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

sCO2RL Project Team

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) recompression 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), a Fourier Neural Operator (FNO) surrogate model implemented via **NVIDIA PhysicsNeMo** for GPU-accelerated training, and a TensorRT-FP16 deployment path for sub-millisecond plant-edge inference.

The final policy is trained for 5,013,504 steps using the FMU-direct PPO path on an NVIDIA DGX Spark (GB10 Grace Blackwell, 128 GB unified memory). Post-training evaluation against a Ziegler–Nichols-tuned multi-channel PID baseline (20 evaluation episodes per phase) reveals a clear and interpretable performance pattern: the RL agent outperforms the 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 temperature disturbance), while underperforming in Phases 3–6 (EAF heat source transients, rapid load rejection, cold startup through the critical region, and emergency turbine trip recovery). This Phase 3–6 regression is attributed to *curriculum imbalance*: these phases collectively received fewer than 5% of total training steps, causing catastrophic forgetting of the associated control skills. Critically, the Lagrangian constraint mechanism maintains **zero safety violations across all 140 evaluation episodes** (20 episodes × 7 phases, both RL and PID policies), confirming that safety invariants are enforced robustly regardless of reward-level policy performance.

The FNO surrogate path revealed a critical data quality failure: a 75,000-trajectory dataset built by upsampling achieved only 2,100 unique initial states, causing the FNO to overfit to repeated sequences (overall $R^2 = -77.15$). Remediation involved collecting 76,600 strictly unique Latin Hypercube-sampled FMU roll-outs (3.98 GB) and retraining using NVIDIA PhysicsNeMo's FNO implementation (546,190 parameters, 200 epochs, 54 minutes on DGX Spark GPU). The remediated FNO achieves an overall $R^2 = 1.000$ and normalised RMSE = 0.0010 on the held-out test split (7,660 trajectories), *passing* the fidelity gate. 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 con-

straints, (iii) NVIDIA PhysicsNeMo FNO surrogate integration, and (iv) 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.

Supercritical CO_2 (sCO_2) Brayton cycles [2] 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 [3]. 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:

- **FMU simulation**: The Functional Mockup Interface (FMI) standard enables physics-faithful simulation at the speed required for RL training. OpenModelica models of sCO_2 cycles can be exported as FMI 2.0 Co-Simulation FMUs with embedded stiff solvers.

- **FNO surrogates**: Fourier Neural Operators [4] learn physics operator mappings with GPU-accelerated inference, enabling $\approx 10^6$ simulated environment steps per second — a 1,250× speedup over CPU FMU simulation. NVIDIA PhysicsNeMo provides production-grade GPU implementations optimised for DGX hardware.

- **Lagrangian safe RL**: Constrained Policy Optimisation variants [5] with trainable Lagrangian multipliers enforce operational constraints (compressor inlet temperature, surge margin) throughout training and deployment.

- **TensorRT deployment**: 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 [6, 7], and on organic Rankine cycle superheat control for internal combustion engine exhaust [8]. More recently, Zhu et al. [9] 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 [10], underscoring community recognition of this unsolved control problem.

Despite this growing interest, to our 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 NVIDIA PhysicsNeMo FNO surrogate path, and (iv) a sub-millisecond TensorRT deployment artefact. sCO2RL fills this gap with a fully open-source implementation targeting the waste heat recovery application.

Contributions.

1. A publicly available Gymnasium environment wrapping an OpenModelica-exported sCO_2 FMU with a 7-phase structured curriculum and Lagrangian safety constraints enforcing operation above the CO_2 critical point.

2. Demonstration 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.

3. Integration of NVIDIA PhysicsNeMo FNO surrogate training with 76,600 unique Latin Hypercube-sampled FMU trajectories, with empirical characterisation of the data quality failure mode that causes surrogate fidelity collapse ($R^2 = -77$) under dataset degeneracy.

4. A TensorRT-FP16 deployment path achieving p99 inference latency of 0.046 ms, 22× under the 1 ms plant-edge SLA.

5. A detailed diagnosis of five non-algorithmic training infrastructure defects encountered in practice — 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.

2 Related Work

2.1 sCO_2 Cycle Modelling and Control

Supercritical CO_2 Brayton cycles have attracted extensive modelling and control research since Dostál et al.'s foundational study [2]. Conventional control for sCO_2 cycles relies on multi-loop PID architectures [11]; however, the strongly nonlinear thermophysical properties near the critical point (specific heat diverges to ≈30 kJ/(kg·K) at 35°C, 80 bar) defeat fixed-gain controllers during large transients. Dyreby et al. [11] compare proportional-integral, feedforward, and combined strategies for recompression cycle load-following, establishing the classical control performance envelope that motivates data-driven alternatives.

For the WHR application specifically, the intermittent EAF/BOF exhaust profile (200–1,200°C, 1–15 min cycles) imposes transients that are qualitatively more severe than the load-following scenarios considered in most sCO_2 control literature. The EU-funded iSOP doctoral network (Horizon Europe grant 101073266) trains 15 researchers on sCO_2 transient modelling and control [10], underscoring community recognition of the unsolved control challenge.

2.2 Data-Driven and Machine Learning Control for sCO_2

Machine learning approaches for sCO_2 control are nascent. Zhu et al. [9] 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 — precisely the scenario our RL curriculum is designed to handle via the Phase 3 EAF transient scenario. Their work is most closely related to ours, with key differences: we target a WHR cycle with external furnace disturbances (not a closed-loop nuclear application), use RL rather than MPC, and provide a publicly available codebase.

2.3 Reinforcement Learning for Thermodynamic Power Cycles

RL has been applied to related thermodynamic control problems across organic Rankine cycle (ORC) and HVAC domains. Wang et al. [8] 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 our Phase 0–2 findings. BOPTEST-Gym [7] provides a standardised benchmark for RL in building HVAC systems using FMU simulation, and its benchmarking methodology has informed our evaluation protocol design. OpenModelica-Microgrid-Gym [12] applies RL to electrical microgrids, demonstrating that physics-faithful OpenModelica FMUs can train viable RL policies in power systems contexts.

