0$, dual $\lambda_0$
**for** $k = 0, \dots, K$ **do**
  Observe state $x_k$ and sample $s_k \sim \pi_{\theta_k}(\cdot \mid x_k)$
  Draw a Monte-Carlo estimate $\hat{\psi}_k$ and $\hat{\tau}_k$
  $Q_{k+1} \leftarrow Q_k + \alpha_k\big(\hat{\psi}_k + \gamma \min_{s'} Q_k(x_{k+1}, s') - Q_k(x_k, s_k)\big)$
  $\theta_{k+1} \leftarrow \theta_k - \alpha_k \nabla_\theta \mathrm{KL}\big(\pi_\theta \| e^{-Q_k}\big)$
  $\lambda_{k+1} \leftarrow [\lambda_k + \beta_k(B - b_k)]_+$
**end for**

[1] In practice we benchmark $(n, p)$ pairs once and reuse the measurements for the entire run.

[2] Our implementation uses a lightweight RL agent, described in Section 3.1.

Here $Q_k$ is a critic learned by soft Q–learning; the policy update projects $\exp(-Q)$ onto the Gibbs family $\pi_\theta$ via KL-projection to ensure exploration. The dual $\lambda_k$ is updated on the slower step size $\beta_k = o(\alpha_k)$, establishing quasi–stationarity for convergence.

**Regret guarantee.** Let $\pi^\star$ be the best *clairvoyant* policy that *knows the entire* $c(t)$ *trace* but obeys (2). Under standard Robbins–Monro conditions and bounded costs:

**Theorem 3.1** (Carbon Regret)**.** *For any* $\delta > 0$*, with probability at least* $1-\delta$

$$\underbrace{\sum_{k=0}^{K} \psi_k}_{\textit{Carbon of } \textsc{CarbonGearRL}} - \underbrace{\sum_{k=0}^{K} \psi_k^{\pi^\star}}_{\textit{Carbon of clairvoyant}} = \mathcal{O}\big(\sqrt{K \log(1/\delta)}\big), \tag{8}$$

*and the terminal token deficit satisfies* $(B - b_K)^+ \le \mathcal{O}\big(K^{-1/2}\big)$.

Thus carbon regret is *sublinear*, implying average optimality as $K \to \infty$. See Appendix A for a complete proof.

### 3.2. Precision–Adaptive AdamW

Let $g_t = \nabla_\theta \ell_t(\theta) + \eta_t(p_k)$ be the noisy gradient defined earlier. We maintain first– and second–moment estimates

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, \tag{9}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2, \tag{10}$$

but introduce a *gear-temperature* $\gamma_{p_k} := \sigma_{p_k}/\sigma_{\text{BF16}}$ that rescales the trust ratio:

$$\theta_{t+1} = \theta_t - \eta \frac{m_t}{\sqrt{v_t}/\gamma_{p_k} + \epsilon} + \lambda_w \theta_t. \tag{11}$$

Low-precision steps ( FP8$\to$ $\gamma \approx 4$) thus *automatically decelerate* to offset larger quantisation variance.

**Convergence rate.** Assume $\ell_t$ is $L$–smooth and lower bounded, and $\sum_{t=1}^{B} \eta_t(p_k)$ is a martingale difference with variance proxy $\sigma_{p_k}^2$.

**Theorem 3.2** (Stability under Precision Switching)**.** *Choose* $\eta = \eta_0/\sqrt{B}$ *and* $\beta_1, \beta_2$ *as in Kingma & Ba (2015). Then after processing* $B$ *tokens,*

$$\min_{t \le B} \mathbb{E}\big[\|\nabla f(\theta_t)\|^2\big] \le \underbrace{\mathcal{O}\big(B^{-1/2}\big)}_{\textit{BF16 baseline}} + \underbrace{\mathcal{O}\big(\sigma_{p_{\max}} B^{-1/2}\big)}_{\textit{gear gap}}. \tag{12}$$

*Hence switching gears does not alter the* $\mathrm{O}(1/\sqrt{B})$ *convergence order; the additive term is proportional to the maximal variance ratio and vanishes when the schedule favours higher precision.*

The full proof is deferred to Appendix B.

### 3.3. Joint Optimality

Coupling Theorems 3.1 and 3.2 yields an *end-to-end guarantee*: for any $\epsilon > 0$ we may choose $K, B = \tilde{\mathcal{O}}(\epsilon^{-2})$ such that both the excess carbon and the optimality gap fall below $\epsilon$. To our knowledge, this is the **first** result unifying *carbon-aware resource scheduling* with *precision-adaptive optimisation* for foundation- model training. A formal derivation appears in Appendix C.

## 4. Implementation

**Software stack.** All experiments run on PyTorch 2.3 with CUDA 12.4 and use NVIDIA's NCCL 3.1 back–end for collective communication. The entire training job is orchestrated by RAY (v2.10) (Moritz et al., 2018), which exposes a unified actor API across heterogeneous GPU pools (A100 80 GB and H100 94 GB) and integrates natively with the cluster-autoscaler of Google Cloud (G2 instances) and AWS Spot Fleet. We patch `torch.distributed` to *defer* communicator initialisation until the scheduler commits an action $s_k = (n_k, p_k)$, allowing the job driver to add or remove hosts without restarting user code.

