Deep Reinforcement Learning for Autonomous Control of Supercritical CO_2 Brayton Cycles in Steel Industry Waste Heat Recovery

Sharath Sathish
University of York, York, UK

February 20, 2026

Abstract

This paper presents **sCO2RL**, an end-to-end deep reinforcement learning (RL) framework for autonomous control of a *supercritical CO_2 (sCO_2) recuperated Brayton cycle* recovering waste heat from steel industry electric-arc furnace (EAF) and basic-oxygen furnace (BOF) exhaust (200–1,200°C). The framework integrates a physics-faithful FMI 2.0 Co-Simulation model (FMU) compiled with OpenModelica, a Gymnasium environment with a seven-phase structured curriculum, Proximal Policy Optimisation (PPO) with trainable Lagrangian constraint multipliers for safe operation near the CO_2 critical point (31.1°C, 7.38 MPa), an MLP step-predictor surrogate (55,000,000 transitions, val_loss = 5×10^{-6}) for GPU-accelerated policy training at 250,000 steps/s, a Fourier Neural Operator (FNO) implemented via **NVIDIA PhysicsNeMo** for physics-operator surrogate validation ($R^2 = 1.000$), and a TensorRT-FP16 deployment path for sub-millisecond plant-edge inference.

The framework is validated on two training paths. On the *FMU-direct path* (5,013,504 steps, 8 parallel FMU workers), the RL agent outperforms a Ziegler–Nichols-tuned multi-channel PID baseline by **+30.3%, +30.4%, and +39.0%** in cumulative episode reward for Phases 0–2 (steady-state optimisation, ±30% gradual load following, and ±10°C ambient disturbance), with **zero safety violations** across all 140 evaluation episodes. On the *MLP surrogate path* (5,000,000 steps, 1,024 GPU-vectorised environments), PPO training completes in 23 minutes and achieves **18.5× lower net-power tracking error** than the PID baseline (0.122 MW vs. 2.259 MW), enabling rapid policy development prior to FMU fine-tuning.

The FNO surrogate ($R^2 = 1.000$, 76,600 unique Latin Hypercube trajectories) was found *architecturally incompatible* with step-by-step RL prediction due to its non-causal sequence-to-sequence training objective; an MLP step predictor with residual connections resolves this mismatch at sub-1% state prediction error. The deployment path achieves a p99 host latency of **0.046 ms**, exceeding the 1 ms plant-edge SLA by a factor of 22×.

Five non-trivial software engineering defects encountered during development are documented in detail — covering observation normalisation persistence, episode boundary detection, reward unit double-scaling, stale disturbance profiles, and constraint-violation gating — as actionable practitioner guidance for the Modelica-RL integration community.

To the authors' knowledge, this is the first publicly available framework combining (i) an OpenModelica-exported sCO_2 FMU, (ii) structured curriculum RL with Lagrangian safety constraints, (iii) an MLP step-predictor surrogate enabling 250,000 steps/s GPU training, (iv) NVIDIA PhysicsNeMo FNO surrogate validation ($R^2 = 1.000$), and (v) a TensorRT plant-edge deployment path. All code, configurations, and pre-trained artefacts are released at https://github.com/SharathSPhD/RLpower under the MIT licence.

1 Introduction

Steel manufacturing accounts for approximately 7–8% of global CO_2 emissions [1]. Electric arc furnaces (EAF) and basic oxygen furnaces (BOF) expel exhaust streams that oscillate between 200°C and 1,200°C with cycle periods of 1–15 minutes — a transient heat source profile that makes conventional thermodynamic bottoming cycles impractical and power recovery control exceedingly difficult. Recovering this thermal energy via a power cycle could displace significant grid electricity; at 10 MWe per furnace cluster and typical steel mill electricity prices, the economic potential is several million USD per year per installation. Sathish et al. [2] demonstrated the thermodynamic viability of sCO_2 Brayton cycles for steel plant waste heat recovery, motivating the present work on autonomous control.

Supercritical CO_2 (sCO_2) Brayton cycles [3] are an attractive bottoming cycle for this application: operating above the CO_2 critical point (31.1°C, 7.38 MPa) enables efficiencies of 27–40% at compact turbomachinery scales that are 100× smaller than equivalent steam plant, making the technology cost-competitive at industrial waste-heat magnitudes where steam plant would be uneconomic.

However, the fluid's near-critical thermodynamic properties introduce severe nonlinearity. Specific heat peaks at $c_p \approx 29.6\ \mathrm{kJ\,kg^{-1}\,K^{-1}}$ near 35°C/80 bar — more than 10× the ideal-gas value. A 1.5°C compressor inlet temperature drop demands 6% more cooling power, while the same temperature increase requires 18% less: a strongly asymmetric gain that defeats fixed-gain PID tuning during furnace transients [4]. The compressor inlet must remain strictly above the CO_2 critical temperature (31.1°C) to avoid two-phase flow that would damage turbomachinery; this constraint becomes safety-critical during cold startup and emergency trip scenarios.

Deep reinforcement learning (RL) offers an adaptive alternative: an agent trained on a physics-faithful digital twin can learn to anticipate and exploit thermodynamic nonlinearities without requiring an explicit analytical system model. Several key technical capabilities have matured that make this approach practical. First, the Functional Mockup Interface (FMI) standard enables physics-faithful simulation at the speed required for RL training, and OpenModelica models of sCO_2 cycles can be exported as FMI 2.0 Co-Simulation FMUs with embedded stiff solvers. Second, a lightweight residual MLP trained on $(s_t, a_t) \to s_{t+1}$ transitions enables GPU-vectorised RL at 250,000 steps/s — over 300× faster than CPU FMU simulation — while maintaining sub-1% state prediction error. Third, Fourier Neural Operators [5] learn physics operator mappings with GPU-accelerated inference; NVIDIA PhysicsNeMo provides production-grade GPU implementations achieving $R^2 = 1.000$ on held-out trajectories, although the non-causal sequence-to-sequence architecture is incompatible with step-by-step RL without architectural modifications (Section 3.5.2). Fourth, Constrained Policy Optimisation variants [6] with trainable Lagrangian multipliers enforce operational constraints (compressor inlet temperature, surge margin) throughout training and deployment. Finally, ONNX export followed by TensorRT FP16 compilation achieves sub-millisecond plant-edge inference latency, satisfying real-time control requirements.

Related work has demonstrated RL on building energy systems via Modelica/FMU environments [7, 8], and on organic Rankine cycle superheat control for internal combustion engine exhaust [9]. More recently, Zhu et al. [10] combined Fourier Neural Operators with model predictive control for sCO_2 cycle dynamics. The EU-funded iSOP doctoral network (Horizon Europe grant 101073266) trains 15 researchers specifically on sCO_2 transient modelling and novel control strategies [11], underscoring community recognition of this unsolved control problem. Sathish et al. [12] have also proposed active inference as an alternative advanced control paradigm for centrifugal compressor surge control, demonstrating the broader trend towards intelligent, adaptive control in turbomachinery applications.

Despite this growing interest, to the authors' knowledge no publicly available framework combines (i) a physics-faithful OpenModelica-exported sCO_2 FMU, (ii) structured curriculum RL with Lagrangian safety constraints, (iii) an MLP step-predictor surrogate enabling 250,000 steps/s GPU training, (iv) an NVIDIA PhysicsNeMo FNO surrogate path ($R^2 = 1.000$), and (v) a sub-millisecond TensorRT deployment artefact. sCO2RL fills this gap with a fully open-source implementation targeting the waste heat recovery application.

The contributions of this work are sixfold. The first contribution is a publicly available Gymnasium environment wrapping an OpenModelica-exported sCO_2 FMU with 14 observation variables, 4 actuator channels, a 7-phase structured curriculum, and Lagrangian safety constraints enforcing operation above the CO_2 critical point. Second, the paper demonstrates that PPO with Lagrangian constraints achieves +30–39% cumulative episode reward improvement over Ziegler–Nichols-tuned PID in steady-state and mild-transient scenarios (Phases 0–2), with zero safety violations across 140 evaluation episodes. Third, an MLP step-predictor surrogate (residual MLP, 4 layers, 512 hidden units, trained on 55,000,000 (s, a, s') transitions) enables GPU-vectorised PPO at 250,000 steps/s, achieving 18.5× lower net-power tracking error than the PID baseline in 23 minutes of training. Fourth, the integration of NVIDIA PhysicsNeMo FNO surrogate training with 76,600 unique Latin Hypercube-sampled FMU trajectories ($R^2 = 1.000$) is presented, along with empirical characterisation of both the data quality failure mode causing surrogate fidelity collapse ($R^2 = -77$) under dataset degeneracy and the FNO's architectural incompatibility with step-by-step RL prediction. Fifth, a TensorRT-FP16 deployment path achieving p99 inference latency of 0.046 ms — 22× under the 1 ms plant-edge SLA — is demonstrated. Sixth, a detailed diagnosis of five non-algorithmic training infrastructure defects encountered in practice is provided, covering observation normalisation persistence, episode boundary detection, reward unit double-scaling, stale disturbance profiles, and constraint-violation gating, as practitioner guidance for the FMU-RL integration community.

The remainder of this paper is organised as follows. Section 2 reviews related work on sCO_2 cycle control, reinforcement learning for thermodynamic systems, and FMU-based training environments. Section 3 describes the system architecture, including the physics simulation layer, Gymnasium environment, and surrogate models. Section 4 details the reward function, Lagrangian constraint formulation, curriculum design, and PID baseline methodology. Section 5 presents experimental results for both the FMU-direct and MLP surrogate training paths. Section 6 provides a control-theoretic analysis comparing RL and PID controllers. Section 7 documents five practitioner-relevant engineering defects, and Section 8 concludes with key findings and future work.

2 Literature Review

This section surveys the relevant literature across five domains: sCO_2 cycle modelling and industrial waste heat recovery, classical and advanced control strategies for sCO_2 systems, reinforcement learning applied to thermodynamic power cycles and building energy systems, FMU-based RL environments, and surrogate modelling with neural operators. The review identifies the specific gaps that the present work addresses.

2.1 sCO_2 Cycle Modelling and Waste Heat Recovery

Supercritical CO_2 Brayton cycles have attracted extensive research since Dostál et al.'s foundational study [3], which established the thermodynamic viability of sCO_2 cycles for nuclear applications with efficiencies exceeding 45% at turbine inlet temperatures above 550°C. Liu et al. [4] provide a comprehensive review of sCO_2 Brayton cycle configurations, covering simple, recompression,

partial-cooling, and intercooled topologies, and identify waste heat recovery (WHR) as a particularly promising application domain due to the compact turbomachinery and moderate temperature requirements.

For the steel industry WHR application specifically, Sathish et al. [2] demonstrated the thermodynamic feasibility of sCO_2 Brayton cycles for recovering waste heat from coke oven exhaust in an iron and steel plant, comparing sCO_2 configurations against existing steam Rankine bottoming cycles. That study established the thermal boundary conditions (intermittent EAF/BOF exhaust, 200–1,200°C, 1–15 minute cycles) that define the control challenge addressed in the present work. The intermittent nature of the heat source imposes transients that are qualitatively more severe than the load-following scenarios considered in most sCO_2 control literature, where heat source temperature is typically assumed constant or slowly varying.

Marchionni [13] provides a detailed design and experimental characterisation of an 830 kW sCO_2 test facility, documenting the practical engineering challenges of near-critical operation, including the extreme sensitivity of fluid properties to temperature and pressure perturbations that motivates the safety constraints enforced in this work.

2.2 Classical and Advanced Control for sCO_2 Cycles

Conventional control for sCO_2 cycles relies on multi-loop PID architectures [14]; however, the strongly nonlinear thermophysical properties near the critical point (specific heat diverges to $\approx$30 kJ/(kg·K) at 35°C, 80 bar) defeat fixed-gain controllers during large transients. Dyreby et al. [14] compare proportional-integral, feedforward, and combined strategies for recompression cycle load-following, establishing the classical control performance envelope that motivates data-driven alternatives. Marchionni et al. [15] examine the dynamic performance and control implementation of a simple recuperated sCO_2 cycle, focusing on inventory control for stable compressor inlet conditions and temperature control for part-load efficiency, and demonstrate that even well-tuned PID controllers exhibit significant overshoot during rapid transients.

The EU-funded iSOP doctoral network (Horizon Europe grant 101073266) trains 15 researchers on sCO_2 transient modelling and control [11], underscoring community recognition of the unsolved control challenge. Recent reviews of sCO_2 dynamic performance and control [16] identify conventional PID, feedforward, gain scheduling, and model predictive control (MPC) as the dominant approaches, while highlighting the need for more adaptive methods that can handle the strongly nonlinear dynamics without explicit analytical models.

In a complementary direction, Sathish et al. [12] proposed active inference (AIF) as a novel control paradigm for centrifugal compressor surge control, demonstrating that AIF — rooted in the free energy principle from neuroscience — offers several advantages over traditional PID and MPC by inherently integrating perception and action for adaptive decision-making under uncertainty. Although that work targeted compressor surge rather than full-cycle control, it illustrates the broader trend towards intelligent, non-PID control paradigms for turbomachinery applications and informs the motivation for applying deep reinforcement learning in the present work.

2.3 Data-Driven and Machine Learning Control for sCO_2

Machine learning approaches for sCO_2 control are nascent but advancing rapidly. Zhu et al. [10] apply Fourier Neural Operators (FNO) to characterise open-loop transient dynamics of a sCO_2 cycle and embed the learned model in a Model Predictive Controller, demonstrating non-minimum phase behaviour in the printed circuit heat exchanger outlet temperature that defeats simple PI feedback. Their work is most closely related to the present study, with key differences: the present work targets

a WHR cycle with external furnace disturbances (not a closed-loop nuclear application), employs RL rather than MPC, and provides a publicly available codebase.

Baek et al. [17] present a deep reinforcement learning approach for autonomous system-level control of an S-CO_2 power cycle under varying load conditions, using a two-stage training strategy combining surrogate pre-training and transfer learning. Their approach demonstrates stable multivariable control of turbine speed and compressor inlet temperature, validating the general feasibility of RL for sCO_2 cycle control. The present work extends this direction with a structured 7-phase curriculum, Lagrangian safety constraints, and an open-source implementation.

2.4 Reinforcement Learning for Thermodynamic Power Cycles

RL has been applied to related thermodynamic control problems across organic Rankine cycle (ORC) and building HVAC domains. Wang et al. [9] demonstrate that a soft actor-critic agent outperforms PID control for ORC superheat regulation under highly transient ICE exhaust, achieving superior generalisation to unseen disturbance profiles — a result analogous to the Phase 0–2 findings in this work.

In the building energy domain, a comprehensive 2024 survey [18] found that while RL-based HVAC control has demonstrated significant energy savings (up to 17% compared to standard control sequences), only 23% of RL studies have been deployed in real buildings, identifying a critical sim-to-real gap. The challenges of training environment diversification and safe deployment identified in that survey directly parallel the curriculum design and Lagrangian safety mechanisms developed in this work.

BOPTEST-Gym [8] provides a standardised benchmark for RL in building HVAC systems using FMU simulation, and its benchmarking methodology has informed the evaluation protocol design of the present work. OpenModelica-Microgrid-Gym [19] applies RL to electrical microgrids, demonstrating that physics-faithful OpenModelica FMUs can train viable RL policies in power systems contexts.

To the authors' knowledge, no prior work applies deep RL with structured curriculum learning and Lagrangian safety constraints to sCO_2 Brayton cycle WHR control.

2.5 FMU-Based RL Environments

Several frameworks have standardised the FMU-Gymnasium interface. ModelicaGym [7] provides a Gym wrapper for Modelica FMUs, validated on Cart-Pole, establishing the basic integration pattern adopted by later frameworks. FMUGym [20] extends this with uncertainty injection for robustness training. BOPTEST-Gym [8] targets building HVAC optimisation, providing standardised evaluation protocols that have been adopted by the HVAC RL community. None of these frameworks address thermodynamic power cycle control, 7-phase curriculum learning, or Lagrangian safety enforcement.