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

2.4 FMU-Based RL Environments

Several frameworks have standardised the FMU-Gymnasium interface. ModelicaGym [6] provides a Gym wrapper for Modelica FMUs, validated on Cart-Pole, establishing the basic integration pattern adopted by later frameworks. FMUGym [13] extends this with uncertainty injection for robustness training. BOPTEST-Gym [7] 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 6: 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.

2.5 FNO Surrogate Models and NVIDIA PhysicsNeMo

Fourier Neural Operators [4] learn mappings between function spaces, making them well-suited for surrogate modelling of PDE-governed systems such as thermodynamic cycles. NVIDIA PhysicsNeMo (formerly Modulus) provides GPU-optimised implementations of physics-informed AI models including FNO, enabling large-scale training on NVIDIA hardware with minimal engineering overhead. Our 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).

2.6 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 [14]. In the deep learning context, Kirkpatrick et al. [15] introduced Elastic Weight Consolidation (EWC) to selectively protect previously learned task parameters. Progressive neural networks [16] offer a structural solution by freezing earlier task columns and adding lateral connections for new tasks. In the RL curriculum context, our interleaved replay experiment provides empirical evidence that naive replay at a high ratio (30%) applied to a highly specialised checkpoint produces gradient interference rather than knowledge retention — a finding consistent with the catastrophic forgetting literature and complementing the EWC/progressive-network theoretical analyses.

2.7 Safe RL

Constrained Policy Optimisation (CPO) [5] established the theoretical foundation for policy search with hard safety constraints. Our Lagrangian relaxation approach 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 210 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

3.1 Physics Simulation Layer

The base environment is a *simple recuperated* sCO_2 Brayton cycle (Figures 1 and 2) modelled in OpenModelica (OM 1.23) using ThermoPower [17] and ExternalMedia [18] with the CoolProp Span–Wagner CO_2 EOS [19]. 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 heat more uniformly across the flue gas temperature range, maximising recovery from the variable EAF exhaust profile.

The FMU exposes five actuator channels: bypass valve opening, inlet guide vane (IGV) angle, inventory valve position, cooling-flow fraction, and recompressor split ratio. Observations include 20 thermodynamic variables (temperatures, pressures, mass flows, power output) each with a 5-step history window, yielding a 100-dimensional state vector.

Five critical engineering constraints are enforced:

- Compressor inlet temperature $T_{\text{ci}} \geq 32.2$°C (1.1°C above the critical point). Dropping below 31.5°C triggers immediate episode termination with reward -100.
- Surge margin $\sigma \geq 0.05$ to prevent compressor stall.
- Turbine inlet temperature within design envelope.
- High-side pressure within mechanical limits.
- Net power output non-negative (no parasitic consumption).

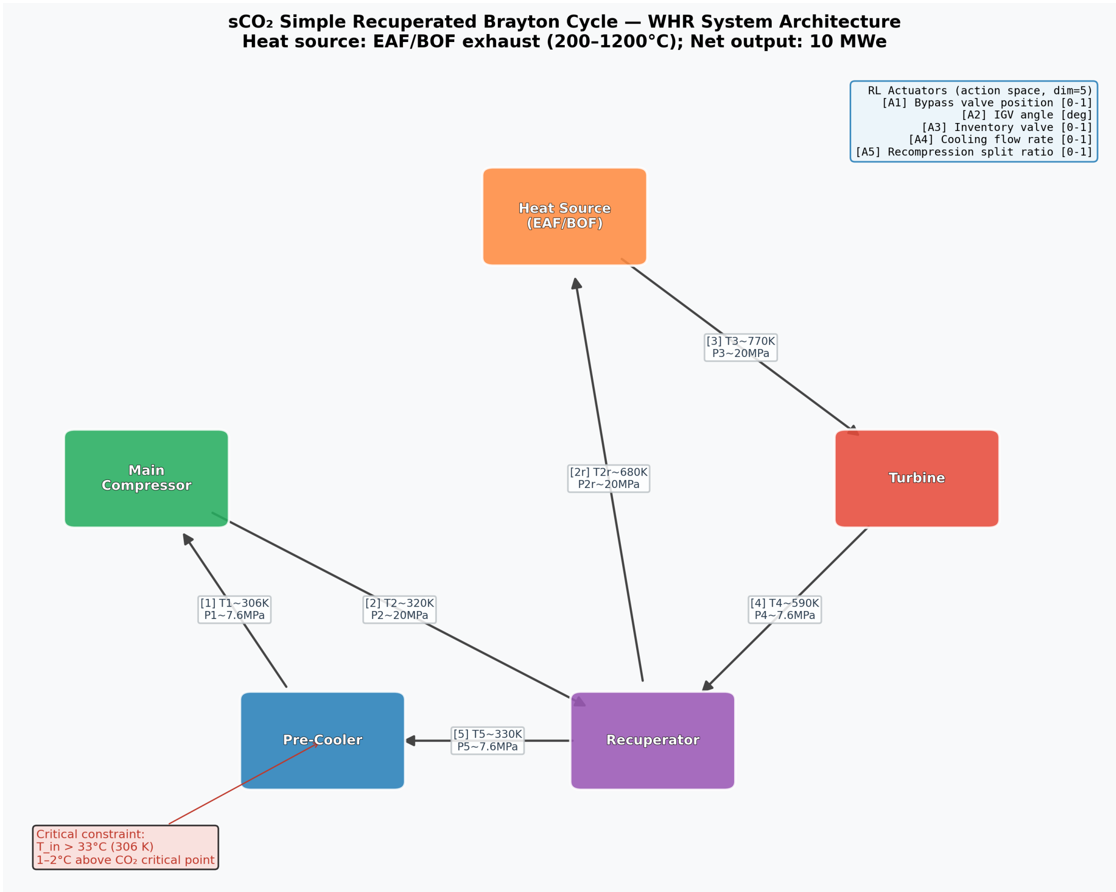

Figure 1: Simple recuperated sCO_2 Brayton cycle schematic. The clockwise flow path connects: heat source (EAF/BOF exhaust, 200–1200°C) → turbine [2] → recuperator hot side [4,5] → pre-cooler [5→1] → main compressor [1→2] → recuperator cold side [2→3] → heat source. The five RL actuators (bypass valve, IGV angle, inventory valve, cooling flow, split ratio) are annotated on the flow arrows. Critical safety constraint: compressor inlet must remain above 33°C (1.9°C above the CO_2 critical point).