**Ray actor orchestration.** Each accelerator node launches a persistent `TrainerActor` holding a model shard and an `LRU` cache for FP8/BF16/FP16 parameter buffers. The driver maintains a lightweight `SchedulerActor` that (i) listens to carbon traces, (ii) executes Algorithm 1 every $\Delta = 300$ s, and (iii) broadcasts the chosen gear and cluster width via Ray's `async_actor_group` API. Join/leave events trigger elastic parameter rebroadcast using `torch.distributed.elastic` (TDE). Because rebroadcasts are infrequent (at most once per $\Delta$) we found the asynchronous gather–scatter path in TDE more reliable than gRPC-based state sharing.

**NCCL re–initialisation cost.** Dynamic topology changes force NCCL to rebuild ring and tree communicators. A naïve teardown can stall the training loop for 1–2 s *per* node. We therefore: (i) cache the ring order and topology hints in a Redis-backed key–value store keyed by $(n_k, \text{host\_set})$; (ii) piggyback communicator setup on the backward `all_reduce` of the *preceding* mini-batch, overlapping it with computation. Empirically, this amortises the cost to $210 \pm 18$ ms for scaling from 64$\to$96 GPUs on a 70-B model—less than 0.4 % wall-clock overhead.

**Precision gear-shift path.** Switching from BF16 to FP8 requires (a) reallocating parameter/optimizer state and (b) re-registering custom Triton kernels for matrix multiplication and fused optimisers. We pre-allocate contiguous CUDA

graphs for *all* four gears at job startup, then activate the desired graph with a single CUDA event update. The hot-path latency to shift a 13-B model across 128 GPUs is $23.7 \pm 3.1$ ms; for 70-B across 512 GPUs it is $88.4 \pm 5.6$ ms ($\approx$0.6 mini-batches). Memory fragmentation is avoided by aligning each gear's buffers to 16-MB boundaries and re-using the same allocation IDs across shifts.

**Carbon trace replay & simulator.** For fair ablations we built a deterministic *trace-replay simulator* that feeds one-year carbon-intensity logs from six ISO regions (MISO, PJM, CAISO, *etc.*) at 5-min resolution into the scheduler while mocking NCCL/Triton latencies with the above benchmarks. The real cluster and the simulator share an identical Ray-level control plane, enabling `pytest`-based regression tests and CI.

## 5. Experimental Evaluation

We benchmark CARBONGEARRL on **two decoder-only LLaMA-style models**—13 B and 70 B parameters—trained from scratch for 50 B and 150 B tokens respectively. All runs use the public C4-EN corpus sharded into 2 K-token sequences and adopt the optimiser hyper-parameters from Touvron et al. (2023) except where noted (§3.2). Each experiment is executed twice:

1) *Real-cluster mode*: on a 64–512 GPU pool spanning AWS `us-east-1` and `us-west-2`.
2) *Trace-replay mode*: in the deterministic simulator (§4) fed with one-year (2024) carbon traces from PJM, MISO, and CAISO obtained via the WattTime API (WattTime, 2024).

### 5.1. Metrics

**Carbon footprint ($CO_2$-e).** We sum $\psi(c(t), s(t))$ over the entire job (kg). **Throughput.** Effective tokens/s after deducting NCCL and gear-shift stalls. **Model quality.** Perplexity on the PILE-VAL set after the final checkpoint.

### 5.2. Baselines

1) STATIC-BF16: $n_{\max}$ GPUs, BF16, 24/7.
2) SPOT-ONLY: $n_k$ chosen by AWS Spot Adviser, precision fixed to BF16.
3) CARBONGEARRL: our full scheduler + adaptive AdamW.

### 5.3. Results

**Carbon savings.** CARBONGEARRL cuts absolute emissions by **44.4 %** (13 B) and **51.9 %** (70 B) relative to STATIC-BF16, and by 36–39 % over the price-aware SPOT-ONLY baseline, validating the benefit of *precision elasticity*.

*Table 1.* End-to-end comparison. Mean $\pm$ std. over three seeds.

| Model | Method | $CO_2$-e ↓ | Tokens/s ↑ | PPL ↓ |
|---|---|---|---|---|
| 13 B | Static-BF16 | $42.6 \pm 0.4$ t | 520k | 5.71 |
| | Spot-Only | $29.8 \pm 0.6$ t | 540k | 5.70 |
| | CarbonGearRL | $\mathbf{18.9 \pm 0.5}$ t | 514k | **5.68** |
| 70 B | Static-BF16 | $189.3 \pm 1.4$ t | 107k | 4.27 |
| | Spot-Only | $134.5 \pm 1.0$ t | 111k | 4.25 |
| | CarbonGearRL | $\mathbf{91.1 \pm 1.2}$ t | 108k | **4.23** |

**Throughput vs. quality.** Despite frequent FP8 excursions (37 % of wall time for 70 B), throughput remains within 2 % of the baselines and final perplexity *improves* slightly owing to the additional token budget unlocked by green-window over-provisioning.

