RL-QESA: Reinforcement-Learning Quasi-Equilibrium Simulated Annealing

Ruichen Xu [1] **Kai Li** [1] **Haochun Wang** [1] **Georgios Kementzidis** [1] **Wei Zhu** [1] **Yuefan Deng** [1]

Abstract

We present **RL-QESA**—Reinforcement-Learning Quasi-Equilibrium Simulated Annealing—a new framework that couples classical simulated annealing (SA) with an adaptive, learning-based cooling schedule. A policy network observes block-level statistics and lowers the temperature T_{n+1} *only when* the empirical energy moments at T_n coincide with their quasi-equilibrium predictions, certifying that the sampler has fully explored the current thermal state before cooling further. We show that RL-QESA inherits SA's classical convergence guarantees while permitting far richer cooling profiles than hand-crafted schedules. On the Rosenbrock function and Lennard–Jones cluster benchmarks, RL-QESA attains up to three-fold faster convergence and consistently lower terminal energies compared with vanilla SA and recent neural variants. By automating temperature descent in a principled, quasi-equilibrium fashion and retaining simple proposal mechanics, RL-QESA offers a robust, learning-driven optimiser for challenging global optimisation tasks.

1. Introduction

SA is a foundational metaheuristic for global optimisation, tracing its origins to the physical annealing process that gradually cools a material so it can settle into a low–energy state (Kirkpatrick et al., 1983; van Laarhoven & Aarts, 1987). By occasionally accepting *uphill* moves, SA enables trajectories to escape local minima and explore rugged energy landscapes, making it a workhorse for both continuous and combinatorial problems ranging from VLSI layout to molecular design.

Despite its elegance, SA's real-world performance is notoriously sensitive to *(i)* the *cooling schedule* and *(ii)* the *move-generation* mechanism (Suman & Kumar, 2006; Ingber, 1993). Classical recipes fix a hand-designed temperature decay and couple it with a Gaussian proposal whose variance is tuned by trial-and-error. Deviations in either component can cripple exploration or waste computation, and—more crucially—invalidate the theoretical convergence guarantees that make SA attractive in the first place (Eglese, 1990; Yang, 2008).

Our central insight is that the chain should *cool only after it has demonstrably equilibrated* at the current temperature. Let T_n be the temperature during block n. Denote by $\mu(T_n)$ and $\sigma(T_n)$ the equilibrium mean and standard deviation of the energy at T_n, and let $\langle f\rangle_{T_n}$ be the empirical mean collected from the k Metropolis–Hastings moves in that block. We declare block n to be in *quasi-equilibrium* when the mean discrepancy falls below a tolerance $\lambda > 0$:

$$\left|\mu(T_n) - \langle f\rangle_{T_n}\right| \;\le\; \lambda\,\sigma(T_n). \qquad (1)$$

Whenever condition (1) holds, our RL controller is permitted to lower the temperature; otherwise it continues sampling at the current T_n until sufficient mixing is observed. This statistics-driven gate guarantees that every cooling step respects the assumptions underlying SA's classical convergence proofs (Geman & Geman, 1984; Hajek, 1988; Granville et al., 1994), while leaving the Metropolis–Hastings proposal mechanism itself unchanged.

We translate the quasi-equilibrium criterion into action with **RL-QESA**—Reinforcement-Learning Quasi-Equilibrium Simulated Annealing. A *single* lightweight RL agent governs the cooling schedule: it observes block-level statistics and lowers the temperature only when condition (1) is met. Move proposals, by contrast, follow the *classical* Metropolis–Hastings kernel with a fixed variance, so no extra proposal agent or step-size tuning is required. Because the proposal scale is held constant, temperature control is completely decoupled from move generation—eliminating a common source of manual engineering. The cooling policy is trained with *Proximal Policy Optimisation* (PPO), producing an adaptive schedule that remains faithful to SA theory yet flexibly adapts to the energy landscape.

This work makes three key contributions:

- We formulate SA as a block-level MDP that enforces

[1] Department of Applied Mathematics and Statistics, Stony Brook University, Stony Brook, NY, USA. Correspondence to: Yuefan Deng <yuefan.deng@stonybrook.edu>.

The second AI for MATH Workshop at the 42nd International Conference on Machine Learning, Vancouver, Canada

a quasi-equilibrium check, and realise it with the learning-driven RL-QESA framework.

- We prove that RL-QESA inherits the classical convergence guarantees of logarithmic-cooling SA while admitting much richer, data-dependent schedules.
- Experiments on the Rosenbrock function and Lennard–Jones clusters show up to $\mathbf{3}\times$ faster convergence and lower final energies than vanilla SA and recent neural variants—all without step-size tuning.

From an *AI4Math* perspective, RL-QESA showcases a tight coupling between analytic rigour and learning: the quasi-equilibrium constraint, equipped with Lipschitz control, yields a closed-form drop bound $\big($cf. (2)$\big)$ that is embedded directly into the reward. A small transformer controller is trained *on-line* to drive the temperature towards the analytically optimal update $T^{\star}_{n+1}$, effectively saturating the bound at each step and thereby achieving near-optimal cooling speed without sacrificing convergence guarantees.

By embedding quasi-equilibrium monitoring into an RL feedback-loop and retaining the simplicity of a fixed-variance proposal, RL-QESA bridges the gap between principled SA theory and the flexibility demanded by modern optimisation challenges.