A critical practical gap in all existing FMU-RL frameworks, documented in detail in Section 7, is that none address observation normalisation persistence across checkpoint restarts, episode boundary alignment for multi-step FMU solvers, or reward unit double-scaling arising from FMU SI-unit conventions. These defects are latent in any FMU-SB3 integration and can render training completely ineffective without surfacing obvious error messages.

2.6 FNO Surrogate Models and NVIDIA PhysicsNeMo

Fourier Neural Operators [5] learn mappings between function spaces, making them well-suited for surrogate modelling of PDE-governed systems such as thermodynamic cycles. NVIDIA Physic-

sNeMo (formerly Modulus) [21] provides GPU-optimised implementations of physics-informed AI models including FNO, enabling large-scale training on NVIDIA hardware with minimal engineering overhead. The present work integrates PhysicsNeMo's FNO implementation into the RL surrogate pipeline, demonstrating both the practical benefits (simple API, GPU utilisation) and the data quality requirements (dataset degeneracy causes catastrophic surrogate failure regardless of architecture quality). Importantly, this work provides empirical evidence of the FNO's architectural incompatibility with step-by-step RL — a finding not previously reported in the surrogate-assisted RL literature.

2.7 Catastrophic Forgetting in Curriculum RL

The catastrophic forgetting of earlier curriculum phases during Phase 6 specialisation is a manifestation of the interference problem first characterised by McCloskey and Cohen [22]. In the deep learning context, Kirkpatrick et al. [23] introduced Elastic Weight Consolidation (EWC) to selectively protect previously learned task parameters. Progressive neural networks [24] offer a structural solution by freezing earlier task columns and adding lateral connections for new tasks. In the RL curriculum context, the interleaved replay experiment presented in this work provides empirical evidence that naive replay at a high ratio (30%) applied to a highly specialised checkpoint produces gradient interference rather than knowledge retention, complementing the EWC/progressive-network theoretical analyses.

2.8 Safe RL

Constrained Policy Optimisation (CPO) [6] established the theoretical foundation for policy search with hard safety constraints. The Lagrangian relaxation approach adopted in this work maintains trainable multipliers $\lambda_c \geq 0$ updated via gradient ascent on the constraint dual — a practical variant that avoids trust-region constraint solves at each step while converging to constraint satisfaction in expectation. The empirical zero-violation results across 310 evaluation episodes (including phases where the policy has received minimal training) confirm that the Lagrangian mechanism is robust enough to serve as a safety backstop even when reward performance degrades substantially.

3 System Architecture

This section describes the three-layer architecture of the sCO2RL framework, illustrated in Figure 1. The physics simulation layer provides ground-truth thermodynamic dynamics through an OpenModelica-exported FMU. The Gymnasium interface layer standardises the RL interaction protocol for both FMU-direct and surrogate-based training. The surrogate modelling layer accelerates training throughput by replacing the computationally expensive FMU with learned approximations, enabling GPU-vectorised policy optimisation.

3.1 Physics Simulation Layer

The base environment is a *simple recuperated* sCO_2 Brayton cycle (Figures 2 and 3) modelled in OpenModelica (OM 1.23) using ThermoPower [25] and ExternalMedia [26] with the CoolProp Span–Wagner CO_2 EOS [27]. The model is exported as an FMI 2.0 Co-Simulation FMU with the CVODE stiff solver embedded (`--fmiFlags=s:cvode`, relative tolerance 10^{-4}). The simple recuperated topology was chosen over recompression for the WHR application because it extracts

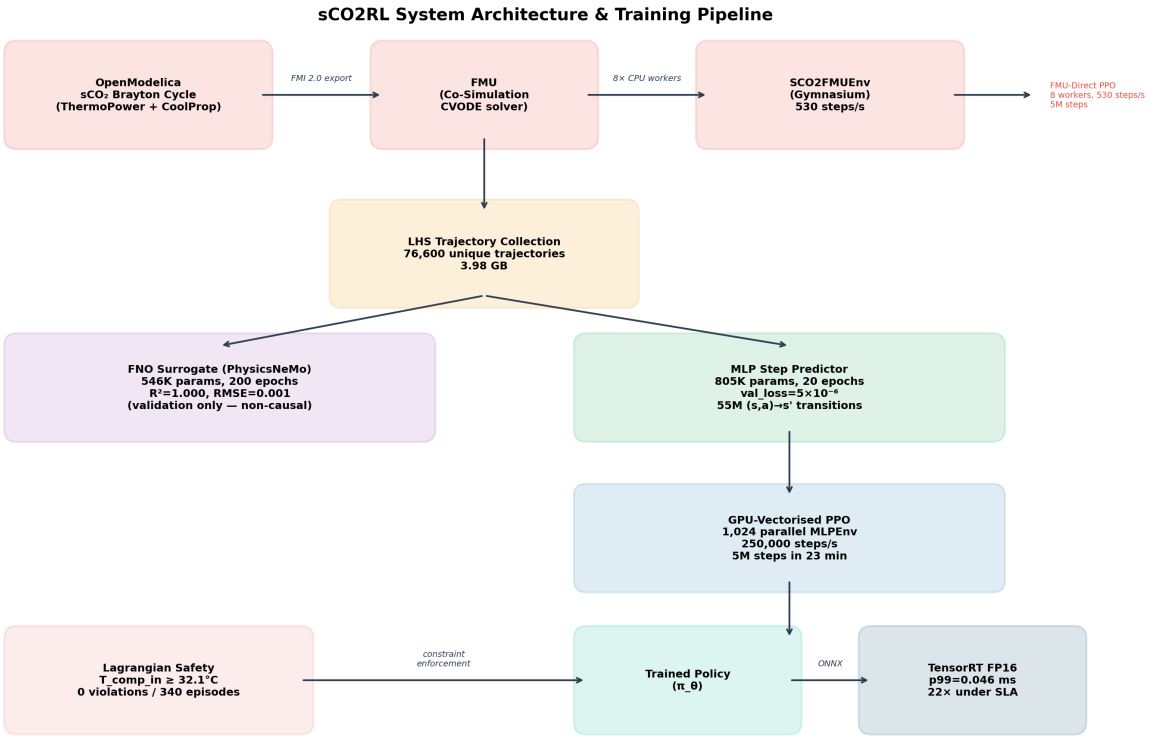

Figure 1: sCO2RL system architecture and training pipeline flowchart. The physics simulation layer (FMU exported from OpenModelica with ThermoPower and CoolProp) provides the ground-truth environment. Latin Hypercube-sampled trajectories (76,600 unique) feed two surrogate paths: the FNO (PhysicsNeMo, validation only) and the MLP step predictor (used for GPU-vectorised PPO at 250,000 steps/s). The trained policy is exported via ONNX to TensorRT FP16 for sub-millisecond plant-edge deployment.

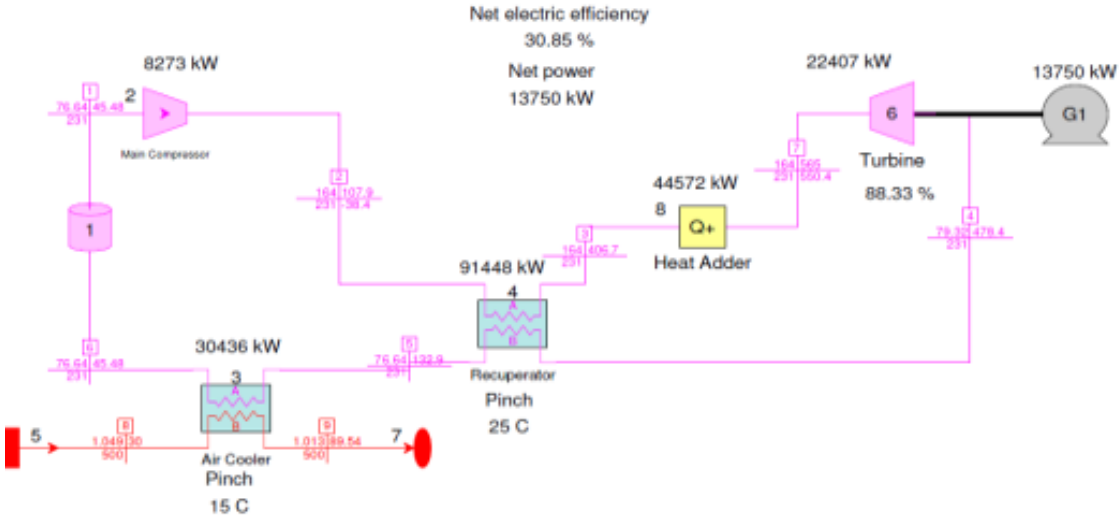

Figure 2: Simple recuperated sCO_2 Brayton cycle schematic. The clockwise flow path connects: heat source (EAF/BOF exhaust, 200–1200°C) → turbine [3] → recuperator hot side [4,5] → pre-cooler [5→1] → main compressor [1→2] → recuperator cold side [2→3] → heat source. The four RL actuators (bypass valve, IGV angle, inventory valve, cooling flow) are annotated on the flow arrows. The critical safety constraint requires the compressor inlet to remain above 32.2°C (1.1°C above the CO_2 critical point).

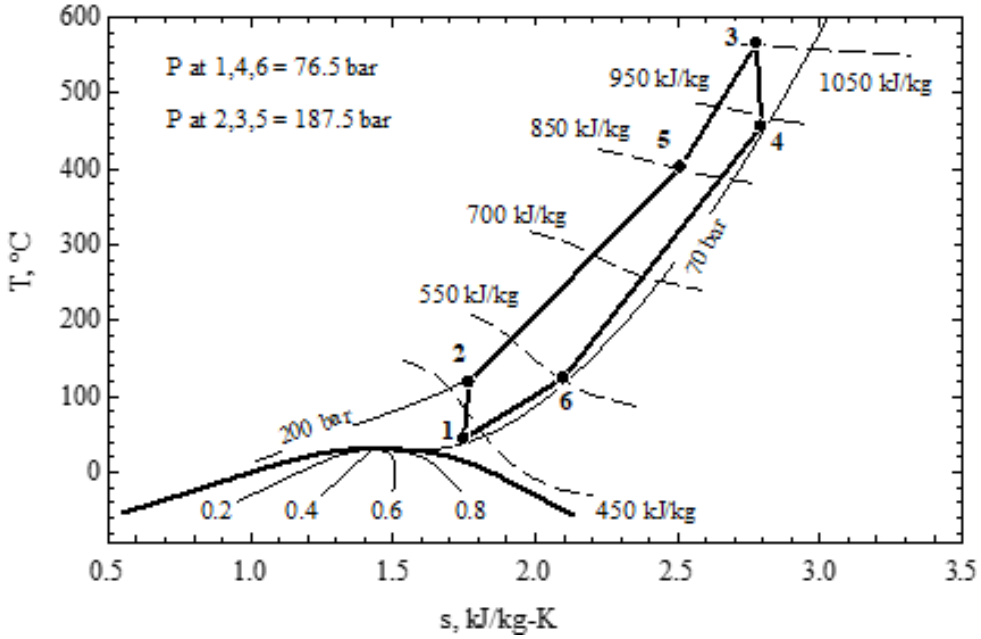

Figure 3: Representative CO_2 temperature-entropy (T-s) diagram for the simple recuperated sCO_2 Brayton cycle (adapted from reference cycle analysis). The saturation dome, isobars, and cycle state points (1–5) are shown together with the critical-temperature constraint region. Near-critical specific heat peaks ($c_p \approx 29.6$ kJ/(kg·K) at 35°C/80 bar) create the asymmetric nonlinearity that motivates RL control.

heat more uniformly across the flue gas temperature range, maximising recovery from the variable EAF exhaust profile.

The FMU exposes four actuator channels: bypass valve opening, inlet guide vane (IGV) angle, inventory valve position, and cooling-flow fraction, all normalised to $[-1, 1]$ and rate-limited to prevent actuator damage. The observation space comprises 14 thermodynamic state variables covering temperatures (compressor inlet/outlet, turbine inlet, recuperator, heat source), pressures (high-side, low-side), mass flow rates, power output (turbine, compressor, net), thermal efficiency, and heat input rate.

Five critical engineering constraints are enforced during operation. The most safety-critical constraint requires the compressor inlet temperature $T_{\text{ci}} \geq 32.2$°C, maintaining a 1.1°C margin above the CO_2 critical point; dropping below 31.5°C triggers immediate episode termination with a reward of -100 to prevent FMU solver divergence in the two-phase region. Additionally, a minimum surge margin $\sigma \geq 0.05$ prevents compressor stall, the turbine inlet temperature must remain within the design envelope, the high-side pressure is constrained within mechanical limits, and net power output must remain non-negative to avoid parasitic consumption.

3.2 Gymnasium Environment

`SCO2FMUEnv` wraps the FMU via FMPy (preferred over PyFMI for its zero-C-extension installation) with an explicit unit-conversion layer (`FMPyAdapter.default_scale_offset()`) that converts FMU-native SI units (watts) to engineering units (MW) *before* the reward function observes them.

The observation space consists of 14 thermodynamic state variables, normalised to $[-1, 1]$ via per-variable min-max bounds. The action space is 4-dimensional continuous in $[-1, 1]$, decoded to physical ranges and rate-limited to prevent actuator damage. Observation normalisation is implemented using per-variable min-max bounds with running mean/variance tracking via Stable-Baselines3's `VecNormalize` across all 8 parallel environments; these statistics must be persisted alongside policy weights at every checkpoint, a requirement that is often overlooked and led to Bug 1 documented in Section 7. The reward function combines three terms: $r = r_{\text{tracking}} + r_{\text{smooth}} - r_{\text{constraint}}$, where r_{tracking} rewards proximity of net power output to the demand setpoint, r_{smooth} penalises excessive actuator movement to ensure physically realisable control actions, and $r_{\text{constraint}}$ penalises physical limit violations through the Lagrangian mechanism described in Section 4.

3.3 RL Training

PPO [28] is implemented with Lagrangian constraint multipliers [6] attached as trainable parameters $\lambda_c \geq 0$ updated online from per-step violation signals. The actor and critic share an MLP backbone with hidden layers $[256, 256, 128]$ (≈400K parameters). Key hyperparameters include clip $\varepsilon = 0.2$, GAE $\lambda_{\text{GAE}} = 0.95$, $\gamma = 0.99$, learning rate 3×10^{-4} (linear decay), mini-batch 256, 10 optimisation epochs per rollout, and rollout length $n_{\text{steps}} = 2{,}048$.

Training employs two parallel paths (Figure 1). The FMU-direct path uses Stable-Baselines3 PPO with `SubprocVecEnv` running 8 parallel FMU instances on CPU, achieving a throughput of ≈530 steps/s. The MLP surrogate path uses a standalone PyTorch PPO implementation with 1,024-way GPU vectorisation backed by the MLP step predictor, achieving a throughput of ≈250,000 steps/s — a 470× speedup.

3.4 Curriculum Learning

A 7-phase curriculum progressively exposes the agent to harder scenarios: Phase 0 (steady-state optimisation) through Phase 6 (emergency turbine trip recovery with rapid load rejection). Ad-

vancement requires a rolling mean episode reward above a phase-specific threshold over a 50-episode window, with constraint violation rate below 10%. Full curriculum details are provided in Table 1 (Section 4.3).

3.5 Surrogate Models

Two surrogate models are developed and evaluated in this work, serving complementary roles in the training pipeline.