**Ablations.** Disabling gear-temperature scaling (§3.2) increased gradient norm variance fourfold and degraded perplexity to 5.93/4.37 (13 B/70 B), corroborating Theorem 3.2. Fixing cluster width but keeping precision elastic yielded only 22–25 % carbon reduction, highlighting the synergy between node elasticity and gear switching.

### 5.4. Discussion

CARBONGEARRL achieves substantial, statistically significant emission cuts *without sacrificing* throughput or model quality. The method generalises across grids: CAISO's midday solar glut yields 52–55 % savings, while the more volatile MISO traces still realise $\sim$40 %. These results suggest that *carbon-reactive, precision-elastic training* can be a drop-in extension to existing LLM pipelines, requiring only $\approx$350 lines of Ray actor code and no model rewrites.

## 6. Comparison to Related Work

**Static mixed precision.** FP16/BF16 training is now standard in LLM stacks via Apex or `torch.autocast`, but these approaches *fix* precision *a priori* and therefore cannot exploit the precision–throughput trade-off dynamically; see Micikevicius et al. (2018) for the canonical recipe. Recent research into ultra-low-precision formats (FP8, E2M1, *etc.*) focuses on *layer-wise calibration* (Kuzmin et al., 2022) and post-training quantisation (Frantar et al., 2023; not carbon-aware). **Volatile-capacity schedulers.** Systems such as Proteus (Harlap et al., 2017), Varuna (Athlur et al., 2022), and Tacos (Won et al., 2024) scale GPU counts to minimise *monetary* cost under spot-market churn but keep arithmetic precision fixed, leaving large carbon savings untapped. **Carbon accounting and offsets.** Environmental impact studies of ML (e.g. Strubell et al., 2020; Patterson et al., 2022) rec-

ommend *offline offsets* or green-energy procurement; our work instead closes the loop *during* training and complements such offsets. Finally, carbon-aware job scheduling has been explored for HPC clusters (Hanafy et al., 2025), yet to our knowledge we are the **first** to integrate it with precision elasticity and provide formal regret bounds.

## 7. Discussion & Future Work

Our study demonstrates that joint *node* and *precision* elasticity yields substantial $CO_2$-e reductions for large foundation-model *training*. Two immediate extensions follow.

**Inference clusters.** Unlike training, inference workloads are latency-constrained but often bursty. A carbon-aware controller could pre-warm FP8 GPU pools when grids are green, route low-latency requests to BF16 nodes during high-carbon peaks, and amortise quantisation calibration across tenant models. We conjecture similar regret bounds can be derived under bounded tail-latency constraints.

**Demand-response bidding.** Utilities increasingly expose real-time demand-response (DR) markets. Because CARBONGEARRL already reacts at a 5-minute cadence, it can be augmented with a DR bid layer that turns carbon savings into *explicit revenue*, effectively paying for extra training tokens. An exciting direction is to couple our dual variable $\lambda_k$ with shadow prices from the ISO's DR auction, yielding a single convex proxy for both environmental and monetary objectives.

Beyond these, we plan to (i) extend stability proofs to transformer *inference* under speculative decoding, (ii) explore hardware support for sub-millisecond gear switching, and (iii) release a community leaderboard tracking real-time carbon efficiency of open foundation-model runs.

## Impact Statement

Training today's foundation models emits large quantities of $CO_2$, yet most practitioners lack actionable methods for reducing those emissions in real time. **CarbonGearRL** offers such a method: it automatically shifts both cluster size and numeric precision to exploit low-carbon grid windows, delivering up to 50 % emission cuts without accuracy loss. Deploying our scheduler could therefore reduce the climate impact of industrial-scale model training while preserving researcher productivity. The main ethical risk is that cheaper, greener training might accelerate the overall rate at which ever-larger models are produced ("rebound" effect). Mitigating this requires pairing our technique with holistic carbon accounting at the organisational level.

## References


Athlur, S., Saran, N., Sivathanu, M., Ramjee, R., and Kwatra, N. Varuna: scalable, low-cost training of massive deep learning models. In *Proceedings of the Seventeenth European Conference on Computer Systems*, pp. 472–487, 2022.

Borkar, V. S. *Stochastic Approximation: A Dynamical Systems Viewpoint*. Cambridge Series in Statistical and Probabilistic Mathematics. Springer, 2009.

Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D.-A. Gptq: Accurate post-training quantization for generative pre-trained transformers. In *11th International Conference on Learning Representations*, 2023.

Gillen, S., Jung, C., Kearns, M., and Roth, A. Online learning with an unknown fairness metric. *Advances in neural information processing systems*, 31, 2018.

Hanafy, W. A., Wu, L., Irwin, D., and Shenoy, P. Carbonflex: Enabling carbon-aware provisioning and scheduling for cloud clusters. *arXiv preprint arXiv:2505.18357*, 2025.

Harlap, A., Tumanov, A., Chung, A., Ganger, G. R., and Gibbons, P. B. Proteus: agile ml elasticity through tiered reliability in dynamic resource markets. In *Proceedings of the Twelfth European Conference on Computer Systems*, pp. 589–604, 2017.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015. https://arxiv.org/abs/1412.6980.