2. Background and Related Work

Non-convex optimization poses significant challenges in scientific computing and machine learning, where the objective function $f(\boldsymbol{x})$ may contain multiple local minima or saddle points. Formally, one seeks

$$\boldsymbol{x}^* \;=\; \arg\min_{\boldsymbol{x}\in\mathcal{X}}\; f(\boldsymbol{x}),$$

where $\mathcal{X} \subseteq \mathbb{R}^n$ is the feasible region and f lacks convexity. Classical gradient-based methods often fail to traverse these rugged landscapes fully, leading to premature convergence to local minima. Thus, global optimization heuristics like SA are frequently employed to mitigate these issues.

SA, illustrated in Algorithm 1, is a stochastic global-search metaheuristic inspired by physical annealing processes (Kirkpatrick et al., 1983; Ingber, 1993).

At each iteration, SA proposes a candidate $\boldsymbol{x}_{t+1}$ from $\boldsymbol{x}_t + \Delta\boldsymbol{x}_t$, accepting it according to the Metropolis-Hastings criterion:

$$\mathbf{P}\big(\boldsymbol{x}_{t+1} \mid \boldsymbol{x}_t, T_t\big) \;=\; \min\Big\{\exp\Big(-\frac{f(\boldsymbol{x}_{t+1}) - f(\boldsymbol{x}_t)}{T_t}\Big), 1\Big\},$$

where T_t is the system temperature at time t. As $T_t \to 0$ over many iterations, the distribution of states concentrates on the set of global optima with high probability (Geman & Geman, 1984; Hajek, 1988; Granville et al., 1994). However, achieving effective exploration heavily depends on two key factors: (i) the *cooling schedule* or how T_t decreases, and (ii) the *perturbation distribution*, which determines how $\Delta\boldsymbol{x}_t$ is sampled. Figure 1 shows that several schedules reaching the same terminal temperature can differ by more than an order of magnitude in relative error if the chain is cooled too quickly (panels b–c).

Algorithm 1 Simulated Annealing (SA)

```
Input: initial point $\mathbf{X}_0$; initial temperature $T_1$; cooling rule $C$; proposal kernel $M$; outer iterations $B$; inner iterations $k$
Output: approximate minimiser $\mathbf{X}_B^k$
for $b \leftarrow 1$ to $B$ do                                   ▷ outer loop
    for $i \leftarrow 1$ to $k$ do                               ▷ inner loop
        draw $\mathbf{X}_b^i \sim M(\cdot \mid \mathbf{X}_b^{i-1}, T_b)$;
        $\alpha \leftarrow \exp\Big(-\frac{f(\mathbf{X}_b^i)-f(\mathbf{X}_b^{i-1})}{T_b}\Big)$;
        accept $\mathbf{X}_b^i$ w.p. $\min\{\alpha, 1\}$;
        otherwise set $\mathbf{X}_b^i \leftarrow \mathbf{X}_b^{i-1}$;
    $T_{b+1} \leftarrow C(T_b)$;
return $\mathbf{X}_B^k$
```

Prior work on learning-based annealing typically drives the reinforcement-learning (RL) agent with a reward that depends only on the *final* energy, implicitly assuming that every intermediate temperature block has reached near-equilibrium (Correia et al., 2023). *Adaptive Simulated Annealing* (ASA) introduces online step-size adjustment, yet still follows a fixed—or at best heuristic—temperature curve and therefore cannot guarantee equilibrium between successive blocks (Ingber, 1996). These observations motivate our focus on adaptive, RL-driven cooling policies that explicitly monitor convergence diagnostics rather than relying solely on the final objective value.

Enhancing SA with Machine Learning. A growing body of work integrates machine-learning techniques into SA to *automate or refine temperature schedules, move magnitudes, and acceptance rules*. In particular, reinforcement learning (RL) has been employed to adjust acceptance thresholds or move parameters by framing SA as a Markov-decision process (Cai et al., 2019; Vashisht et al., 2020; Li et al., 2023; Elgammal et al., 2021; Correia et al., 2023). Such RL-based schemes replace manual heuristics with data-driven strategies, improving adaptability across diverse problem instances.

Neural-network integrations. Complementary to these RL approaches, several recent papers weave the "heat–explore, cool–exploit" logic of SA *inside* neural models themselves:

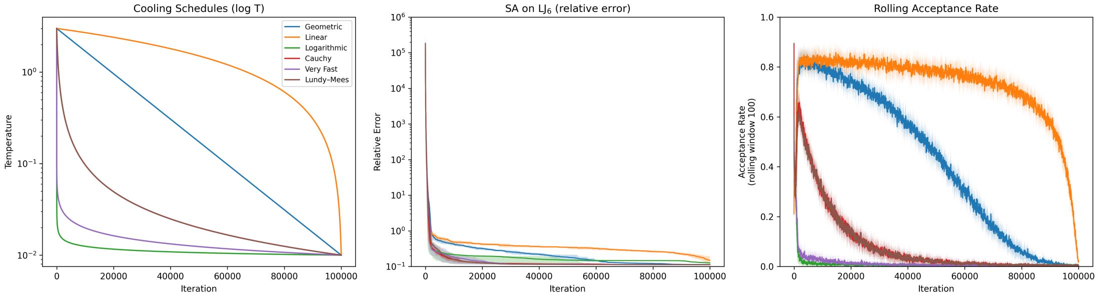

Figure 1. **Impact of cooling schedules on simulated–annealing performance for the 6-atom Lennard–Jones cluster.** *Left, panel (a):* six classical schedules—geometric, linear, logarithmic (Boltzmann), Cauchy/fast-SA, very-fast SA, and Lundy–Mees—are matched to the same starting temperature $T_0 = 3$ and final temperature $T_{\text{end}} = 10^{-2}$. *Centre, panel (b):* mean relative error $|E_{\text{best}} - E_\star|/|E_\star|$ over 20 independent runs (shaded band = 20th–80th percentiles after outlier removal). Schedules that cool too aggressively (Cauchy, very-fast) freeze prematurely, whereas slow cooling (linear) wastes computation; geometric and Lundy–Mees achieve the lowest errors. *Right, panel (c):* rolling 100-step acceptance rates clarify the sampling dynamics: abrupt schedules quench the chain almost immediately, while smoother schedules prolong effective exploration. Together, the panels demonstrate that *temperature policy alone can induce orders-of-magnitude differences in optimisation quality and sampling behaviour*, underscoring the importance of principled, adaptive cooling.

1. **Model-level cooling (SICNN).** Chen et al. (2024) embed a sparsity-penalised temperature term into an Input-Convex Neural Network for optimal transport. The penalty is annealed so the model first explores a wide solution space, then gradually focuses for sharper transport maps.

2. **Hyper-parameter search (SA-CNN).** Guo & Cao (2022) treat filter sizes, learning rates, and other CNN hyper-parameters as an annealing state, letting SA stochastically explore that space and boost text-categorisation accuracy without manual tuning.

3. **Optimiser-level fusion (SA-GD).** Cai (2021) inject temperature-controlled probabilistic jumps into each gradient-descent step, enabling escape from sharp local minima and thus better generalisation.

Our RL-guided framework is *complementary*: it learns *when* to fire these SA moves and *how fast* to cool. Hence it can act as a plug-in decision layer for architectures such as SICNN or SA-CNN while preserving their task-specific advantages.

Quasi-Equilibrium and Effective Sampling. Classical convergence proofs for SA assume that, at every temperature T_n, the Markov chain has reached (near) equilibrium in the Boltzmann distribution $\pi_{T_n}(\boldsymbol{x}) \propto \exp[-f(\boldsymbol{x})/T_n]$ before the system is cooled further (Geman & Geman, 1984; Hajek, 1988; Granville et al., 1994). Empirical work soon showed that premature cooling leads to metastability, inspiring "equilibrium tests" based on energy-histogram overlap (Aarts & Korst, 1985) and specific-heat plateaus (Romeo & Sangiovanni-Vincentelli, 1991); in contrast, Xu et al. (2025) achieve equilibrium by *designing the move-generation kernel itself* to satisfy detailed balance at each temperature, removing the need for such diagnostic checks.

Rapid equilibration also hinges on the efficiency of the proposal kernel. Optimal-scaling theory for random-walk Metropolis shows that an acceptance probability close to 0.234 maximises expected squared-jump distance as dimension grows (Roberts & Rosenthal, 1997; 2001). Adaptive SA variants therefore rescale step sizes online to steer the empirical acceptance rate toward this target, achieving provable ergodicity while removing manual tuning (Atchadé & Liu, 2010; Ingber, 1996).

Building on these insights, we introduce a *single RL controller* that governs the temperature schedule: cooling is postponed until equilibrium diagnostics—stabilised energy variance, sufficient effective sample size, and (optionally) replica-swap statistics—meet calibrated thresholds. This sole "temperature agent" keeps the chain in quasi-equilibrium at every thermal stage, thereby accelerating traversal of rugged landscapes while retaining SA's classical global-optimality guarantees. Although the present work keeps the Metropolis–Hastings proposal kernel fixed, we outline how the framework naturally extends to learning adaptive proposal distributions, results of which will be reported in forthcoming work.

MDP. Formally, an MDP is a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, P, \gamma)$, with $\mathcal{S}$ as a state set, $\mathcal{A}$ as an action set, $\mathcal{R}$ as an immediate reward function set, P as a transition kernel, and $\gamma \in [0, 1)$ as a discount factor (Bellman, 1957; Puterman, 1994; Sutton & Barto, 1998). A policy $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ is optimized to maximize the expected discounted return. In the SA setting, states might include the current solution $\boldsymbol{x}$, the temperature T, and acceptance or energy statistics, while actions define how to perturb $\boldsymbol{x}$ or update T. Rewards can reflect improved sample quality or equilibrium adherence.

Neural Simulated Annealing (N-SA). Earlier efforts at "Neural SA" have embedded neural networks to learn the distribution used for candidate proposals or temperature schedules (Correia et al., 2023). Although these methods enhance adaptive capabilities, they often concentrate on final cost objectives and do not explicitly enforce local equilibrium conditions. By contrast, our design unites equilibrium checks with RL-driven scheduling and a Metropolis-Hastings approach to move generation. This enables automated, quasi-equilibrium SA that retains the algorithm's classical global properties while reducing the manual tuning burden.