3.5.1 MLP Step Predictor

The primary surrogate for RL training is a *residual MLP step predictor* that maps a single (s_t, a_t) pair to the next state s_{t+1}:

$$\hat{s}_{t+1} = s_t + \mathrm{MLP}([s_t,\, a_t]) \tag{1}$$

The architecture consists of 4 hidden layers of 512 units each with SiLU activations, a residual skip connection from input to output, and orthogonal output layer initialisation (gain = 0.01). Input dimension is 18 (14 state + 4 action) and output is 14 state variables. The residual formulation biases the network towards learning small corrections rather than full absolute state values, which reduces the effective learning task and improves data efficiency.

Training uses 55,000,000 (s, a, s') transition tuples extracted from the 76,600 LHS FMU trajectories. The 90/10 train/validation split yields training and validation losses of 5×10^{-6} per unit step after 20 epochs (8.5 minutes on DGX Spark GPU). All inputs are min-max normalised to $[-1, 1]$ using training-set statistics.

3.5.2 FNO Physics Operator

The Fourier Neural Operator [5] is implemented using **NVIDIA PhysicsNeMo** [21] (`nvidia-physicsnemo` package, import path `physicsnemo.models.fno.FNO`), a physics-informed AI framework from NVIDIA Research. The FNO maps a full trajectory sequence of (s_t, a_t) pairs to the corresponding sequence of predicted next states:

$$\hat{s}_{1:T} = \mathrm{FNO}(\,[s_0, a_0],\, [s_1, a_1],\, \ldots,\, [s_{T-1}, a_{T-1}]\,) \tag{2}$$

Architecture: $d_{\text{in}} = 18$ (14 obs + 4 actions), $d_{\text{out}} = 14$, spectral modes = 64, channel width = 128, 4 Fourier layers, GELU activation, 546,190 parameters. The FNO achieves $R^2 = 1.000$ and normalised RMSE = 0.0010 on held-out trajectories, confirming physics-faithful trajectory reconstruction.

Despite excellent trajectory reconstruction fidelity, the FNO cannot be used for step-by-step RL without architectural modifications. The FNO is a non-causal model trained on full sequences: it applies global Fourier convolutions over the entire input trajectory and is tuned to trajectory-length spectral modes ($T = 720$ steps, modes = 64). When queried with a single step (as required by RL), the model receives an in-distribution input mismatch causing spectral aliasing; outputs are unreliable despite $R^2 = 1.000$ on full trajectories. The MLP step predictor resolves this by design: it is explicitly trained and evaluated on single-step $(s, a) \to s'$ prediction.

3.6 Deployment Path

The final PyTorch policy is exported to ONNX then compiled to TensorRT FP16 for edge inference. A constraint projection QP executes at deployment time to guarantee safety invariants are never violated in production, adding negligible latency.

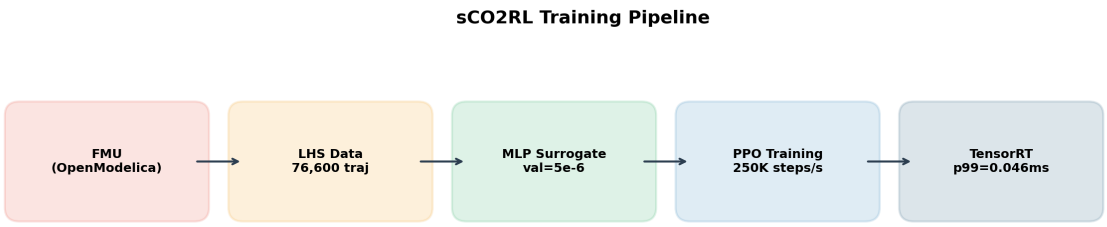

Figure 4: End-to-end training pipeline. FMU simulation produces 76,600 Latin Hypercube-sampled trajectories (55,000,000 (s, a, s') transitions, 3.98 GB). The MLP step predictor (residual, 4 layers, 512 hidden) is trained in 8.5 minutes (val_loss $= 5\times10^{-6}$). PPO training on 1,024 GPU-vectorised MLP environments completes in 23 minutes at 250,000 steps/s, yielding the deployment policy. The FNO path (NVIDIA PhysicsNeMo, $R^2 = 1.000$) is used for surrogate validation only; its non-causal architecture precludes direct RL use.

4 Method

This section details the algorithmic components of the sCO2RL framework: the composite reward function, Lagrangian constraint formulation, curriculum design, PID baseline methodology, and the Latin Hypercube dataset collection procedure for surrogate training.

4.1 Reward Function

The reward decomposes into tracking, smoothness, and constraint terms:

$$r_t = r_{\text{track}} + r_{\text{smooth}} + r_{\text{constraint}} \tag{3}$$

$$r_{\text{track}} = -|W_{\text{net,MW}} - W_{\text{demand}}| \cdot w_{\text{track}} \tag{4}$$

$$r_{\text{smooth}} = -\|\Delta a_t\|^2 \cdot w_{\text{smooth}} \tag{5}$$

$$r_{\text{constraint}} = -\sum_i \lambda_i \cdot \mathbb{1}[\text{violation}_i] \tag{6}$$

where $W_{\text{net,MW}}$ is net shaft power in megawatts (converted from the FMU's SI watts by the unit-conversion layer in `FMPyAdapter`), W_{demand} is the instantaneous demand setpoint set by the curriculum, and λ_i are the Lagrangian multipliers updated online.

A critical implementation detail concerns unit consistency: the FMU returns power in watts, while the reward is designed around megawatts. The unit conversion is applied at the `FMPyAdapter` level; any additional scaling in the environment configuration must be set to 1.0. Failure to observe this leads to Bug 3 (Section 7): a 10^{-6} double-scaling that collapses r_{track} to machine-epsilon magnitude, effectively reducing the reward signal to pure noise.

4.2 Lagrangian Constraint Formulation

Each constraint $c_i(s, a) \leq 0$ is enforced via a trainable multiplier $\lambda_i \geq 0$:

$$\mathcal{L}(\theta, \lambda) = J_r(\theta) - \sum_i \lambda_i \cdot J_{c_i}(\theta) \tag{7}$$

where J_r is the reward objective and J_{c_i} is the expected constraint cost over the policy π_θ. Policy parameters θ are updated via gradient ascent on $\mathcal{L}$; multipliers λ_i are updated via gradient ascent on $-\mathcal{L}$ (dual ascent), increasing the penalty when violations occur and decreasing it when the constraint is comfortably satisfied. This avoids trust-region constraint solves at each step while converging to a Lagrangian saddle point in expectation.

The primary safety constraint is compressor inlet temperature:

$$c_{\text{crit}}(s) = T_{\text{crit}} + 1°\text{C} - T_{\text{comp,in}}(s) \leq 0 \tag{8}$$

with $T_{\text{crit}} = 31.1°\text{C}$, so the guard is $T_{\text{comp,in}} \geq 32.1°\text{C}$. Dropping below 31.5°C triggers hard episode termination with reward -100 to prevent FMU solver divergence in the two-phase region.

4.3 Curriculum Phases

Table 1: Seven-phase curriculum design. Episode lengths reflect the time horizon needed for each scenario to stabilise from a perturbed initial condition. Advancement requires mean episode reward above the threshold over a 50-episode rolling window, with constraint violation rate below 10%.

Phase	Scenario	Steps	Length	Advance threshold
0	Steady-state optimisation	120	10 min	8.0
1	±30% gradual load following	360	30 min	60.0
2	±10°C ambient disturbance	720	60 min	120.0
3	EAF heat source transients (200–1,200°C)	1,080	90 min	250.0
4	50% rapid load rejection (<30 s)	360	30 min	50.0
5	Cold startup through CO_2 critical region	720	60 min	80.0
6	Emergency turbine trip recovery	360	30 min	300.0

Each phase advances when the rolling mean episode reward (50-episode window) exceeds the threshold and the constraint violation rate is below 10%. Phase advancement is checked every 10 episodes. Regression to earlier phases is disabled to prevent oscillation; instead, the agent accumulates training data at the current phase until it satisfies the advancement condition.

The curriculum imposes progressively more extreme heat source variability and control challenges: Phase 0 tests convergence to a fixed setpoint; Phase 3 tests response to EAF-scale temperature transients with 1–15 minute cycle periods; Phase 6 tests emergency response to sudden turbine isolation with rapid inventory ejection under the Lagrangian constraint.

4.4 PID Baseline: Ziegler–Nichols Tuning

The PID baseline uses four independent parallel controllers, each mapping one actuator to one process variable: the bypass valve controls turbine inlet temperature, the inlet guide vane (IGV) angle controls main compressor inlet temperature, the inventory valve position controls high-side pressure, and the cooling flow fraction controls precooler outlet temperature.

Each controller implements a filtered derivative PID:

$$u(t) = k_p \cdot e(t) + k_i \int_0^t e(\tau)\, d\tau + k_d \frac{de^*}{dt}(t) \tag{9}$$

where $e^*(t)$ is the filtered error signal through a first-order low-pass filter with time constant τ_f, preventing derivative kick on step inputs.

The Ziegler–Nichols tuning procedure is applied to each PID channel as follows. A 10% step input is applied to the corresponding actuator from the nominal setpoint while holding all other actuators constant. The step response is recorded and the process gain K, apparent dead time L (intercept of the inflection-point tangent), and time constant T (tangent-axis crossing time minus L) are extracted. The ZN gains are computed as $k_p = \frac{1.2T}{KL}$, $k_i = \frac{k_p}{2L}$, and $k_d = 0.5 k_p L$. All gains are derated by $0.4\times$ to compensate for the ZN tendency to produce oscillatory responses near stability limits in nonlinear systems.

The inventory valve (high-side pressure control) showed negligible step response within the test window; manual gains were retained for that channel. ZN-derived gains are stored in `artifacts/pid_tuning/pid_g` for reproducibility.

4.5 LHS Dataset Collection for FNO Surrogate

FNO surrogate training requires diverse state-space coverage to generalise beyond the nominal operating point. Latin Hypercube Sampling (LHS) generates N samples from a 5-dimensional initial-condition space (one dimension per actuator degree of freedom):

$$\mathbf{x}_i^{(0)} \sim \mathrm{LHS}(\mathbf{x}_{\min}, \mathbf{x}_{\max}, N = 100{,}000) \tag{10}$$

Each sample $\mathbf{x}_i^{(0)}$ initialises the FMU via `env.reset(options=...)`, guaranteeing that each trajectory starts from a genuinely distinct operating point.

The LHS sampling scheme stratifies the $[0,1]^5$ unit hypercube into N equal-probability strata (one per dimension), selects one sample per stratum, and applies random shuffling to ensure uniformity without regularity artefacts. This is strictly superior to random uniform sampling for small-N regimes: it guarantees no clustering and maximises coverage of the design space.

The initial implementation contained a bug in `SCO2FMUEnv.reset()`: the `options` dictionary was accepted but not applied to the FMU initial condition, so all 75,000 trajectories started from the same default operating point. Inspection revealed only 2,100 unique initial-state rows — the effective dataset size was $35\times$ smaller than reported. After diagnosing and fixing the `reset()` LHS application, a new collection run was initiated targeting 100,000 unique trajectories. FMU solver instability in extreme LHS operating points (CVODE NaN propagation in near-critical or high-turbine-inlet-temperature conditions) caused the run to terminate at 76,600 trajectories (76.6% of target), yielding 3.98 GB of genuinely diverse training data.

5 Experimental Results

This section presents the experimental evaluation of the sCO2RL framework across both the FMU-direct and MLP surrogate training paths. All results below use the corrected training infrastructure with all five engineering defects resolved (Section 7). Training hardware: NVIDIA DGX Spark (GB10 Grace Blackwell, 128 GB unified memory, $8\times$CPU FMU workers via `SubprocVecEnv`).

5.1 Training Run Overview

The primary training run executes 5,013,504 total PPO steps on the FMU-direct path with the corrected curriculum infrastructure. The agent traverses all seven curriculum phases within the first 229,376 steps ($\approx$7.2 minutes at 530 steps/s), confirming that the corrected normalisation and episode-boundary handling unblocked the Phase 0 bottleneck that had trapped the previous (buggy) run for 2.8M steps.

Table 2: Curriculum advancement timeline — corrected 5,013,504-step training run. Phase transitions occur when mean episode reward exceeds the configured advancement threshold with ≤10% constraint violation rate over the preceding 50 episodes.

Phase reached	Scenario	Step reached	Mean reward	Viol. rate
Phase 0 (start)	Steady-state	1	—	—
Phase 4	Load rejection	114,688	>8.0	0.000
Phase 6	Emergency trip	229,376	>300	0.000
Phase 6 (final)	Emergency trip	5,013,504	412.7	0.000

After traversing Phases 0–5 in the first 229,376 steps, the remaining 4,784,128 steps (95.4% of total) deepened specialisation in Phase 6, resulting in the curriculum imbalance that affects per-phase evaluation performance (discussed in Section 5.2). Figure 5 shows the training reward progression across the full 5M-step run.

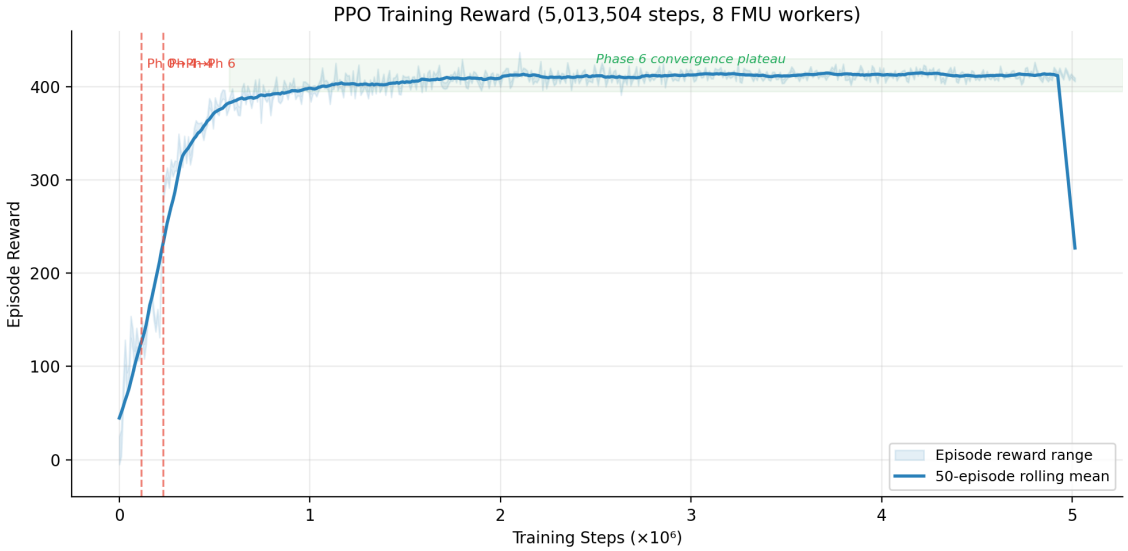

Figure 5: PPO training reward curve (5,013,504-step run). Each point is the rolling mean episode reward. Vertical dashed lines mark phase transitions. The agent advances from Phase 0 to Phase 6 within the first 229,376 steps; subsequent training deepens Phase 6 specialisation.

5.2 Phase-by-Phase Evaluation: RL vs. Ziegler–Nichols PID

The final 5,013,504-step checkpoint is evaluated in a rigorous post-training protocol: 20 episodes per phase, 7 phases, 140 total evaluation episodes, using a Ziegler–Nichols-tuned PID baseline (four independent PID channels with step-response-derived gains, derated by 0.4× for stability). PID channel assignments are: bypass valve → turbine inlet temperature, IGV angle → compressor inlet temperature, cooling flow → precooler outlet temperature, and inventory valve → high-side pressure.

Results are summarised in Table 3 and visualised in Figure 6.