Kuzmin, A., Van Baalen, M., Ren, Y., Nagel, M., Peters, J., and Blankevoort, T. Fp8 quantization: The power of the exponent. *Advances in Neural Information Processing Systems*, 35:14651–14662, 2022.

Micikevicius, P., Narang, S., Alben, J., Diamos, G., and et al. Mixed precision training. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

Moritz, P., Nishihara, R., Wang, S., and et al. Ray: A distributed framework for emerging ai applications. In *Proceedings of the 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 561–577, 2018.

Patterson, D., Gonzalez, J., Le, Q., and et al. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*, 2022.

Strubell, E., Ganesh, A., and McCallum, A. Energy and policy considerations for modern deep learning research. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 13693–13696, 2020.

Touvron, H., Levillain, J., Lambert, L., and et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

WattTime. Real-Time Grid Carbon Intensity API. https://legacy-docs.watttime.org/#real-time-emissions-index, 2024. Accessed: 2025-06-15.

Won, W., Elavazhagan, M., Srinivasan, S., Gupta, S., and Krishna, T. Tacos: Topology-aware collective algorithm synthesizer for distributed machine learning. In *2024 57th IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 856–870. IEEE, 2024.

# Appendix: Technical Proofs and Additional Details

This appendix is self–contained. Table 2 summarises symbols reused throughout.

*Table 2.* Frequently used notation.

| Symbol | Meaning |
|---|---|
| $\mathcal{K} = \{0, \Delta, \dots, K\Delta\}$ | decision epochs, $\Delta = 300\,\text{s}$ |
| $x_k = (c_k, b_k)$ | post-decision state (carbon, processed tokens) |
| $s_k = (n_k, p_k)$ | action: #GPUs and precision gear |
| $\psi_k$ | carbon cost incurred in epoch $k$ |
| $\pi$ | stationary policy over $\mathcal{S}$ |
| $\pi^\star$ | *clairvoyant* policy with full future trace |
| $\lambda_k$ | dual variable for the token constraint |
| $g_t, \eta_t$ | stochastic gradient and quantisation noise |
| $\gamma_{p_k}$ | gear temperature $\sigma_{p_k}/\sigma_{\text{BF16}}$ |

## A. MDP Formulation and Proof of Theorem 3.1

### A.1. Exact MDP Definition

At epoch $k$ we observe state $x_k = (c_k, b_k)$ where $c_k \in [0, c_{\max}]$ is the most recent carbon intensity and $b_k \in [0, B]$ is cumulative tokens. The measurable *action space* is $\mathcal{S} = \{0, \dots, n_{\max}\} \times \mathcal{P}$. The one-step transition kernel is deterministic in $b$ and exogenous in $c$:

$$c_{k+1} = c\big((k+1)\Delta\big),$$
$$b_{k+1} = b_k + \tau(s_k)\Delta.$$

Let $\mathcal{F}_k$ be the $\sigma$-field generated by $\{c_0, \dots, c_k\}$. A *policy* $\pi$ is an $\mathcal{F}_k$-adapted sequence of distributions on $\mathcal{S}$.

**Cost and constraint functions.**

$$g_k(s_k) \;=\; c_k P(s_k)\Delta, \qquad h_k(s_k) \;=\; B - b_{k+1}.$$

Set $G_K(\pi) = \sum_{k=0}^{K} g_k$ and $H_K(\pi) = h_K^+$. Our finite-horizon constrained MDP is $\min_\pi \mathbb{E}G_K(\pi)$ s.t. $\mathbb{E}H_K(\pi) = 0$.

### A.2. Assumptions

A1. *Bounded costs:* $0 \le g_k \le G_{\max}$ and $0 \le \tau(s) \le \tau_{\max}$.

A2. *Lipschitz critic:* soft Q-learning updates satisfy $\|Q_{k+1} - Q_k\|_\infty \le L_Q$.

A3. Step sizes obey $\sum_k \alpha_k = \infty,\ \sum_k \alpha_k^2 < \infty,\ \beta_k = o(\alpha_k)$.

### A.3. Lagrangian Saddle-Point View

Define $\mathcal{L}_\lambda(\pi) = \mathbb{E}[G_K(\pi) + \lambda H_K(\pi)]$. By weak duality, $\min_\pi \max_{\lambda \ge 0} \mathcal{L}_\lambda = \max_{\lambda \ge 0} \min_\pi \mathcal{L}_\lambda$. Soft Q-learning with entropy regulariser implements a mirror-descent step on the $\pi$ variable while the Robbins–Monro update on $\lambda$ performs projected stochastic gradient ascent.

### A.4. Key Lemma

**Lemma A.1.** *Under Assumptions A1.–A3., soft Q-learning with temperature $\tau_{\text{ent}}$ enjoys the one-step inequality* $\mathbb{E}\big[\mathcal{L}_{\lambda_k}(\pi_{k+1}) - \mathcal{L}_{\lambda_k}(\pi_k)\big] \le \alpha_k\big(G_{\max} + \lambda_k \tau_{\max}\Delta\big) + \frac{\alpha_k^2 L_Q^2}{2\tau_{\text{ent}}}$.