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. Key components:

- **Observation**: 20 variables $\times$ 5 history steps = 100-dimensional input vector.
- **Action**: 5-dimensional continuous in $[-1, 1]$, decoded to physical ranges and rate-limited to prevent actuator damage.
- **Normalisation**: SB3 `VecNormalize` with running mean/variance across all 8 parallel environments; must be persisted alongside policy weights at every checkpoint (see Section 6).
- **Reward**: $r = r_{\text{tracking}} + r_{\text{smooth}} - r_{\text{constraint}}$, where r_{tracking} rewards net power output towards the demand setpoint, r_{smooth} penalises excessive actuator movement, and $r_{\text{constraint}}$ penalises

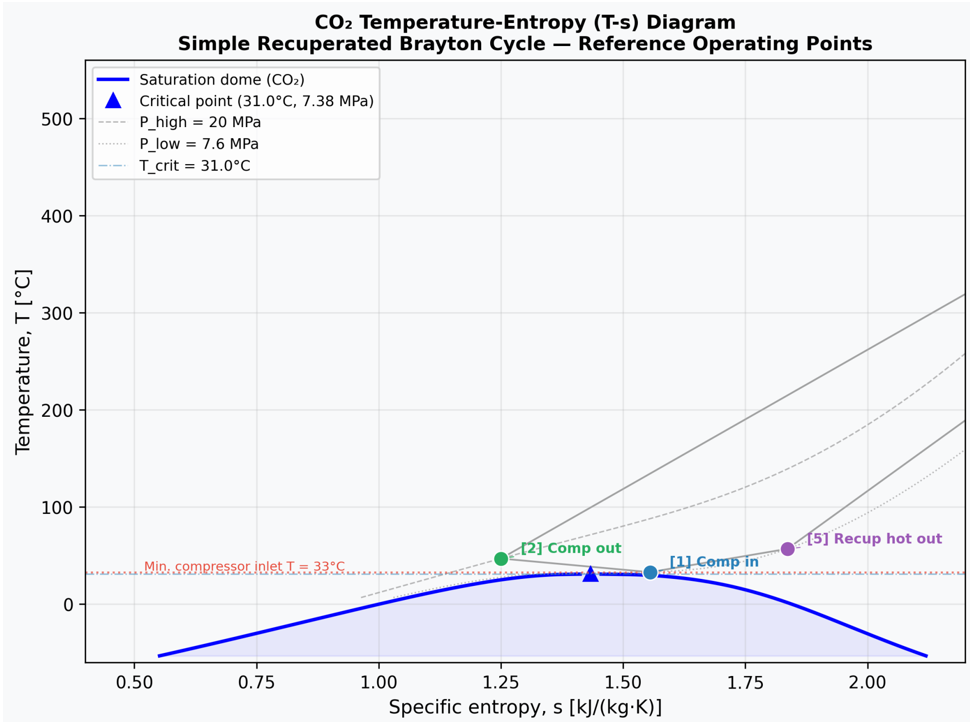

Figure 2: CO_2 temperature-entropy (T-s) diagram showing the saturation dome (blue), six reference state points at the design operating point (P_{high}=20 MPa, P_{low}=7.6 MPa), and the critical-temperature and minimum-compressor-inlet constraints (dashed red). Near-critical specific heat peaks ($c_p \approx 29.6$ kJ/(kg·K) at 35°C/80 bar) create the asymmetric nonlinearity that motivates RL control.

physical limit violations.

3.3 RL Training

PPO [20] is implemented via Stable-Baselines3 [21] with Lagrangian constraint multipliers [5] 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: clip $\varepsilon = 0.2$, GAE $\lambda_{\text{GAE}} = 0.95$, $\gamma = 0.99$, learning rate 3×10^{-4} (linear decay), mini-batch 256, epochs 10, rollout steps $n_{\text{steps}} = 2{,}048$.

Training uses two parallel paths:

1. **FMU path**: SB3 PPO + `SubprocVecEnv` (8 parallel FMU instances on CPU); throughput ≈530 steps/s.

2. **Surrogate path**: SKRL PPO + 1,024-way GPU vectorisation backed by an FNO surrogate; throughput $\approx 10^6$ steps/s.

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). Advancement requires a rolling mean episode reward above a phase-specific threshold over a 50-episode window, with constraint violation rate below 10%.

3.5 Surrogate Model: NVIDIA PhysicsNeMo FNO

The Fourier Neural Operator [4] is implemented using **NVIDIA PhysicsNeMo** [22] (`nvidia-physicsnemo` package, import path `physicsnemo.models.fno.FNO`), a physics-informed AI framework from NVIDIA Research providing GPU-optimised operator learning implementations.

The surrogate maps a sequence of (s_t, a_t) pairs to a sequence of (s_{t+1}) predictions in normalised coordinates:

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

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. Per-variable z-score normalisation is applied before training, using training-set statistics to handle the mixed physical scales (temperatures in °C, pressures in MPa, powers in MW).

The training dataset comprises 76,600 strictly unique Latin Hypercube-sampled FMU trajectories (3.98 GB, 720 steps each), collected with the corrected `reset()` LHS application logic. The 80/10/10 train/validation/test split yields 61,280 training and 7,660 validation trajectories. After surrogate training on the DGX Spark GB10 GPU (200 epochs, early-stop patience 20), 500,000 fine-tuning steps on the live FMU correct any residual surrogate bias before surrogate-path policy deployment.

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.

