

Safe Contextual Tube-MPC for Microgrid Energy Management

Tinko Sebastian Bartels*, Ruixiang Wu*, Jiahao Ai*, and Tongxin Li

*Equal contribution.

This document accompanies the main article and contains the deployment verification details (Section S1), workload-effort annotation records (Section S2), and replay provenance (Section S3). All notation follows the main article.

S1 Deployment verification and online responses

The checks below distinguish a fitted-model assessment from a physical deployment certificate. A fitted-model check supports only a simulation claim; a physical certificate requires bounds and reachability tests validated for the deployed inverter.

S1.1 Error sets and tube propagation

Offline verification. Estimate mode- and horizon-conditioned residual envelopes $\mathcal{D}_{j|t}$ on data disjoint from model fitting. Enlarge them for sensor quantization, missing samples, and the chosen confidence or hard engineering margin. Verify that the set propagator outer-bounds every tested nonlinear successor.

Online response. Monitor whether measurements remain inside the declared envelope. An excursion invalidates the certificate and invokes a separately engineered emergency procedure. Replanning alone does not restore the deterministic guarantee.

S1.2 Terminal family and backup extension

Offline verification. Enumerate the finite command/dwell states and verify terminal invariance of $\mathcal{X}_f(r)$ and a full- $\mathcal{W}$ backup extension under $\mu_f(r)$ satisfying the tube propagation and tightening conditions.

Online response. Keep the shifted feasible scenario tree in memory. Preserve stored forecast envelopes and append the verified full- $\mathcal{W}$ backup. Safety fallback alone does not certify completion of recovery.

S1.3 Synchronization and contraction

Offline verification. For every candidate, bound pair-state reachability to the synchronization relation $\mathcal{S}_i(\delta_i)$, requiring a common discrete state and local neighborhood. Verify the contraction and cost-Lipschitz inequalities in the main article’s assumption entitled “Finite tube synchronization and local contraction.”

Online response. Execute the certified recovery policy without extending its deadline. Failures outside its certificate preserve safety only and fall outside the main article’s controller-switching regret theorem.

S1.4 Parallel-rollout losses and stage-cost bound

Offline verification. Bound the fitted-model stage-cost range L_C and verify numerical rollout costs. Record approximation and clipping discrepancies ϵ_t^C ; assess physical model fidelity separately on held-out trajectories.

Online response. Update all rollouts with the same revealed disturbance and causal information. Missing full-information losses require a partial-feedback algorithm and do not satisfy the main article’s controller-switching regret theorem.

S1.5 Context-free candidate and input boundary

Offline verification. Freeze the template questions and answer schema. Authenticate schedule/log sources and reject generated actions, malformed numerical features, and direct register requests. Include at least one self-contained controller that receives no text-derived feature.

Online response. Treat accepted language only as forecast evidence. The safety filter remains authoritative. Under its stated conditions, the main article’s context-free fallback corollary bounds long-run excess cost relative to the context-free candidate under misleading context.

S2 Workload-effort annotation records

S2.1 Provenance and use in the replay

The CPU test-suite labels were generated offline with GPT-5.4 Thinking using the prompt reproduced in Section S2.2. Non-library entries such as `index.html` were manually removed from the output. The transcript retains the original wording, order, and spelling, including all 156 comma-separated input entries; only line wrapping has been adjusted.

For CPU-compilation and GPU-training records, the archived field `llm_estimated_task_effort` contains low, medium, or high labels, mapped to 1/3, 2/3, or 1, respectively. Activity-weighted aggregation gives three semantic coordinates: CPU effort on each of two workstations and GPU effort. Missing annotations contribute zero. The labels remain fixed during replay; the numerical readout is updated from completed measurement intervals. No LLM response is used on the replay’s actuation path.

S2.2 Exact CPU-effort prompt

Question

For each of those boost library, list whether the computational effort for running the unit test suite with b2 will be low, medium, or high?