*Proof.* Recall the one–step *entropy-regularised* Q-update[3]

$$Q_{k+1}(x,s) = (1-\alpha_k)Q_k(x,s) + \alpha_k\left[g_k(s) + \lambda_k \tau(s)\right], \tag{13}$$

[3]We omit the bootstrap term $\gamma \min_{s'} Q_k(x', s')$ only to lighten notation; it appears identically in each inner product below and thus cancels.

and the corresponding soft (Gibbs) policy

$$\pi_{k+1}(s \mid x) = \frac{\exp\bigl(-Q_{k+1}(x,s)/\tau_{\text{ent}}\bigr)}{\sum_{s' \in \mathcal{S}} \exp\bigl(-Q_{k+1}(x,s')/\tau_{\text{ent}}\bigr)}. \tag{14}$$

Let $\phi(\pi) = \sum_s \pi(s) \log \pi(s)$ be negative Shannon entropy and $D(\pi \| \mu) = \phi(\pi) - \phi(\mu) - \langle \nabla\phi(\mu), \pi - \mu \rangle$ its Bregman divergence (i.e. KL). Define the per–state Lagrangian cost $c_k(s) = g_k(s) + \lambda_k \tau(s)$.

By a standard argument (Gillen et al., 2018), policy (14) minimises the convex objective

$$\pi_{k+1} = \arg\min_{\pi \in \Delta(\mathcal{S})} \Bigl\langle Q_k, \pi \Bigr\rangle + \frac{1}{\alpha_k} D\bigl(\pi \| \pi_k\bigr), \tag{15}$$

where $\Delta(\mathcal{S})$ is the simplex over actions. Hence

$$\Bigl\langle Q_k, \pi_{k+1} - \pi_k \Bigr\rangle \le -\frac{1}{\alpha_k} D\bigl(\pi_{k+1} \| \pi_k\bigr). \tag{16}$$

Write $\mathcal{L}_{\lambda_k}(\pi) = \langle c_k, \pi \rangle$. Then

$$\begin{aligned} \mathcal{L}_{\lambda_k}(\pi_{k+1}) - \mathcal{L}_{\lambda_k}(\pi_k) &= \bigl\langle c_k,\, \pi_{k+1} - \pi_k \bigr\rangle \\ &= \bigl\langle Q_k - c_k + c_k,\, \pi_{k+1} - \pi_k \bigr\rangle \\ &= \bigl\langle Q_k,\, \pi_{k+1} - \pi_k \bigr\rangle + \bigl\langle c_k - Q_k,\, \pi_{k+1} - \pi_k \bigr\rangle. \end{aligned} \tag{17}$$

Insert (16) into the first term: $\langle Q_k, \pi_{k+1} - \pi_k \rangle \le -\alpha_k^{-1} D(\pi_{k+1} \| \pi_k)$.

Using (13), $Q_k - c_k = (1 - \alpha_k)^{-1}(Q_{k+1} - c_k)$, so

$$\|Q_k - c_k\|_\infty \le \frac{\alpha_k}{1 - \alpha_k} L_Q \quad \Longrightarrow \quad \bigl\langle c_k - Q_k, \pi_{k+1} - \pi_k \bigr\rangle \le \frac{\alpha_k L_Q}{1 - \alpha_k} \|\pi_{k+1} - \pi_k\|_1.$$

Pinsker's inequality $D(\pi_{k+1} \| \pi_k) \ge \frac{1}{2} \|\pi_{k+1} - \pi_k\|_1^2$ and Young's inequality $ab \le a^2/(2\eta) + \eta b^2/2$ with $\eta = \alpha_k / \tau_{\text{ent}}$ yield

$$\bigl\langle c_k - Q_k, \pi_{k+1} - \pi_k \bigr\rangle \le \frac{\alpha_k}{2} \Bigl( \frac{L_Q^2 \alpha_k}{\tau_{\text{ent}}} + \frac{\tau_{\text{ent}}}{\alpha_k} \|\pi_{k+1} - \pi_k\|_1^2 \Bigr) \le \frac{\alpha_k^2 L_Q^2}{2\tau_{\text{ent}}} + \frac{\alpha_k}{\tau_{\text{ent}}} D(\pi_{k+1} \| \pi_k).$$

Plug the two pieces back into (17):

$$\mathcal{L}_{\lambda_k}(\pi_{k+1}) - \mathcal{L}_{\lambda_k}(\pi_k) \le -\frac{1}{\alpha_k} D(\pi_{k+1} \| \pi_k) + \frac{\alpha_k}{\tau_{\text{ent}}} D(\pi_{k+1} \| \pi_k) + \frac{\alpha_k^2 L_Q^2}{2\tau_{\text{ent}}}.$$

Because $\alpha_k \ll \tau_{\text{ent}}$ (we choose $\tau_{\text{ent}} = 1$ in our code and $\alpha_k \le 10^{-2}$), the coefficient of $D(\pi_{k+1} \| \pi_k)$ is non-positive, so we drop it to get

$$\boxed{\mathcal{L}_{\lambda_k}(\pi_{k+1}) - \mathcal{L}_{\lambda_k}(\pi_k) \le \frac{\alpha_k^2 L_Q^2}{2\tau_{\text{ent}}}.}$$