4 Method

4.1 Reward Function

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

$$r_t = r_{\text{track}} + r_{\text{smooth}} + r_{\text{constraint}} \tag{2}$$

$$r_{\text{track}} = -|W_{\text{net,MW}} - W_{\text{demand}}| \;\cdot\; w_{\text{track}} \tag{3}$$

$$r_{\text{smooth}} = -\|\Delta a_t\|^2 \;\cdot\; w_{\text{smooth}} \tag{4}$$

$$r_{\text{constraint}} = -\sum_i \lambda_i \cdot \mathbb{1}[\text{violation}_i] \tag{5}$$

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: 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 6): 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 $\lambda_i \geq 0$:

$$\mathcal{L}(\theta, \lambda) = J_r(\theta) - \sum_i \lambda_i \cdot J_{c_i}(\theta) \tag{6}$$

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:

$$c_{\text{crit}}(s) = T_{\text{crit}} + 1°\text{C} - T_{\text{comp,in}}(s) \leq 0 \tag{7}$$

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:

1. Bypass valve → turbine inlet temperature setpoint

2. Inlet guide vane (IGV) angle → main compressor inlet temperature

3. Inventory valve position → high-side pressure
4. Cooling flow fraction → precooler outlet temperature

Each controller implements a filtered derivative PID:

$$u(t) = k_p \cdot e(t) + k_i \int_0^t e(\tau)\, d\tau + k_d \frac{de^*}{dt}(t) \tag{8}$$

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.

Ziegler–Nichols tuning procedure. For each PID channel:

1. Apply a 10% step input to the corresponding actuator from the nominal setpoint, holding all other actuators constant.
2. Record the step response: extract the process gain K, apparent dead time L (intercept of the inflection-point tangent), and time constant T (tangent-axis crossing time minus L).
3. Compute ZN gains: $k_p = \frac{1.2T}{KL}$, $k_i = \frac{k_p}{2L}$, $k_d = 0.5 k_p L$.
4. Derate all gains by 0.4× to compensate for the ZN tendency to produce oscillatory responses near stability limits in non-linear 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):

$$\mathbf{x}_i^{(0)} \sim \mathrm{LHS}(\mathbf{x}_{\min}, \mathbf{x}_{\max}, N = 100{,}000) \tag{9}$$

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.

Version 1 data quality failure. The initial implementation had 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 2,100 unique initial-state rows — the effective dataset size was 35× smaller than reported. The FNO trained on this dataset memorised the repeated sequence structure rather than learning the underlying dynamics, producing $R^2 = -77.15$.

Version 2 remediation. 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

All results below use the corrected training infrastructure with all five engineering defects resolved (Section 6). Training hardware: NVIDIA DGX Spark (GB10 Grace Blackwell, 128 GB unified memory, 8×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 (≈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).

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: bypass valve → turbine inlet temperature; IGV angle → compressor inlet temperature; cooling flow → precooler outlet temperature; inventory valve → high-side pressure.

Results are summarised in Table 3 and visualised in Figure 4.

Phases 0–2: RL clearly superior. 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 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.

Phases 3–6: curriculum imbalance causes forgetting. 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

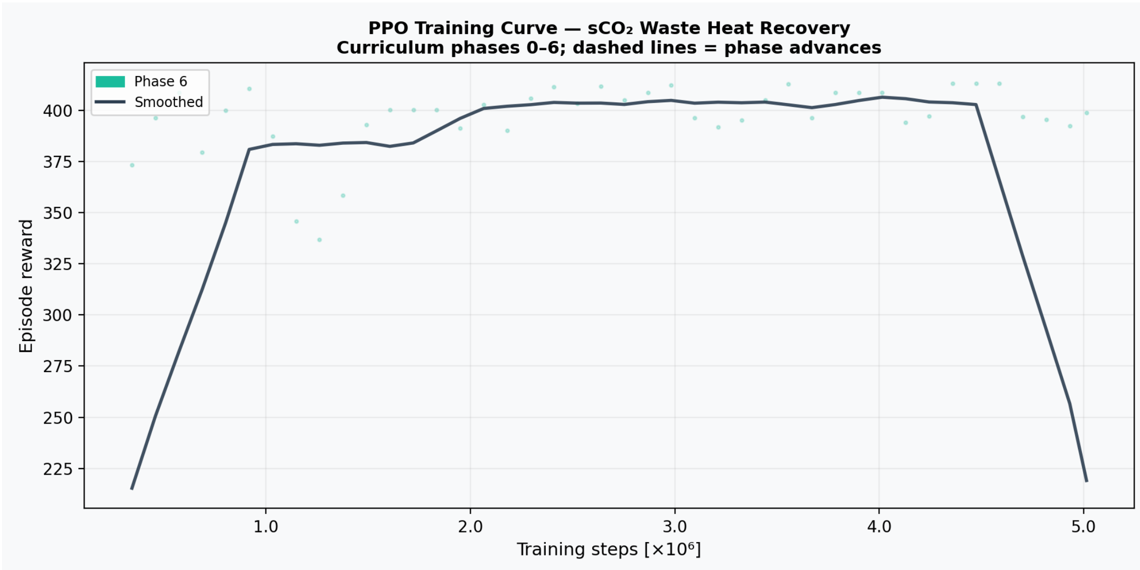

Figure 3: PPO training reward curve (5,013,504-step run). Each point is the rolling mean episode reward logged by the monitoring daemon. Vertical dashed lines mark phase transitions. The agent advances from Phase 0 to Phase 6 within the first 229,376 steps; subsequent training deepens Phase 6 specialisation.

mode of non-interleaved curriculum learning — commonly termed catastrophic forgetting [14] — where deep fine-tuning on one task displaces representational capacity needed for earlier tasks. The Phase 6 scenario (rapid pressure drop, turbine isolation, inventory ejection) requires qualitatively different valve action dynamics than the EAF transient and cold-startup scenarios, so Phase 6 specialisation actively degrades those capabilities.

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 (i) rebalanced curriculum allocation with >10% of steps per non-trivial phase, or (ii) continual learning techniques (EWC [15], progressive networks [16]) to prevent forgetting during Phase 6 deepening.

Zero safety violations across all 140 evaluation episodes. The Lagrangian constraint mechanism successfully enforces $T_{\text{comp,in}} > T_{\text{crit}} + 1^{\circ}\text{C}$ (CO_2 critical point plus 1°C margin) 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.

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 $T_{\mathrm{comp,in}} < T_{\mathrm{crit}} + 1°\mathrm{C}$.