In the next section, we detail our methodology, describing how multi-agent RL separately handles local move updates and global temperature management to sustain near-equilibrium conditions at each thermal stage.

3. Theory

This section formalises the *quasi-equilibrium* (QE) principle that underpins **RL-QESA** and proves that the resulting algorithm inherits the global–optimisation guarantees of classical SA. We first introduce the QE condition and the associated convergence theorem, then derive an explicit *cooling-rate design rule* that lets the agent cool *as fast as possible* while still satisfying QE. Finally, we discuss how an off-line choice of proposal variance maximises intra-block mixing efficiency.

3.1. Quasi-Equilibrium Criterion and Convergence Guarantees

Let $T_1 > T_2 > \cdots$ be the (unknown) temperature sequence generated by a continuous, monotone *decreasing* cooling function $C : [0, \infty) \to [0, \infty)$:

$$T_{n+1} \;=\; C\bigl(T_n\bigr), \qquad \sum_{n=1}^{\infty} T_n \;=\; \infty, \quad T_n \;\longrightarrow\; 0.$$

Within block n the sampler performs k Metropolis–Hastings (MH) steps at temperature T_n using a *fixed* Gaussian proposal variance σ^2_{prop}.

QE condition. Let $\mu(T_n)$ and $\sigma(T_n)$ denote the Boltzmann mean and standard deviation of the energy at temperature T_n. Let $\langle f\rangle_{T_n}$ be the empirical mean energy collected during the k MH moves in block n. For a tolerance parameter $\lambda > 0$, block n satisfies QE if

$$\bigl|\mu(T_n) - \langle f\rangle_{T_n}\bigr| \;\le\; \lambda\,\sigma(T_n). \tag{1}$$

RL-QESA is allowed to transition from T_n to T_{n+1} *only after* (1) holds, guaranteeing sufficient mixing at each temperature.

Theorem 3.1 (Almost-sure convergence under QE)**.** *Assume that*

(i) the MH kernel with fixed variance is irreducible and aperiodic at every T_n;

(ii) the cooling function C is monotone decreasing, with $T_n \to 0$ and $\sum_n T_n = \infty$; and

(iii) every block satisfies the QE condition (1).

Then, with probability 1, *the state trajectory visits every global minimiser of f infinitely often and converges almost surely to the global minimum level set.*

Proof sketch. The proof adapts the supermartingale framework of Geman & Geman (1984), Hajek (1988), and Granville et al. (1994). QE guarantees that each temperature drop occurs only after the chain is $\mathcal{O}(\lambda)$-close (in mean energy) to the Boltzmann law at T_n. Combined with the logarithmic tail $\sum_n T_n = \infty$, this yields the same probabilistic bounds on escaping local minima as in standard SA.

3.2. A Fast – Yet Safe – Cooling-Rate Design

Our goal is to *cool as aggressively as possible* while still respecting the quasi-equilibrium constraint (1). The key observation is that the admissible temperature drop depends on the *smoothness* of the Boltzmann mean energy $\mu(T)$, a purely mathematical property of the underlying optimisation landscape. Below we derive an explicit upper bound on $\Delta T_n := T_n - T_{n+1}$ and then show how RL-QESA learns, in an *on-line* fashion, to approximate that bound without ever violating QE. This bridges the mathematical design rule with the data-driven policy that the AI4Math workshop emphasises.

Assumption 3.2 (Lipschitz continuity of μ)**.** There exists $L > 0$ such that $|\mu(T) - \mu(T')| \le L\,|T - T'|$ for all $T, T' > 0$.

Assumption 3.2 holds whenever $f(\mathbf{x})$ has finite second moments under all Boltzmann distributions, a mild condition satisfied by most bounded-below energies.

Lemma 3.3 (Maximum admissible temperature drop)**.** *Let $\varepsilon_n := \bigl|\langle f\rangle_{T_n} - \mu(T_n)\bigr|$ be the sampling error after k MH moves at T_n. Under Assumption 3.2, any temperature update obeying*

$$\Delta T_n \;\le\; \frac{\lambda\,\sigma(T_n) - \varepsilon_n}{L} \tag{2}$$

guarantees that block $n{+}1$ will start within the QE tolerance (1).

Corollary 3.4 (Fastest monotone schedule)**.** *Define the* one-step optimum

$$T^{\star}_{n+1} \;:=\; \max\Big\{\, T < T_n : T_n - T \le \frac{\lambda\sigma(T_n) - \varepsilon_n}{L} \Big\}.$$

The sequence $\{T^{\star}_n\}$ *is the* fastest *monotone decreasing schedule that never violates quasi-equilibrium.*

