PNAS Nexus, 2023, 2, 1–16
https://doi.org/10.1093/pnasnexus/pgad248
Advance access publication 2 August 2023
Research Report

Model-free optimization of power/efficiency tradeoffs in quantum thermal machines using reinforcement learning

Paolo A. Erdman[a,*] and Frank Noé[a,b,c,d,*]

[a]Department of Mathematics and Computer Science, Freie Universität Berlin, Arnimallee 6, 14195 Berlin, Germany
[b]Microsoft Research AI4Science, Karl-Liebknecht Str. 32, 10178 Berlin, Germany
[c]Department of Physics, Freie Universität Berlin, Arnimallee 6, 14195 Berlin, Germany
[d]Department of Chemistry, Rice University, Houston, TX 77005, USA
*To whom correspondence should be addressed. Email: p.erdman@fu-berlin.de; frank.noe@fu-berlin.de
Edited By: A. Editor

Abstract

A quantum thermal machine is an open quantum system that enables the conversion between heat and work at the micro or nano-scale. Optimally controlling such out-of-equilibrium systems is a crucial yet challenging task with applications to quantum technologies and devices. We introduce a general model-free framework based on reinforcement learning to identify out-of-equilibrium thermodynamic cycles that are Pareto optimal tradeoffs between power and efficiency for quantum heat engines and refrigerators. The method does not require any knowledge of the quantum thermal machine, nor of the system model, nor of the quantum state. Instead, it only observes the heat fluxes, so it is both applicable to simulations and experimental devices. We test our method on a model of an experimentally realistic refrigerator based on a superconducting qubit, and on a heat engine based on a quantum harmonic oscillator. In both cases, we identify the Pareto-front representing optimal power-efficiency tradeoffs, and the corresponding cycles. Such solutions outperform previous proposals made in the literature, such as optimized Otto cycles, reducing quantum friction.

Keywords: quantum thermal machines, reinforcement learning, quantum optimal control, quantum thermodynamics, quantum technologies

Significance Statement

Thermal machines, such as heat engines and refrigerators, allow the conversion between heat and work. Quantum thermal machines are micro or nano-scale thermal machines that operate exploiting quantum effects, potentially improving their performance. However, the optimal control of such quantum devices is an extremely challenging task with application to quantum technologies in general. Here, we develop a reinforcement learning framework to optimally control quantum thermal machines as to maximize their power-efficiency tradeoff. The method does not require any knowledge of the innerworkings of the device, nor of its state, making it potentially applicable directly to experimental devices. It only monitors the heat currents through the device. We apply our method to two prototypical setups, outperforming previous proposals made in the literature.

A driving force of the research field of quantum thermodynamic is the quest of understanding and designing quantum thermal machines (QTMs), i.e. devices that convert between heat and work at the micro or nanoscale exploiting quantum effects (1–5). Such devices could be operated as heat engines, which convert heat into work, or refrigerators, that extract heat from a cold bath. Recent experiments have measured the heat flowing across these devices (6–9), and early experimental realizations of QTMs have been reported (10–17).

However, the optimal control of such devices, necessary to reveal their maximum performance, is an extremely challenging task that could find application in the control of quantum technologies and devices beyond QTMs. The difficulties include: (i) having to operate in finite time, the state can be driven far from equilibrium, where the thermal properties of the system are model-specific; (ii) the optimization is a search over the space of all possible time-dependent controls, which increases exponentially with the number of time points describing the cycle; (iii) in experimental devices, often subject to undesired effects such as noise and decoherence (18), we could have a limited knowledge of the actual model describing the dynamics of the QTM.

A further difficulty (iv) arises in QTMs, since the maximization of their performance requires a multiobjective optimization.

Competing Interest: The authors declare no competing nonfinancial interests but the following Competing Financial Interest. P.A.E. and F.N. are authors of a pending patent application containing aspects of the model-free reinforcement learning approach to quantum thermal machines here described (Application to the European Patent Office, file number: 21 191 966.7).
Received: July 13, 2023. **Revised:** July 13, 2023. **Accepted:** July 25, 2023

Indeed, the two main quantities that describe the performance of a heat engine (refrigerator) are the extracted power (cooling power) and the efficiency (coefficient of performance). The optimal strategy to maximize the efficiency consists of performing reversible transformations (19) which are, however, infinitely slow, and thus deliver vanishing power. Conversely, maximum power is typically reached at the expense of reduced efficiency. Therefore, one must seek optimal tradeoffs between the two.

The theoretical optimization of QTMs is typically carried out making restrictive assumptions on the cycle. For example, optimal strategies have been derived assuming the driving speed of the control to be slow (20–29) or fast (30–32) compared to the thermalization time. Other approaches consists of assuming a priori a specific shape of the cycle structure (33–40), such as the Otto cycle (41–56). Shortcuts to adiabaticity (57–65) and variational strategies (66–68) have also been employed.

In general, aside from variational approaches, there is no guarantee that these regimes and cycles are optimal. Recently, reinforcement learning (RL) has been used to find cycles that maximize the power of QTMs without making assumptions on the cycle structure (69). However, this approach requires a model of the system and the knowledge of the quantum state of the system, which restricts its practical applicability. This calls for the development of robust and general strategies that overcome all above-mentioned difficulties (i–iv).

We propose a RL-based method with the following properties: (i) it finds cycles yielding near Pareto-optimal tradeoffs between power and efficiency, i.e. the collection of cycles such that it is not possible to further improve either power or efficiency, without decreasing the other one. (ii) It only requires the heat currents as input, and not the quantum state of the system. (iii) It is completely model-free. (iv) It does not make any assumption on the cycle structure, nor on the driving speed. The RL method is based on the Soft Actor-Critic algorithm (70, 71), introduced in the context of robotics and video-games (72, 73), generalized to combined discrete and continuous actions and to optimize multiple objectives. RL has received great attention for its success at mastering tasks beyond human-level such as playing games (74–76), and for robotic applications (77). RL has been recently used for quantum control (78–87), outperforming previous state-of-the-art methods (88, 89), for fault-tolerant quantum computation (90, 91), and to minimize entropy production in closed quantum systems (92).

We prove the validity of our approach finding the full Pareto-front, i.e. the collection of all Pareto-optimal cycles describing optimal power-efficency tradeoffs, in two paradigmatic systems that have been well studied in the literature: a refrigerator based on an experimentally realistic superconducting qubit (6, 49), and a heat engine based on a quantum harmonic oscillator (43). In both cases, we find elaborate cycles that outperform previous proposal mitigating quantum friction (43, 49, 55, 66, 93–95), i.e. the detrimental effect of the generation of coherence in the instantaneous eigenbasis during the cycle. Remarkably, we can also match the performance of cycles found with the RL method of Ref. (69) that, as opposed to our model-free approach, requires monitoring the full quantum state and only optimizes the power.

Setting: black-box quantum thermal machine

We describe a QTM by a quantum system, acting as a "working medium", that can exchange heat with a hot (H) or cold (C) thermal bath characterized by inverse temperatures $\beta_H < \beta_C$ (Fig. 1). Our method can be readily generalized to multiple baths, but we focus the description on two baths here.

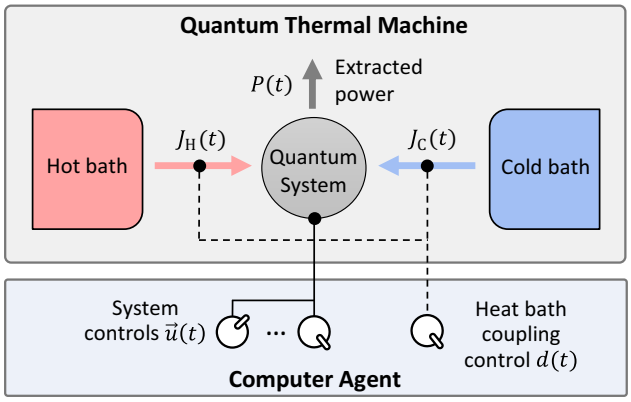

Fig. 1. Schematic representation of a quantum thermal machine controlled by a computer agent. A quantum system (gray circle in the center) can be coupled to a hot (cold) bath at inverse temperature β_H (β_C), represented by the red square to the left (blue square to the right), enabling a heat flux $J_H(t)$ ($J_C(t)$). The quantum system is controlled by the computer agent through a set of experimental control parameters $\vec{u}(t)$, such as an energy gap or an oscillator frequency, that control the power exchange $P(t)$, and through a discrete control $d(t) = \{\text{Hot, Cold, None}\}$ that determines which bath is coupled to the quantum system.

We can control the evolution of the quantum system and exchange work with it through a set of time-dependent continuous control parameters $\vec{u}(t)$ that enter in the Hamiltonian $H[\vec{u}(t)]$ of the quantum system (96), and through a discrete control $d(t) = \{\text{Hot, Cold, None}\}$ that determines which bath is coupled to the system. $J_H(t)$ and $J_C(t)$ denote the heat flux flowing out, respectively, from the hot and cold bath at time t.

Our method only relies on the following two assumptions:

(i) the RL agent can measure the heat fluxes $J_C(t)$ and $J_H(t)$ (or their averages over a time period Δt);

(ii) $J_C(t)$ and $J_H(t)$ are functions of the control history $(\vec{u}(t-\tau), d(t-\tau)), \ldots, (\vec{u}(t), d(t))$, where τ is the timescale over which the QTM remembers past controls.

In contrast to previous work (69), the RL optimization algorithm does not require any knowledge of the microscopic model of the inner workings of the quantum system, nor of its quantum state; it is only provided with the values of the heat fluxes $J_C(t)$ and $J_H(t)$. These can be either computed from a theoretical simulation of the QTM (69), or measured directly from an experimental device whenever the energy change in the heat bath can be monitored without influencing the energetics of the quantum system (see e.g. experimental demonstrations (6–9)). In this sense, our quantum system is treated as a "black-box," and our RL method is "model-free." Any theoretical model or experimental device satisfying these requirements can be optimized by our method, including also classical stochastic thermal machines. The timescale τ is finite because of energy dissipation and naturally emerges by making the minimal assumption that the coupling of the quantum system to the thermal baths drives the system towards a thermal state within some timescale τ. Such a timescale can be rigorously identified e.g. within the weak system-bath coupling regime, and in the reaction coordinate framework that can describe non-Markovian and strong-coupling effects (97). In a Markovian setting, τ is related to the inverse of the characteristic thermalization rate.

The thermal machines we consider are the heat engine and the refrigerator. Up to an internal energy contribution that vanishes after each repetition of the cycle, the instantaneous power of a heat engine equals the extracted heat:

$$P_{\text{heat}}(t) = J_C(t) + J_H(t), \tag{1}$$

and the cooling power of a refrigerator is:

$$P_{\text{cool}}(t) = J_C(t). \tag{2}$$

The entropy production is given by

$$\Sigma(t) = -\beta_C J_C(t) - \beta_H J_H(t), \tag{3}$$

where we neglect the contribution of the quantum system's entropy since it vanishes after each cycle.