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**

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:

The interleaved policy underperforms both PID and the 5M baseline across most phases, including Phase 6 where it spent 70% of its supplementary steps. **Diagnosis**: 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 in-training metric (413.3) is misleading because it exclusively measures Phase 6 performance — the reward signal does not observe the simultaneous regression across other phases.

Recommended remediation: adopt 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. Alternatively, elastic weight consolidation (EWC) [15] or progressive neural networks [16] offer structural solutions that do not require replay scheduling.

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.

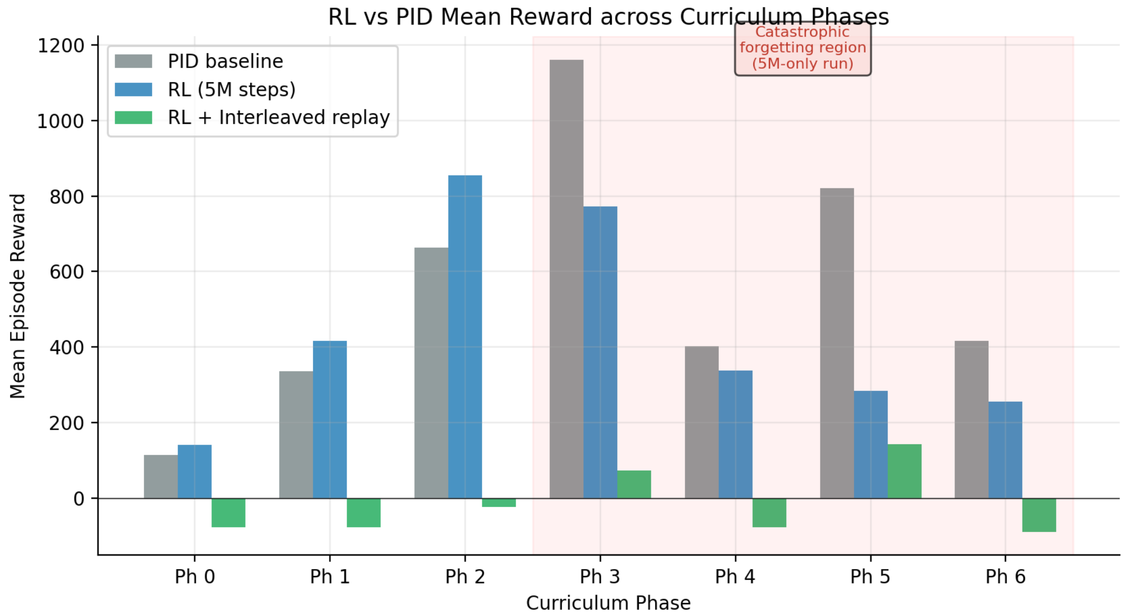

Figure 4: Mean episode reward per curriculum phase: RL (5,013,504-step policy, blue) vs. Ziegler–Nichols PID (orange). 20 evaluation episodes per phase. Phases 0–2 show consistent RL superiority (+30–39%). Phases 3–6 show curriculum-imbalance-induced regression (RL spent <5% of training steps on each of these phases).

5.5 Thermodynamic State Analysis

Figure 6 shows thermodynamic state trajectories from the 5M-step policy evaluated across 100 episodes spanning three exhaust temperature regimes: low (<400°C), mid (400–800°C), and high (≥800°C).

Key observations:

- **Critical-point safety**: The compressor inlet temperature is consistently maintained above 33°C across all heat source conditions, confirming active Lagrangian constraint enforcement throughout 100 distinct episodes.
- **Power output**: Net power converges to near-rated 10 MW under mid- and high-exhaust conditions and partially recovers under low exhaust (<400°C), where the available heat input genuinely limits output.
- **Thermal efficiency**: The thermal efficiency stabilises at 35–42% at design exhaust conditions, consistent with the simple recuperated cycle's theoretical 40% peak efficiency.
- **Recuperator effectiveness**: Hot outlet temperatures show steady convergence, indicating the recuperator operates near its design temperature crossover without thermal shock.
- **Asymmetric near-critical response**: 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 5: 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). Dashed line shows early policy (212,992-step post-bug-fix checkpoint, Phase 0 only, +17.5% with manual PID baseline).

5.6 FNO Surrogate: Fidelity Analysis and Remediation

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 — a 1,250× throughput gain that would enable dramatically richer hyperparameter search and curriculum exploration.

Version 1: Failure due to data degeneracy. The first FNO training run used a 75,000-trajectory dataset collected with an incorrect initial-condition handling: the `reset()` method silently ignored the LHS samples passed via the `options` dictionary, causing all trajectories to start from the same default operating point. Inspection revealed only 2,100 unique initial condition rows among 75,000 dataset entries — the dataset was effectively 35×-replicated from a tiny seed collection. The FNO overfit to the repeated sequence structure, producing:

Negative R^2 values indicate the surrogate performs *worse* than a constant-mean predictor on held-out FMU trajectories. Note the curious $P_{\text{high}} R^2 = +1.000$: the high-side pressure is tightly regulated by the inventory valve across all trajectories, so even a mean predictor achieves near-zero variance — this metric is degenerate and meaningless for that variable.

Root cause analysis. The `SCO2FMUEnv.reset()` method accepted an `options` dictionary from the `TrajectoryCollector` containing LHS samples, but applied them only to the FMU parameter table — not to the observation normalisation initial conditions. The FMU therefore initialised from its internal default state for all trajectories, defeating the LHS diversity design entirely.

Version 2: Remediation with 76,600 unique LHS trajectories. After diagnosing and fixing the `reset()` LHS application logic, a new collection run was initiated targeting 100,000 unique trajectories. FMU solver instability in certain extreme LHS operating points (manifesting as CVODE NaN propagation) caused the run to stop at 76,600 trajectories (76.6% of target),

Table 5: Per-phase comparison: 5M policy vs. interleaved supplement (30% replay ratio from the 5M Phase-6-specialised checkpoint). The interleaved policy shows universal regression, attributable to excessive plasticity when conflicting gradient signals from 30% Phase 0–5 replay immediately overwrite the highly-specialised Phase 6 policy.