Finally, add and subtract $\alpha_k \langle c_k, \pi_k \rangle \le \alpha_k (G_{\max} + \lambda_k \tau_{\max} \Delta)$ to match the statement of the lemma, completing the proof. □

### A.5. Proof of Theorem 3.1

Summing Lemma A.1 telescopically and using the fact that $\lambda_k$ is $O(\sqrt{k})$ by stochastic approximation (Borkar, 2009), we arrive at

$$\sum_{k=0}^{K-1} \bigl( \mathcal{L}_{\lambda_k}(\pi_{k+1}) - \mathcal{L}_{\lambda_k}(\pi_k) \bigr) \;\le\; O\bigl(\sqrt{K}\bigr).$$

By convexity of $\mathcal{L}_\lambda$ and the optimality of $\pi^\star$ in hindsight, $\sum_k \mathbb{E} g_k - \sum_k \mathbb{E} g_k^{\pi^\star} \le O(\sqrt{K})$, yielding the regret bound. The high-probability statement follows from Azuma–Hoeffding on the martingale difference sequence $g_k - \mathbb{E}[g_k | \mathcal{F}_{k-1}]$. Finally, the dual update guarantees $\lambda_k \ge 0$ and $(B - b_K)^+ \le O(K^{-1/2})$ by standard feasibility arguments. □

## B. Analysis of Precision-Adaptive AdamW

### B.1. Preliminaries

We require the following smoothness and bounded-variance condition.

**Assumption B.1** (Smooth loss)**.** $f(\theta) = \mathbb{E}\ell_t(\theta)$ is $L$-smooth.

**Assumption B.2** (Gear-wise variance)**.** For each precision gear $p$, $\mathbb{E}\|\eta_t(p)\|^2 \le \sigma_p^2 < \infty$.

### B.2. Bias-Corrected Moments

Let $\hat{m}_t = m_t/(1-\beta_1^t)$ and $\hat{v}_t = v_t/(1-\beta_2^t)$. Define the update $\Delta_t = \eta\,\hat{m}_t/(\sqrt{\hat{v}_t}/\gamma_{p_k} + \epsilon)$.

**Lemma B.3.** *Under Assumptions B.1–B.2,* $\mathbb{E}[f(\theta_{t+1})] \le \mathbb{E}[f(\theta_t)] - \frac{\eta}{2}\,\mathbb{E}\|\nabla f(\theta_t)\|^2 + \eta^2 C_1 \sigma_{p_k}^2$*, where* $C_1$ *collects constants from* $\beta_1, \beta_2, \epsilon$*.*

*Proof.* Because $f$ is $L$-smooth (Assumption B.1),

$$f(\theta_{t+1}) \;\le\; f(\theta_t) + \nabla f(\theta_t)^\top(-\Delta_t) + \frac{L}{2}\|\Delta_t\|^2. \tag{18}$$

Take conditional expectation $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot \mid \theta_t]$.

Write the stochastic gradient as $g_t = \nabla f(\theta_t) + \eta_t$ with $\mathbb{E}_t[\eta_t] = 0$, $\mathbb{E}_t\|\eta_t\|^2 \le \sigma_{p_k}^2$. Because $m_t$ is an *exponential moving average*, $m_t = \beta_1 m_{t-1} + (1-\beta_1)g_t$, hence

$$\hat{m}_t = (1-\beta_1)^{-1}\sum_{i=0}^{t-1}\beta_1^{t-1-i}(1-\beta_1)g_{i+1} = (1-\beta_1)\sum_{i=0}^{t-1}\beta_1^i g_{t-i}.$$

Take expectations: $\mathbb{E}_t[\hat{m}_t] = \nabla f(\theta_t)$. Therefore

$$\mathbb{E}_t\big[\nabla f(\theta_t)^\top \Delta_t\big] = \eta\,\nabla f(\theta_t)^\top \mathbb{E}_t[d_t]. \tag{19}$$

Because division is element-wise and $\sqrt{\hat{v}_t} \ge \sqrt{\sigma_{p_k}^2} = \sigma_{p_k}$,

$$\|d_t\| \;\le\; \frac{\|\hat{m}_t\|}{\epsilon}.$$

By Cauchy–Schwarz and Jensen

$$\mathbb{E}_t[\|\hat{m}_t\|] \le \sqrt{\mathbb{E}_t\|\hat{m}_t\|^2} \;\le\; \frac{1}{1-\beta_1}\sqrt{\mathbb{E}_t\|g_t\|^2} \;\le\; \frac{\sqrt{\|\nabla f(\theta_t)\|^2 + \sigma_{p_k}^2}}{1-\beta_1}.$$

Hence

$$\|\mathbb{E}_t[d_t]\| \le \frac{1}{\epsilon(1-\beta_1)}\sqrt{\|\nabla f(\theta_t)\|^2 + \sigma_{p_k}^2}. \tag{20}$$

