Supplementary Information

Verified Self-Improving Learning of Large Language Models for Multi-Objective Point-Spec Circuit Design

Juzheng Zhang[1,†], Kehao Zhang[1,‡], Yangfan Tang[1,‡]
Jinwen Liu[1,‡], Lijuan Li[1,*], Hongliang Liu[1,*], You Chen[2,*]

[1]Xiangtan University, Xiangtan 411105, China

[2]School of Housing, Building and Planning, Universiti Sains Malaysia, Penang 11700, Malaysia

[†]Juzheng Zhang is the sole first author.

[‡]Kehao Zhang, Yangfan Tang, and Jinwen Liu contributed equally as co-second authors.

[*]Correspondence: lilj@xtu.edu.cn

lhl@xtu.edu.cn

vxr4175@163.com

Contents

Supplementary Table S1: Family overview: subclasses, operating points τ, and headline measurements.

Category	**Subclass** f	**Operating point** τ **and measurements**
DC–DC	buck–boost, buck, boost, SEPIC	$\tau = (V_{\text{in}}, R_{\text{load}}, V_{\text{out}}^{\star})$; measurements: V_{avg}, η, ripple v_{ripple}, overshoot v_{over}.
Amplifier	RF power amplifier, two-stage op-amp	$\tau = (G^{\star}, f_{-3\text{dB}}^{\star})$; measurements: gain G, bandwidth $f_{-3\text{dB}}$, phase margin PM, static power P_{static}.
Oscillator	LC, RC, ring, Wien	$\tau = (V_{DD}, f_{\text{osc}}^{\star})$; measurements: oscillation frequency f_{osc}, amplitude A_{pp}, static power P_{static}, convergence check.

1 Additional Supplementary Details

DSL and validity. INC DSL enforces a restricted node set and a fixed format; each generated sample must be parsable, compilable into SPICE, and able to complete simulation. In the Grid77 evaluation reported in this paper, the simulation completion rate is 1.0000.

1.1 Task families and operating points

1.2 DC–DC Ablations and Tolerance Sensitivity (Supplement)

Supplementary Table S2: DC–DC tolerance sensitivity (open-loop/raw; Grid77; 77 tasks×10 generations; sample-level counts with duplicates).

	AR@1% ↑	AR@2% ↑	AR@5% ↑
Overall	0.9234	0.9390	0.9610
Buck	0.8808	0.9154	0.9577
Boost	0.8529	0.8706	0.9059
SEPIC	0.9941	0.9941	0.9941
Buck–Boost	0.9882	0.9882	0.9882

1.3 Constraints, verifiers, and active cores (unified)

This subsection only supplements **implementation details** not expanded in the main text: fixed node sets/node pools, and the provided active-core SPICE fragments used by amplifier/oscillator benchmarks. Definitions of OK/AR/CE, point-spec errors, and the closed-loop formulation are summarized in the main paper.

Supplementary Table S3: Amplifier/oscillator tolerance sensitivity after restricted repair (sample-level counts with duplicates).

Category	Subclass	AR@1% ↑	AR@5% ↑
Amplifier	Overall	0.3200	0.8600
	Two-stage op-amp (OpTwo)	0.3000	0.8200
	RFPA	0.3400	0.9000
Oscillator	Overall	0.6950	0.9500
	LC	0.8600	1.0000
	RC	0.6200	0.9600
	Ring	0.6600	0.8800
	Wien	0.6400	0.9600

Node sets and node pools.

- **DC–DC**: must include nodes $\{\texttt{vin}, \texttt{sw}, \texttt{out}, 0\}$; may additionally use a bounded helper-node pool $\{\texttt{nd1}, \ldots, \texttt{nd16}, \texttt{nsn1}, \ldots, \texttt{nsn4}\}$.
- **Amplifiers**: fixed nodes $\{\texttt{vin}, \texttt{inv}, \texttt{out}, \texttt{vdd}, 0\}$; the active part is provided by an active core (below).
- **Oscillators**: base nodes $\{\texttt{out}, \texttt{vdd}, 0\}$ plus a small set of auxiliary nodes (e.g., `n1,n2,...`); the active startup core is fixed (below).