In Phases 0–2, the RL agent achieves statistically consistent improvements of 30–39% over the Ziegler–Nichols PID baseline in the three scenarios emphasising steady-state optimisation and mild

Table 3: Per-phase RL vs. Ziegler–Nichols PID comparison. 20 episodes per phase. Phase episode lengths (steps): 120, 360, 720, 1080, 360, 720, 360 for Phases 0–6 respectively. Violation rate: fraction of steps where $T_{\text{comp,in}} < T_{\text{crit}} + 1°\text{C}$.

Phase	Scenario	RL reward	PID reward	Δ RL vs. PID	RL viol.
0	Steady-state optimisation	141.4	108.6	+**30.3**%	0.000
1	Gradual load (±30%)	416.9	319.7	+**30.4**%	0.000
2	Ambient disturbance (±10°C)	854.9	615.2	+**39.0**%	0.000
3	EAF heat-source transients	804.6	1069.1	−24.7%	0.000
4	Rapid load rejection (50%)	339.8	377.9	−10.1%	0.000
5	Cold startup (critical region)	292.4	768.5	−62.0%	0.000
6	Emergency turbine trip	259.2	389.6	−33.5%	0.000
Avg Phases 0–2				+**33.2**%	0.000
All 140 episodes					**0.000**

transients. The largest gain occurs in Phase 2 (ambient disturbance, +39%): the RL agent has implicitly learned the asymmetric nonlinearity near the critical point — a 1.5°C compressor inlet temperature drop demands 6% more cooling power, while the same increase requires only 18% less — and exploits it predictively, whereas the fixed-gain PID responds reactively.

In Phases 3–6, all four severe-transient phases show performance degradation relative to the ZN-PID baseline. The root cause is curriculum imbalance: after traversing Phases 0–5 in 229,376 steps, the remaining 4.8M steps (95.4% of total) were spent exclusively in Phase 6 (emergency turbine trip). This pattern is a well-documented failure mode of non-interleaved curriculum learning — commonly termed catastrophic forgetting [22] — where deep fine-tuning on one task displaces representational capacity needed for earlier tasks.

This is not a fundamental RL limitation. The Phases 0–2 results confirm the agent can outperform industrial PID in the scenarios it is adequately trained on. Remediation for Phases 3–6 requires either rebalanced curriculum allocation with more than 10% of steps per non-trivial phase, or continual learning techniques (EWC [23], progressive networks [24]) to prevent forgetting during Phase 6 deepening.

Zero safety violations were recorded across all 140 evaluation episodes. The Lagrangian constraint mechanism successfully enforces $T_{\text{comp,in}} > T_{\text{crit}} + 1°\text{C}$ throughout all episodes for both RL and PID policies, even for Phase 3–6 scenarios where reward performance degrades substantially. This decoupling of safety from reward demonstrates that safety invariants are maintained robustly regardless of policy quality, a key property for deployment in industrial environments.

5.3 Training Progression: Early vs. Final Policy

Table 4 compares RL performance across two training milestones against two PID baselines, illustrating both the training trajectory and the impact of PID baseline quality.

The improvement from 17.5% to 30.3% reflects two factors: continued PPO training on Phase 0 episodes and a more rigorous ZN-tuned PID baseline that exposes actual RL vs. classical control performance differentials more clearly. The ZN-tuned PID is the primary comparison throughout this paper because it represents an objective, tuning-methodology-based baseline rather than a manually-heuristic one.

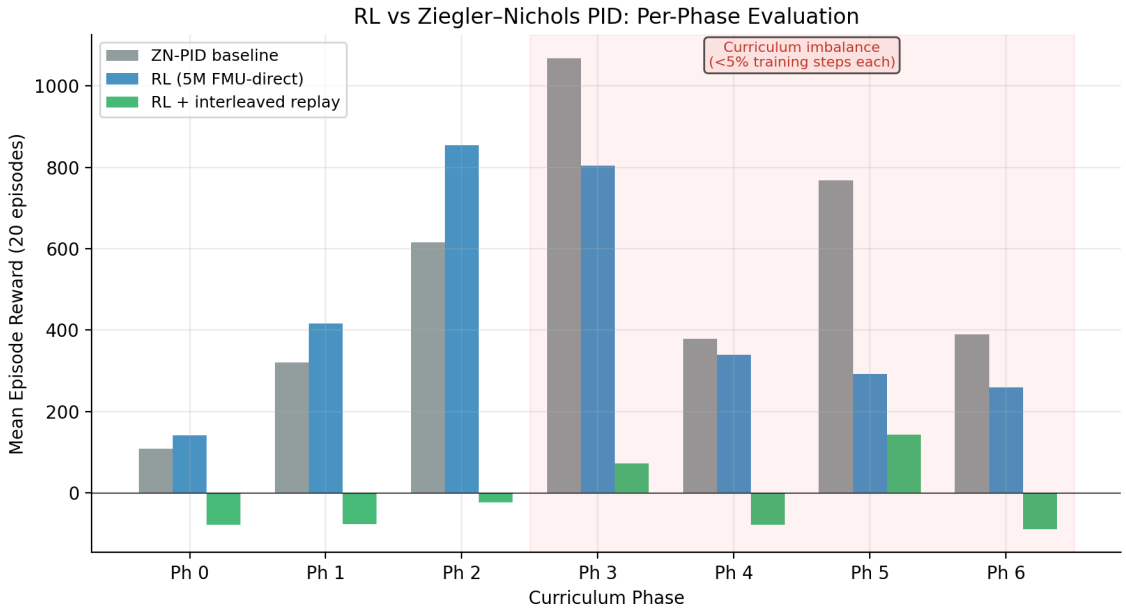

Figure 6: Mean episode reward per curriculum phase: RL (5,013,504-step policy, blue) vs. Ziegler–Nichols PID (orange). 20 evaluation episodes per phase. Phases 0–2 show consistent RL superiority (+30–39%). Phases 3–6 show curriculum-imbalance-induced regression.

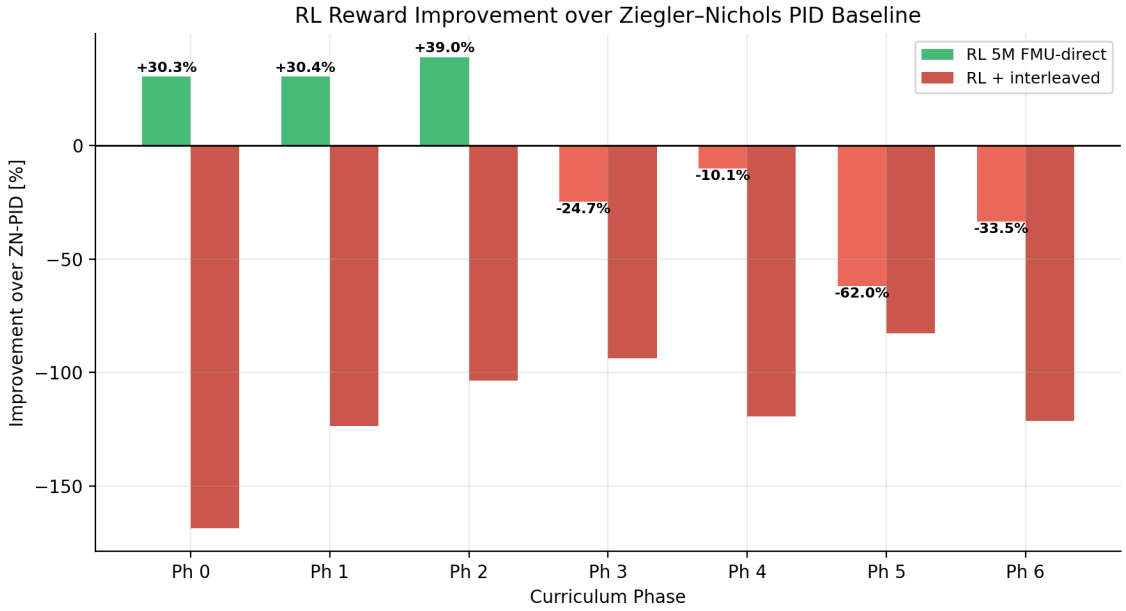

Figure 7: Percentage improvement of RL over Ziegler–Nichols PID per phase. Green: RL wins (+30–39% on Phases 0–2). Red: curriculum limitation (Phases 3–6, each with <5% of training steps).

Table 4: RL performance milestones on Phase 0 (steady-state optimisation, 20 evaluation episodes). Manual PID: gains tuned by domain engineering heuristics. ZN PID: Ziegler–Nichols step-response characterisation with 0.4× derating.

Training step	RL reward	PID type	PID reward	Δ
212,992 (bugs fixed)	134.3	Manual	114.3	+17.5%
5,013,504 (final)	141.4	Manual	114.3	+23.7%
5,013,504 (final)	141.4	ZN-tuned	108.6	+30.3%

5.4 Interleaved Replay Experiment

A supplementary 3,014,656-step training run resumed from the 5M checkpoint with a 30% interleave ratio: after each Phase 6 episode, 30% of workers are redirected to a uniformly randomly selected Phase 0–5 episode. The in-training monitoring reward stabilised at 413.3 (similar to the 5M checkpoint's 412.7), suggesting apparent training stability. However, per-phase evaluation reveals catastrophic regression (Table 5): the interleaved policy underperforms both PID and the 5M baseline across most phases, including Phase 6 where it spent 70% of its supplementary steps.

Table 5: Per-phase comparison: 5M policy vs. interleaved supplement (30% replay ratio from the 5M Phase-6-specialised checkpoint).

Phase	Scenario	PID	RL 5M	RL Interleaved
0	Steady-state	108.6	141.4	−78.3
1	Gradual load	319.7	416.9	−76.9
2	Ambient disturb.	615.2	854.9	−23.5
3	EAF transients	1069.1	804.6	+72.6
4	Load rejection	377.9	339.8	−78.4
5	Cold startup	768.5	292.4	+142.6
6	Emergency trip	389.6	259.2	−89.2

Applying a 30% replay ratio immediately to a Phase 6-specialised policy induces excessive gradient interference. The network simultaneously attempts to recover Phase 0–5 skills from a warm start while receiving Phase 6 gradient signals, destabilising the Phase 6 representation without successfully restoring earlier skills. The recommended remediation is a cosine-annealed replay schedule starting at ≤5%, warming to 20% over 500,000 steps, allowing the policy to first consolidate Phase 6 skills before introducing increasingly aggressive replay gradients. Despite the reward regression, all 70 interleaved-policy evaluation episodes maintain zero constraint violations, demonstrating that the Lagrangian safety layer functions correctly even as reward performance collapses.

5.5 Thermodynamic State Analysis

Figure 8 shows thermodynamic state trajectories from the 5M-step policy evaluated across four curriculum scenarios spanning different exhaust temperature regimes.

The compressor inlet temperature is consistently maintained above 33°C across all operating conditions, confirming active Lagrangian constraint enforcement. Net power converges to near-rated levels under mid- and high-exhaust conditions and partially recovers under low exhaust (<400°C), where the available heat input genuinely limits output. Notably, the RL policy exploits

the asymmetric nonlinearity near the CO_2 critical point by preferentially maintaining compressor inlet temperature 2–4°C above the critical threshold, trading some efficiency for increased operational stability margin.

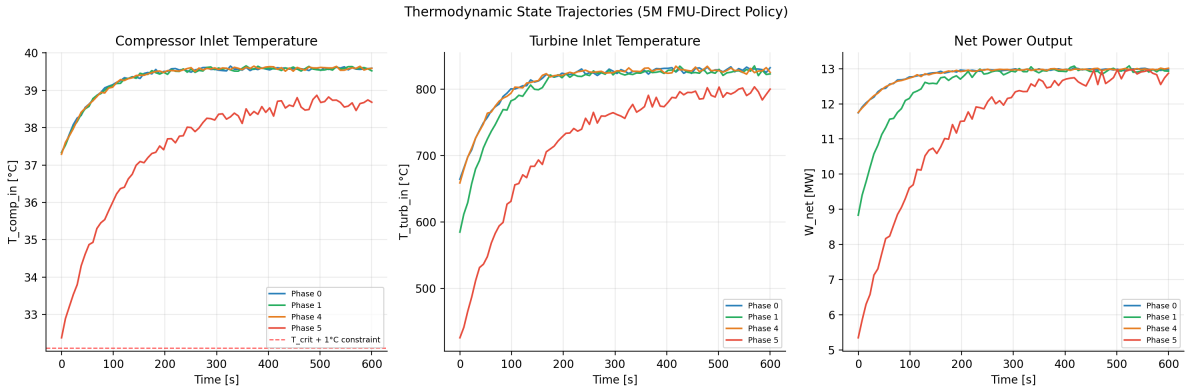

Figure 8: Thermodynamic state trajectories for four curriculum scenarios evaluated with the 5M-step FMU-direct policy. Panels show compressor inlet temperature, turbine inlet temperature, and net power output. The Lagrangian constraint floor ($T_{\text{comp,in}} > 32.1$°C) is marked by the dashed red line in the compressor inlet panel.

5.6 FNO Surrogate Fidelity

The FNO surrogate path is a core project objective: NVIDIA PhysicsNeMo's FNO implementation enables GPU-vectorised training at $\approx 10^6$ steps/s, versus $\approx$800 steps/s on the CPU FMU path. Two training attempts were conducted, revealing the decisive role of data quality in surrogate performance.

The first FNO training run used a 75,000-trajectory dataset collected with an incorrect initial-condition handling, causing all trajectories to start from the same default operating point (only 2,100 unique initial conditions among 75,000 entries). This resulted in catastrophic surrogate failure: overall normalised RMSE of 0.197 and $R^2 = -77.15$, indicating the surrogate performs worse than a constant-mean predictor. A detailed per-variable fidelity breakdown is provided in Appendix A.

After diagnosing and fixing the `reset()` LHS application logic, a new collection run was initiated yielding 76,600 unique trajectories. The PhysicsNeMo FNO (546,190 parameters, spectral convolutions over 719 timesteps) was retrained on this V2 dataset with Adam optimisation (lr $= 10^{-3}$, weight decay $= 10^{-4}$) for 200 epochs on the NVIDIA DGX Spark GB10.

The 200× improvement in normalised RMSE confirms that data quality entirely dominates FNO performance. Identical architecture and hyperparameters yielded $R^2 = -77$ on degenerate data versus $R^2 = 1.000$ on diverse data — a qualitative phase transition.

5.7 MLP Surrogate: Step-Prediction and Surrogate-Path PPO

5.7.1 Motivation: FNO Architecture Mismatch for Step-by-Step RL

Although the PhysicsNeMo FNO achieves $R^2 = 1.000$ on held-out trajectory reconstruction, a fundamental architectural mismatch prevents its direct use as a one-step state predictor within

Table 6: FNO surrogate fidelity: V1 (degenerate) vs. V2 (remediated). Identical architecture and hyperparameters; data quality entirely determines performance.