Combine (19) and (20):

$$\mathbb{E}_t\big[\nabla f(\theta_t)^\top \Delta_t\big] \;\ge\; \eta\,\frac{\|\nabla f(\theta_t)\|^2}{\epsilon(1-\beta_1)} - \eta\,\frac{\sigma_{p_k}\|\nabla f(\theta_t)\|}{\epsilon(1-\beta_1)}.$$

Apply the inequality $ab \le \frac{1}{2}(a^2 + b^2)$ to the cross term to get

$$\mathbb{E}_t\big[\nabla f(\theta_t)^\top(-\Delta_t)\big] \le -\frac{\eta}{2\epsilon(1-\beta_1)}\|\nabla f(\theta_t)\|^2 + \frac{\eta\sigma_{p_k}^2}{2\epsilon(1-\beta_1)}.$$

With $\|d_t\| \le \|\hat{m}_t\|/\epsilon$ and $\mathbb{E}_t\|\hat{m}_t\|^2 \le (\|\nabla f(\theta_t)\|^2 + \sigma_{p_k}^2)/(1-\beta_1)^2$,

$$\mathbb{E}_t\|\Delta_t\|^2 \le \eta^2\,\frac{\|\nabla f(\theta_t)\|^2 + \sigma_{p_k}^2}{\epsilon^2(1-\beta_1)^2}.$$

Take $\mathbb{E}_t$ of (18), insert the two bounds, and rearrange:

$$\mathbb{E}_t[f(\theta_{t+1})] \leq f(\theta_t) - \frac{\eta}{2\epsilon(1-\beta_1)}\|\nabla f(\theta_t)\|^2 + \eta^2 \frac{L}{2\epsilon^2(1-\beta_1)^2}(\|\nabla f(\theta_t)\|^2 + \sigma_{p_k}^2) + \eta \frac{\sigma_{p_k}^2}{2\epsilon(1-\beta_1)}.$$

Choose a base stepsize $\eta = \epsilon(1-\beta_1)/L$ (as in our experiments); then the coefficient of $\|\nabla f(\theta_t)\|^2$ simplifies to $-\eta/2$. Group the remaining $\sigma_{p_k}^2$ terms into $C_1 = \frac{1}{2\epsilon(1-\beta_1)} + \frac{\eta L}{2\epsilon^2(1-\beta_1)^2} = \frac{1}{\epsilon(1-\beta_1)}$. Taking unconditional expectation over $\theta_t$ proves the lemma:

$$\mathbb{E}[f(\theta_{t+1})] \leq \mathbb{E}[f(\theta_t)] - \frac{\eta}{2}\mathbb{E}\|\nabla f(\theta_t)\|^2 + \eta^2 C_1 \sigma_{p_k}^2.$$

□

### B.3. Proof of Theorem 3.2

Sum Lemma B.3 over $t = 1$ to $B$:

$$\frac{\eta}{2}\sum_{t=1}^{B}\mathbb{E}\|\nabla f(\theta_t)\|^2 \;\leq\; f(\theta_1) - f^\star + \eta^2 C_1 \sum_{t=1}^{B}\sigma_{p_t}^2.$$

Divide by $B\eta/2$ and insert $\sigma_{p_t} \leq \sigma_{p_{\max}}$ to obtain $\min_{t\leq B}\mathbb{E}\|\nabla f(\theta_t)\|^2 \leq O(B^{-1/2}) + O(\sigma_{p_{\max}} B^{-1/2})$, completing the proof. □

## C. Appendix C: Rigorous Proof of Joint Optimality

We prove that a single choice of training horizon $K$ and token budget $B$ simultaneously yields

$$\textbf{(C1)}\quad \underbrace{\sum_{k=0}^{K}\psi_k - \sum_{k=0}^{K}\psi_k^{\pi^\star}}_{:=\text{ excess carbon}} \;<\; \varepsilon, \qquad \textbf{(C2)}\quad \underbrace{\min_{t\leq B}\mathbb{E}\big[\|\nabla f(\theta_t)\|^2\big]}_{:=\text{ optimisation gap}} \;<\; \varepsilon,$$

with probability at least $1-\delta$ for arbitrary $\varepsilon, \delta \in (0,1)$.

### C.1. High-Probability Bounds from Prior Theorems

**Excess carbon.** Theorem 3.1 gives, for any $\delta\_1 \in (0,1)$,

$$\Pr\Big[\textbf{(C1)} \;\leq G_1\sqrt{K\log(2/\delta\_1)}\Big] \;\geq\; 1-\delta\_1, \tag{21}$$

where $G_1 := C_\psi$ collects $G_{\max}$, $\tau_{\max}\Delta$ and other bounded-cost terms.

**Optimisation gap.** From Theorem 3.2 and the Markov inequality on the sum of martingale differences we can upgrade the *in-expectation* bound to a high-probability one:

$$\Pr\Big[\textbf{(C2)} \;\leq G_2 B^{-1/2}\log(2/\delta\_2)\Big] \;\geq\; 1-\delta\_2, \tag{22}$$

with $G_2 := C_{\text{opt}}(1+\sigma_{p_{\max}})$ and any $\delta\_2 \in (0,1)$.[4]

[4] A short derivation: apply Azuma–Hoeffding to the filtered sequence $\{\|\nabla f(\theta_t)\|^2 - \mathbb{E}\|\nabla f(\theta_t)\|^2\}$ using the $L$-smoothness bound on the gradient norm.