Minimum component-count constraints follow the main paper.

pass flags (implementation). Besides OK/AR/CE, we log fine-grained pass flags for diagnosis and decomposition: for amplifiers, goal attainment corresponds to `pass_gain` & `pass_bw`, while CE further depends on `pass_pm` and `pass_p`; for oscillators, goal attainment corresponds to `pass_freq`, while CE further checks `pass_vpp` and `pass_p`.

1.4 Measurement stacks per family

DC–DC measurements. Each Grid77 converter task specifies $(V_{\text{in}}, R_{\text{load}}, V_{\text{out}}^{\star})$ with tolerances on the DC output. The verifier records V_{avg}, ripple amplitude V_{ripple}, overshoot V_{over}, efficiency η, and imposes CE thresholds on efficiency/ripple. Goal attainment uses the normalized error $|V_{\text{avg}} - V_{\text{out}}^{\star}|/|V_{\text{out}}^{\star}| \le \epsilon$.

Amplifier measurements. Amplifier tasks target a low-frequency gain $G^{\star}$, $-3\,\text{dB}$ bandwidth $f_{-3\text{dB}}^{\star}$, and bounds on phase margin / static power. We extract the AC transfer curve to measure $(G, f_{-3\text{dB}})$, compute phase margin from the open-loop response, and integrate quiescent current for P_{static}. Goal attainment requires both gain and bandwidth errors to be within tolerance (`pass_gain` & `pass_bw`); CE additionally enforces PM and P_{static} limits.

Oscillator measurements. Oscillator tasks provide a supply V_{DD} and target frequency $f^{\star}_{\text{osc}}$. The verifier runs a transient sim, checks convergence, and measures steady-state frequency, peak-to-peak amplitude A_{pp}, and static power. Goal attainment hinges on the relative frequency error; CE enforces amplitude and power envelopes. Summary statistics appear in Table S6.

1.5 Verifier definitions and continuous spec errors

DC–DC. For each operating point, we run transient simulation and compute V_{avg} in a fixed steady-state window. We define the point-spec relative error $e_{\text{spec}} = \frac{|V_{\text{avg}} - V^{\star}|}{\max(10^{-9}, |V^{\star}|)}$; goal attainment (AR) is determined by checking whether point-spec errors are within tolerance. We also report efficiency, ripple, and overshoot as additional quality/stability signals for Pareto ranking among feasible samples.

Amplifiers. We run AC sweeps, compute low-frequency gain G (dB), and extract $f_{-3\text{dB}}$ from the frequency response. We convert to linear ratio $A = 10^{G/20}$ and define relative errors on (A, f):

$$e_G = \frac{|A - A^{\star}|}{\max(10^{-9}, A^{\star})}, \quad e_f = \frac{|f_{-3\text{dB}} - f^{\star}|}{\max(10^{-9}, f^{\star})}, \quad e_{\text{spec}} = \tfrac{1}{2}(e_G + e_f). \tag{1}$$

Goal attainment requires both gain/bandwidth to satisfy tolerances (implemented as `pass_gain` & `pass_bw`). CE is defined by engineering constraints (e.g., `pass_pm` & `pass_p`); the corresponding continuous measurements or violation terms are used in Pareto comparison among attained samples.

Oscillators. We run transient simulation and estimate the dominant oscillation frequency f_{osc} via FFT peak detection, with steady-state oscillation checks (amplitude and convergence). The verifier returns the relative frequency error $e_f = \frac{|f_{\text{osc}} - f^{\star}|}{\max(10^{-9}, f^{\star})}$ and stability indicators; after goal attainment, these can be used for Pareto ranking and quality analysis.

1.6 Self-Improving logs and artifacts

Each verified sample records: (i) DSL validity, (ii) simulation completion, (iii) component count, (iv) spec error e_{spec}, (v) family-specific quality metrics. The evaluation pipeline writes per-task metric files (one JSON per operating point) and summary files, enabling downstream analyses such as operating-point heatmaps and ablations.