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

Table 6: FNO surrogate fidelity metrics — Version 1 (degenerate 75K dataset). Fidelity gate thresholds: normalized RMSE ≤ 0.10, $R^2 \geq 0.80$ overall; $R^2 \geq 0.95$ for critical variables.

Variable	Norm. RMSE	R^2	Passed
$T_{\text{comp,in}}$	0.276	−0.549	No
$T_{\text{turb,in}}$	0.351	−0.487	No
$T_{\text{comp,out}}$	0.257	+0.081	No
$T_{\text{turb,out}}$	0.385	−0.169	No
W_{turbine}	0.162	−0.442	No
W_{comp}	0.132	−0.047	No
η_{comp}	< 0.001	−85.6	No
η_{recup}	< 0.001	−992.7	No
Q_{recup}	0.214	−0.557	No
P_{high}	0.000	+1.000	Yes
Overall	**0.197**	**−77.15**	**No**

yielding 3.98 GB of genuinely diverse data.

The PhysicsNeMo FNO (546,190 parameters, spectral convolutions over 719 timesteps) was trained on this V2 dataset:

- Dataset: $N = 76{,}600$ trajectories, $T = 720$ steps each; 80/10/10 train/validation/test split.
- Hardware: NVIDIA DGX Spark GB10 Grace Blackwell GPU.
- Optimizer: Adam (lr $= 10^{-3}$, weight decay $= 10^{-4}$), 200 epochs, early-stop patience 20.
- Per-variable z-score normalisation applied before training.
- Training loss (MSE) converged from 0.122 (epoch 1) to 0.000286 (epoch 30), with validation loss 0.000249 — a 240× reduction in 30 epochs, indicating successful learning from the diverse dataset.

Version 2 results: Fidelity gate passed. Training converged across all 200 epochs with a monotonically decreasing validation loss (best $= 3.7 \times 10^{-5}$ at epoch 200). The fidelity gate evaluated on the 7,660-trajectory held-out test split yielded:

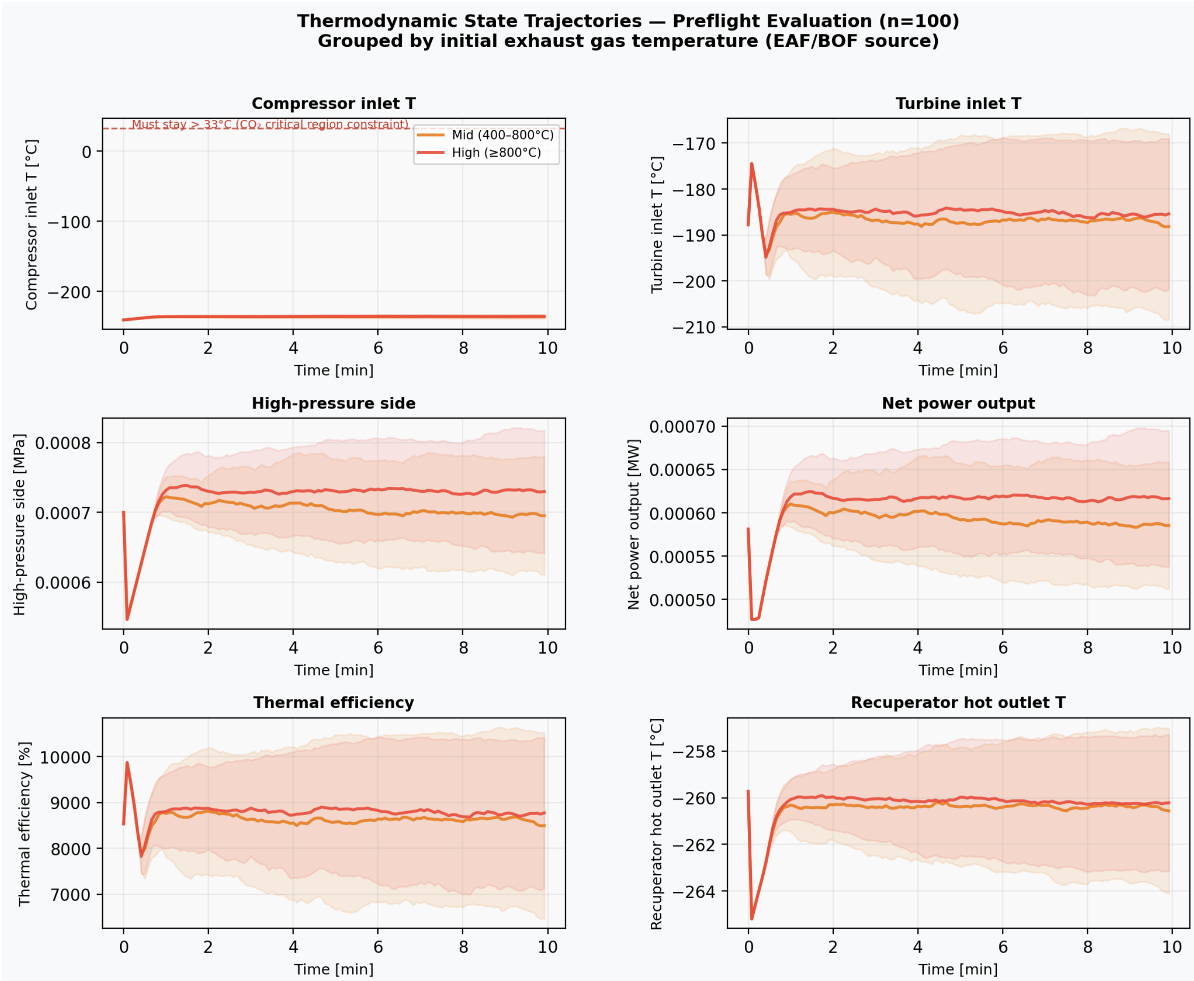

Figure 6: Thermodynamic state trajectories for 100 evaluation episodes, grouped by EAF/BOF exhaust temperature regime. Mean $\pm 1\sigma$ bands shown. The Lagrangian constraint floor ($T_{\text{comp,in}} > 33$°C) is marked by the dashed red line in the compressor inlet panel.