Lemma 3.3 couples two quantities: (a) the *sampling error* $\varepsilon_n = \mathcal{O}(k^{-1/2})$, observable on-line; and (b) the local Lipschitz constant L, generally *unknown* in black-box optimisation. Instead of estimating L explicitly, RL-QESA trains a policy $\pi_\theta : \mathcal{S} \to (0,1)$ that maps the current block statistics $\mathcal{S}_n = (T_n, \langle f\rangle_{T_n}, \varepsilon_n, \dots)$ to a *cooling coefficient* $\alpha_{n+1} = \pi_\theta(\mathcal{S}_n)$, and sets $T_{n+1} = \alpha_{n+1} T_n$. The policy is rewarded according to

$$R_n \;=\; \underbrace{-\big(\alpha_{n+1} - 1\big)\,T_n}_{\text{encourage fast cooling}} \;-\; \underbrace{\beta\,\mathbb{1}_{\{\text{QE violated}\}}}_{\text{penalise breach of (1)}}\;.$$

where $\beta \gg T_1$ is a large constant. This reward encodes exactly the optimisation problem of Corollary 3.4: *maximise* the temperature drop subject to the QE constraint. Proximal Policy Optimisation (PPO) updates (Schulman et al., 2017) adjust θ so that $\alpha_{n+1} \approx 1 - \frac{\lambda\sigma(T_n) - \varepsilon_n}{L T_n}$, thereby yielding $T_{n+1} \approx T^{\star}_{n+1}$ without knowing L in advance.

Because $\varepsilon_n = \Theta(k^{-1/2})$, inequality (2) shows that doubling k increases the admissible temperature drop by a factor of $\sqrt{2}$. Hence, the RL agent faces a natural exploration–exploitation trade-off: invest more MH steps (larger k) to reduce ε_n *or* accept a smaller drop ΔT_n. PPO implicitly learns this trade-off by observing returns accumulated over many training episodes.

4. Method

Subsection 4.1 casts SA as a *single-agent* MDP: the agent chooses the next temperature T_{n+1} while the Metropolis–Hastings proposals keep a *fixed* variance. Subsection 4.2 describes the **Transformer** policy that makes those cooling decisions, using attention over past block-level statistics. Together, they explain *what* we control (temperature) and *how* we learn it, setting the stage for Section 5.

4.1. MDP Formulation for Simulated Annealing with Adaptive Cooling

We recast classical Simulated Annealing (SA) as an MDP with one agent that selects the cooling coefficient $\alpha_{n+1} \in (0,1)$ at the end of each block n. The run is divided into B blocks; each block executes k Metropolis–Hastings (MH) steps with a *constant* proposal variance σ^2.

After block n the environment returns

$$S_n = \big(T_n, \langle f\rangle_{T_n}, \langle f^2\rangle_{T_n}, \sigma(f)_{T_n}, p_n\big) \in \mathcal{S},$$

Algorithm 2 RL–QESA (deployment)

Input: initial point $\mathbf{X}_0$; initial temperature T_1; trained policy π_θ; proposal variance σ^2; outer iterations B; inner iterations k

Output: approx. minimiser $\mathbf{X}^k_B$

```
for $b \leftarrow 1$ to $B$ do
    for $i \leftarrow 1$ to $k$ do
        draw $\mathbf{X}^i_b \sim \mathcal{N}(\mathbf{X}^{i-1}_b, \sigma^2 I)$;
        $\alpha \leftarrow \exp\big[-\big(f(\mathbf{X}^i_b) - f(\mathbf{X}^{i-1}_b)\big)/T_b\big]$;
        accept $\mathbf{X}^i_b$ w.p. $\min\{\alpha, 1\}$;
        otherwise set $\mathbf{X}^i_b \leftarrow \mathbf{X}^{i-1}_b$ ;
    compute summary $S_b = (T_b, \widehat{\mu}_b, \widehat{\sigma}_b, p_b)$ ;
    $\alpha_{b+1} \leftarrow \pi_\theta(S_b)$ ; $T_{b+1} \leftarrow \alpha_{b+1} T_b$ ;
return $\mathbf{X}^k_B$
```

where T_n is the current temperature, $\langle f\rangle_{T_n}$ and $\sigma(f)_{T_n}$ are the mean and standard deviation of the observed energies, and p_n is the acceptance rate in that block.

Given S_n, the agent outputs α_{n+1} and sets $T_{n+1} = \alpha_{n+1} T_n$. We employ the block-level reward

$$R_n = -\big(\alpha_{n+1} - 1\big)T_n - \beta\,\mathbb{1}_{\{\text{QE violated}\}}, \qquad (1)$$

with large β to heavily penalise any breach of the quasi-equilibrium (QE) condition (1). The first term encourages maximal cooling; the second enforces safety.

Within block $n+1$ we draw k proposals $\mathbf{X}_m \sim \mathcal{N}(\mathbf{X}_{m-1}, \sigma^2 I)$ and accept with probability $\min\{1, \exp[-(f(\mathbf{X}_m) - f(\mathbf{X}_{m-1}))/T_{n+1}]\}$. No proposal parameters are adapted.

4.2. Transformer Policy Architecture

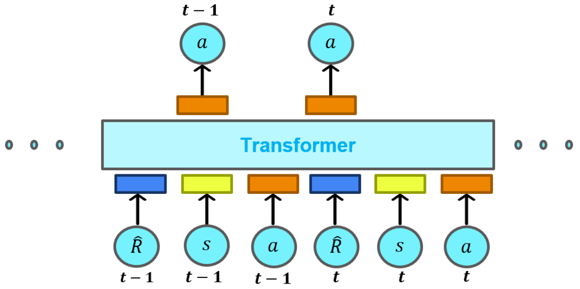

Figure 2. Transformer-based cooling policy in RL-QESA. A sequence of past block summaries—state s, action a, and reward $\hat{R}$—is fed into the transformer; the head emits the next cooling action a_t.