Machine learning problem

Our goal is to identify cycles, i.e. periodic functions $\vec{u}(t)$ and $d(t)$, that maximize a tradeoff between power and efficiency *on the long run*. Since power and efficiency cannot be simultaneously optimized, we use the concept of Pareto-optimality (98, 99). Pareto-optimal cycles are those where power or efficiency cannot be further increased without sacrificing the other one. The Pareto-front, defined as the collection of power-efficiency values delivered by all Pareto-optimal cycles, represents all possible optimal tradeoffs. To find the Pareto-front, we define the *reward* function $r_c(t)$ as:

$$r_c(t) = c\frac{P(t)}{P_0} - (1-c)\frac{\Sigma(t)}{\Sigma_0}, \tag{4}$$

where $P(t)$ is the power of a heat engine (Eq. 1) or cooling power of a refrigerator (Eq. 2), P_0, Σ_0 are reference values to normalize the power and entropy production, and $c \in [0, 1]$ is a weight that determines the tradeoff between power and efficiency. As in Ref. (69), we are interested in cycles that maximize the long-term performance of QTMs; we thus maximize the *return* $\langle r_c \rangle(t)$, where $\langle\cdot\rangle(t)$ indicates the exponential moving average of future values:

$$\langle r_c \rangle(t) = \kappa \int_0^\infty e^{-\kappa\tau} r_c(t+\tau)\, d\tau. \tag{5}$$

Here, κ is the inverse of the averaging timescale, that will in practice be chosen much longer than the cycle period, such that $\langle r_c \rangle(t)$ is approximately independent of t.

For $c = 1$, we are maximizing the average power $\langle r_1 \rangle = \langle P \rangle / P_0$. For $c = 0$, we are minimizing the average entropy production $\langle r_0 \rangle = -\langle \Sigma \rangle / \Sigma_0$, which corresponds to maximizing the efficiency. For intermediate values of c, the maximization of $\langle r_c \rangle$ describes tradeoffs between power and efficiency (see "Optimizing the entropy production" in Materials and Methods for details). Interestingly, if convex, it has been shown that the full Pareto-front can be identified repeating the optimization of $\langle r_c \rangle$ for many values of c (98, 100).

Results

Deep RL for black-box QTMs

In RL, a *computer agent* must learn to master some task by repeated interactions with some *environment*. Here, we develop an RL approach where the agent maximizes the return 5 and the environment is the QTM with its controls (Fig. 2A). To solve the RL problem computationally, we discretize time as $t_i = i\Delta t$. By time-discretizing the return 5, we obtain a discounted return whose discount factor $\gamma = \exp(-\kappa\Delta t)$ determines the averaging timescale and expresses how much we are interested in future or immediate rewards (see "Reinforcement Learning Implementation" in Materials and Methods for details).

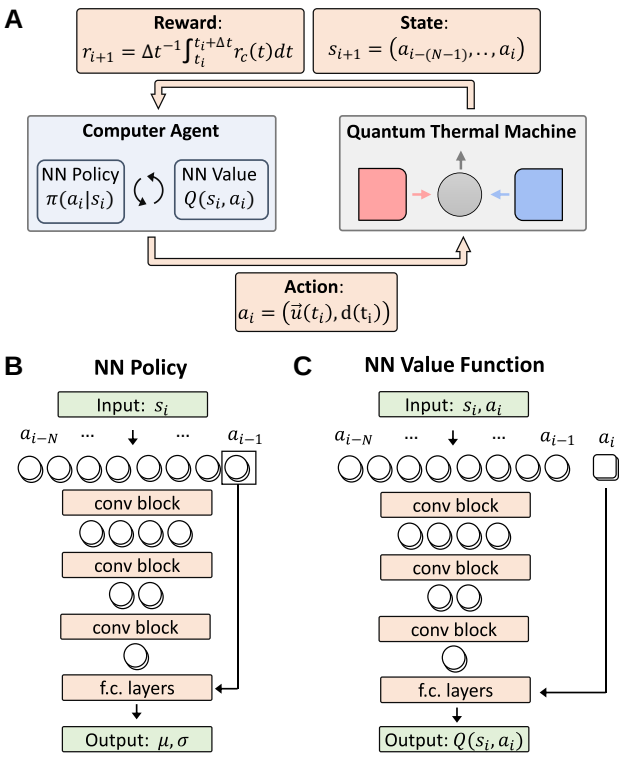

Fig. 2. A) Schematic representation of the learning process. A computer agent (center left blue box) chooses an action a_i at time-step i based on the current state s_i of the QTM (center right gray box) through the policy function $\pi(a_i \mid s_i)$. The action, that encodes the control $(\vec{u}(t_i), d(t_i))$, is passed to the QTM (lower arrow). The new state s_{i+1}, composed of the time-series of the last N actions, and the reward r_{i+1} are returned to the agent (upper arrow), which uses this information to improve $\pi(a \mid s)$ using the soft actor-critic algorithm, which learns also the values function $Q(s, a)$. This process is reiterated until convergence of the policy. B,C) Schematic representation of the NN architectures used to parameterize the policy (B) and the value function (C). The action time-series in s_i is processed using multiple 1D convolution blocks, each one halving the length of the series. The final output is produced by fully connected (f.c.) layers.

At each time step t_i, the agent employs a policy function $\pi(a \mid s)$ to choose an action $a_i = \{\vec{u}(t_i), d(t_i)\}$ based on the state s_i of the environment. Here, the policy function $\pi(a \mid s)$ represents the probability of choosing action a, given that the environment is in state s, $\vec{u}(t)$ are the continuous controls over the quantum system, and $d(t_i) \in$ {Hot, Cold, None} is a discrete control that selects the bath the system is coupled to. All controls are considered to be constant during time step of duration Δt. The aim of RL is to learn an optimal policy function $\pi(a \mid s)$ that maximizes the return.

In order to represent a black-box quantum system whose inner mechanics are unknown, we define the control history during a time interval of length T as the observable state:

$$s_i = (a_{i-N}, a_{i-N+1}, \ldots, a_{i-1}), \tag{6}$$

where $N = T/\Delta t$. Therefore, the state of the quantum system is implicitly defined by the sequence of the agent's N recent actions.

To find an optimal policy we employ the soft actor-critic algorithm, that relies on learning also a value function $Q(s, a)$, generalized to a combination of discrete and continuous actions (70–73). The policy function $\pi(a \mid s)$ plays the role of an "actor" that chooses the actions to perform, while a value function

$Q(s, a)$ plays the role of a "critic" that judges the choices made by the actor, thus providing feedback to improve the actor's behavior. We further optimize the method for a multiobjective setting by introducing a separate critic for each objective, i.e. one value function for the power, and one for the entropy production. This allow us to vary the weight c during training, thus enhancing convergence (see "Reinforcement Learning Implementation" in Materials and Methods for details).

We learn the functions $\pi(a \mid s)$ and $Q(s, a)$ using a deep NN architecture inspired by WaveNet, an architecture that was developed for processing audio signals (101) (See Fig. 2B and C). We introduce a "convolution block" to efficiently process the time-series of actions defining the state s_i. It consists of a 1D convolution with kernel size and stride of 2, such that it halves the length of the input. It is further equipped with a residual connection to improve trainability (102) (see "Reinforcement Learning Implementation" in Materials and Methods for details). The policy $\pi(a_i \mid s_i)$ is described by a NN that takes the state s_i as input, and outputs parameters μ and σ describing the probability distribution from which action a_i is sampled (Fig. 2B). The value function $Q(s_i, a_i)$ is computed by feeding (s_i, a_i) into a NN, and outputting $Q(s_i, a_i)$ (Fig. 2C). Both $\pi(a_i \mid s_i)$ and $Q(s_i, a_i)$ process the state by feeding it through multiple convolution blocks (upper orange boxes in Fig. 2B and C), each one halving the length of the time-series, such that the number of blocks and of parameters in the NN is logarithmic in N. Then a series of fully connected layers produce the final output.

The policy and value functions are determined by minimizing the loss functions in Eqs. 39 and 49 using the ADAM optimization algorithm (103). The gradient of the loss functions is computed off-policy, over a batch of past experience recorded in a replay buffer, using back-propagation (see "Reinforcement Learning Implementation" in Materials and Methods for details).

Pareto-optimal cycles for a superconducting qubit refrigerator

We first consider a refrigerator based on an experimentally realistic system: a superconducting qubit coupled to two resonant circuits that behave as heat baths (49) (Fig. 3A). Such a system was experimentally studied in the steady-state in Ref. (6). The system Hamiltonian is given by (49, 55, 63):

$$\hat{H}[u(t)] = -E_0[\Delta\hat{\sigma}_x + u(t)\hat{\sigma}_z], \tag{7}$$

where E_0 is a fixed energy scale, Δ characterizes the minimum gap of the system, and $u(t)$ is our control parameter. In this setup the coupling to the baths, described by the commonly employed Markovian master equation (104–107), is fixed, and cannot be controlled. However, the qubit is resonantly coupled to the baths at different energies. The u-dependent coupling strength to the cold (hot) bath is described by the function $\gamma_u^{(C)}$ ($\gamma_u^{(H)}$), respectively (Fig. 3F). As in Ref. (63), the coupling strength is, respectively, maximal at $u = 0$ ($u = 1/2$), with a resonance width determined by the "quality factor" Q_C (Q_H) (see "Physical model" in Materials and Methods for details). This allows us to choose which bath is coupled to the qubit by tuning $u(t)$.