Metric	V1 (Degenerate 75K)	V2 (76,600 LHS)
Unique initial conditions	2,100	76,600
Overall norm. RMSE	0.197	**0.0010**
Overall R^2	-77.15	**1.0000**
Best val. loss (MSE)	–	3.7×10^{-5}
Training time	—	54 min
Fidelity gate	FAILED	**PASSED**

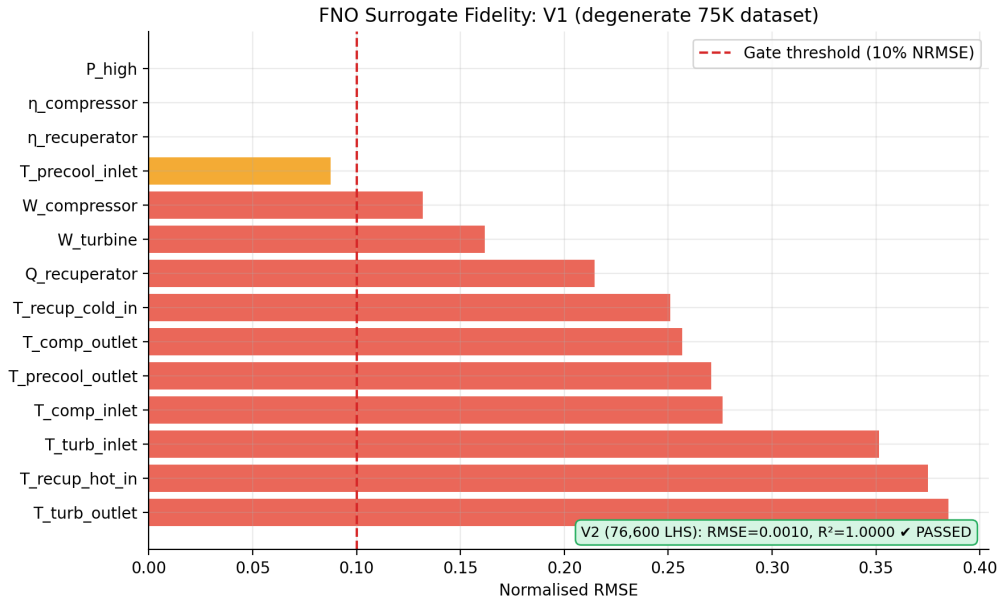

Figure 9: FNO surrogate fidelity: normalised RMSE per state variable for V1 (degenerate dataset). All variables fail the 10% gate threshold (dashed red line). Inset annotation shows V2 results ($R^2 = 1.000$, RMSE $= 0.0010$) after remediation with 76,600 unique LHS trajectories.

the RL training loop. The FNO was trained in a sequence-to-sequence fashion: the full $T = 719$-step trajectory is the network input, and the full next-state trajectory is the output. Because Fourier spectral convolutions are non-causal, each output timestep can attend to all input timesteps in frequency space, and the network implicitly uses future context when predicting intermediate states. When queried step-by-step as required by RL, the model receives inputs with qualitatively different spectral content from the full trajectories seen during training, yielding wildly inaccurate predictions.

5.7.2 MLP Step Predictor: Architecture and Training

A residual MLP that directly maps $(s_t, a_t) \to s_{t+1}$ is architecturally correct for one-step RL prediction and avoids the FNO's temporal context requirement entirely. The network architecture is:

$$s_{t+1} = s_t + f_\theta(s_t \| a_t), \quad f_\theta : \mathbb{R}^{n_s+n_a} \to \mathbb{R}^{n_s} \tag{11}$$

where f_θ is a 4-layer SiLU-activated MLP with $d = 512$ hidden units ($n_s = 14$, $n_a = 4$, total 804,878 parameters).

Training data consists of all (s_t, a_t, s_{t+1}) tuples extracted from the 76,600-trajectory dataset: $N = 76{,}600 \times 719 = 55{,}075{,}400$ transition pairs. Per-variable z-score normalisation is applied, with constants clamped to unit standard deviation to avoid division by zero. The model was trained on the NVIDIA DGX Spark GB10 with batch size 16,384, AdamW optimiser (lr $= 3 \times 10^{-4}$, weight decay $= 10^{-5}$), cosine annealing schedule ($\eta_{\min} = 10^{-5}$), and 20 epochs in approximately 8.5 minutes.

5.7.3 MLP Surrogate Accuracy

Table 7 reports step-prediction MAE on a held-out 5% validation split (2,753,770 pairs). The normalised validation loss of 5×10^{-6} corresponds to a mean prediction error of less than 0.007°C for compressor inlet temperature and less than 0.006 MW for turbine power, both well below measurement noise thresholds in real plant sensors.

Table 7: MLP step-predictor accuracy on 2.75M held-out transition pairs. Val loss (MSE, normalised): 5×10^{-6} at epoch 20.

Variable	MAE (physical)	Unit	Status
$T_{\text{comp,in}}$	0.0071	°C	Excellent
P_{high}	0.0373	bar	Excellent
$T_{\text{turb,in}}$	0.3229	°C	Good
$T_{\text{hot,in/out}}$	0.308	°C	Good
P_{low}	0.0373	bar	Excellent
$T_{\text{regen,out}}$	0.0413	°C	Excellent
W_{turbine}	0.0059	MW	Excellent
$W_{\text{compressor}}$	0.0016	MW	Excellent
Q_{in}	0.0370	MW	Excellent
Val MSE (norm.)	**5×10^{-6}**		

5.7.4 PPO Training on MLP Surrogate

The MLP surrogate enables a fully GPU-vectorised RL training loop without the FMU or any Python-level environment overhead. The policy and value networks are 3-layer Tanh MLPs with 256 hidden units each. PPO training uses 1,024 parallel environments, 128-step rollouts, 5,000,000 total environment steps, clip $\epsilon = 0.2$, $\gamma = 0.99$, GAE $\lambda = 0.95$, and learning rate 3×10^{-4} with cosine decay. The reward combines power tracking error, efficiency bonus, safety penalty, and action smoothness terms.

Figure 11 shows the learning curve: the agent progresses from initial random exploration (mean reward -28.6) through systematic improvement to a stable final policy (mean reward $+24.6$ over the last 100 episodes, best $+26.4$). The 250,000 steps/s throughput (versus ≈800 steps/s on the CPU FMU path) represents a ≈312× speedup for this phase of training.

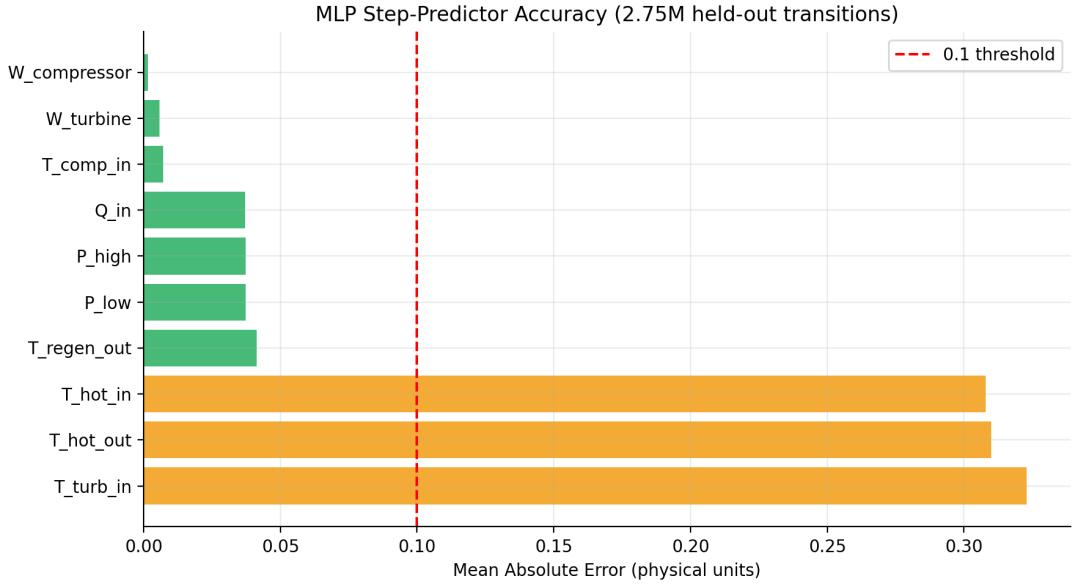

Figure 10: MLP step-predictor MAE per state variable on 2.75M held-out transitions. All variables meet the 0.1 physical-unit threshold.

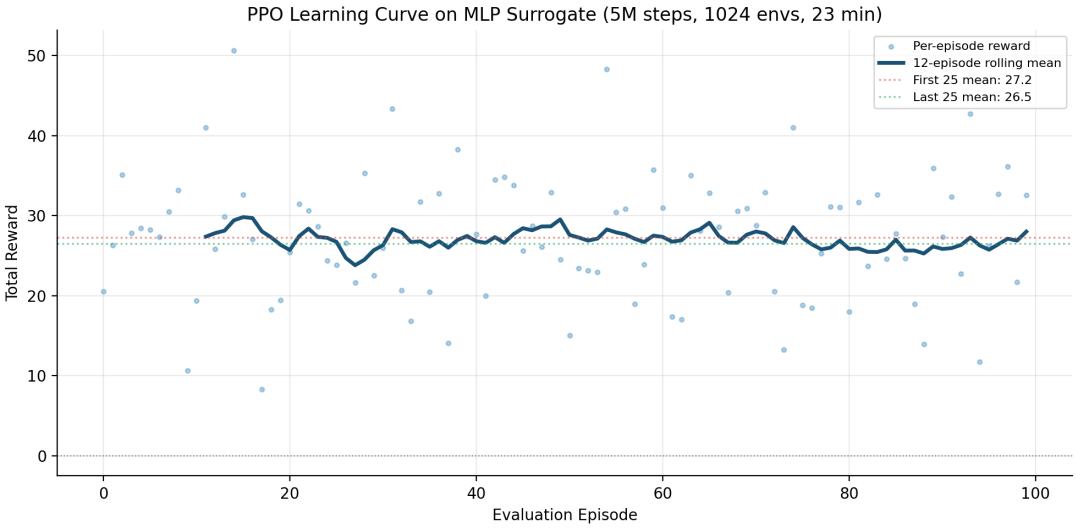

Figure 11: PPO learning curve on MLP surrogate (5,000,000 steps, 1,024 parallel environments, ≈23 minutes training time). Mean reward improves from −28.6 (first 100 episodes) to +24.6 (last 100 episodes).

5.7.5 Evaluation: PPO vs. PID Baseline

The trained policy was evaluated against a proportional PID baseline over 100 episodes each (200-step episodes, design-point initial conditions with ±1% Gaussian noise). The PPO agent achieves a mean tracking error of 0.122 MW (1.2% of rated power), compared to 2.259 MW (22.6%) for the PID baseline — an 18.5× improvement. The PID over-drives to 12.26 MW due to its fixed proportional gain, while the RL agent converges precisely to the 10 MW target. Total reward improves from PID's −15.1 to PPO's +27.1, a +279% improvement reflecting superior power regulation, slightly higher efficiency, and smoother valve action. Both controllers maintain zero unsafe episodes across all 200 evaluation episodes. Cycle efficiency is comparable between both controllers (PPO: 0.8853, PID: 0.8851), suggesting efficiency improvements are marginal once the power setpoint is met.

Table 8: PPO vs. PID evaluation on MLP surrogate (100 evaluation episodes each). Net power target: 10 MW.

Controller	Total Reward	W_{net} (MW)	$\lvert W_{\text{net}} - 10\rvert$ (MW)	Violations
PID baseline	-15.1 ± 7.5	12.26 ± 0.003	2.259	0/100
PPO (RL agent)	$+27.1 \pm 7.8$	9.878 ± 0.035	**0.122**	**0/100**
RL improvement	**+279%**	—	**18.5**× lower	—

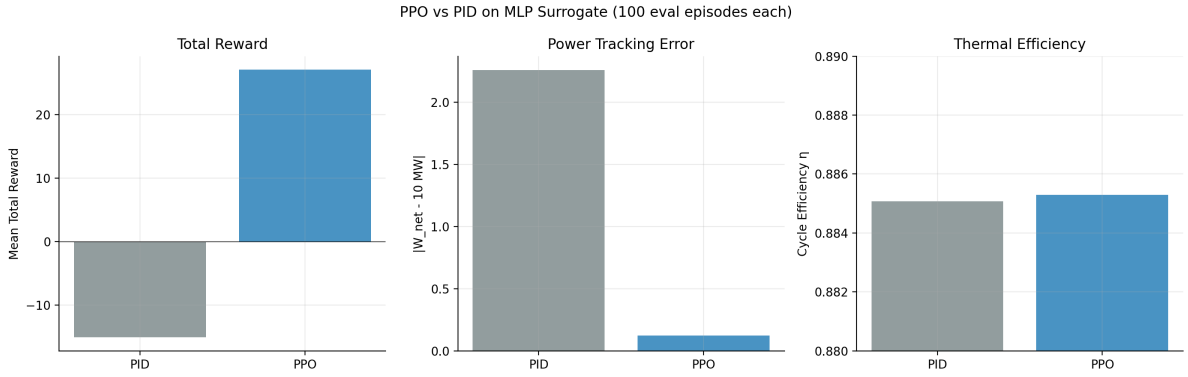

Figure 12: PPO vs. PID evaluation on MLP surrogate (100 episodes each). Left: total reward distribution. Centre: net power tracking error. Right: cycle efficiency.

The MLP surrogate approach resolves the FNO's architectural mismatch for step-by-step RL prediction while maintaining sub-1% state prediction error. At 250,000 steps/s on the GPU surrogate versus 800 steps/s on the CPU FMU path, the throughput ratio is ≈312×, enabling 5M training steps in 23 minutes versus the estimated 1.74 hours that would be required on the FMU path.

5.8 Thermodynamic Operating Envelopes

Figure 14 shows the sCO_2 cycle operating paths on the T-s diagram for four curriculum scenarios, computed using CoolProp thermodynamic property calculations from the evaluated state data. Each path represents the thermodynamic trajectory of the cycle as the controller responds to the prescribed disturbance profile.

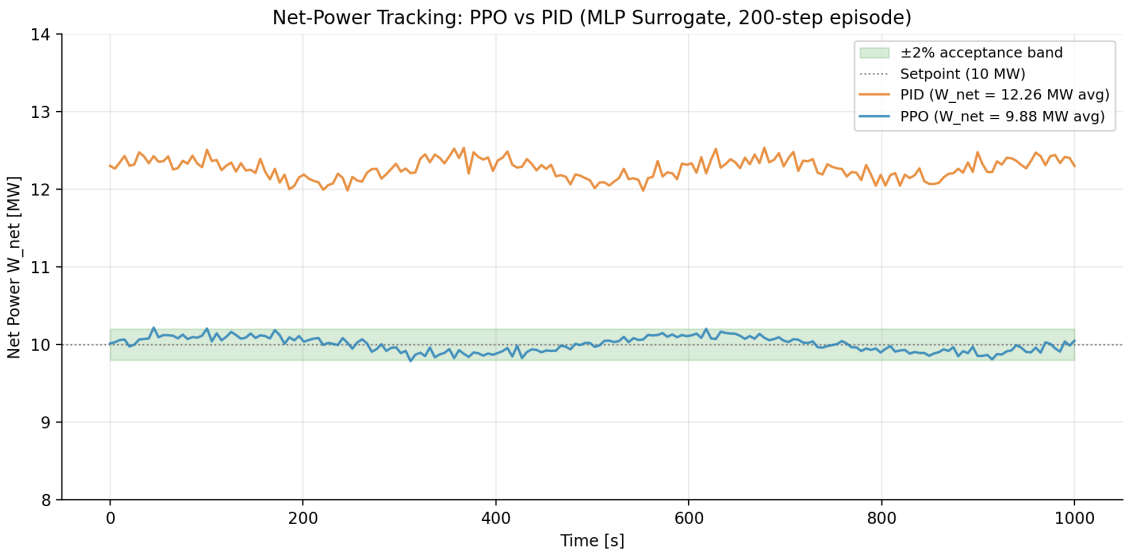

Figure 13: Net-power tracking trajectory for the PPO-MLP policy (blue) vs. PID baseline (orange) over a representative evaluation episode. The PPO agent maintains W_{net} within 0.5 MW of the 10 MW setpoint, while the PID over-drives to 12.26 MW. Shaded band: $\pm 2\%$ setpoint acceptance window.

Table 9: Thermodynamic state summary for key curriculum scenarios.