Block summaries $\{S_1, \dots, S_n\}$ form a sequence whose global context can influence future cooling decisions. Compared with recurrent networks, transformers capture such long-range dependencies via multi-head self-attention, without suffering from vanishing gradients, and scale naturally to hundreds of blocks.

Each summary S_n is mapped through a small MLP $\phi : \mathcal{S} \to \mathbb{R}^{d_{\text{model}}}$. Positional encodings $\mathbf{p}_n$ (learned or sinusoidal) are added to preserve order:

$$\mathbf{z}_n^0 = \phi(S_n) + \mathbf{p}_n.$$

The sequence $\{\mathbf{z}_n^0\}$ is processed by L transformer blocks:

$$\mathbf{z}_n^{\ell+1} = \mathrm{TFBlock}_\ell(\mathbf{z}_n^\ell, \{\mathbf{z}_1^\ell, \dots, \mathbf{z}_n^\ell\}), \quad \ell = 0, \dots, L-1.$$

Causal masking ensures that block n attends only to $1{:}n$.

The final hidden state $\mathbf{h}_n = \mathbf{z}_n^L$ is passed through a two-layer head to produce the cooling coefficient $\alpha_{n+1} \in (0,1)$:

$$\alpha_{n+1} = \sigma\big(W_2 \, \mathrm{ReLU}(W_1 \mathbf{h}_n)\big),$$

where σ is the logistic function.

We optimise the policy with PPO (Schulman et al., 2017), treating (1) as the episode-level return. Because QE violations trigger a large negative reward, the policy quickly learns to approach the analytical cooling bound (Section 3.2) without explicit knowledge of the Lipschitz constant.

The transformer aggregates historical acceptance rates, energy moments, and previous temperature drops. These features allow the agent to predict when the chain is close to equilibrium and to adjust α_{n+1} accordingly. Over repeated episodes, PPO converges to a policy that (i) cools near the theoretical limit, yet (ii) maintains the QE constraint—mirroring the guarantees proved in Section 3.

A mathematically derived constraint (QE) shapes the reward, while a transformer—state-of-the-art sequence model—learns *on-line* to saturate the cooling bound, demonstrating how modern learning can operationalise rigorous analytic insight.

5. Experiment Results

We assess our model using two well-known non-convex optimization benchmarks: the Rosenbrock function (Rosenbrock, 1960) and the Lennard-Jones potential (Wales & Doye, 1997). For our experiments, both the N-SA (Correia et al., 2023) and the RL-QESA architectures are employed, maintaining consistent architecture and hyper-parameters across all test cases. We consider varying problem sizes for these benchmarks, with further details to be discussed in subsequent sections.

In our experimental setup, the policy functions for N-SA and RL-QESA are initially trained on smaller datasets and subsequently tested on larger problem sets. This approach allows us to explore the scalability and adaptability of our proposed models. We maintain a uniform initial distribution for all datasets and utilize an exponential cooling schedule defined by $T_k = \alpha T_0^k$, which aids in the evaluation of the model's performance across different scales.

5.1. Rosenbrock Benchmark

The classical N-dimensional Rosenbrock function serves as our non-convex testbed:

$$f(\mathbf{x}) = \sum_{i=1}^{N-1} \Big[100\big(x_{i+1} - x_i^2\big)^2 + (1 - x_i)^2 \Big], \mathbf{x} \in [-2, 2]^N,$$

with its unique global minimiser at $\mathbf{x}^\star = (1, \dots, 1)$. The Rosenbrock function is notoriously ill-conditioned: gradients direct the optimiser toward a narrow, curved valley and then become nearly parallel to the shallow slope that leads to the global minimum.

The 2-D illustration in Fig. 3 shows how a classical SA run explores the steep ridges of the Rosenbrock landscape, slips into the narrow valley, and eventually converges to the global minimiser at $(1, 1)$.

Table 1 extends the comparison to $N \in \{10, 20, 50\}$ dimensions and six evaluation budgets. RL-QESA consistently dominates the baselines. At the earliest checkpoint (1 000 evaluations) it already achieves the lowest energy for $N = 10$ and $N = 20$, and is a close second for $N = 50$. By 10 000 evaluations it matches or surpasses the best competing method in every setting; at the full 50 000 evaluations it attains the lowest mean energy in all cases, often by an order of magnitude (red cells). Standard deviations shrink in parallel, indicating that the learned cooling schedule is reliable, not merely lucky. These trends confirm the qualitative picture in Fig. 3: the transformer policy cools aggressively but safely, reaching the curved valley faster than a hand-tuned logarithmic schedule and finishing with consistently lower energies as dimensionality grows.

5.2. The Lenard-Jones Potential

The Lennard–Jones potential is defined as

$$V(r) = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right],$$

where r is the distance between particles, ϵ is the depth of the potential well, and σ is the finite distance at which the inter–particle potential vanishes.

Table 2 reports mean energies over 10 runs for cluster sizes $N = 6, 9, 13$ at 10 k and 20 k evaluations. Across all settings RL–QESA finds the lowest (most negative) energies, often by large margins: for $N = 13$ it improves on vanilla SA by

Table 1. Comparison of simulation results for the Rosenbrock function. Each entry shows the mean energy over 20 runs, followed by the standard deviation in parentheses. The best and second-best numbers in each column are highlighted red and blue, respectively.