In Fig. 3, we show an example of our training procedure to optimize the return $\langle r_c \rangle$ at $c = 0.6$ using $N = 128$ steps determining the RL state, and varying c during training from 1 to 0.6 (Fig. 3C). In the early stages of the training, the return $\langle r_c \rangle_i$, computed as in Eq. 28 but over past rewards, and the running averages of the cooling power $\langle P_{\text{cool}} \rangle_i$ and of the negative entropy production $-\langle \Sigma \rangle_i$ all start off negative (Fig. 3B), and the corresponding actions are random (left panel of Fig. 3D). Indeed, initially the RL agent has no

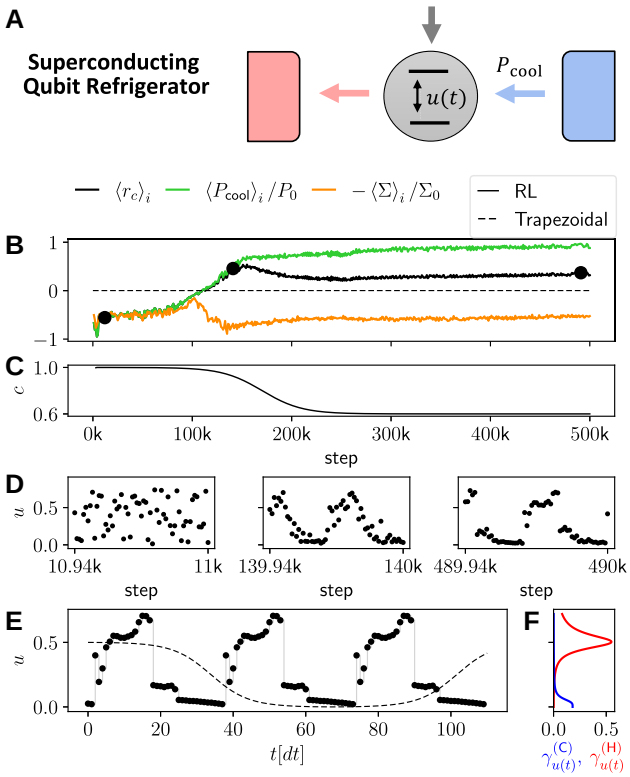

Fig. 3. Training of the superconducting qubit refrigerator model to optimize $\langle r_c \rangle$ at $c = 0.6$. A) Schematic representation of the energy levels of the qubit (horizontal black lines) that are controlled by $u(t)$. The vertical gray arrow represents the input power, while the colored horizontal arrows represent the heat fluxes. B) Return $\langle r_c \rangle_i$ computed over past rewards (black curve), running average of the cooling power $\langle P_{\text{cool}} \rangle_i / P_0$ (green curve), and of the negative entropy production $-\langle \Sigma \rangle_i / \Sigma_0$ (orange curve), as a function of the training step. The dashed line represents the value of the return found optimizing the period of a smoothed trapezoidal cycle. C) Value of the weight c as a function of the step. It is varied during training from 1 to the final value 0.6 to improve convergence. D) Actions chosen by the agent, represented by the value of u, as a function of step, zoomed around the three black circles in panel (B). E) Final deterministic cycle found by the agent (thick black dots) and smoothed trapezoidal cycle (thin dashed line) whose return is given by the dashed line in panel (B), as a function of time. F) coupling strength $\gamma_u^{(C)}$ (blue curve) and $\gamma_u^{(H)}$ (red curve) as a function of u (on the y-axis). The parameters used for training are $N = 128$, $g_H = g_C = 1$, $\beta_H = 10/3$, $\beta_C = 2\beta_H$, $Q_H = Q_C = 4$, $E_0 = 1$, $\Delta = 0.12$, $\omega_H = 1.028$, $\omega_C = 0.24$, $\mathcal{U} = [0, 0.75]$, $\Delta t = 0.98$, $\gamma = 0.997$, $P_0 = 6.62 \cdot 10^{-4}$, and $\Sigma_0 = 0.037$.

experience controlling the QTM, so random actions are performed, resulting in heating the cold bath, rather than cooling it, and in a large entropy production. However, with increasing steps, the chosen actions exhibit some structure (Fig. 3D), and the return $\langle r_c \rangle_i$ increases (Fig. 3B). While both the power and the negative entropy production initially increase together, around step 100k we see that $-\langle \Sigma \rangle_i$ begins to decrease. This is a manifestation of the fact that power and entropy production cannot be simultaneously optimized. Indeed, the agent learns that in order to further increase the return, it must "sacrifice" some entropy production to produce a positive and larger cooling power. In fact, the only way to achieve positive values of $\langle r_c \rangle_i$ is to have a positive cooling power, which inevitably requires producing entropy. Eventually all quantities in Fig. 3B reach a maximum value, and the corresponding final deterministic cycle (i.e. the cycle generated by policy switching off stochasticity, see "Reinforcement Learning Implementation" in Materials and Methods for details) is shown in Fig. 3E as thick black dots.

For the same system, Ref. (63) proposed a smoothed trapezoidal cycle $u(t)$ oscillating between the resonant peaks at $u = 0$ and $u = 1/2$ and optimized the cycle time (Fig. 3E, dashed line). While this choice outperformed a sine and a trapezoidal cycle (49), the cycle found by our RL agent produces a larger return (Fig. 3B). The optimal trapezoidal cycle found for $c = 0.6$ is shown in Fig. 3E as a dashed line (see "Comparing with other methods" in Materials and Methods for details).

Fig. 4 compares optimal cycles for different tradeoffs between cooling power and coefficient of performance η_{cool}, the latter defined as the ratio between the average cooling power, and the average input power. This is achieved by repeating the optimization for various values of c. To demonstrate the robustness of our method, the optimization of $\langle r_c \rangle$ was repeated five times for each choice of c (variability shown with error bars in Fig.4A, and as separate points in Fig.4B). The RL method substantially outperforms the trapezoidal cycle by producing larger final values of the return $\langle r_c \rangle$ at all values of c (Fig. 4A), and by producing a better Pareto front (Fig. 4B). The RL cycles simultaneously yield higher power by more than a factor of 10, and a larger η_{cool}, for any choice of the power-efficiency tradeoff. The model-free RL cycles can also deliver the same power at a substantially higher COP (roughly 10 times larger) when compared with the cycle found with the RL method of Ref. (69), which only optimizes the power. This is remarkable since, as opposed to the current model-free method, the method in Ref. (69) has access to the full quantum state of the system, and not only to the heat currents (see "Comparing with other methods" in Materials and Methods for details). This also shows that a large efficiency improvement can be achieved by sacrificing very little power.

As expected, the period of the RL cycles increases as c decreases and the priority shifts from high power to high η_{cool} (Fig. 4C to F, black dots). However, the period is much shorter than the corresponding optimized trapezoidal cycle (dashed line), and the optimal control sequence is quite unintuitive, even going beyond the resonant point at $u = 1/2$. As argued in (49, 55, 63), the generation of coherence in the instantaneous eigenbasis of the quantum system, occurring because $[\hat{H}(u_1), \hat{H}(u_2)] \neq 0$ for $u_1 \neq u_2$, causes power losses that increase with the speed of the cycle. We find that we can interpret the power enhancement achieved by our cycle as a mitigation of such detrimental effect: indeed, we find that trapezoidal cycles operated at the same frequency as the RL cycle generate twice as much coherence as the RL cycles (see "Generation of coherence" in Materials and Methods for details). In either case, cycles with higher power tend to generate more coherence.

Given the stochastic nature of RL, we also compared the cycles obtained across the five independent training runs, finding that cycles are typically quite robust, displaying only minor changes (see Fig. 8 of Methods for four cycles found in independent training runs corresponding to Fig. 4C to F).

Pareto-optimal cycles for a quantum harmonic oscillator engine

We now consider a heat engine based on a collection of noninteracting particles confined in a harmonic potential (43) (Fig. 5A). The Hamiltonian is given by

$$\hat{H}[u(t)] = \frac{1}{2m}\hat{p}^2 + \frac{1}{2}m(u(t)w_0)^2\hat{q}^2, \tag{8}$$

where m is the mass of the system, w_0 is a reference frequency and $\hat{p}$ and $\hat{q}$ are the momentum and position operators. The control parameter $u(t)$ allows us to change the frequency of the oscillator.

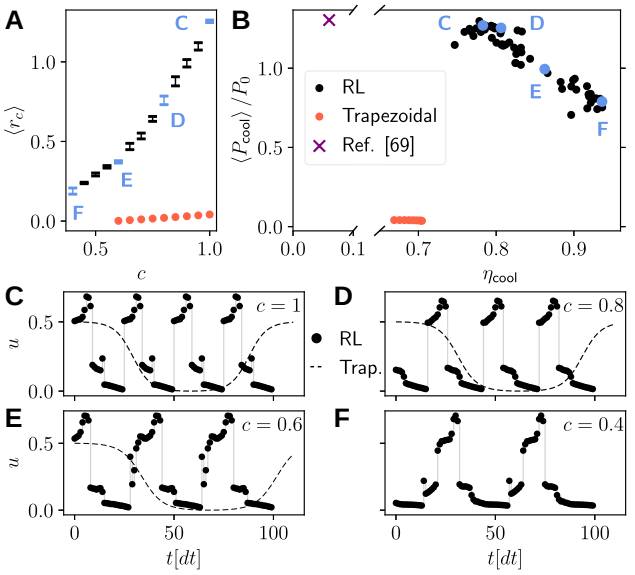

Fig. 4. Results for the optimization of the superconducting qubit refrigerator model. A) Final value of the return $\langle r_c \rangle$, as a function of c, found using the RL method (black and blue points), and optimizing the period of a trapezoidal cycle (red dots). The error bars represent the standard deviation of the return computed over 5 independent training runs. B) Corresponding values of the final average cooling power $\langle P_{\text{cool}} \rangle$ and of the coefficient of performance η_{cool} found using the RL method (black and blue dots), optimizing the trapezoidal cycle (red dots), and using the RL method of Ref. (69) (purple cross). Results for each of the 5 repetitions are shown as separate points to visualize the variability across multiple trainings. C–F) Final deterministic cycles identified by the RL method (thick black dots), as a function of time, corresponding to the blue points in panels (A) and (B) (respectively, for $c = 1, 0.8, 0.6, 0.4$ choosing the training run with the largest return). The dashed line represents the trapezoidal cycle that maximizes the return for the same value of c [not shown in panel (F) since no cycle yields a positive return]. The parameters used for training are chosen as in Fig. 3.

Here, at every time step, we let the agent choose which bath (if any) to couple to the oscillator. The coupling to the baths, characterized by the thermalization rates Γ_α, is modeled using the Lindblad master equation as in Ref. (43) (see "Physical model" in Materials and Methods for details). In contrast to the superconducting qubit case, c is held constant during training.

Fig. 5 reports the results on the optimal tradeoffs between extracted power and efficiency η_{heat}, the latter defined as the ratio between the extracted power and the input heat, in the same style of Fig. 4. In this setup, we compare our RL-based results to the well-known Otto cycle. The authors of Ref. (43) study this system by optimizing the switching times of an Otto cycle, i.e. the duration of each of the four segments, shown as a dashed lines in Fig. 5D and E, composing the cycle (see "Comparing with other methods" in Materials and Methods for details).