Supplementary Algorithm S1 MO-SIMIclosed-loop training (one round)

Require: Task set $\mathcal{T}$, guard subset $\mathcal{G}$, random spot-check subset $\mathcal{R}$, tolerance ϵ, initial reference model π_0, reference model π_{ref}, current model π_t, group size K

1: **(A) Self-Improving + simulation:** for each $\tau \in \mathcal{T}$, sample $\{x_i\}_{i=1}^{K} \sim \pi_t(\cdot \mid \tau)$ and simulate; obtain OK, AR, CE and quality metrics

2: **Restricted repair:** if a near-miss sample exists, apply the family-specific $\mathcal{R}_f$ only on its predeclared editable knobs, re-simulate, and add the repaired sample only if it flips AR to 1

3: **(B) Pareto ranking:** compute Pareto ranks only for feasible samples (OK $= 1$, AR $= 1$); use crowding distance to break ties within the same rank

4: **(C) PVPO:** construct within-task preference pairs (x^+, x^-) from Pareto ranks and optimize the preference objective (reference: π_{ref}) to obtain $\tilde{\pi}_t$

5: **(D) Safe-PPO:** initialize PPO from $\tilde{\pi}_t$; after each update, evaluate on $\mathcal{G}$ (hard gate) and spot-check on $\mathcal{R}$; rollback if guard-AR or spot-check AR decreases, otherwise accept the checkpoint

6: **return** Best checkpoint as π_{t+1}

Supplementary Table S4: Key training hyperparameters for DC–DC (Grid77).

Item	Value
Tolerance ϵ	0.01 (relative error of V_{out})
Minimum #components	20 (`INC` lines)
Self-Improving group size K	8
PVPO steps	200, followed in the DC–DC main run by 400 SFT-regularization steps
Safe-PPO steps	80 (guard evaluation + rollback)
PPO learning rate	5×10^{-6}
PPO target KL	0.03
PPO clip ϵ	0.2
Max generation tokens	320

1.7 Closed-loop pseudo-code

1.8 Training hyperparameters

1.9 Family-level metric decomposition (subclasses)

Unified summary table. For unified comparison across three families (10 subclasses), Table S6 summarizes the raw open-loop metrics before restricted repair, including sample-level (duplicates counted) OK/AR/CE, diversity (overall dedup ratio), and the paired Acc values before/after restricted repair. We use 1% tolerance for DC–DC (AR@1%) and 5% for amplifiers/oscillators (AR@5%). In the table, AR is the sample-level attainment rate (duplicates counted), the unique ratio is the canonical-hash dedup ratio **regardless of passing**, and Acc is reported both before/after the same restricted-repair rule summarized in Supplementary Table S7.

Supplementary Table S5: Pilot rationale for the Safe-PPO simulation batch layout.

Layout	Observed trade-off	Decision
4×8	Fewer simulator calls per PPO step, but poorer simulator utilization and noisier model-update estimates.	Rejected
8×8	Balanced simulator throughput, update stability, and guard-check latency under the available GPU/SPICE workers.	Used
16×8	Larger per-step budget and longer wall-clock latency; pilot checks did not show a guard-AR benefit over 8×8.	Rejected

Supplementary Table S6: Raw open-loop evaluation summary across three families (10 subclasses), with sample-level counts including duplicates.