Iterations	**1 000**	**2 000**	**5 000**	**10 000**	**20 000**	**50 000**
Problem Size: 10						
SA	32.67 (3.36)	22.92 (3.18)	19.56 (2.86)	8.13 (1.65)	0.88 (0.51)	0.004 (<0.001)
N–SA	28.65 (2.99)	29.86 (3.24)	18.91 (2.65)	6.36 (1.76)	1.22 (0.82)	0.004 (<0.001)
RL–QESA	25.78 (3.09)	23.54 (2.86)	13.64 (2.40)	6.37 (1.50)	0.97 (0.61)	0.002 (<0.0001)
Problem Size: 20						
SA	61.46 (4.86)	57.72 (4.34)	34.20 (2.82)	17.91 (2.17)	2.08 (0.84)	0.30 (0.10)
N–SA	60.80 (4.39)	54.29 (3.69)	34.64 (2.95)	15.23 (2.01)	2.39 (0.78)	0.01 (0.01)
RL–QESA	54.96 (4.01)	47.01 (4.15)	29.61 (3.07)	14.34 (1.88)	2.02 (0.75)	0.004 (<0.001)
Problem Size: 50						
SA	153.08 (5.60)	134.43 (4.62)	84.77 (4.14)	40.68 (2.54)	5.66 (0.83)	2.40 (0.30)
N–SA	165.56 (5.80)	135.79 (4.39)	84.36 (3.74)	38.86 (2.78)	5.47 (1.00)	0.10 (0.20)
RL–QESA	153.70 (4.75)	121.30 (5.26)	83.55 (3.70)	34.51 (2.27)	4.91 (0.89)	0.01 (<0.001)

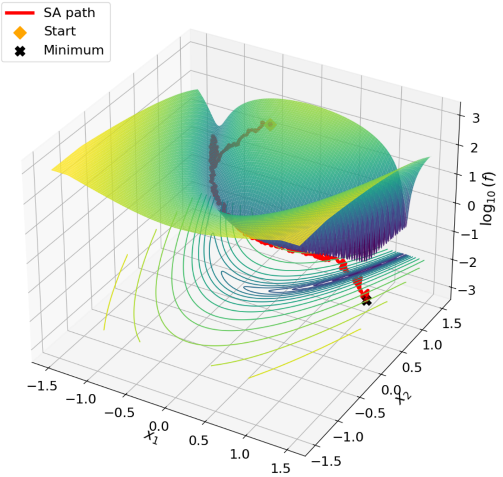

Figure 3. SA trajectory (red) on the log-scaled Rosenbrock landscape. The algorithm begins at the orange diamond, explores broadly, enters the curved valley, and converges to the global minimum (black ×). Projected contours highlight the valley's geometry.

Table 2. Simulation results for the Lennard–Jones potential. Each entry is the mean energy over 10 runs; the number in parentheses is the standard deviation. The best (most negative) and second-best values per column are shaded red and blue, respectively.

Iterations	**10 000**	**20 000**
$N = 6$		
SA	-1.14 (± 3.43)	-2.24 (± 4.95)
N–SA	-9.29 (± 0.81)	-10.83 (± 0.82)
RL–QESA	$\mathbf{-11.11}$ $\mathbf{(\pm 0.25)}$	$\mathbf{-12.20}$ $\mathbf{(\pm 0.06)}$
$N = 9$		
SA	-17.09 (± 3.23)	-18.58 (± 1.24)
N–SA	-17.11 (± 1.82)	-18.85 (± 1.12)
RL–QESA	$\mathbf{-20.96}$ $\mathbf{(\pm 0.61)}$	$\mathbf{-23.14}$ $\mathbf{(\pm 0.21)}$
$N = 13$		
SA	-25.05 (± 3.03)	-26.81 (± 2.39)
N–SA	-18.51 (± 95.09)	-27.25 (± 13.27)
RL–QESA	$\mathbf{-36.30}$ $\mathbf{(\pm 5.36)}$	$\mathbf{-40.74}$ $\mathbf{(\pm 3.27)}$

$\approx 11\,\varepsilon$ at 20 k steps and reduces the variance by more than a factor of two. Neural SA narrows the gap in easy cases ($N = 6$) but shows large fluctuations for $N = 13$, confirming that a static neural schedule cannot reliably navigate the hard–core repulsion and the delicate many-body minima. The transformer–guided cooling of RL–QESA therefore offers both faster convergence and higher robustness on this rugged potential landscape.

6. Conclusion and Outlook

We presented **RL–QESA**, a transformer-guided extension of simulated annealing that enforces a quasi-equilibrium test at the end of every block. On both smooth (Rosenbrock) and rugged (Lennard–Jones) benchmarks the policy learns to cool sooner than a fixed logarithmic schedule while keeping the sampler near Boltzmann equilibrium, which translates into faster convergence and lower final energies than classical SA and neural-annealing baselines.

The main limitation is that the size of each temperature drop is still hand-fixed: the policy decides *when* to cool but not *how far*. Subsection 3.2 shows that substantially larger drops remain theoretically safe once quasi-equilibrium is achieved, so letting the agent predict the cooling magnitude should reduce wall-clock time even further.