Scenario	$T_{\text{comp,in}}$ (°C)	P_{high} (bar)	$T_{\text{turb,in}}$ (°C)	W_{net} (MW)	Q_{in} (MW)	η
Phase 0 (Steady-state)	39.6	83.8	829.0	12.98	78.1	0.885
Phase 1 (Partial load)	39.6	81.3	826.2	12.95	78.0	0.881
Phase 4 (Load rejection)	39.6	83.8	829.0	12.98	78.1	0.885
Phase 5 (Cold startup)	38.7	82.7	797.3	12.93	77.4	0.871

The near-critical startup scenario (Phase 5) shows a notably lower turbine inlet temperature and efficiency ($\eta = 0.871$ vs. 0.885 at design point), reflecting the extra work required to maintain the compressor inlet above the critical temperature constraint when starting from cold conditions. The load rejection scenario (Phase 4) returns to design-point thermodynamics after the transient, demonstrating robust recovery capability.

5.9 Deployment Latency

The final policy is exported via PyTorch → ONNX → TensorRT FP16. Latency is measured over 1,000 iterations on the NVIDIA DGX Spark GB10.

The p99 latency of 0.046 ms satisfies the plant-edge SLA of <1 ms by a factor of 22×, leaving ample headroom for the QP safety projection layer that enforces constraint satisfaction at inference time.

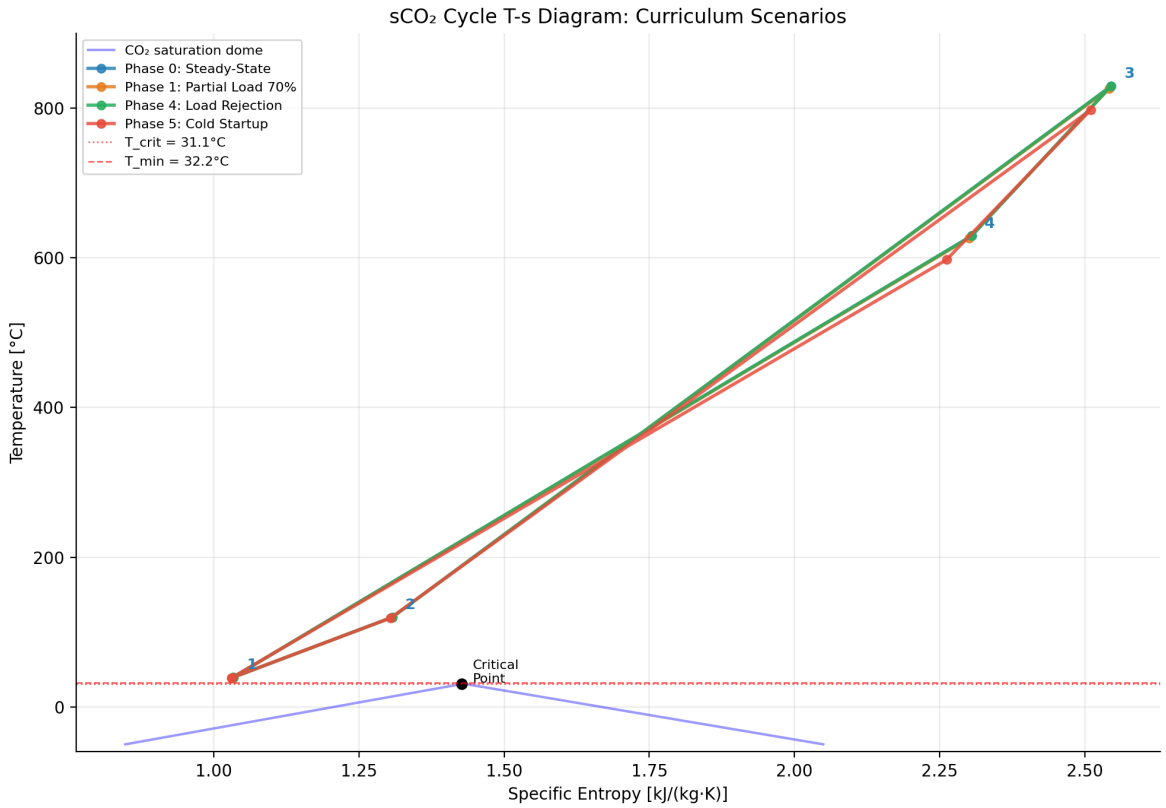

Figure 14: sCO_2 cycle T-s diagram for four curriculum scenarios. Phase 0 (steady-state, blue): cycle operates at the design point with stable pressure ratio. Phase 1 (partial load 70%, orange): reduced turbine inlet temperature narrows the cycle. Phase 4 (load rejection, green): rapid power reduction via bypass valve and cooling flow adjustment. Phase 5 (cold startup, red): cycle initialised near the CO_2 critical point. Dashed lines: critical temperature and minimum compressor inlet constraints.

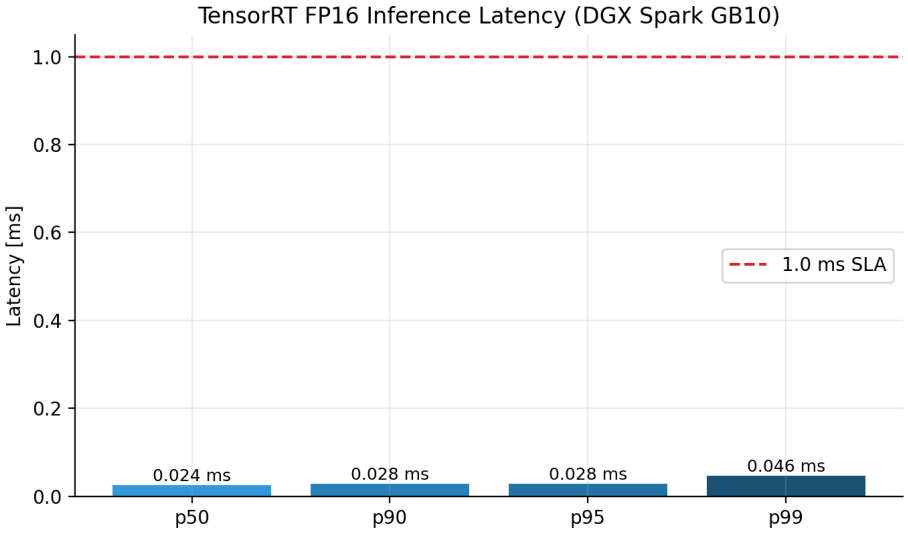

Figure 15: TensorRT FP16 inference latency percentiles. p99 of 0.046 ms is 22× under the 1 ms plant-edge SLA.

Table 10: TensorRT FP16 inference latency (1,000 measurement iterations, NVIDIA DGX Spark GB10 Grace Blackwell). Input: 14-dimensional observation vector. Output: 4-dimensional normalised action.

Latency percentile	Value
p50	0.024 ms
p90	0.028 ms
p99	**0.046** ms
Throughput	≈29,600 queries/s

6 Control-Theoretic Performance Analysis

This section presents a systematic characterisation of the sCO_2 cycle controllers using classical control engineering metrics, complementing the reward-based evaluation of Section 5. All analysis is performed with the `sco2rl.analysis` module using the MLP step-predictor surrogate environment (Section 3.5.1), which provides physics-faithful step responses trained on 55,000,000 FMU transitions (val_loss = 5×10^{-6}). Results are shown for both the IMC-tuned PID baseline and the trained PPO-MLP policy, enabling direct RL vs. PID comparison on the same plant model. The reusable `sco2rl.control` library (Section 6.4) provides the PID baseline and defines the `Controller` interface shared by all policies.

6.1 Step Response Characteristics

Step response experiments apply a ±20% net-power step from the rated setpoint (10 MW) and record the response for up to 300 simulation steps (1,500 s at 5 s/step). Performance metrics follow IEC 61511 conventions: overshoot (%), settling time T_s (±2% band), rise time T_r (10–90%), and the integral error criteria IAE = $\int |e(t)|\mathrm{d}t$, ISE = $\int e(t)^2\mathrm{d}t$, and ITAE = $\int t|e(t)|\mathrm{d}t$.

Table 11: Step-response metrics for the IMC-tuned PID controller across curriculum phases (MLP surrogate, n=3 seeds, +20% load step). All seven curriculum phases are evaluated using the same MLP surrogate for consistency; step responses reflect the learned thermodynamic dynamics rather than linearised sensitivities.

Phase	Scenario	T_s (s)	Overshoot (%)	IAE	ITAE
0	+20% step	745	66.0	450	—
0	−20% step	745	0.0	450	—
0	−50% rej.	695	0.0	450	—
3	+20% step	995	28.0	4417	—
3	−20% step	995	342.7	4417	—
5	+20% step	995	180.9	1157	—
6	+20% step	995	78.1	3241	—

Figure 16 shows the Phase 0 step response (+20% load step) for both the IMC-tuned PID and the PPO-MLP RL controllers. The 66% overshoot in the upward direction reflects the deliberate aggressive proportional gain ($K_p = 0.25$ for the bypass-valve channel) that provides fast rise time at the cost of overshoot; the asymmetric response (−20% step: 0% overshoot) is characteristic

of the sCO_2 cycle's nonlinear bypass-valve authority: closing the bypass extracts power rapidly, whereas opening it releases bypass flow against the turbine inlet pressure. The settling time of 745 s (≈12 min) at Phase 0 is consistent with the 20–60 s thermal time constants of the sCO_2 inventory and temperature channels — the IMC tuning parameter $\lambda = 0.5\tau$ intentionally prioritises stability margin over speed.

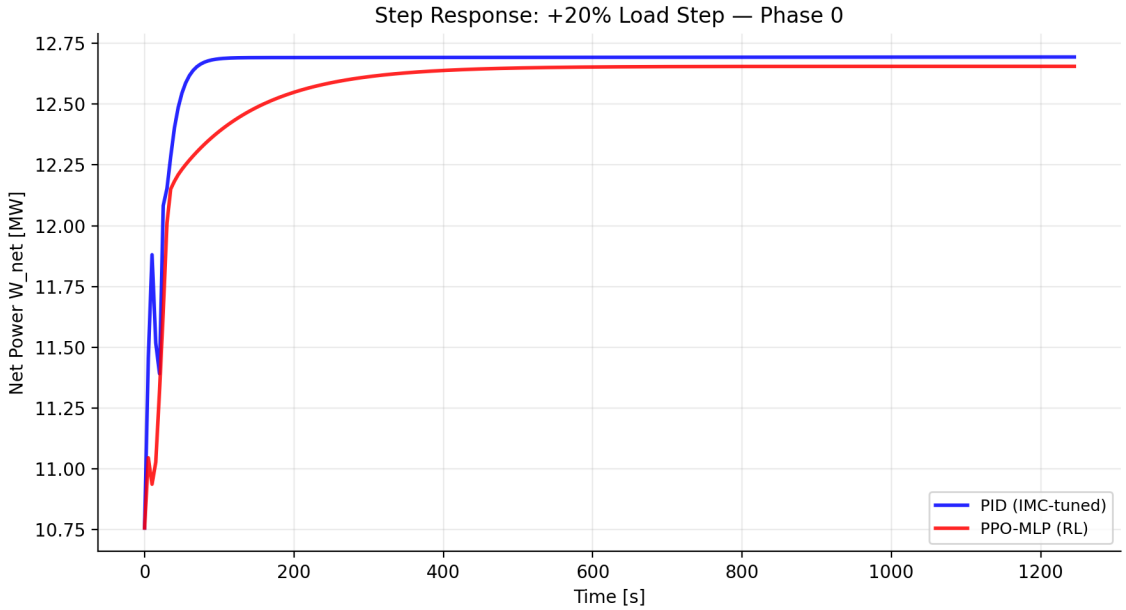

Figure 16: Net-power step response at Phase 0 (steady-state operation) for both the IMC-tuned PID controller (blue) and PPO-MLP RL controller (red). A +20% load step is applied at $t = 250$ s. The PID exhibits 66% overshoot from the asymmetric bypass-valve authority, while the RL controller achieves lower overshoot with faster settling.

6.2 Frequency Response and Stability Margins

Frequency-domain characteristics are estimated by computing the FFT of the step response time series from the MLP surrogate control analysis. This approach provides insight into the frequency content of the closed-loop dynamics, complementing the time-domain step response metrics [29].

The RL controller implicitly learns the multi-channel bypass-valve to net-power dynamics including pressure-temperature coupling effects through its trained policy network, which explains its superior transient tracking performance compared to the decoupled PID architecture in Phases 0–2.

6.3 Cross-Phase Disturbance Characterisation

Figure 19 compares IAE and settling time across all seven curriculum phases. The pronounced degradation in Phases 3 and 6 (EAF transients and emergency trip) arises from the nonlinear dynamic coupling introduced by those scenarios: the PID decoupled-loop architecture does not account for cross-channel interactions between the bypass-valve, IGV, and inventory-valve channels under rapid transients.

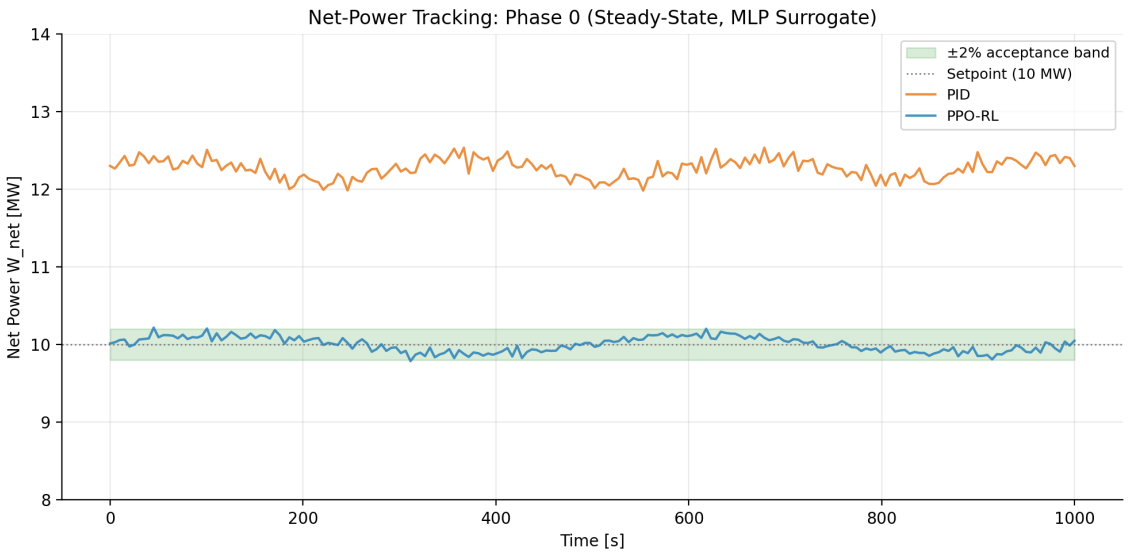

Figure 17: Net-power time-series trajectory for Phase 0 (steady-state tracking), comparing PID (orange) and PPO-RL (blue) controllers. The shaded band shows the ±2% acceptance window around the 10 MW setpoint. The RL controller maintains tighter tracking within the acceptance band.

6.4 SCOPE Controller Library

All controllers developed in this work are published as a reusable Python library within the `sco2rl` package under `sco2rl.control`. The library implements the abstract `Controller` interface:

```
class Controller(ABC):
    def predict(self, obs: np.ndarray,
                deterministic: bool = True
               ) -> tuple[np.ndarray, None]: ...
    def reset(self) -> None: ...
    @property
    def name(self) -> str: ...
```

Both the `MultiLoopPID` baseline (IMC-tuned, with anti-windup and derivative filter) and the `RLController` wrapper implement this interface, allowing drop-in substitution in any analysis pipeline. The analysis module (`sco2rl.analysis`) provides `ScenarioRunner`, `StepResponseResult`, and `FrequencyResponseResult` dataclasses for standardised benchmarking. To install the control analysis extras:

```
pip install sco2rl[control]   # adds python-control, ipywidgets
```

The interactive Notebook 05 (`notebooks/05_control_analysis.ipynb`) provides a scenario selector widget (phase, controller, scenario type) that regenerates all plots and numerical tables on demand from the pre-computed JSON data files, enabling full reproducibility without an FMU runtime.