Cat.	Sub.	#T	#S	OK ↑	AR ↑	CE ↑	$\mathrm{Acc}_{\mathrm{raw}}$ ↑	$\mathrm{Acc}_{\mathrm{rep}}$ ↑	Uniq.↑	Rep.%	Succ.%
DC–DC	Overall	77	770	1.0000	0.9234	0.8247	0.9886	0.9915	0.5247	0.0766	0.7119
DC–DC	buck	26	260	1.0000	0.8808	0.7615	0.9898	0.9964	0.5769	0.1192	1.0000
DC–DC	boost	17	170	1.0000	0.8529	0.7176	0.9737	0.9747	0.7353	0.1471	0.3200
DC–DC	SEPIC	17	170	1.0000	0.9941	0.9235	0.9933	0.9948	0.6412	0.0059	1.0000
DC–DC	buck–boost	17	170	1.0000	0.9882	0.9294	0.9968	0.9976	0.1176	0.0118	1.0000
Amplifier	Overall	10	100	0.6800	0.5400	0.6800	0.7698	0.8412	0.7900	0.3200	1.0000
Amplifier	OpTwo	5	50	0.7000	0.5200	0.7000	0.7090	0.7939	0.9600	0.3000	1.0000
Amplifier	RFPA	5	50	0.6600	0.5600	0.6600	0.8342	0.8885	0.6200	0.3400	1.0000
Oscillator	Overall	20	200	0.7650	0.4900	0.7650	0.7029	0.9513	1.0000	0.5200	0.9038
Oscillator	LC	5	50	0.8000	0.3800	0.8000	0.6296	0.9941	1.0000	0.6800	1.0000
Oscillator	RC	5	50	0.6400	0.5000	0.6400	0.8274	0.9506	1.0000	0.3000	0.8667
Oscillator	Ring	5	50	0.6600	0.3000	0.6600	0.5300	0.8905	1.0000	0.7800	0.8462
Oscillator	Wien	5	50	0.9600	0.7800	0.9600	0.8207	0.9706	1.0000	0.3200	0.8750

Family-specific sub-metrics (split out from Table S6).

Category	Subclass	Family-specific sub-metrics (sample-level with duplicates; DC–DC quality metrics on hit)
DC–DC	Overall	$\bar{\eta}_{\text{hit}}$=0.8453; ripple=79.01 mV; over=4.95 mV; #elems=28.02
DC–DC	buck	$\bar{\eta}_{\text{hit}}$=0.8604; ripple=2.01 mV; over=2.68 mV; #elems=27.99
DC–DC	boost	$\bar{\eta}_{\text{hit}}$=0.8332; ripple=249.19 mV; over=8.71 mV; #elems=28.21
DC–DC	SEPIC	$\bar{\eta}_{\text{hit}}$=0.8391; ripple=33.32 mV; over=5.35 mV; #elems=27.94
DC–DC	buck–boost	$\bar{\eta}_{\text{hit}}$=0.8413; ripple=84.02 mV; over=4.44 mV; #elems=28.00
Amplifier	Overall	pass_gain=0.2800; pass_bw=0.0500; pass_pm=0.5900; pass_p=0.6800
Amplifier	OpTwo	pass_gain=0.0000; pass_bw=0.1000; pass_pm=0.5200; pass_p=0.7000
Amplifier	RFPA	pass_gain=0.5600; pass_bw=0.0000; pass_pm=0.6600; pass_p=0.6600
Oscillator	Overall	pass_freq=0.3550; pass_vpp=0.7400; pass_p=0.7650
Oscillator	LC	pass_freq=0.2000; pass_vpp=0.7400; pass_p=0.8000
Oscillator	RC	pass_freq=0.5000; pass_vpp=0.6400; pass_p=0.6400
Oscillator	Ring	pass_freq=0.0400; pass_vpp=0.6200; pass_p=0.6600
Oscillator	Wien	pass_freq=0.6800; pass_vpp=0.9600; pass_p=0.9600

Detailed continuous metrics across 10 subclasses (quality metrics).

Amplifier low-level metrics (example). Besides binary pass flags, the amplifier verifier logs low-level measurements (gain, bandwidth, phase margin, static power, etc.), which together determine **goal attainment**/CE: goal attainment is determined by point-spec errors of gain and bandwidth (`pass_gain` & `pass_bw`), while CE is determined by engineering constraints such as stability and power (`pass_pm` & `pass_p`).

Oscillators: four topologies. Oscillators include LC/RC/Ring/Wien topologies; open-loop metrics are unified in Table S6.

1.10 Runtime Environment and Software Versions

Hardware. Training and evaluation are conducted on Ubuntu 22.04.5 LTS cloud servers: CPU is 2× Intel Xeon Platinum 8470Q (52 cores/socket; 104 physical cores, 208 threads in total) with 1 TiB memory. PVPO and Safe-PPO use multi-GPU data parallelism (8 GPUs); simulation and verification are mainly executed by a CPU worker pool in parallel.