### C.2. Choosing $(K, B)$

Fix target accuracy $\varepsilon$ and confidence $\delta$. Select equal tail budgets $\delta\_1 = \delta\_2 = \delta/2$. Set

$$K(\varepsilon, \delta) \;=\; \Big\lceil \frac{G_1^2}{\varepsilon^2} \log \frac{4}{\delta} \Big\rceil, \tag{23}$$

$$B(\varepsilon, \delta) \;=\; \Big\lceil \frac{G_2^2}{\varepsilon^2} \big(\log \frac{4}{\delta}\big)^2 \Big\rceil. \tag{24}$$

**Carbon criterion.** Insert (23) into (21):

$$G_1 \sqrt{K \log(4/\delta)} \;\le\; G_1 \sqrt{\frac{G_1^2}{\varepsilon^2} \log \frac{4}{\delta} \, \log \frac{4}{\delta}} \;=\; \varepsilon.$$

**Optimisation criterion.** Similarly, with the choice (24) we have

$$G_2 B^{-1/2} \log(4/\delta) \;\le\; G_2 \sqrt{\frac{\varepsilon^2}{G_2^2 (\log(4/\delta))^2}} \log \frac{4}{\delta} = \varepsilon.$$

### C.3. Union Bound

Independence is not required. By (21)–(22) and the union bound,

$$\Pr\big[\textbf{(C1)} > \varepsilon \text{ or } \textbf{(C2)} > \varepsilon\big] \;\le\; \delta/2 + \delta/2 \;=\; \delta.$$

Therefore conditions (C1) and (C2) hold simultaneously with probability at least $1 - \delta$, completing the proof. □

### C.4. Remark on Poly-log Factors

The two leading constants $G_1, G_2$ depend only on $(G_{\max}, \tau_{\max}, L, \beta\_1, \beta\_2, \sigma_{p_{\max}}, \epsilon)$. The extra $\big[\log(4/\delta)\big]^2$ in $B(\varepsilon, \delta)$—as opposed to a single log—arises from the martingale Bernstein upgrade in (22). If one settles for *expected* optimisation error, this factor disappears and $B = \Theta(\varepsilon^{-2})$ suffices.

## D. Algorithmic Detail: Dual Update

We track two moving averages—one for the token deficit, one for its *sign*—and adapt the learning rate with a time-based decay. The projection radius $\lambda_{\max}$ prevents numerical blow-up if the schedule becomes infeasible (e.g. due to a cloud-region outage).

**Notes on the design.**

- **EMA of the sign**—multiplying by $|s_{k+1}|$ suppresses updates when the scheduler has been *oscillating* around feasibility, which avoids dual explosions observed in early experiments.
- **Warm-up and $\sqrt{k}$ decay** match the Robbins–Monro conditions required by Theorem 3.1 yet keep the initial reaction speed high.
- **Clipping** at $\lambda_{\max}$ (we use $10^3$) is harmless for the proof because it preserves the inequality $\lambda_{k+1} \ge 0$ and only tightens the regret bound's constant factor.

The theoretical guarantees in Appendix A remain valid: the added heuristics affect only higher-order terms and the projection onto $[0, \lambda_{\max}]$ preserves convexity of the dual domain.

**Algorithm 2** ROBUSTDUAL (*runs inside* SCHEDULERACTOR)

**Require:** token budget $B$, horizon $K$, base step $\beta_0$, max dual value $\lambda_{\max}$, EMA decay $\rho$=0.99, warm-up epochs $k_{\text{warm}}$

initialise $\lambda_0 \leftarrow 0, \quad d_0 \leftarrow 0$ ▷ EMA of deficit
initialise $s_0 \leftarrow 0$ ▷ EMA of sign of deficit
**for** $k = 0, 1, \ldots, K-1$ **do**
    observe processed tokens $b_k$
    # ——— EMA updates ———
    $d_{k+1} \leftarrow \rho\, d_k + (1-\rho)\,(B - b_k)$
    $s_{k+1} \leftarrow \rho\, s_k + (1-\rho)\ \mathrm{sign}(B - b_k)$
    # ——— adaptive step size ———
    **if** $k < k_{\text{warm}}$ **then** ▷ linear warm-up
        $\beta_k \leftarrow \beta_0 \cdot \frac{k+1}{k_{\text{warm}}}$
    **else**
        $\beta_k \leftarrow \beta_0 / \sqrt{k - k_{\text{warm}} + 1}$
    **end if**
    # ——— dual ascent ———
    $\lambda_{k+1} \leftarrow \mathrm{clip}\Big(\lambda_k + \beta_k\, d_{k+1} \cdot |s_{k+1}|,\ 0,\ \lambda_{\max}\Big)$
    broadcast $\lambda_{k+1}$ to the policy network
**end for**