The RL method produces cycles with a larger return and with a better power-efficiency Pareto-front with respect to the Otto cycle (Fig. 5B and C). The cycles found by the RL method significantly outperforms the Otto engine in terms of delivered power. For $c = 1$, a high-power cycle is found (Fig. 5D and corresponding blue dots in Figs. 5B-C) but at the cost of a lower efficiency than the Otto cycles. However, at $c = 0.5$, the RL method finds a cycle that matches the maximum efficiency of the Otto cycles, while delivering a $\sim 30\%$ higher power (Fig. 5E and corresponding blue dots in Fig. 5B and C) Remarkably, our model-free RL method also finds cycles with nearly the same power as the RL method of Ref. (69), but at almost twice the efficiency (see "Comparing with other

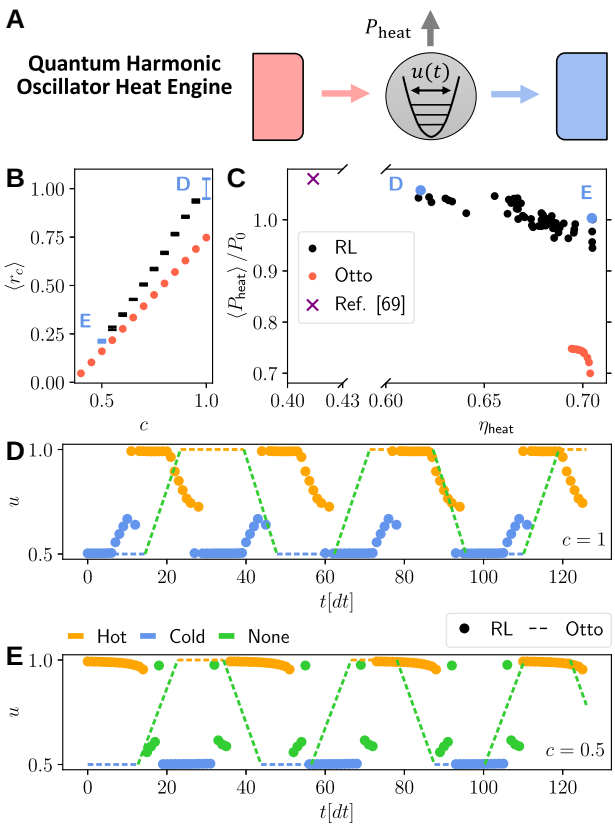

Fig. 5. Results for the optimization of the harmonic oscillator heat engine model. A) Schematic representation of the energy levels of the particles (black horizontal lines) trapped in a harmonic potential (parabolic curve) whose amplitude is controlled by $u(t)$. The vertical gray arrow represents the extracted power, while the colored horizontal arrows represent the heat fluxes. B) Final value of $\langle r_c \rangle$, as a function of c, found using the RL method (black and blue dots), and optimizing the Otto cycle (red dots). The error bars represent the standard deviation of the return computed over five independent training runs. C) Corresponding values of the average power $\langle P_{\text{heat}} \rangle / P_0$ and of the efficiency η_{heat} found using the RL method (black and blue dots), optimizing the Otto cycle (red dots), and using the RL method of Ref. (69) (purple cross). Results for each of the five repetitions are shown as separate points to visualize the variability across multiple trainings. D,E) Final deterministic cycle identified by the RL method (thick dots), as a function of time, corresponding to the blue points in panels (B) and (C) (respectively, $c = 1$, 0.5 choosing the training run with the largest return). The color corresponds to the discrete choice $d = \{\text{Hot, Cold, None}\}$ (see legend). The dashed line represents the Otto cycle that maximizes the return for the same value of c. The parameters used for training are $N = 128$, $\Gamma^{(\text{H})} = \Gamma^{(\text{C})} = 0.6$, $\beta_{\text{H}} = 0.2$, $\beta_{\text{C}} = 2$, $w_0 = 2$, $\mathcal{U} = [0.5, 1]$ [to enable a fair comparison with Ref. (43)], $\Delta t = 0.2$, $\gamma = 0.999$, $P_0 = 0.175$, and $\Sigma_0 = 0.525$.

methods" in Materials and Methods for details). As in Fig. 4, we see that a very small decrease in power can lead to a large efficiency increase.

Interestingly, as shown in Fig. 5D and E, the cycles found by the RL agent share many similarities with the Otto cycle: both alternate between the hot and cold bath (orange and blue portions) with a similar period. However, there are some differences: at $c = 1$, the RL cycle ramps the value of u while in contact with the bath, eliminating the unitary stroke (Fig. 5D). Instead, at $c = 0.5$, the RL agent employs a unitary stroke that is quite different respect to a linear ramping of u (Fig. 5E, green dots). As in the superconducting qubit case, the enhanced performance of the RL cycle may be interpreted as a mitigation of quantum friction (43, 93).

Also in this setup, we verified that the discovered cycles are quite robust across the five independent training runs, displaying only minor changes (see Fig. 9 of Methods for two cycles found in independent training runs corresponding to Fig. 5D and E).

Discussion

We introduced a model-free framework, based on RL, to discover Pareto-optimal thermodynamic cycles that describe the best possible tradeoff between power and efficiency of out-of-equilibrium QTMs (heat engines and refrigerators). Our algorithm only requires monitoring the heat fluxes of the QTM, making it a model-free approach. It can therefore be used both for the theoretical optimization of known systems, and potentially for the direct optimization of experimental devices for which no model is known, and in the absence of any measurement performed on the quantum system. Using state-of-the-art machine learning techniques, we demonstrate the validity of our method applying it to two different prototypical setups. Our black-box method discovered elaborate cycles that outperform previously proposed cycles and are on par with a previous RL method that observes the full quantum state (69). Up to minor details, the cycles found by our method are reproducible across independent training runs. Physically, we find that Otto cycles, commonly studied in the literature, are not generally optimal, and that optimal cycles balance a fast operation of the cycle, with the mitigation of quantum friction.

Our method paves the way for a systematic use of RL in the field of quantum thermodynamics. Future directions include investing larger systems to uncover the impact of quantum many-body effects on the performance of QTMs, optimizing systems in the presence of noise, and optimizing tradeoffs that include power fluctuations (99, 108–110).

Materials and methods

In this section, we provide details on the optimization of the entropy production, on the RL implementation, on the physical model used to describe the QTMs, on the training details, on the convergence of the method, on the comparison with other methods, and on the computation of the generation of coherence during the cycles. We also provide access to the full code that was used to generate the results presented in the manuscript, and the corresponding data.

Optimizing the entropy production

Here, we discuss the relation between optimizing the power and the entropy production, or the power and the efficiency. We start by noticing that we can express the efficiency of a heat engine η_{heat} and the coefficient of performance of a refrigerator η_{cool} in terms of the averaged power and entropy production, i.e.

$$\eta_\nu = \eta_\nu^{(\text{c})} \left[1 + \langle \Sigma \rangle / (\beta_\nu \langle P_\nu \rangle)\right]^{-1}, \tag{9}$$

where $\nu = \text{heat, cool}$, $\eta_{\text{heat}}^{(\text{c})} \equiv 1 - \beta_{\text{H}}/\beta_{\text{C}}$ is the Carnot efficiency, $\eta_{\text{cool}}^{(\text{c})} \equiv \beta_{\text{H}}/(\beta_{\text{C}} - \beta_{\text{H}})$ is the Carnot coefficient of performance, and where we defined $\beta_{\text{heat}} \equiv \beta_{\text{C}}$ and $\beta_{\text{cool}} \equiv \beta_{\text{C}} - \beta_{\text{H}}$. We now show that, thanks to this dependence of η_ν on $\langle P_\nu \rangle$ and $\langle \Sigma \rangle$, if a cycle is a Pareto-optimal tradeoff between high power and high efficiency, then it is also a Pareto-optimal tradeoff between high power and low entropy-production up to a change of c. This

means that if we find all optimal tradeoffs between high power and low entropy-production (as we do with our method if the Pareto-front is convex), we will have necessarily also found all Pareto-optimal tradeoffs between high power and high efficiency.

Mathematically, we want to prove that the cycles that maximize

$$\langle G_v(c)\rangle \equiv c\langle P_v\rangle + (1-c)\eta_v \tag{10}$$

for some value of $c \in [0, 1]$, also maximize the return in Eq. 5 for some (possibly different) value of $c \in [0, 1]$. To simplify the proof and the notation, we consider the following two functions

$$\begin{aligned} F(a, b, \theta) &= aP(\theta) - b\Sigma(P(\theta), \eta(\theta)),\\ G(a, b, \theta) &= aP(\theta) + b\eta(\theta), \end{aligned} \tag{11}$$

where $P(\theta)$ and $\eta(\theta)$ represent the power and efficiency of a cycle parameterized by a set of parameters θ, $a > 0$ and $b > 0$ are two scalar quantities, and

$$\Sigma(P, \eta) = \frac{\eta_v^{(c)} - \eta}{\eta}\beta_v P \tag{12}$$

is obtained by inverting Eq. 9.

We wish to prove the following. Given some weights $a_1 > 0$ and $b_1 > 0$, let θ_1 be the value of θ that locally maximizes $G(a_1, b_1, \theta)$. Then, it is always possible to identify positive weights $a_2 > 0$, $b_2 > 0$ such that the same parameters θ_1 (i.e. the same cycle) is a local maximum for $F(a_2, b_2, \theta)$. In the following, we will use that

$$\partial_P\Sigma \geq 0, \quad \partial_\eta\Sigma < 0, \tag{13}$$

and that the Hessian $H^{(\Sigma)}$ of $\Sigma(P, \eta)$ is given by

$$H^{(\Sigma)} = \begin{pmatrix} 0 & -\beta_v\dfrac{\eta_v^{(c)}}{\eta^2} \\ -\beta_v\dfrac{\eta_v^{(c)}}{\eta^2} & 2\beta_v P\dfrac{\eta_v^{(c)}}{\eta^3} \end{pmatrix}. \tag{14}$$

Proof: by assumption, θ_1 is a local maximum for $G(a_1, b_1, \theta)$. Denoting with ∂_i the partial derivative in $(\theta)_i$, we thus have

$$0 = \partial_i G(a_1, b_1, \theta_1) = a_1\partial_i P(\theta_1) + b_1\partial_i\eta(\theta_1). \tag{15}$$

Now, let us compute the derivative in θ of $F(a_2, b_2, \theta_1)$, where $a_2 > 0$ and $b_2 > 0$ are two arbitrary positive coefficients. We have

$$\partial_i F(a_2, b_2, \theta_1) = (a_2 - b_2\partial_P\Sigma)\partial_i P(\theta_1) - (b_2\partial_\eta\Sigma)\partial_i\eta(\theta_1). \tag{16}$$

Therefore, if we choose a_2 and b_2 such that

$$\begin{pmatrix} a_1 \\ b_1 \end{pmatrix} = \begin{pmatrix} 1 & -\partial_P\Sigma \\ 0 & -\partial_\eta\Sigma \end{pmatrix}\begin{pmatrix} a_2 \\ b_2 \end{pmatrix}, \tag{17}$$

thanks to Eq. 15 we have that

$$0 = \partial_i F(a_2, b_2, \theta_1), \tag{18}$$

meaning that the same parameters θ_1 that nullifies the gradient of G, nullifies also the gradient of F at a different choice of the weights, given by Eq. 17. The invertibility of Eq. 17 (i.e. a nonnull determinant of the matrix) is guaranteed by Eq. 13. We also have to make sure that if $a_1 > 0$ and $b_1 > 0$, then also $a_2 > 0$ and $b_2 > 0$. To do this, we invert Eq. 17, finding

$$\begin{pmatrix} a_2 \\ b_2 \end{pmatrix} = \begin{pmatrix} 1 & -\partial_P\Sigma/(\partial_\eta\Sigma) \\ 0 & -1/(\partial_\eta\Sigma) \end{pmatrix}\begin{pmatrix} a_1 \\ b_1 \end{pmatrix}. \tag{19}$$

It is now easy to see that also the weights a_2 and b_2 are positive using Eq. 13.