Software. We use Python 3.12.3 and PyTorch 2.7.0 (CUDA 12.8 / cuDNN 9.7) for training; the circuit simulator is ngspice-36. The `.tex` sources in this paper compile under XeLaTeX (TeX Live 2024).

1.11 Distributed learning and parallel simulation

Parallel verification. The main cost of training and evaluation comes from simulation. We shard verification across a CPU worker pool, cache compiled artifacts as much as possible, and batch simulator calls to reduce per-sample overhead. Guard evaluation uses a fixed stratified subset to control variance while maintaining family/operating-range coverage.

Multi-GPU learning. PVPO and Safe-PPO run in data-parallel fashion on multiple GPUs and use gradient accumulation for stable effective batch sizes. Acceptance rules (guard non-degradation) are independent of hardware scale: a checkpoint is accepted only if both guard attainment and random spot-check attainment do not decrease; if both attainment checks tie, we break ties by guard CE.

1.12 Duty-cycle adaptation (DC–DC)

Motivation. In DC–DC converters, duty cycle acts as a continuous knob to tune V_{out} under an ideal-switch model. However, with non-ideal switch/diode models and parasitics, analytic mappings break down, and transient-waveform verification remains necessary to satisfy point targets.

Usage. We allow an *optional* duty-cycle adapter to adjust duty within a bounded range to reduce e_{spec} for a given generated netlist. This is not template fallback: circuit structure (`INC` lines) remains unchanged, and candidates must still satisfy all DSL constraints and pass transient simulation. Evaluation artifacts may optionally include both raw and restricted-repair summaries for sanity checks; in the current manuscript, the cross-family main comparison in the main paper uses the same restricted-repair setting described here, while the DC–DC baseline/ablation table remains raw (open-loop). Supplementary Table S7 details the corresponding restricted-repair results under the same generation budget and shows how limited additional simulator calls can flip some near-miss failures into successes.

1.13 Artifact format examples

Per-task metric files. For each operating point, we write a JSON metric file containing the task definition (family and targets), per-sample records (DSL validity, simulation

status, component count, measurements, and pass flags), and task-level aggregations (e.g., pass@k).

Summary files. We also write a summary JSON that reports overall sample/pass@k rates, family-level breakdowns, and consistency checks (e.g., minimum component constraints, no-template-fallback rate). These artifacts enable regenerating all figures and tables in the paper from logs.

1.14 Guard set construction and hard-point mining

Guard set. The guard set $\mathcal{G}$ is a fixed subset of tasks used to (i) reduce evaluation variance during RL and (ii) enforce the non-degradation acceptance rule. We stratify-sample $\mathcal{G}$ by circuit family and operating ranges and keep it fixed within a training run. Guard evaluation uses the same feasibility/quality metrics as full evaluation but with fewer tasks and fewer samples.

Hard-point mining. After feasibility improves, later PPO batches can be dominated by easier operating points. To focus learning on bottlenecks, we maintain a moving feasibility estimate for each task and form a family-specific hard-task pool from guard evaluation: tasks whose guard pass rate falls below 0.9 are added to the pool. During Safe-PPO, each 8-task PPO batch draws from the corresponding hard-task pool with probability 0.5 and otherwise from the ordinary family task pool. This mechanism changes only training-time Safe-PPO sampling and does not alter evaluation protocols.