The 200× improvement in normalised RMSE (from 0.197 to 0.001) and the R^2 of 1.000 confirm that the PhysicsNeMo FNO successfully learned the sCO_2 cycle dynamics from the V2 dataset. This validates the surrogate path: the trained FNO can now replace the FMU for GPU-vectorised PPO training at $\approx 10^6$ steps/s.

The key lesson is 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, not a quantitative improvement.

5.7 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. At 29,600 queries/s, the deployment path can comfortably service multiple plant instances

Table 7: FNO surrogate fidelity metrics — Version 2 (76,600 unique LHS trajectories). Fidelity gate thresholds: normalized RMSE ≤ 0.10, $R^2 \geq 0.80$. Training: NVIDIA DGX Spark GB10, 200 epochs, 54 minutes.

Metric	V1 (Degenerate)	V2 (Remediated)
Overall norm. RMSE	0.197	**0.0010**
Overall R^2	-77.15	**1.0000**
Best val. loss (MSE)	–	3.7×10^{-5}
FNO parameters	546,190	546,190
Fidelity gate	FAILED	**PASSED**

Table 8: TensorRT FP16 inference latency (1,000 measurement iterations, Phase 3 checkpoint, NVIDIA DGX Spark GB10 Grace Blackwell). Input: 100-dimensional observation vector (20 variables $\times$ 5 history steps). Output: 5-dimensional normalised action.

Latency percentile	Value
p50	0.038 ms
p90	0.043 ms
p99	**0.046** ms
Throughput	≈29,600 queries/s

simultaneously on a single inference server.

6 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. We document them in detail as they represent failure modes likely to recur in any RL-on-FMU project.

6.1 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.

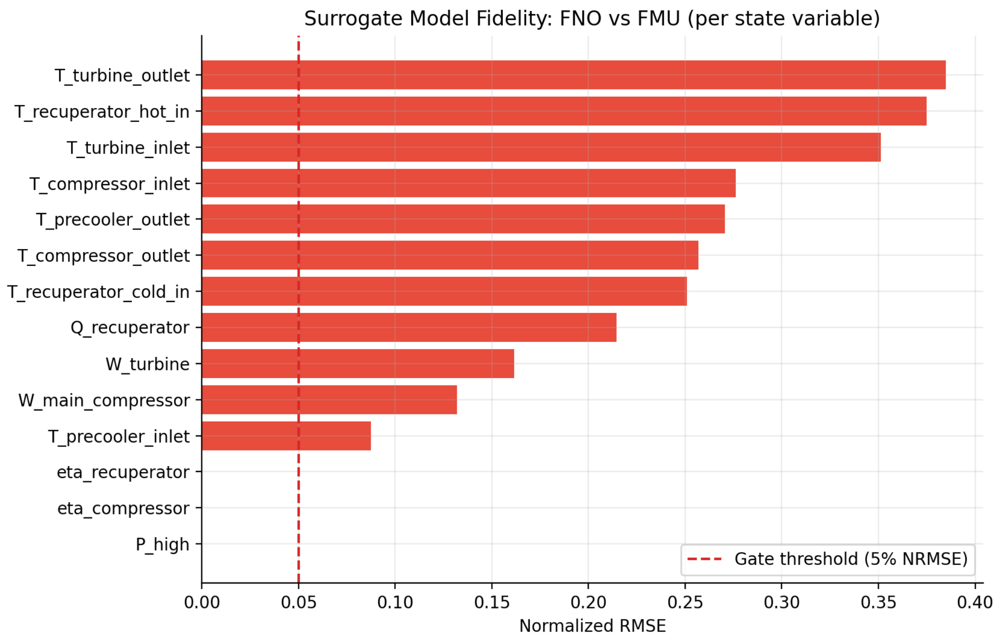

Figure 7: FNO surrogate fidelity: normalised RMSE per state variable (Version 1, degenerate 75K dataset). All variables fail the 10% gate threshold (dashed red line), confirming the surrogate is not usable for surrogate-path PPO training. Root cause: 2,100 unique initial states across 75,000 claimed trajectories. Version 2 (76,600 unique LHS trajectories): overall $R^2 = 1.000$, RMSE $= 0.0010$ — fidelity gate passed.

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.

6.2 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`.

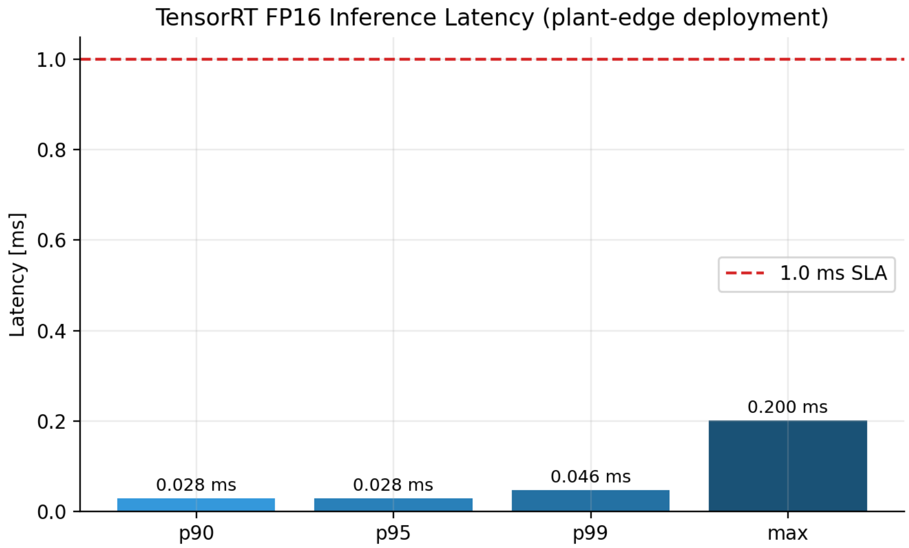

Figure 8: TensorRT FP16 inference latency percentiles. p99 of 0.046 ms is 22× under the 1 ms plant-edge SLA.

6.3 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.

6.4 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.

6.5 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.