To conclude the proof, we show that θ_1 is a local maximum for $F(a_2, b_2, \theta)$ by showing that its Hessian is negative semi-definite. Since, by hypothesis, θ_1 is a local maximum for $G(a_1, b_1, \theta)$, we have that the Hessian matrix

$$H_{ij}^{(G)} \equiv \partial_{ij}G(a_1, b_1, \theta_1) = a_1\partial_{ij}P + b_1\partial_{ij}\eta \tag{20}$$

is negative semi-definite. We now compute the Hessian $H^{(F)}$ of $F(a_2, b_2, \theta)$ in $\theta = \theta_1$:

$$H_{ij}^{(F)} = a_2\partial_{ij}P - b_2[\partial_P\Sigma\,\partial_{ij}P + \partial_\eta\Sigma\,\partial_{ij}\eta + Q_{ij}], \tag{21}$$

where

$$Q_{ij} = (\partial_i P \quad \partial_i\eta)H^{(\Sigma)}\begin{pmatrix} \partial_j P \\ \partial_j\eta \end{pmatrix}, \tag{22}$$

and $H^{(\Sigma)}$ is the Hessian of $\Sigma(P, \eta)$ computed in $P(\theta_1)$ and $\eta(\theta_1)$. Since we are interested in studying the Hessian of $F(a_2, b_2, \theta_1)$ in the special point (a_2, b_2) previously identified, we substitute Eq. 19 into Eq. 21, yielding

$$H_{ij}^{(F)} = H_{ij}^{(G)} + \frac{b_1}{\partial_\eta\Sigma}Q_{ij}. \tag{23}$$

We now prove that $H_{ij}^{(F)}$ is negative semi-definite since it is the sum of negative semi-definite matrices. By hypothesis $H_{ij}^{(G)}$ is negative semi-definite. Recalling Eq. 13 and that $b_1 > 0$, we now need to show that Q_{ij} is positive semi-definite. Plugging Eq. 14 into Eq. 22 yields

$$Q_{ij} = \beta_{[v]}\frac{\eta_{[v]}^{(c)}}{\eta^2}\partial_i\eta\partial_j\eta R_{ij}, \tag{24}$$

where

$$R_{ij} \equiv 2\frac{P}{\eta} + S_{ij} + S_{ij}^T, \quad S_{ij} = -\frac{\partial_i P}{\partial_i\eta}. \tag{25}$$

We now show that if R_{ij} is positive semi-definite, then also Q_{ij} is positive semi-definite. By definition, Q_{ij} is positive semidefinite if, for any set of coefficient a_i, we have that $\sum_{ij} a_i Q_{ij} a_j \geq 0$. Assuming R_{ij} to be positive semi-definite, and using that $\beta_{[v]}$, $\eta_{[v]}^{(c)}$, $\eta > 0$, we have

$$\sum_{ij} a_i Q_{ij} a_j = \beta_{[v]}\frac{\eta_{[v]}^{(c)}}{\eta^2}\sum_{ij} x_i R_{ij} x_j \geq 0, \tag{26}$$

where we define $x_i \equiv \partial_i\eta a_i$. We thus have to prove the positivity of R_{ij}. We prove this showing that it is the sum of 3 positive semi-definite matrices. Indeed, the first term in Eq. 25, $2P/\eta$, is proportional to a matrix with 1 in all entries. Trivially, this matrix has 1 positive eigenvalue, and all other ones are null, so it is positive semi-definite. At last, S_{ij} and its transpose have the same positivity, so we focus only on S_{ij}. S_{ij} is a matrix with all equal columns. This means that it has all null eigenvalues, except for a single one that we denote with λ. Since the trace of a matrix is equal to the sum of the eigenvalues, we have $\lambda = \mathrm{Tr}[S] = \sum_i S_{ii}$. Using the optimality condition in Eq. 15, we see that each entry of S is positive, i.e. $S_{ij} > 0$. Therefore $\lambda > 0$, thus S is positive semi-definite, concluding the proof that $H_{ij}^{(F)}$ is negative semi-definite.

To conclude, we notice that we can always renormalize a_2 and b_2, preserving the same exact optimization problem. This way, a value of $c \in [0, 1]$ can be identified.

RL implementation

As discussed in the main text, our goal is to maximize the return $\langle r_c\rangle(t)$ defined in Eq. 5. To solve the problem within the RL

framework, we discretize time as $t_i = i\Delta t$. At every time-step t_i, the aim of the agent is to learn an optimal policy that maximizes, in expectation, the time-discretized return $\langle r_c \rangle_i$. The time-discrete reward and return functions are given by:

$$r_{i+1} = \Delta t^{-1} \int_{t_i}^{t_i+\Delta_t} r_c(t)\, dt, \tag{27}$$

$$\langle r_c \rangle_i = (1-\gamma) \sum_{j=0}^{\infty} \gamma^j r_{i+1+j}. \tag{28}$$

Equation 28 is the time-discrete version of Eq. 5, where the *discount factor* $\gamma = \exp(-\kappa \Delta t)$ determines the averaging timescale and expresses how much we are interested in future or immediate rewards.

To be precise, plugging Eq. 27 into Eq. 28 gives $\langle r_c \rangle(t)$ (up to an irrelevant constant prefactor) only in the limit of $\Delta t \to 0$. However, also for finite Δt, both quantities are time-averages of the reward, so they are equally valid definitions to describe a long-term tradeoff maximization.

As in Ref. (69), we use a generalization of the soft-actor critic (SAC) method, first developed for continuous actions (70, 71), to handle a combination of discrete and continuous actions (72, 73). We further tune the method to stabilize the convergence in a multiobjective scenario. We here present an overview of our implementation of SAC putting special emphasis on the differences with respect to the standard implementation. However, we refer to (70–73) for additional details. Our method, implemented with PyTorch, is based on modifications and generalizations of the SAC implementation provided by Spinning Up from OpenAI (111). All code and data to reproduce the experiments is available online (see Data Availability and Code Availability sections).

The SAC algorithm is based on policy iteration, i.e. it consists of iterating multiple times over two steps: a *policy evaluation step*, and a *policy improvement step*. In the policy evaluation step, the value function of the current policy is (partially) learned, whereas in the policy improvement step a better policy is learned by making use of the value function. We now describe these steps more in detail.

In typical RL problems, the optimal policy $\pi^*(s \mid a)$ is defined as the policy that maximizes the expected return defined in Eq. 28, i.e.:

$$\pi^* = \arg\max_{\pi} \mathop{\mathrm{E}}_{s\sim\mu_\pi} \left[\sum_{k=0}^{\infty} \gamma^k r_{k+1} \,\middle|\, s_0 = s \right], \tag{29}$$

where E_π denotes the expectation value choosing actions according to the policy π. The initial state $s_0 = s$ is sampled from μ_π, i.e. the steady-state distribution of states that are visited by π. In the SAC method, balance between exploration and exploitation (112) is achieved by introducing an Entropy-Regularized maximization objective. In this setting, the optimal policy π^* is given by

$$\pi^* = \arg\max_{\pi} \mathop{\mathrm{E}}_{s\sim\mathcal{B}} \left[\sum_{k=0}^{\infty} \gamma^k (r_{k+1} + \alpha H[\pi(\cdot \mid s_k)]) \,\middle|\, s_0 = s \right], \tag{30}$$

where $\alpha \geq 0$ is known as the "temperature" parameter that balances the tradeoff between exploration and exploitation, and

$$H[P] = \mathop{\mathrm{E}}_{x\sim P} [-\log P(x)] \tag{31}$$

is the entropy of the probability distribution P. Notice that we replaced the unknown state distribution μ_π with $\mathcal{B}$, which is a replay buffer populated during training by storing the observed one-step transitions $(s_k, a_k, r_{k+1}, s_{k+1})$.

Developing on Ref. (69), we generalize such approach to a combination of discrete and continuous actions in the following way. Let us write an arbitrary action a as $a = (u, d)$, where u is the continuous action and d is the discrete action (for simplicity, we describe the case of a single continuous action, though the generalization to multiple variables is straightforward). From now on, all functions of a are also to be considered as functions of u, d. We decompose the joint probability distribution of the policy as

$$\pi(u, d \mid s) = \pi_{\mathrm{D}}(d \mid s) \cdot \pi_{\mathrm{C}}(u \mid d, s), \tag{32}$$

where $\pi_{\mathrm{D}}(d \mid s)$ is the marginal probability of taking discrete action d, and $\pi_{\mathrm{C}}(u \mid d, s)$ is the conditional probability density of choosing action u, given action d (D stands for "discrete", and C for "continuous"). Notice that this decomposition is an exact identity, thus allowing us to describe correlations between the discrete and the continuous action. With this decomposition, we can write the entropy of a policy as

$$H[\pi(\cdot \mid s)] = H^\pi_{\mathrm{D}}(s) + H^\pi_{\mathrm{C}}(s), \tag{33}$$

where

$$H^\pi_{\mathrm{D}}(s) = H[\pi_{\mathrm{D}}(\cdot \mid s)], \quad H^\pi_{\mathrm{C}}(s) = \sum_d \pi_{\mathrm{D}}(d \mid s) H[\pi_{\mathrm{C}}(\cdot \mid d, s)], \tag{34}$$

correspond, respectively, to the entropy contribution of the discrete (D) and continuous (C) part. These two entropies take on values in different ranges: while the entropy of a discrete distribution with $|D|$ discrete actions is nonnegative and upper bounded by $\log(|D|)$, the (differential) entropy of a continuous distribution can take on any value, including negative values (especially for peaked distributions). Therefore, we introduce a separate temperature for the discrete and continuous contributions replacing the definition of the optimal policy in Eq. 30 with

$$\pi^* = \arg\max_{\pi} \mathop{\mathrm{E}}_{s\sim\mathcal{B}} \left[\sum_{k=0}^{\infty} \gamma^k (r_{k+1} + \alpha_{\mathrm{D}} H^\pi_{\mathrm{D}}(s_k) + \alpha_{\mathrm{C}} H^\pi_{\mathrm{C}}(s_k)) \,\middle|\, s_0 = s \right], \tag{35}$$

where $\alpha_{\mathrm{C}} \geq 0$ and $\alpha_{\mathrm{D}} \geq 0$ are two distinct "temperature" parameters. This is one of the differences with respect to Refs. (69–71). Equation 35 defines our optimization objective. Accordingly, we define the value function $Q^\pi(s, a)$ of a given policy π as

$$Q^\pi(s, a) = \mathrm{E}_\pi \left[r_1 + \sum_{k=1}^{\infty} \gamma^k (r_{k+1} + \alpha_{\mathrm{D}} H^\pi_{\mathrm{D}}(s_k) + \alpha_{\mathrm{C}} H^\pi_{\mathrm{C}}(s_k)) \,\middle|\, s_0 = s,\, a_0 = a \right]. \tag{36}$$