1.15 DC–DC: a two-parameter restricted-repair example

We provide a real sample from training-stage restricted repair: in a Boost task ($V_{\text{in}} = 12\text{V}$, $V_{\text{out}}^{\star} = 18\text{V}$, $\epsilon = 1\%$), the pre-repair netlist is simulatable but fails `pass_CV`; the restricted repair rule tries bounded scalings of existing passive values, and one successful edit scales L_1 by 0.5 and C_1 by 2.0, modifying only two passive-component values to flip `pass_CV` from False to True (the rest of the netlist remains unchanged).

```
# before (x): pass_CV = False
INC L1 vin sw 47u
INC S1 sw 0 Sstd
INC D1 sw out Dstd
INC C1 out 0 47u
INC C2 vin 0 47u
INC R1 sw nsn1 5
INC C3 nsn1 0 0.1
```

```
INC R2 sw nsn2 10
INC C4 nsn2 0 1
INC R3 out nd1 0.2
INC C5 nd1 0 0.47
INC R4 out nd2 0.5
INC C6 nd2 0 1
INC R5 out nd3 0.2
INC C7 nd3 0 0.47
INC R6 out nd4 0.5
INC C8 nd4 0 1
INC R7 out nd5 0.2
INC C9 nd5 0 0.47
INC R8 out nd6 0.5
INC C10 nd6 0 1
INC R9 out nd7 0.2
INC C11 nd7 0 0.47
INC R10 out nd8 0.5
INC C12 nd8 0 1
INC R11 out nd9 0.2
INC C13 nd9 0 0.47
INC R12 out nd10 0.5

# after (x'): pass_CV = True  (restricted repair: L1*0.5, C1*2.0)
INC L1 vin sw 2.35e-05
INC S1 sw 0 Sstd
INC D1 sw out Dstd
INC C1 out 0 9.4e-05
INC C2 vin 0 47u
INC R1 sw nsn1 5
INC C3 nsn1 0 0.1
INC R2 sw nsn2 10
INC C4 nsn2 0 1
INC R3 out nd1 0.2
INC C5 nd1 0 0.47
INC R4 out nd2 0.5
INC C6 nd2 0 1
INC R5 out nd3 0.2
INC C7 nd3 0 0.47
INC R6 out nd4 0.5
```

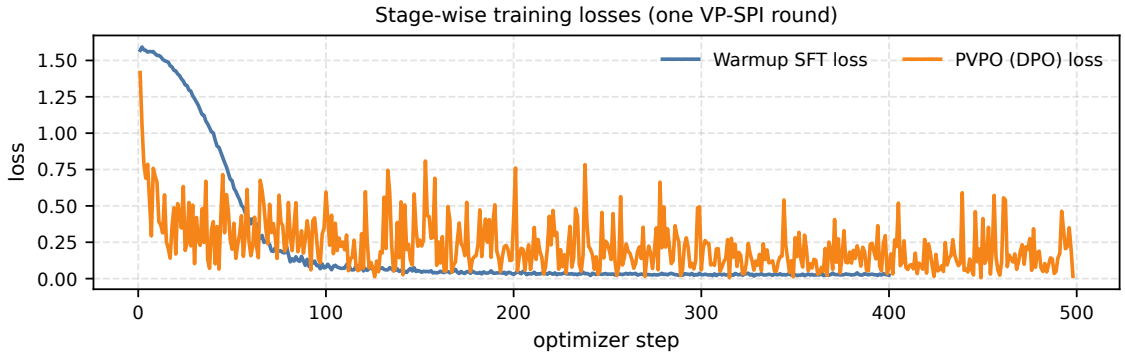
```
INC C8 nd4 0 1
INC R7 out nd5 0.2
INC C9 nd5 0 0.47
INC R8 out nd6 0.5
INC C10 nd6 0 1
INC R9 out nd7 0.2
INC C11 nd7 0 0.47
INC R10 out nd8 0.5
INC C12 nd8 0 1
INC R11 out nd9 0.2
INC C13 nd9 0 0.47
INC R12 out nd10 0.5
```

1.16 PVPO and Safe-PPO details

PVPO construction. Within each task group, we construct preference pairs only when the chosen sample **attains the goal** and dominates (or is close to the Pareto front compared with) weaker candidates. For the DC–DC main run, we first perform 200 PVPO updates on the current round's valid preference pairs and then append a separate 400-step SFT-regularization phase on the same Xyce-curated SFT corpus before Safe-PPO starts. This is a short standalone regularization stage rather than a fixed-percentage SFT mixture inside every PVPO mini-batch, and its only purpose is to keep syntax and family coverage stable.