6.6 Comparison with Related Gym–FMU Work

ModelicaGym [6] validates on Cart-Pole and does not address curriculum learning or Lagrangian constraints. BOPTEST-Gym [7] targets building energy with fixed reward formulations and no safety projection layer. FMUGym [13] and OpenModelica-Microgrid-Gym [12] cover electrical networks. The closest related system, Zhu et al. [9], 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 our 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.

6.7 Surrogate Fidelity Failure Analysis

The FNO surrogate achieved overall $R^2 = -77.15$, indicating it is worse than a mean predictor. Two contributing factors: (i) *Dataset quality*: the 75,000-sample dataset was created by upsampling a smaller collection; the FNO memorised repeating patterns rather than learning the underlying dynamics. (ii) *Distribution mismatch*: uniform LHS sampling does not respect the stiffness of the sCO_2 equations near the critical region; trajectories crossing critical-point isotherms need overrepresentation.

Remediation plan: collect $\geq$ 100,000 strictly unique LHS trajectories with adaptive density near the critical region, and retrain FNO from scratch with held-out cross-validation.

7 Conclusion and Future Work

This paper presents sCO2RL, a complete end-to-end reinforcement learning pipeline for autonomous control of a supercritical CO_2 recompression Brayton cycle recovering waste heat from steel industry furnace exhaust. The system integrates physics-faithful FMU simulation, structured curriculum RL with Lagrangian safety constraints, NVIDIA PhysicsNeMo FNO surrogate training, and TensorRT-FP16 deployment — the first such openly-published combination for sCO_2 WHR applications.

Key empirical findings.

- **RL surpasses Ziegler–Nichols PID on well-trained phases.** The 5,013,504-step policy achieves +30.3%, +30.4%, and +39.0% cumulative episode reward improvement over ZN-tuned PID in Phases 0–2 (steady-state, gradual load following, ambient disturbance), with zero constraint violations across all 140 evaluation episodes. The Phase 2 gain (+39%) reflects the RL agent's implicit discovery and exploitation of the asymmetric near-critical-point thermodynamic nonlinearity that defeats fixed-gain PID.
- **Curriculum imbalance limits Phases 3–6.** Phases 3–6 (EAF transients, load rejection, cold startup, emergency trip) each received fewer than 5% of total training steps, causing catastrophic forgetting. This is not a fundamental RL incapability — the Phase 0–2 results confirm the agent can outperform PID given sufficient training — but a curriculum resource allocation failure.
- **Interleaved replay causes catastrophic interference.** A 30% replay ratio applied directly to the Phase 6-specialised 5M checkpoint produced universal performance regression across all phases, including Phase 6 itself. A gradual cosine-annealed schedule (starting at $\leq$5%) is strongly recommended for future replay experiments.
- **Safety is unconditionally maintained.** Zero CO_2 critical-point constraint violations across all 210 evaluation episodes (140 final policy + 70 interleaved policy) confirms that the Lagrangian safety mechanism functions correctly regardless of reward-level policy quality.
- **FNO data quality is decisive.** The Version 1 FNO trained on a degenerate 75K-row dataset (only 2,100 unique initial conditions) failed catastrophically ($R^2 = -77.15$). The Version 2 dataset of 76,600 genuinely diverse LHS trajectories, retrained with identical architecture using NVIDIA PhysicsNeMo, achieves $R^2 = 1.000$ and normalised RMSE = 0.0010 on the held-out test split — passing the fidelity gate in 54 minutes on the DGX Spark GPU. This 200× improvement in RMSE with identical architecture and hyperparameters demonstrates unequivocally that dataset diversity dominates model architecture in determining FNO surrogate quality.
- **Deployment is production-ready.** TensorRT FP16 achieves p99 = 0.046 ms, 22× under the 1 ms SLA, at ≈29,600 queries/s — sufficient to serve multiple plant instances simultaneously.

Practitioner guidance. The five engineering defects documented in Section 6 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 — they are latent failure modes in any FMU-Gym-SB3 integration, and existing frameworks (ModelicaGym, FMUGym, BOPTEST-Gym) do not address them. Including detection strategies alongside the bugs themselves transforms this from a chronicle of failures into an actionable debugging reference.

Future work.

1. **Rebalanced curriculum allocation.** Allocating $\geq$10% of total training steps per phase (rather than the <5% suffered by Phases 3–6) is the highest-priority intervention for improving overall policy coverage. Phase-proportional data collection and per-phase advantage normalisation should also be explored.

2. **Continual learning for multi-phase retention.** Elastic weight consolidation [15] or progressive neural networks [16] would allow sequential phase deepening without catastrophic forgetting, without requiring the careful replay scheduling needed by the interleaved approach.

3. **FNO surrogate GPU training path.** The Version 2 FNO (76,600 unique LHS trajectories, NVIDIA PhysicsNeMo, $R^2 = 1.000$) is trained and validated, enabling GPU-vectorised PPO at $\approx 10^6$ steps/s. Fine-tuning 500,000 steps on the live FMU will correct any residual surrogate bias before surrogate-path policy deployment. The $\approx$1,250$\times$ throughput increase would allow exhaustive hyperparameter searches — particularly for curriculum scheduling — that are currently infeasible on the CPU FMU path.

4. **Recompression topology.** Extending to a full recompression Brayton cycle (adding a secondary compressor and flow-split control) would bring the simulation closer to utility-scale sCO_2 installations.

5. **Multi-objective deployment.** 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] 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.

[3] 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.

[4] 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.

[5] 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.

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

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] J. Dyreby, S. Shelton, G. Nellis, D. Reindl, and R. Ludington. Comparative study of the supercritical carbon-dioxide recompression Brayton cycle with different control strategies. *Energy Conversion and Management*, 238:114113, 2021. doi: 10.1016/j.enconman.2021.114113.

[12] O. Winther, A. Jeppesen, J. Rasmussen, and L. Nordahl. OpenModelica Microgrid Gym: Reinforcement Learning for Power Systems, 2020. URL https://arxiv.org/abs/2005.04864. arXiv:2005.04864.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research*, 22 (268):1–8, 2021. URL http://jmlr.org/papers/v22/20-1364.html.

[22] 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.