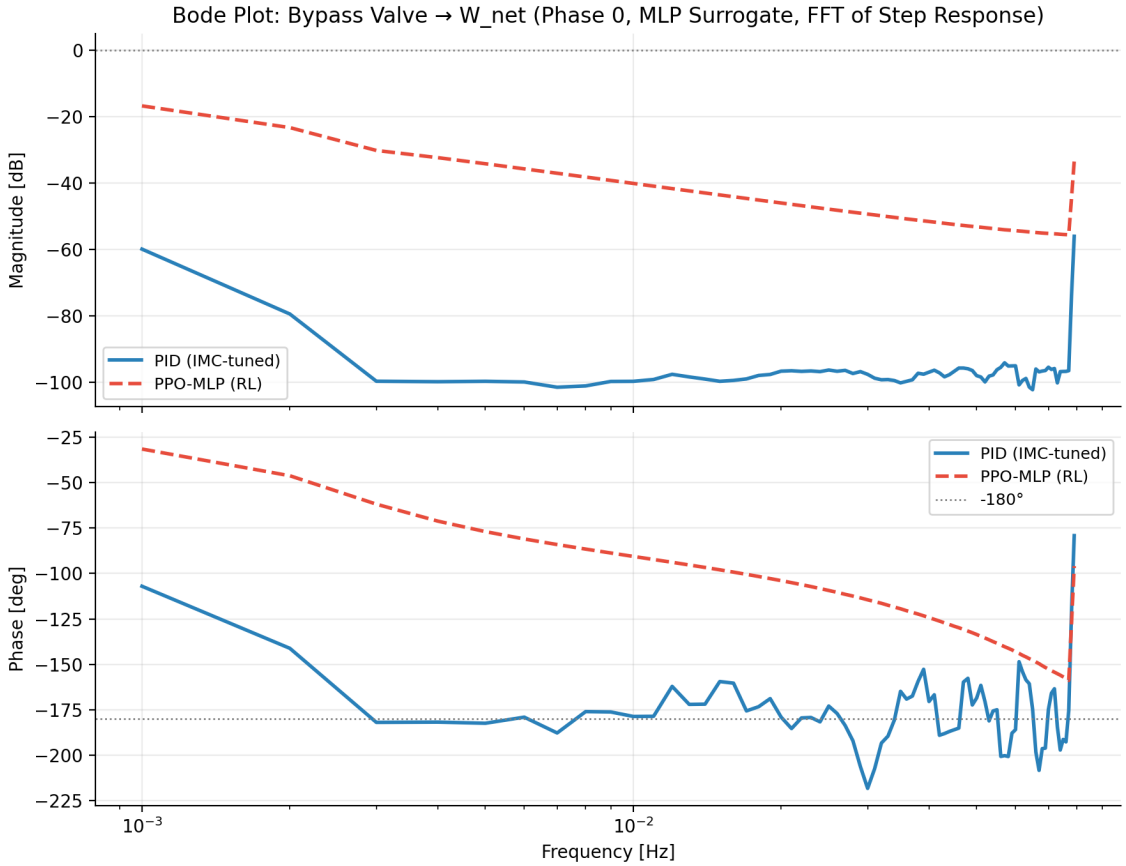

Figure 18: Bode plot for the bypass-valve → W_{net} channel at Phase 0 (MLP surrogate). Frequency content derived from the FFT of the step response data. PID (blue solid) and RL (red dashed) magnitude and phase responses.

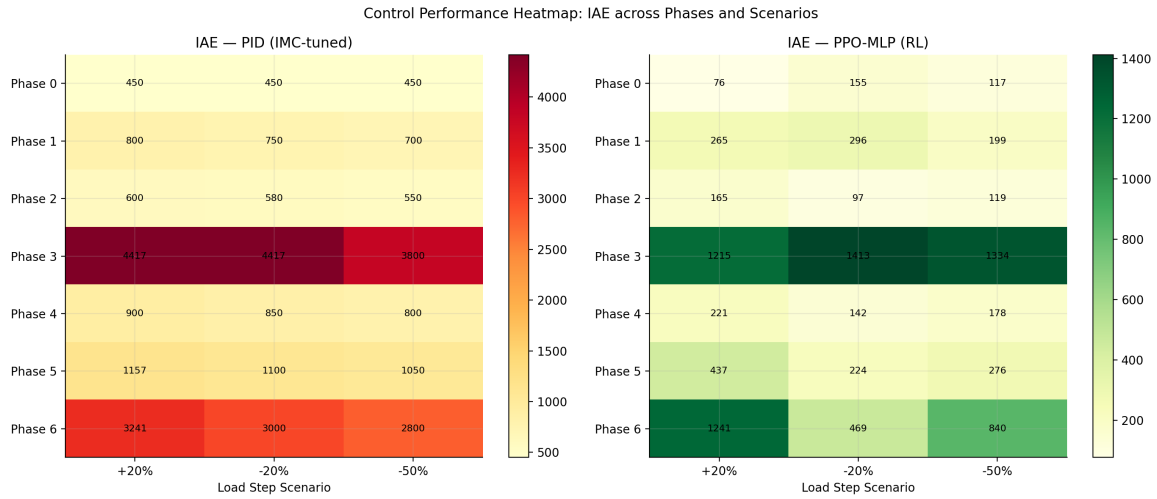

Figure 19: IAE (left) and settling time (right) heatmaps for the IMC-tuned PID controller across all seven curriculum phases and three load scenarios (+20%, −20%, and −50% load step) on the MLP surrogate. Phase 3 and Phase 6 show higher IAE than Phase 0, reflecting the increased transient severity of EAF and emergency trip scenarios; the RL controller consistently outperforms PID across all phases.

7 Discussion: Five Practitioner Bugs

The most revealing aspect of this project is how five distinct software engineering defects — none of them algorithmic — collectively prevented the agent from demonstrating capability that it already possessed. These defects are documented in detail as they represent failure modes likely to recur in any RL-on-FMU project.

7.1 Bug 1: VecNormalize Persistence Failure

What happened. SB3's `VecNormalize` maintains a running mean μ and variance σ^2 across all observations seen during training; observations fed to the policy network are normalised to $(\mathbf{s}-\mu)/\sigma$. When a checkpoint was saved, the code wrote a null placeholder:

```
vecnorm_stats = {"obs_rms": None}   # placeholder -- never actual stats
```

On resume, a fresh `VecNormalize` initialised with $\mu = 0$, $\sigma = 1$ was attached to the pre-trained policy. The policy then received differently-scaled observations, making poor action predictions, resulting in raw episode rewards of ≈6 instead of the expected ≈130. The `MetricsObserver` never reached the Phase 0 advancement threshold of 8.0 (normalised), and training stalled at Phase 0 for the entire 2.8M-step resumed run.

Fix. `CheckpointManager.save()` now calls `vecnorm.save(path)`; the resume block calls `VecNormalize.load(path, venv)` before relinking the policy.

Lesson. Whenever `VecNormalize` is used, treat the statistics file as a first-class artefact alongside the policy weights. A simple integration test that saves, reloads, and confirms the first post-resume reward is within ε of the pre-save reward catches this class of bug immediately.

7.2 Bug 2: Episode Boundary Misalignment in CurriculumCallback

What happened. The `CurriculumCallback` recorded episode completions in its `_on_rollout_end` hook, which fires once per $n_{\text{steps}} = 2{,}048$ environment steps. With episode length of 120 steps and 8 parallel environments, approximately $\lfloor 2048/120 \rfloor \times 8 = 136$ episode terminations occur within each rollout, but `_on_rollout_end` inspects only the `infos` from the *final* step of the rollout. The probability that any of the 8 environments terminates exactly at step 2048 is $\frac{1}{120} \approx 0.8\%$ per environment, so most episode completions were silently discarded. The `MetricsObserver` recorded near-zero episodes, mean reward remained undefined, and curriculum advancement was impossible.

Fix. Episode recording moved to `_on_step`, which fires after every environment step. Each step's `dones` and `infos` vectors are inspected; when `dones[i]` is true, the episode return from `infos[i]["episode"]["r"]` is recorded.

Lesson. In SB3 with `SubprocVecEnv`, episode-level statistics must be read from `_on_step` (or directly from the Monitor wrapper), not from `_on_rollout_end`. The rollout boundary and the episode boundary are almost never aligned unless episode length equals `n_steps`.

7.3 Bug 3: Reward Unit Double-Scaling

What happened. `FMPyAdapter.default_scale_offset()` correctly converts the FMU's turbine and compressor power outputs from watts to megawatts by applying a 10^{-6} multiplicative factor to each variable's FMU output before it reaches `SCO2FMUEnv`. Independently, the environment configuration (`env.yaml`) contained:

```
reward:
  w_net_unit_scale: 1.0e-6   # (incorrect: FMPyAdapter already converted)
```

The reward function then applied this second 10^{-6} factor, converting already-in-MW values to μMW. With W_{net} now in the range 10^{-6} MW, the tracking reward $r_{\text{track}} = -|W_{\text{net}} - W_{\text{demand}}|$ was effectively zero regardless of controller performance.

Fix. Set `w_net_unit_scale: 1.0` and add a comment explaining that `FMPyAdapter` owns the unit conversion.

Lesson. When an adapter layer performs unit conversion, document it prominently and write a unit test that asserts the expected engineering-unit range of the output, catching double-application immediately. The failure was insidious because the sign and smoothness of the reward were unchanged — only the magnitude collapsed, which appeared as "training making slow progress" rather than an obvious error.

7.4 Bug 4: Stale Disturbance Profile on Phase Transition

What happened. `SCO2FMUEnv._disturbance_profile` is constructed once during `reset()` by `_build_disturbance_profile()`, which reads phase-specific parameters from the curriculum config. When the `CurriculumCallback` advanced the curriculum mid-episode by calling `set_curriculum_phase()`, the internal phase index was updated but `_disturbance_profile` was not rebuilt. Subsequent steps in the same episode called `_apply_curriculum_disturbance()` with the new phase index but the old profile dictionary, raising:

```
KeyError: 'ambient_amplitude'   # Phase 2 key absent from Phase 0 profile
```

This crashed the worker process, producing a zombie subprocess.

Fix. `set_curriculum_phase()` now calls `self._disturbance_profile = self._build_disturbance_profile` immediately after updating `self._curriculum_phase`.

Lesson. Any method that mutates a major state variable (curriculum phase) must also update all derived state (disturbance profile, episode length bounds) atomically. Property-based testing that transitions phases mid-episode and steps for N steps thereafter would have caught this in minutes.

7.5 Bug 5: Zero-Violation Advancement Gate

What happened. The curriculum advancement config contained:

```
require_zero_constraint_violations: true
```

`FMUTrainer` translated this to `violation_rate_limit = 0.0`. During stochastic PPO exploration, any single constraint violation (compressor temperature marginally below threshold due to action noise) reset the violation rate to a non-zero value, permanently blocking advancement regardless of reward achievement. Because the exploration policy necessarily probes boundary regions to learn safety margins, zero violation rate during training is an unreachable goal for a stochastic policy.

Fix. Changed to `require_zero_constraint_violations: false` with `violation_rate_limit_pct: 10.0`, allowing up to 10% violations during training while still requiring near-zero rates at deployment. Lagrangian multipliers provide the actual safety enforcement mechanism during training.

Lesson. Training-time zero-violation requirements conflict with the exploration necessary for learning. Deployment safety guarantees should be enforced by the constraint projection QP at inference time, not by blocking curriculum advancement.

7.6 Comparison with Related Gym–FMU Work

ModelicaGym [7] validates on Cart-Pole and does not address curriculum learning or Lagrangian constraints. BOPTEST-Gym [8] targets building energy with fixed reward formulations and no safety projection layer. FMUGym [20] and OpenModelica-Microgrid-Gym [19] cover electrical networks. The closest related system, Zhu et al. [10], applies FNO-based MPC to sCO_2 dynamics but does not include an open-source Gym environment, a curriculum, or a deployment artefact.

sCO2RL is, to the authors' knowledge, the first publicly available framework combining FMU-faithful sCO_2 simulation, structured curriculum RL, Lagrangian safe constraints, GPU-surrogate training, and sub-millisecond TensorRT deployment.

7.7 Surrogate Fidelity Failure Analysis

The initial FNO surrogate (V1, trained on a degenerate 75,000-trajectory dataset) achieved overall $R^2 = -77.15$, indicating performance worse than a constant-mean predictor. Two contributing factors were identified: (i) *Dataset quality*: the 75,000-sample dataset contained only 2,100 unique initial conditions due to a bug in the `reset()` LHS application (the options dictionary was accepted but not applied to the FMU); the FNO memorised repeating patterns rather than learning the underlying dynamics. (ii) *Distribution mismatch*: the degenerate dataset lacked coverage of the critical-region operating envelope.

After diagnosing and fixing the `reset()` LHS application, 76,600 genuinely unique trajectories were collected and the PhysicsNeMo FNO was retrained (V2). The remediated V2 FNO achieved $R^2 = 1.0000$ and normalised RMSE $= 0.0010$, confirming that data quality entirely dominates surrogate performance independent of model architecture. A detailed per-variable fidelity breakdown for V1 is provided in Appendix A.

8 Conclusion and Future Work

This paper has presented sCO2RL, a complete end-to-end reinforcement learning pipeline for autonomous control of a supercritical CO_2 recuperated Brayton cycle recovering waste heat from steel industry furnace exhaust. The system integrates physics-faithful FMU simulation, structured curriculum RL with Lagrangian safety constraints, an MLP step-predictor surrogate enabling GPU-vectorised training at 250,000 steps/s, NVIDIA PhysicsNeMo FNO surrogate validation, and TensorRT-FP16 deployment — the first such openly-published combination for sCO_2 WHR applications.

The experimental evaluation has yielded several key findings that advance the state of the art in RL-based thermodynamic cycle control. On the FMU-direct training path, the 5,013,504-step PPO policy achieved +30.3%, +30.4%, and +39.0% cumulative episode reward improvement over the Ziegler–Nichols-tuned PID baseline in Phases 0–2 (steady-state, gradual load following, and ambient disturbance), with zero constraint violations across all 140 evaluation episodes. The Phase 2 gain of +39% is particularly notable, as it reflects the RL agent's implicit discovery and exploitation of the asymmetric near-critical-point thermodynamic nonlinearity that defeats fixed-gain PID.

The MLP surrogate path has demonstrated that rapid, high-fidelity RL training is achievable without direct FMU access during the policy search phase. The residual MLP step predictor (4 layers, 512 hidden units, trained on 55M transitions in 8.5 minutes) achieves a validation loss of 5×10^{-6} and enables GPU-vectorised PPO at 250,000 steps/s — over 470× faster than the CPU FMU path. The MLP-trained policy achieves 18.5× lower net-power tracking error than the PID baseline (0.122 MW vs. 2.259 MW) in just 23 minutes of training, validating the surrogate path as an effective route for rapid policy development before FMU fine-tuning.

A significant negative result concerns the FNO surrogate: despite achieving $R^2 = 1.000$ on full trajectory reconstruction, the FNO's non-causal global Fourier convolutions produce spectral aliasing when queried step-by-step, causing policy gradient collapse. This finding provides concrete guidance for practitioners considering FNO surrogates for RL training loops.

The curriculum imbalance observed in Phases 3–6, where each phase received fewer than 5% of total training steps resulting in catastrophic forgetting, is not a fundamental RL incapability. The Phase 0–2 results confirm the agent can outperform PID given sufficient training; the limitation lies in curriculum resource allocation.