Its recursive Bellman equation therefore reads

$$Q^\pi(s, a) = \mathop{\mathrm{E}}_{\substack{s_1 \\ a_1\sim\pi(\cdot\mid s_1)}} [r_1 + \gamma(Q^\pi(s_1, a_1) + \alpha_{\mathrm{D}} H^\pi_{\mathrm{D}}(s_1) + \alpha_{\mathrm{C}} H^\pi_{\mathrm{C}}(s_1)) \mid s_0 = s,\, a_0 = a]. \tag{37}$$

As in Ref. (70, 71), we parameterize $\pi_{\mathrm{C}}(u \mid d, s)$ as a squashed Gaussian policy, i.e. as the distribution of the variable

$$\tilde{u}(\xi \mid d, s) = u_{\mathrm{a}} + \tfrac{u_{\mathrm{b}} - u_{\mathrm{a}}}{2} [1 + \tanh(\mu(d, s) + \sigma(d, s) \cdot \xi)], \quad \xi \sim \mathcal{N}(0, 1), \tag{38}$$

where $\mu(d, s)$ and $\sigma(d, s)$ represent, respectively, the mean and standard deviation of the Gaussian distribution, $\mathcal{N}(0, 1)$ is the normal distribution with zero mean and unit variance, and where we assume that $\mathcal{U} = [u_{\mathrm{a}}, u_{\mathrm{b}}]$. This is the so-called reparameterization trick.

We now describe the policy evaluation step. In the SAC algorithm, we learn two value functions $Q_{\phi_i}(s, a)$ described by the learnable parameters ϕ_i, for $i = 1, 2$. $Q_\phi(s, a)$ is a function approximator, e.g. a neural network. Since $Q_{\phi_i}(s, a)$ should satisfy the Bellman Eq. 37, we define the loss function for $Q_{\phi_i}(s, a)$ as the mean square

difference between the left and right-hand side of Eq. 37, i.e.

$$L_{\mathrm{Q}}(\boldsymbol{\phi}_{\mathrm{i}}) = \mathop{\mathrm{E}}_{(s,a,r,s')\sim\mathcal{B}}[(Q_{\boldsymbol{\phi}_{\mathrm{i}}}(s,a) - y(r,s'))^2], \tag{39}$$

where

$$y(r,s') = r + \gamma \mathop{\mathrm{E}}_{a'\sim\boldsymbol{\pi}(\cdot\,|\,s')}\left[\min_{j=1,2} Q_{\boldsymbol{\phi}_{\mathrm{targ},j}}(s',a') + \alpha_{\mathrm{D}} H_{\mathrm{D}}(s') + \alpha_{\mathrm{C}} H_{\mathrm{C}}(s')\right]. \tag{40}$$

Notice that in Eq. 40 we replaced $Q^{\boldsymbol{\pi}}$ with $\min_{j=1,2} Q_{\boldsymbol{\phi}_{\mathrm{targ},j}}$, where $\boldsymbol{\phi}_{\mathrm{targ},j}$, for $j = 1, 2$, are target parameters which are not updated when minimizing the loss function; instead, they are held fixed during backpropagation, and then they are updated according to Polyak averaging, i.e.

$$\boldsymbol{\phi}_{\mathrm{targ,i}} \leftarrow \rho_{\mathrm{polyak}}\boldsymbol{\phi}_{\mathrm{targ,i}} + (1-\rho_{\mathrm{polyak}})\boldsymbol{\phi}_{\mathrm{i}}, \tag{41}$$

where ρ_{polyak} is a hyperparameter. This change was shown to improve learning (70, 71). In order to evaluate the expectation value in Eq. 40, we use the decomposition in Eq. 32 to write

$$\mathop{\mathrm{E}}_{a'\sim\boldsymbol{\pi}(\cdot\,|\,s')}[\cdot] = \sum_{d'} \boldsymbol{\pi}_{\mathrm{D}}(d'\,|\,s') \mathop{\mathrm{E}}_{u'\sim\boldsymbol{\pi}_{\mathrm{C}}(\cdot\,|\,d',s')}[\cdot], \tag{42}$$

where we denote $a' = (u', d')$. Plugging Eq. 42 into Eq. 40 and writing the entropies explicitly as expectation values yields

$$\begin{aligned} y(r,s') = r + \gamma \sum_{d'} \boldsymbol{\pi}_{\mathrm{D}}(d'\,|\,s') \\ \cdot\Bigg(\mathop{\mathrm{E}}_{u'\sim\boldsymbol{\pi}_{\mathrm{C}}(\cdot\,|\,d',s')}\left[\min_{j=1,2} Q_{\boldsymbol{\phi}_{\mathrm{targ},j}}(s',d',u') - \alpha_{\mathrm{C}}\log\boldsymbol{\pi}_{\mathrm{C}}(u'\,|\,d',s')\right] \\ -\alpha_{\mathrm{D}}\log\boldsymbol{\pi}_{\mathrm{D}}(d'\,|\,s')\Bigg). \end{aligned} \tag{43}$$

We then replace the expectation value over u' in Eq. 43 with a single sampling $u' \sim \boldsymbol{\pi}_{\mathrm{C}}(\cdot\,|\,d', s')$ (therefore one sampling for each discrete action) performed using Eq. 38. This corresponds to performing a full average over the discrete action, and a single sampling of the continuous action.

We now turn to the policy improvement step. Since we introduced two separate temperatures, we cannot use the loss function introduced in Refs. (70, 71). Therefore, we proceed in two steps. Let us define the following function

$$Z_{\boldsymbol{\pi}}(s) = -\mathop{\mathrm{E}}_{a\sim\boldsymbol{\pi}(\cdot\,|\,s)}[Q^{\boldsymbol{\pi}^{\mathrm{old}}}(s,a)] - \alpha_{\mathrm{D}} H^{\boldsymbol{\pi}}_{\mathrm{D}}(s) - \alpha_{\mathrm{C}} H^{\boldsymbol{\pi}}_{\mathrm{C}}(s), \tag{44}$$

where $Q^{\boldsymbol{\pi}^{\mathrm{old}}}(s,a)$ is the value function of some given "old policy" $\boldsymbol{\pi}^{\mathrm{old}}$, and $\boldsymbol{\pi}$ is an arbitrary policy. First, we prove that if a policy $\boldsymbol{\pi}^{\mathrm{new}}$ satisfies

$$Z_{\boldsymbol{\pi}^{\mathrm{new}}}(s) \le Z_{\boldsymbol{\pi}^{\mathrm{old}}}(s) \tag{45}$$

for all values of s, then $\boldsymbol{\pi}^{\mathrm{new}}$ is a better policy than $\boldsymbol{\pi}^{\mathrm{old}}$ as defined in Eq. 35. Next, we will use this property to define a loss function that implements the policy improvement step. Equation 45 implies that

$$\begin{aligned} &\mathop{\mathrm{E}}_{a\sim\boldsymbol{\pi}^{\mathrm{old}}(\cdot\,|\,s)}[Q^{\boldsymbol{\pi}^{\mathrm{old}}}(s,a)] + \alpha_{\mathrm{D}} H^{\boldsymbol{\pi}^{\mathrm{old}}}_{\mathrm{D}}(s) + \alpha_{\mathrm{C}} H^{\boldsymbol{\pi}^{\mathrm{old}}}_{\mathrm{C}}(s) \\ &\le \mathop{\mathrm{E}}_{a\sim\boldsymbol{\pi}^{\mathrm{new}}(\cdot\,|\,s)}[Q^{\boldsymbol{\pi}^{\mathrm{old}}}(s,a)] + \alpha_{\mathrm{D}} H^{\boldsymbol{\pi}^{\mathrm{new}}}_{\mathrm{D}}(s) + \alpha_{\mathrm{C}} H^{\boldsymbol{\pi}^{\mathrm{new}}}_{\mathrm{C}}(s). \end{aligned} \tag{46}$$

We now use this inequality to show that $\boldsymbol{\pi}^{\mathrm{new}}$ is a better policy. Starting from the Bellmann equation 37 for $Q^{\boldsymbol{\pi}^{\mathrm{old}}}$, we have Eq. 47.

$$\begin{aligned} Q^{\boldsymbol{\pi}^{\mathrm{old}}}(s,a) &= \mathop{\mathrm{E}}_{\substack{s_1\\ a_1\sim\boldsymbol{\pi}^{\mathrm{old}}(\cdot|s_1)}}\left[r_1 + \gamma(Q^{\boldsymbol{\pi}^{\mathrm{old}}}(s_1,a_1) + \alpha_{\mathrm{D}} H^{\boldsymbol{\pi}^{\mathrm{old}}}_{\mathrm{D}}(s_1) + \alpha_{\mathrm{C}} H^{\boldsymbol{\pi}^{\mathrm{old}}}_{\mathrm{C}}(s_1)) \,|\, s_0 = s, a_0 = a\right] \\ &\le \mathop{\mathrm{E}}_{\substack{s_1\\ a_1\sim\boldsymbol{\pi}^{\mathrm{new}}(\cdot|s_1)}}\left[r_1 + \gamma(Q^{\boldsymbol{\pi}^{\mathrm{old}}}(s_1,a_1) + \alpha_{\mathrm{D}} H^{\boldsymbol{\pi}^{\mathrm{new}}}_{\mathrm{D}}(s_1) + \alpha_{\mathrm{C}} H^{\boldsymbol{\pi}^{\mathrm{new}}}_{\mathrm{C}}(s_1)) \,|\, s_0 = s, a_0 = a\right] \\ &= \mathop{\mathrm{E}}_{\substack{s_1\\ a_1\sim\boldsymbol{\pi}^{\mathrm{new}}(\cdot|s_1)}}\left[r_1 + \gamma(\alpha_{\mathrm{D}} H^{\boldsymbol{\pi}^{\mathrm{new}}}_{\mathrm{D}}(s_1) + \alpha_{\mathrm{C}} H^{\boldsymbol{\pi}^{\mathrm{new}}}_{\mathrm{C}}(s_1)) \,|\, s_0 = s, a_0 = a\right] \\ &\quad + \gamma \mathop{\mathrm{E}}_{\substack{s_1\\ a_1\sim\boldsymbol{\pi}^{\mathrm{new}}(\cdot|s_1)}}\left[Q^{\boldsymbol{\pi}^{\mathrm{old}}}(s_1,a_1) \,|\, s_0 = s, a_0 = a\right] \\ &\le \cdots \le Q^{\boldsymbol{\pi}^{\mathrm{new}}}(s,a). \end{aligned} \tag{47}$$

Using a strategy similar to that described in Refs. (70, 112), in Eq. 47, we make a repeated use of inequality 46 and of the Bellmann equation for $Q^{\boldsymbol{\pi}^{\mathrm{old}}}(s,a)$ to prove that the value function of $\boldsymbol{\pi}^{\mathrm{new}}$ is better or equal to the value function of $\boldsymbol{\pi}^{\mathrm{old}}$.