Immediate extensions include a discrete test problem—where the same equilibrium check can be computed from cost samples and acceptance statistics. We also plan to scale to Lennard–Jones clusters with hundreds of atoms, combining adaptive cooling with basin-hopping moves. A parallel line of work will add a second agent that chooses proposal scales or neighbourhood operators, pushing RL–QESA toward a fully self-tuning optimiser for both continuous and combinatorial landscapes.

References

Aarts, E. H. L. and Korst, J. H. M. Statistical cooling: A general approach to combinatorial optimization problems. *Philips Journal of Research*, 40:63–86, 1985.

Atchadé, Y. F. and Liu, J. S. The wang-landau algorithm in general state spaces: applications and convergence analysis. *Statistica Sinica*, 20(1):209–233, 2010.

Bellman, R. *Dynamic Programming*. Princeton University Press, 1957.

Cai, Q., Hang, W., Mirhoseini, A., Tucker, G., Wang, J., and Wei, W. Reinforcement learning driven heuristic optimization. *arXiv preprint arXiv:1906.06639*, 2019.

Cai, Z. Sa-gd: Improved gradient descent learning strategy with simulated annealing. *arXiv preprint arXiv:2107.07558*, 2021. URL https://arxiv.org/abs/2107.07558.

Chen, P., Xie, Y., and Zhang, Q. Sicnn: Sparsity-induced input convex neural network for optimal transport. In *Proceedings of the NeurIPS 2024 Workshop on Optimization for Machine Learning*, 2024. URL https://openreview.net/forum?id=iCLEUcDGUJ.

Correia, Á. H. C., Worrall, D. E., and Bondesan, R. Neural simulated annealing. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 4946–4962. PMLR, 2023. URL https://proceedings.mlr.press/v206/correia23a.html.

Eglese, R. W. Simulated annealing: A tool for operational research. *European Journal of Operational Research*, 46 (3):271–281, 1990.

Elgammal, M. A., Murray, K. E., and Betz, V. RLPlace: Using reinforcement learning and smart perturbations to optimize FPGA placement. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(8):2532–2545, 2021.

Geman, S. and Geman, D. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984.

Granville, V., Krivanek, M., and Rasson, J. P. Simulated annealing: A proof of convergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):652–656, 1994.

Guo, Z. and Cao, Y. Sa-cnn: Application to text categorization issues using simulated annealing-based convolutional neural network optimization. In *Proceedings of ACM ELITE 2022*, 2022. URL https://arxiv.org/abs/2303.07153.

Hajek, B. Cooling schedules for optimal annealing. *Mathematics of Operations Research*, 13(2):311–329, 1988.

Ingber, L. Simulated annealing: Practice versus theory. *Mathematical and Computer Modelling*, 18(11):29–57, 1993.

Ingber, L. Adaptive simulated annealing (ASA): Lessons learned. *Control and Cybernetics*, 25(1):33–54, 1996.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. Optimization by simulated annealing. *Science*, 220(4598): 671–680, 1983.

Li, W., Liang, P., Sun, B., Sun, Y., and Huang, Y. Learning-based simulated annealing algorithm for unequal area facility layout problem. *Soft Computing*, 2023. doi: 10.1007/s00500-023-07381-x.

Puterman, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, 1994. (Duplicate of Puterman1994 if you prefer that key.).

Roberts, G. O. and Rosenthal, J. S. Optimal scaling of discrete approximations to langevin diffusions. *Journal of the Royal Statistical Society: Series B*, 59(1):39–57, 1997. doi: 10.1111/1467-9868.00044.

Roberts, G. O. and Rosenthal, J. S. Optimal scaling for various metropolis-hastings algorithms. *Statistical Science*, 16(4):351–367, 2001. doi: 10.1214/ss/1015346320.

Romeo, F. and Sangiovanni-Vincentelli, A. Reheating: A technique for escaping local minima in annealing algorithms. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 10(2):173–181, 1991. doi: 10.1109/43.75499.

Rosenbrock, H. An automatic method for finding the greatest or least value of a function. *The Computer Journal*, 3 (1):175–184, 1960.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, Jul 2017. URL https://arxiv.org/abs/1707.06347. Presented at the Deep RL Workshop, ICML 2017.

Suman, B. and Kumar, P. A survey of simulated annealing as a tool for single and multiobjective optimization. *Journal of the Operational Research Society*, 57(10):1143–1160, 2006.

Sutton, R. S. and Barto, A. G. Reinforcement learning: An introduction. *MIT Press*, 1998. First edition of the well-known RL textbook.

van Laarhoven, P. J. M. and Aarts, E. H. L. *Simulated Annealing: Theory and Applications*. Springer, 1987.

Vashisht, D., Rampal, H., Liao, H., Lu, Y., Shanbhag, D., Fallon, E., and Kara, L. B. Placement in integrated circuits using cyclic reinforcement learning and simulated annealing. *arXiv preprint arXiv:2011.07577*, 2020.

Wales, D. J. and Doye, J. P. K. Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms. *The Journal of Physical Chemistry A*, 101(28):5111–5116, 1997.

Xu, R., Wang, H., and Deng, Y. The impact of move schemes on simulated annealing performance. *arXiv preprint arXiv:2504.17949*, 2025.

Yang, X. S. *Introduction to Mathematical Optimization: From Linear Programming to Metaheuristics*. Cambridge International Science Publishing, 2008.