Complete input list (original order)

conversion, hash2, move, property_map, test, accumulators, convert, headers, mp11, property_map_parallel, thread, algorithm, core, heap, mpi, property_tree, throw_exception, align, coroutine, histogram, mpl, proto, timer, any, coroutine2, hof, mqtt5, ptr_container, tokenizer, array, crc, icl, msm, python, tti, asio, date_time, index.html, multi_array, qvm, tuple, assert, describe, integer, multi_index, random, type_erasure, assign, detail, interprocess, multiprecision, range, type_index, atomic, dll, intrusive, mysql, ratio, type_traits, beast, dynamic_bitset, io, nowide, rational, typeid, bimap, endian, iostreams, numeric, redis, units, bind, exception, iterator, optional, regex, unordered, bloom, fiber, json, outcome, safe_numerics, url, callable_traits, filesystem, lambda, parameter, scope, utility, charconv, flyweight, lambda2, parameter_python, scope_exit, uuid, chrono, foreach, leaf, parser, serialization, variant, circular_buffer, format, lexical_cast, pfr, signals2, variant2, cobalt, function, libraries.htm, phoenix, smart_ptr, vmd, compat, function_types, local_function, platform_maintainers.txt, sort, wave, compute, functional, locale, poly_collection, spirit, winapi, concept_check, fusion, lockfree, polygon, stacktrace, xpressive, config, geometry, log, pool, statechart, yap, container, gil, logic, predef, static_assert, container_hash, graph, preprocessor, static_string, context, graph_parallel, math, process, stl_interfaces, contract, hana, metaparse, program_options, system

S2.3 GPU-effort question

For GPU training, an analogous question requested low, medium, or high computational effort for each vision-model and optimizer combination. The available record describes this question but does not preserve a verbatim GPU prompt transcript. The description above therefore does not reconstruct or substitute for the original wording.

S3 Replay provenance and interpretation

These details are recorded in the README and provenance files of the continuous-week experiment archive.

S3.1 Study selection and continuity

The replay covers December 19–25, 2025, with all seven days retained in their original order. This complete week was selected from an exploratory suitability sweep and then evaluated using 256 paired selector seeds per context condition, disjoint from the screening seeds. These runs quantify selector randomization on one measured weather/load week; they are not 256 independent measured weeks.

Measured CPU/GPU load from two workstations is replayed on the fitted plant with 1.5-kWh storage and initial SoC of 65%. The running SoC penalty is 2, the MPC horizon is 24 hours, commands are updated every 15 minutes, and plant transitions use one-minute steps. Predictor, plant, and selector states continue across midnight. The selector uses $\eta = 8$ and loss bound 10.

S3.2 Information available to the predictors

Both predictors receive the same recorded workload activity/end schedule and known available PV. The structured predictor uses an intercept, three activity coordinates, CPU cores and log-transformed file counts for each workstation, GPU parameter count, and optimizer code. The semantic predictor adds only the three archived effort coordinates described in Section S2.

Coefficient updates use only past completed intervals, with a four-hour observation half-life and ridge regularization of 0.1. Forecast and PV rows extending beyond the episode cutoff repeat the final day’s pattern.

S3.3 Attack selection

The selected `future_high_peak` attack begins after four hours. It uses the current readout coefficients to select valid extreme future effort inputs that raise peak-period forecasts and lower other future forecasts. Current-interval features and numerical fields remain truthful. The attack was selected from four predefined policies using the fixed semantic controller’s cost; selector outcomes were not used in this selection.

S3.4 Reading the cost and final-choice comparisons

Figure 3 of the main article reports state-matched, within-day gaps to numerical offline optimum estimates. Each reference optimizes the complete day from the corresponding run’s actual initial full plant state and requires at least that run’s actual ending SoC. References therefore differ across trajectories. Signed component differences are retained, and the mean uses the ratio of summed costs rather than the mean of daily percentages.

Figure 4 uses direct realized-cost comparisons. A daily final-choice value refers to the last applied controller at that day’s endpoint. The whole-week final-choice value refers to the final decision of the continuous week; it is neither controller occupation averaged over time nor an average of the daily final-choice values. Negative observations are retained.

S3.5 Variation across selector seeds

The uncertainty of a mean cost estimate is smaller than the variation of individual selector trajectories. Table S1 reports both quantities as percentages of the mean weekly cost for the corresponding condition. Each condition uses 256 runs.

Table S1: Weekly cost variation across selector seeds.

Context condition	Individual-run standard deviation	Monte Carlo standard error of mean
Truthful	1.51%	0.0945%
Adversarial	1.70%	0.1061%