Pareto objectives (DC–DC). For DC–DC, the simulated metric vector includes spec error e_{spec}, efficiency, stability proxies (ripple and overshoot), and structural constraints (component count). We apply Pareto ranking only among feasible (goal-attaining) samples and treat infeasible samples as strictly worse, avoiding "normalizing failures into positive rewards". When multiple non-dominated feasible candidates exist, we break ties by a diversity term (crowding distance) to avoid collapsing into a narrow design family.

Safe-PPO. We run PPO under KL control with a reference model, but gate updates using guard evaluation. Each step evaluates the current model on a fixed guard set $\mathcal{G}$ and a random spot-check set $\mathcal{R}$, and accepts a checkpoint only if neither attainment value decreases; otherwise we rollback to the best checkpoint and increase conservativeness (e.g., larger KL penalties / smaller updates). This avoids common "RL degradation": losing rare feasible behaviors while optimizing noisy proxies.



Supplementary Figure S1: Stage-loss curves from the DC–DC main training run. These curves accompany the Safe-PPO acceptance traces reported in the main paper and show the stage-wise optimization behavior recorded during the protected training loop.

Supplementary Table S7: Restricted-repair summary under the same generation budget (sample-level with duplicates counted).

Family	Setting	OK ↑	Acc ↑	AR@1% ↑	AR@5% ↑	CE ↑
DC–DC (Overall)	raw→ restricted repair	1.0000	0.9886 → 0.9915	0.9234 → 0.9779	0.9610 → 0.9779	0.8247 → 0.8247
buck	raw→ restricted repair	1.0000	0.9898 → 0.9964	0.8808 → 1.0000	0.9577 → 1.0000	0.7615
boost	raw→ restricted repair	1.0000	0.9737 → 0.9747	0.8529 → 0.9000	0.9059 → 0.9000	0.7176
SEPIC	raw→ restricted repair	1.0000	0.9933 → 0.9948	0.9941 → 1.0000	0.9941 → 1.0000	0.9235
buck–boost	raw→ restricted repair	1.0000	0.9968 → 0.9976	0.9882 → 1.0000	0.9882 → 1.0000	0.9294
Amplifier (Overall)	raw→ restricted repair	0.6800 → 1.0000	0.7698 → 0.8412	0.0000 → 0.3200	0.5400 → 0.8600	0.6800 → 1.0000
Two-stage amp	raw→ restricted repair	0.7000 → 1.0000	0.7090 → 0.7939	0.0000 → 0.3000	0.5200 → 0.8200	0.7000 → 1.0000
RF amp	raw→ restricted repair	0.6600 → 1.0000	0.8342 → 0.8885	0.0000 → 0.3400	0.5600 → 0.9000	0.6600 → 1.0000
Oscillator (Overall)	raw→ restricted repair	0.9350 → 0.9950	0.7029 → 0.9513	0.2300 → 0.6950	0.4800 → 0.9500	0.9300 → 0.9900
LC	raw→ restricted repair	1.0000 → 1.0000	0.6296 → 0.9941	0.1800 → 0.8600	0.3200 → 1.0000	1.0000 → 1.0000
RC	raw→ restricted repair	0.9400 → 1.0000	0.8274 → 0.9506	0.3600 → 0.6200	0.7000 → 0.9600	0.9200 → 0.9800
Wien	raw→ restricted repair	0.9200 → 0.9800	0.8207 → 0.9706	0.3800 → 0.6400	0.6800 → 0.9600	0.9200 → 0.9800
Ring	raw→ restricted repair	0.8800 → 1.0000	0.5300 → 0.8905	0.0000 → 0.6600	0.2200 → 0.8800	0.8800 → 1.0000

1.17 Restricted repair summary under the same generation budget

This section summarizes the setup and results of the same restricted-repair rule used for the cross-family main summary in the main paper, and provides the corresponding family/subclass breakdowns. The goal here is to report the resulting repairability and cost boundaries under a fixed generation budget, rather than to introduce an additional training-stage component beyond the loop already described in the main paper.