Let $\boldsymbol{\pi}_\theta(a\,|\,s)$ be a parameterization of the policy function that depends on a set of learnable parameters $\boldsymbol{\theta}$. We define the following loss function

$$L_{\boldsymbol{\pi}}(\boldsymbol{\theta}) = \mathop{\mathrm{E}}_{\substack{s\sim\mathcal{B}\\ a\sim\boldsymbol{\pi}_\theta(\cdot|s)}}[-Q^{\boldsymbol{\pi}^{\mathrm{old}}}(s,a) - \alpha_{\mathrm{D}} H^{\boldsymbol{\pi}_\theta}_{\mathrm{D}}(s) - \alpha_{\mathrm{C}} H^{\boldsymbol{\pi}_\theta}_{\mathrm{C}}(s)]. \tag{48}$$

Thanks to Eqs. 44 and 45, this choice guarantees us to find a better policy by minimizing $L_{\boldsymbol{\pi}}(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$. In order to evaluate the expectation value in Eq. 48, as before we explicitly average over the discrete action and perform a single sample of the continuous action, and we replace $Q^{\boldsymbol{\pi}^{\mathrm{old}}}$ with $\min_j Q_{\boldsymbol{\phi}_j}$. Recalling the parameterization in Eq. 38, this yields

$$\begin{aligned} L_{\boldsymbol{\pi}}(\boldsymbol{\theta}) = \mathop{\mathrm{E}}_{s\sim\mathcal{B}}\Bigg[\sum_d \boldsymbol{\pi}_{\mathrm{D},\theta}(d\,|\,s)\Big(\alpha_{\mathrm{D}}\log\boldsymbol{\pi}_{\mathrm{D},\theta}(d\,|\,s) \\ + \alpha_{\mathrm{C}}\log\boldsymbol{\pi}_{\mathrm{C},\theta}(\tilde{u}_\theta(\xi\,|\,d,s)\,|\,d,s) - \min_{j=1,2} Q_{\boldsymbol{\phi}_j}(s,\tilde{u}_\theta(\xi\,|\,d,s),d)\Big)\Bigg], \\ \xi\sim\mathcal{N}(0,1). \end{aligned} \tag{49}$$

We have defined and shown how to evaluate the loss functions $L_{\mathrm{Q}}(\boldsymbol{\phi})$ and $L_{\boldsymbol{\pi}}(\boldsymbol{\theta})$ that allow us to determine the value function and the policy [see Eqs. 39, 43 and 49]. Now, we discuss how to automatically tune the temperature hyperparameters α_{D} and α_{C}. Ref. (71) shows that constraining the average entropy

of the policy to a certain value leads to the same exact SAC algorithm with the addition of an update rule to determine the temperatures. Let $\bar{H}_D$ and $\bar{H}_C$ be, respectively, the fixed average values of the entropy of the discrete and continuous part of the policy. We can then determine the corresponding temperatures α_D and α_C minimizing the following two loss functions

$$\begin{aligned} L_D(\alpha_D) &= \alpha_D \mathop{\mathrm{E}}_{s\sim\mathcal{B}}[H_D^\pi(s) - \bar{H}_D], \\ L_C(\alpha_C) &= \alpha_C \mathop{\mathrm{E}}_{s\sim\mathcal{B}}[H_C^\pi(s) - \bar{H}_C]. \end{aligned} \tag{50}$$

As usual, we evaluate the entropies by explicitly taking the average over the discrete actions, and taking a single sample of the continuous action. To be more specific, we evaluate L_D by computing

$$L_D(\alpha_D) = \alpha_D \mathop{\mathrm{E}}_{s\sim\mathcal{B}}\left[-\sum_d \pi_D(d\,|\,s)\log\pi_D(d\,|\,s) - \bar{H}_D\right], \tag{51}$$

and L_C by computing

$$L_C(\alpha_C) = \alpha_C \cdot \mathop{\mathrm{E}}_{s\sim\mathcal{B}}\left[-\sum_d \pi_D(d\,|\,s) \mathop{\mathrm{E}}_{u\sim\pi_C(\cdot\,|\,d,\,s)}[\log\pi_C(u\,|\,d,\,s)] - \bar{H}_C\right] \tag{52}$$

and replacing the expectation value over u with a single sample.

To summarize, the SAC algorithm consists of repeating over and over a policy evaluation step, a policy improvement step, and a step where the temperatures are updated. The policy evaluation step consists of a single optimization step to minimize the loss functions $L_Q(\phi_i)$ (for $i = 1, 2$), given in Eq. 39, where $y(r, s')$ is computed using Eq. 43. The policy improvement step consists of a single optimization step to minimize the loss function $L_\pi(\theta)$ given in Eq. 49. The temperatures are then updated performing a single optimization step to minimize $L_D(\alpha_D)$ and $L_C(\alpha_C)$ given, respectively, in Eqs. 51 and 52. In all loss functions, the expectation value over the states is approximated with a batch of experience sampled randomly from the replay buffer $\mathcal{B}$.

We now detail how we parameterize $\pi(a\,|\,s)$ and $Q(s, a)$. The idea is to develop an efficient way to process the state that can potentially be a long time-series of actions. To this aim, we introduce a "convolution block" as a building element for our NN architecture. The convolution block, detailed in Fig. 6, takes an input of size (C_{in}, L_{in}), where C_{in} is the number of channels (i.e. the number of parameters determining an action at every time-step) and L_{in} is the length of the time-series, and produces an output of size $(C_{out}, L_{out} = L_{in}/2)$, thus halving the length of the time-series. Notice that we include a skip connection (right branch in Fig. 6) to improve trainability (102).

Using the decomposition in Eq. 32 and the parameterization in Eq. 38, the quantities that need to be parameterized are the discrete probabilities $\pi_D(d\,|\,s)$, the averages $\mu(d, s)$ and the variances $\sigma(d, s)$, for $d = 1, \ldots, |D|$, $|D| = 3$ being the number of discrete actions. The architecture of the neural network that we use for the policy function is shown in Fig. 7A. The state, composed of the time-series $s_i = (a_{i-N}, \ldots, a_{i-1})$ which has shape $(C_{in}, L_{in} = N)$, is fed through a series of $\ln_2(N)$ convolutional blocks, which produce an output of length $(C_{out}, L = 1)$. The number of input channels C_{in} is determined by stacking the components of $\vec{u}$ (which, for simplicity, is a single real number u in this appendix) and by using a one-hot encoding of the discrete actions. We then feed this output, together with the last action which has a privileged position, to a series of fully connected NNs with ReLU activations. Finally, a linear network outputs $W(d\,|\,s)$, $\mu(d, s)$ and $\log(\sigma(d, s))$, for all $d = 1, \ldots, |D|$. The probabilities $\pi_D(d\,|\,s)$ are then produced applying the softmax operation to $W(d\,|\,s)$.

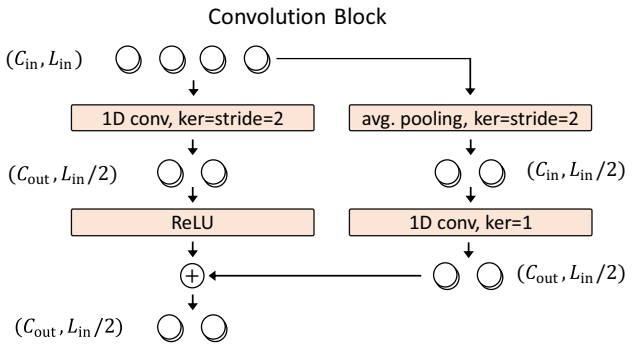

Fig. 6. Schematic representation of the convolution block that takes as input a 1D time-series of size (C_{in}, L_{in}), where L_{in} is the length of the series and C_{in} is the number of channels, and produces an output of size $(C_{out}, L_{in}/2)$. In this image $L_{in} = 4$. The output is produced by stacking a 1D convolution of kernel size and stride of 2, and a nonlinearity (left branch). A residual connection (right branch), consisting only of linear operations, is added to improve trainability.

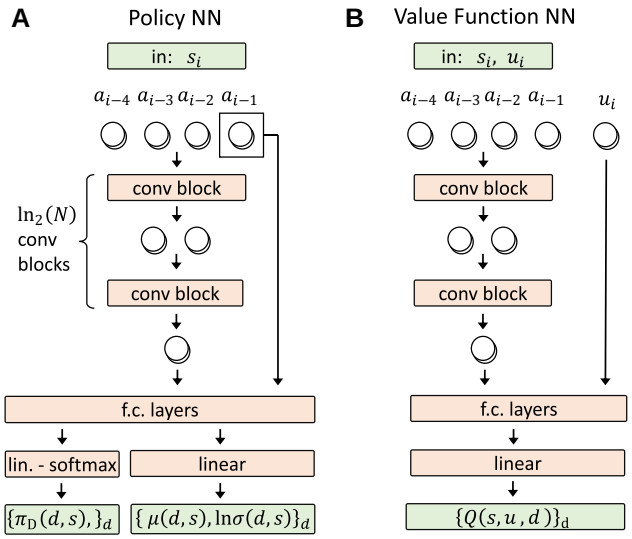

Fig. 7. Neural network architecture used to parameterize the policy $\pi(\vec{u}, d\,|\,s)$ A) and to parameterize the value function $Q(s, \vec{u}, d)$ B).