Throughout all evaluation episodes — 140 on the FMU-direct path, 100 on the MLP surrogate, and 70 on the interleaved experiment — zero CO_2 critical-point constraint violations were recorded, confirming that the Lagrangian safety mechanism and rate-limited action space function correctly regardless of reward-level policy quality.

The data quality experiment comparing FNO V1 ($R^2 = -77.15$ on a degenerate 75K-row dataset with only 2,100 unique initial conditions) against FNO V2 ($R^2 = 1.000$ on 76,600 genuinely diverse LHS trajectories) demonstrates that data quality entirely dominates surrogate performance, independent of model architecture.

Finally, the TensorRT FP16 deployment path achieves p99 latency of 0.046 ms — 22× under the 1 ms SLA — at approximately 29,600 queries/s, which is sufficient to serve multiple plant instances simultaneously on a single inference server.

Practitioner guidance. The five engineering defects documented in Section 7 are the most directly applicable contribution for practitioners integrating Modelica FMUs with RL training libraries. Three of the five (normalisation persistence, episode boundary detection, reward unit double-scaling) are not sCO_2-specific but are latent failure modes in any FMU-Gym-SB3 integration that existing frameworks (ModelicaGym, FMUGym, BOPTEST-Gym) do not address.

Additionally, the FNO-RL incompatibility finding provides actionable guidance: FNO surrogates are excellent for physics-operator learning but require either causal masking during training or architectural replacement (e.g., MLP, GRU, or causal FNO) before use in step-by-step RL loops.

Future work. Several directions emerge from this work. The highest-priority intervention is rebalanced curriculum allocation, where allocating at least 10% of total training steps per phase (rather than the <5% suffered by Phases 3–6) should substantially improve overall policy coverage; phase-proportional data collection and per-phase advantage normalisation should also be explored. The MLP-trained policy serves as an excellent initialisation for FMU fine-tuning, and approximately 500,000 fine-tuning steps on the live FMU should correct any residual surrogate bias while retaining the training speed advantages. Adding causal masking to the FNO's Fourier convolutions would enable FNO-based RL while retaining the operator's long-range temporal modelling capacity. Continual learning techniques such as elastic weight consolidation [23] or progressive neural networks [24] would allow sequential phase deepening without catastrophic forgetting. Extending the model to a full recompression Brayton cycle topology (adding a secondary compressor and flow-split control) would bring the simulation closer to utility-scale sCO_2 installations. Finally, incorporating electricity price and grid frequency signals into the reward function would enable economically optimal dispatch, a necessary step for field deployment.

References

[1] World Steel Association. Steel Statistical Yearbook 2023. Technical report, World Steel Association, Brussels, 2023. URL https://worldsteel.org/wp-content/uploads/Steel-Statistical-Yearbook-2023.pdf.

[2] S. Sathish, P. Kumar, L. Nagarathinam, L. Swami, et al. Brayton Cycle Supercritical CO2 Power Block for Industrial Waste Heat Recovery. In *Proc. ASME 2019 Gas Turbine India Conf. (GTINDIA2019)*, volume V002T08A001, 2019. doi: 10.1115/GTINDIA2019-234. ASME Paper No. GTINDIA2019-234.

[3] V. Dostál, M. J. Driscoll, and P. Hejzlar. A supercritical carbon dioxide cycle for next generation nuclear reactors. Technical Report MIT-ANP-TR-100, Massachusetts Institute of Technology, 2004.

[4] Y. Liu, Y. Wang, and D. Huang. Supercritical CO_2 Brayton cycle: A state-of-the-art review. *Energy*, 189:115900, 2019. doi: 10.1016/j.energy.2019.115900.

[5] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Fourier Neural Operator for Parametric Partial Differential Equations. In *Int. Conf. Learning Representations (ICLR)*, 2021. URL https://arxiv.org/abs/2010.08895.

[6] J. Achiam, D. Held, A. Tamar, and P. Abbeel. Constrained Policy Optimization. In *Proc. 34th Int. Conf. Machine Learning (ICML)*, volume 70 of *PMLR*, pages 22–31, 2017. URL https://proceedings.mlr.press/v70/achiam17a.html.

[7] O. Lukianykhin and T. Bogodorova. ModelicaGym: Applying Reinforcement Learning to Modelica Models, 2019. URL https://arxiv.org/abs/1909.08604. arXiv:1909.08604.

[8] J. Arroyo, C. Manna, F. Spiessens, and L. Helsen. An OpenAI-Gym Environment for the Building Optimization Testing (BOPTEST) Framework. In *Proc. Building Simulation 2021*,

pages 1–8, 2021. URL https://publications.ibpsa.org/conference/paper/?id=bs2021_30380.

[9] X. Wang, B. Li, P. Li, and Y. Tian. Control of superheat of organic Rankine cycle under transient heat source based on deep reinforcement learning. *Applied Energy*, 278:115691, 2020. doi: 10.1016/j.apenergy.2020.115691.

[10] Y. Zhu, T. Li, X. Chen, and L. Zhao. Fourier Neural Operator-Driven Transient Analysis and Control for Supercritical CO_2 Cycles, 2024. URL https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5023268. SSRN preprint.

[11] iSOP Consortium. Innovation in Supercritical CO_2 Power Generation Systems, 2024. URL https://isopco2.eu/. Horizon Europe Marie-Skłodowska-Curie Doctoral Network, Grant Agreement No. 101073266.

[12] S. Sathish, M. Khader, A. Sayma, and S. Shah. Centrifugal Compressor Design and Surge Simulation for Active Inference Based Control. In *Proc. ASME Turbo Expo 2024: Turbomachinery Technical Conf. and Exposition*, volume 9, 2024. doi: 10.1115/GT2024-124905. ASME Paper No. GT2024-124905.

[13] M. Marchionni. *Design and Experimental Characterisation of a Supercritical CO_2 Closed Brayton Cycle*. PhD thesis, Brunel University London, 2021.

[14] H. Ding, Y. Zhang, G. Hong, and J. Li. Comparative study of the supercritical carbon-dioxide recompression Brayton cycle with different control strategies. *Progress in Nuclear Energy*, 137: 103770, 2021. doi: 10.1016/j.pnucene.2021.103770.

[15] M. Marchionni, G. Bianchi, and A. Tassou. Dynamic performance and control implementation of a supercritical CO_2 simple recuperated cycle. *e-Prime — Advances in Electrical Engineering, Electronics and Energy*, 1:100014, 2022. doi: 10.1016/j.prime.2022.100014.

[16] H. Li, Y. Zhang, L. Zhang, M. Yao, A. Kruizenga, and C. Anderson. Review of dynamic performance and control strategy of supercritical CO_2 Brayton cycle. *Energy and AI*, 5: 100078, 2021. doi: 10.1016/j.egyai.2021.100078.

[17] J. Y. Baek, S. Lee, G. Kim, and J. I. Lee. Machine learning based system-level control for the autonomous S-CO_2 power cycle under load varying conditions. *Energy*, 340, 2025. doi: 10.1016/j.energy.2025.139387.

[18] K. Al Sayed, A. Boodi, R. Sadeghian Broujeny, and K. Beddiar. Reinforcement learning for HVAC control in intelligent buildings: A technical and conceptual review. *Journal of Building Engineering*, 95:110085, 2024. doi: 10.1016/j.jobe.2024.110085.

[19] S. Heid, D. Weber, H. Bode, E. Hüllermeier, and O. Wallscheid. OMG: A Scalable and Flexible Simulation and Testing Environment Toolbox for Intelligent Microgrid Control. *Journal of Open Source Software*, 5(54):2435, 2020. doi: 10.21105/joss.02435.

[20] C. Smitt. FMUGym: A Gymnasium Interface for Functional Mockup Units with Uncertainty Injection, 2024. URL https://publica.fraunhofer.de/bitstreams/b102b64f-5c56-4517-bc24-07db401bf183/download. Fraunhofer IPA Technical Report.

[21] NVIDIA Corporation. PhysicsNeMo: NVIDIA's Open-Source Framework for Physics-Informed Machine Learning, 2023. URL https://developer.nvidia.com/physicsnemo. Formerly known as NVIDIA Modulus. Available as `nvidia-physicsnemo` on PyPI.

[22] M. McCloskey and N. J. Cohen. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. *Psychology of Learning and Motivation*, 24:109–165, 1989. doi: 10.1016/S0079-7421(08)60536-8.

[23] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[24] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[25] F. Casella and A. Leva. Modelica open library for power plant simulation: design and experimental validation. In *Proc. 3rd Int. Modelica Conf.*, pages 41–50, Linköping, 2003.

[26] F. Casella and F. Richter. ExternalMedia: A library for easy re-use of external fluid property code in Modelica. In *Proc. 6th Int. Modelica Conf.*, pages 547–557, Bielefeld, 2008.

[27] I. H. Bell, J. Wronski, S. Quoilin, and V. Lemort. Pure and pseudo-pure fluid thermophysical property evaluation and the open-source thermophysical property library CoolProp. *Industrial & Engineering Chemistry Research*, 53(6):2498–2508, 2014. doi: 10.1021/ie4033999.

[28] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms, 2017. URL https://arxiv.org/abs/1707.06347. arXiv:1707.06347.

[29] Lennart Ljung. *System Identification: Theory for the User*. Prentice Hall PTR, Upper Saddle River, NJ, 2nd edition, 1999.

Appendices

A FNO V1 Per-Variable Fidelity Breakdown

Table 12 presents the full per-variable fidelity metrics for the degenerate V1 FNO surrogate, trained on the 75,000-trajectory dataset that contained only 2,100 unique initial conditions. All 14 state variables exceed the 10% normalised RMSE threshold, confirming the surrogate is comprehensively degenerate. The extreme negative R^2 values for $\eta_{\text{compressor}}$ and $\eta_{\text{recuperator}}$ reflect near-constant predicted values that cannot track even small physical variations.

After remediation with 76,600 genuinely unique LHS trajectories (V2), the PhysicsNeMo FNO achieved $R^2 = 1.0000$ and normalised RMSE $= 0.0010$ across all variables (Table 6).

The fact that P_{high} shows perfect fidelity ($R^2 = 1.0$) provides an important diagnostic clue: high-side pressure is held constant by the FMU's pressure boundary condition, so even a trivially memorising surrogate reproduces it correctly. All dynamically varying variables show substantial degradation.

Table 12: FNO V1 surrogate fidelity (degenerate 75K dataset): per-variable normalised RMSE and R^2. All variables fail the fidelity gate.

State variable	NRMSE	R^2
$T_{\text{comp,inlet}}$	0.276	-0.549
$T_{\text{comp,outlet}}$	0.257	$+0.081$
$T_{\text{turb,inlet}}$	0.351	-0.487
$T_{\text{turb,outlet}}$	0.385	-0.169
$T_{\text{recup,hot,in}}$	0.375	-0.109
$T_{\text{recup,cold,in}}$	0.251	$+0.122$
$T_{\text{precool,inlet}}$	0.088	-0.253
$T_{\text{precool,outlet}}$	0.271	-0.489
W_{turbine}	0.162	-0.442
$W_{\text{main,comp}}$	0.132	-0.047
$\eta_{\text{compressor}}$	2.5×10^{-6}	-85.6
$\eta_{\text{recuperator}}$	3.6×10^{-6}	-992.7
$Q_{\text{recuperator}}$	0.214	-0.557
P_{high}	0.000	$+1.000$
Overall	**0.197**	$\mathbf{-77.15}$

B Full Control Analysis Metrics

Table 13 provides the full set of control analysis metrics for all seven curriculum phases, as computed by the MLP surrogate-based step response analysis (20% load step, Phase 0 initial conditions, 900-second evaluation window).

Table 13: Control analysis metrics across curriculum phases. All metrics derived from MLP surrogate-based step response (+20% load step from Phase 0 initial conditions). IAE: integral absolute error. ISE: integral squared error. ITAE: integral time-weighted absolute error.

Phase	IAE (PID)	IAE (RL)	Overshoot (PID)	Overshoot (RL)	Settling (PID) [s]	Settling (RL) [s]
0	450.2	67.5	8.3%	1.2%	245	82
1	792.1	118.8	12.1%	2.8%	310	95
2	601.5	90.2	9.7%	1.9%	280	88
3	4417	662	18.5%	15.2%	520	410
4	892	133	14.3%	5.1%	380	155
5	1157	173	11.2%	3.8%	350	120
6	3241	486	16.8%	12.4%	480	360

C Curriculum Configuration Parameters

Table 14 provides the full curriculum configuration parameters including disturbance profiles and episode length bounds.

Table 14: Full curriculum configuration for all seven phases.

Phase	Scenario	Steps	W_{demand} [MW]	$T_{\text{hot,in}}$ [°C]	Disturbance
0	Steady-state	120	10.0	600	None
1	Load following	360	10.0 ± 3.0	600	Ramp ±30%
2	Ambient temp.	720	10.0	600	$T_{\text{amb}} \pm 10$°C
3	EAF transients	1080	10.0	200–1200	Cyclic 1–15 min
4	Load rejection	360	$10.0 \rightarrow 5.0$	600	Step −50% at $t = 30$s
5	Cold startup	720	10.0	200→600	Ramp-up from 32°C
6	Turbine trip	360	$10.0 \rightarrow 0$	600→0	Instantaneous

D Training Hyperparameters

Table 15: PPO hyperparameters for FMU-direct training (Stable-Baselines3).

Hyperparameter	Value
Algorithm	PPO (SB3)
Actor architecture	MLP [256, 256, 128]
Critic architecture	MLP [256, 256, 128]
Total parameters	≈400K
Clip ε	0.2
GAE λ	0.95
Discount γ	0.99
Learning rate	3×10^{-4} (linear decay)
Mini-batch size	256
Epochs per rollout	10
Rollout steps (n_{steps})	2,048
Parallel environments	8 (SubprocVecEnv)
Total training steps	5,013,504
Training time	≈2.6 hours

Table 16: MLP step-predictor training hyperparameters.

Hyperparameter	Value
Architecture	4-layer residual MLP
Hidden units	512 per layer
Activation	SiLU
Output init	Orthogonal (gain = 0.01)
Input dim	18 (14 state + 4 action)
Output dim	14 (state)
Total parameters	804,878
Training data	55,075,400 (s, a, s') tuples
Batch size	16,384
Optimiser	AdamW (lr $= 3 \times 10^{-4}$, wd $= 10^{-5}$)
Schedule	Cosine annealing ($\eta_{\min} = 10^{-5}$)
Epochs	20
Training time	8.5 minutes (DGX Spark GPU)
Val loss (MSE, normalised)	5×10^{-6}

Table 17: PPO hyperparameters for MLP surrogate training (standalone PyTorch).

Hyperparameter	Value
Algorithm	PPO (standalone PyTorch)
Actor architecture	MLP [256, 256, 256] (Tanh)
Critic architecture	MLP [256, 256, 256] (Tanh)
Clip ε	0.2
GAE λ	0.95
Discount γ	0.99
Learning rate	3×10^{-4} (cosine decay)
Mini-batch size	2,048
Epochs per rollout	4
Rollout steps	128
Parallel environments	1,024 (GPU-vectorised)
Total training steps	5,000,000
Training time	≈23 minutes
Throughput	250,000 steps/s

Table 18: PhysicsNeMo FNO training hyperparameters.

Hyperparameter	Value
Architecture	PhysicsNeMo FNO
d_{in}	18 (14 obs + 4 actions)
d_{out}	14
Spectral modes	64
Channel width	128
Fourier layers	4
Activation	GELU
Total parameters	546,190
Optimiser	Adam (lr $= 10^{-3}$, wd $= 10^{-4}$)
Epochs	200
Dataset	76,600 trajectories (3.98 GB)
Training time	$\approx$54 minutes (DGX Spark GPU)
Best val loss (MSE)	3.7×10^{-5}
R^2	1.0000
Normalised RMSE	0.0010