We parameterize the value function $Q_\phi(s, u, d)$ as in Fig. 7B. As for the policy function, the state s is fed through $\ln_2(N)$ stacked convolution blocks which reduce the length of the input to $(C_{out}, L = 1)$. This output, together with the action u, is fed into a series of fully connected layers with ReLU activations. We then add a linear layer that produces $|D|$ outputs, corresponding to the value of $Q(s, u, d)$ for each $d = 1, \ldots, |D|$.

At last, we discuss a further change to the current method that we implemented in the superconducting qubit refrigerator case to improve the converge. This idea is the following. The return $\langle r_c \rangle$ is a convex combination of the power and of the negative entropy production. The first term is positive when the system is delivering the desired power, while the second term is strictly negative. Therefore, for c close to 1, the optimal value of the return is some positive quantity. Instead, as c decreases, the optimal value of the return decreases, getting closer to zero (this can be seen explicitly in Figs. 4A and 5B). However, a null return can also be achieved by a trivial cycle that consists of doing nothing, i.e. of keeping the control constant in time. Indeed, this yields both

zero power, and zero entropy production. Therefore, as c decreases, it becomes harder and harder for the RL agent to distinguish good cycles from these trivial solutions. We thus modify our method to allow us to smoothly change the value of c during training from 1 to the desired final value, which allows to tackle an optimization problem by "starting from an easier problem" ($c = 1$), and gradually increasing its difficulty. This required the following modifications to the previously described method.

We introduce two separate value functions, one for each objective (P for the power, and Σ for the entropy production)

$$\begin{aligned} Q^{\pi}_{\mathrm{P}}(s,a) &= \mathrm{E}_{\pi}\Big[r^{(\mathrm{P})}_1 + \sum_{k=1}^{\infty}\gamma^k\big(r^{(\mathrm{P})}_{k+1} + \alpha_{\mathrm{D}}H^{\pi}_{\mathrm{D}}(s_k) \\ &\quad + \alpha_{\mathrm{C}}H^{\pi}_{\mathrm{C}}(s_k)\big)\Big|\, s_0 = s,\, a_0 = a\Big], \\ Q^{\pi}_{\Sigma}(s,a) &= \mathrm{E}_{\pi}\Big[r^{(\Sigma)}_1 + \sum_{k=1}^{\infty}\gamma^k\big(r^{(\Sigma)}_{k+1} + \alpha_{\mathrm{D}}H^{\pi}_{\mathrm{D}}(s_k) \\ &\quad + \alpha_{\mathrm{C}}H^{\pi}_{\mathrm{C}}(s_k)\big)\Big|\, s_0 = s,\, a_0 = a\Big], \end{aligned} \tag{53}$$

where

$$r^{(\mathrm{P})}_{i+1} \equiv \frac{1}{\Delta t}\int_{t_i}^{t_i+\Delta t}\frac{P(\tau)}{P_0}\,d\tau, \quad r^{(\Sigma)}_{i+1} \equiv \frac{1}{\Delta t}\int_{t_i}^{t_i+\Delta t}\frac{\Sigma(\tau)}{\Sigma_0}\,d\tau, \tag{54}$$

represent, respectively, the normalized average power and average entropy production during each time-step. Since the value functions in Eq. 53 are identical to Eq. 36 up to a change of the reward, they separately satisfy the same Bellmann equation as in Eq. 37, with r_1 replaced respectively with $r^{(\mathrm{P})}_1$ and $r^{(\Sigma)}_1$. Therefore, we learn each value functions minimizing the same loss function L_{Q} given in Eq. 39, with r_i replaced with $r^{(\mathrm{P})}_1$ or $r^{(\Sigma)}_1$. Both value functions are parameterized using the same architecture, but separate and independent parameters. We now turn to the determination of the policy. Comparing the definition of r_i given in the main text with Eq. 54, we see that $r_{i+1} = cr^{(\mathrm{P})}_{i+1} - (1-c)r^{(\Sigma)}_{i+1}$. Using this property, and comparing Eq. 36 with Eq. 53, we see that

$$Q^{\pi}(s,a) = cQ^{\pi}_{\mathrm{P}}(s,a) - (1-c)Q^{\pi}_{\Sigma}(s,a). \tag{55}$$

Therefore, we learn the policy minimizing the same loss function as in Eq. 49, using Eq. 55 to compute the value function. To summarize, this method allows us to vary c dynamically during training. This requires learning two value functions, one for each objective, and storing in the replay buffer the two separate rewards $r^{(\mathrm{P})}_i$ and $r^{(\Sigma)}_i$.

At last, when we refer to "final deterministic cycle", we are sampling from the policy function "switching off the stochasticity", i.e. choosing continuous actions u setting $\xi = 0$ in Eq. 38, and choosing deterministically the discrete action with the highest probability.

Physical model

As discussed in the main text, we describe the dynamics of the two analyzed QTMs employing the Lindblad master equation that can be derived also for nonadiabatic drivings (107), in the weak system-bath coupling regime performing the usual Born-Markov and secular approximation (104–106) and neglecting the Lamb-shift contribution. This approach describes the time-evolution of the reduced density matrix of the quantum system, $\hat{\rho}(t)$, under the assumption of weak system-bath interaction. Setting $\hbar = 1$, the master equation reads

$$\frac{\partial}{\partial t}\hat{\rho}(t) = -\mathrm{i}[\hat{H}[\vec{u}(t)], \hat{\rho}(t)] + \sum_{\alpha}\mathcal{D}^{(\alpha)}_{\vec{u}(t),d(t)}[\hat{\rho}(t)], \tag{56}$$

where $\hat{H}[\vec{u}(t)]$ is the Hamiltonian of the quantum system that depends explicitly on time via the control parameters $\vec{u}(t)$, $[\cdot,\cdot]$ denotes the commutator, and $\mathcal{D}^{(\alpha)}_{\vec{u}(t),d(t)}[\cdot]$, known as the dissipator, describes the effect of the coupling between the quantum system and bath $\alpha = \mathrm{H}, \mathrm{C}$. We notice that since the RL agent produces piece-wise constant protocols, we are not impacted by possible inaccuracies of the master equation subject to fast parameter driving (113), provided that Δt is not smaller than the bath timescale. Without loss of generality, the dissipators can be expressed as (105, 106)

$$\begin{aligned} \mathcal{D}^{(\alpha)}_{\vec{u}(t),d(t)} = \lambda_{\alpha}[d(t)]\sum_{k}\gamma^{(\alpha)}_{k,\vec{u}(t)}\Big(&\hat{\mathrm{A}}^{(\alpha)}_{k,\vec{u}(t)}\hat{\rho}\hat{\mathrm{A}}^{(\alpha)\dagger}_{k,\vec{u}(t)} \\ &-\frac{1}{2}\hat{\mathrm{A}}^{(\alpha)\dagger}_{k,\vec{u}(t)}\hat{\mathrm{A}}^{(\alpha)}_{k,\vec{u}(t)}\hat{\rho} - \frac{1}{2}\hat{\rho}\hat{\mathrm{A}}^{(\alpha)\dagger}_{k,\vec{u}(t)}\hat{\mathrm{A}}^{(\alpha)}_{k,\vec{u}(t)}\Big), \end{aligned} \tag{57}$$

where $\lambda_{\alpha}[d(t)] \in \{0,1\}$ are functions that determine which bath is coupled the quantum system, $\hat{\mathrm{A}}^{(\alpha)}_{k,\vec{u}(t)}$ are the Lindblad operators, and $\gamma^{(\alpha)}_{k,\vec{u}(t)}$ are the corresponding rates. In particular, $\lambda_{\mathrm{H}}(\mathrm{Hot}) = 1$, $\lambda_{\mathrm{C}}(\mathrm{Hot}) = 0$, while $\lambda_{\mathrm{H}}(\mathrm{Cold}) = 0$, $\lambda_{\mathrm{C}}(\mathrm{Cold}) = 1$, and $\lambda_{\mathrm{H}}(\mathrm{None}) = \lambda_{\mathrm{C}}(\mathrm{None}) = 0$. Notice that both the Lindblad operators and the rates can depend on time through the value of the control $\vec{u}(t)$. Their explicit form depends on the details of the system, i.e. on the Hamiltonian describing the dynamics of the *overall system* including the bath and the system-bath interaction. Below, we provide the explicit form of $\hat{\mathrm{A}}^{(\alpha)}_{k,\vec{u}(t)}$ and $\gamma^{(\alpha)}_{k,\vec{u}(t)}$ used to model the two setups considered in the manuscript. We adopt the standard approach to compute the instantaneous power and heat currents (114)

$$\begin{aligned} P(t) &\equiv -\mathrm{Tr}\Big[\hat{\rho}(t)\frac{\partial}{\partial t}\hat{H}[\vec{u}(t)]\Big], \\ J_{\alpha}(t) &\equiv \mathrm{Tr}\Big[\hat{H}[\vec{u}(t)]\,\mathcal{D}^{(\alpha)}_{\vec{u}(t),d(t)}\Big], \end{aligned} \tag{58}$$

that guarantees the validity of the first law of thermodynamics $\partial U(t)/(\partial t) = -P(t) + \sum_{\alpha}J_{\alpha}(t)$, the internal energy being defined as $U = \mathrm{Tr}[\hat{\rho}(t)\hat{H}[\vec{u}(t)]]$.

In the superconducting qubit refrigerator, we employ the model first put forward in Ref. (49), and further studied in Refs. (55, 63). In particular, we consider the following Lindblad operators and corresponding rates (identifying $k = \pm$):

$$\hat{\mathrm{A}}^{(\alpha)}_{+,u(t)} = -\mathrm{i}|e_{u(t)}\rangle\langle g_{u(t)}|, \quad \hat{\mathrm{A}}^{(\alpha)}_{-,u(t)} = +\mathrm{i}|g_{u(t)}\rangle\langle e_{u(t)}|, \tag{59}$$

where $|g_{u(t)}\rangle$ and $|e_{u(t)}\rangle$ are, respectively, the instantaneous ground state and excited state of Eq. 7. The corresponding rates are given by $\gamma^{(\alpha)}_{\pm,u(t)} = S_{\alpha}[\pm\Delta\epsilon_{u(t)}]$, where $\Delta\epsilon_{u(t)}$ is the instantaneous energy gap of the system, and

$$S_{\alpha}(\Delta\epsilon) = \frac{g_{\alpha}}{2}\frac{1}{1+\mathrm{Q}^2_{\alpha}(\Delta\epsilon/\omega_{\alpha} - \omega_{\alpha}/\Delta\epsilon)^2}\frac{\Delta\epsilon}{e^{\beta_{\alpha}\Delta\epsilon}-1} \tag{60}$$

is the noise power spectrum of bath α. Here ω_{α}, Q_{α} and g_{α} are the base resonance frequency, quality factor and coupling strength of the resonant circuit acting as bath $\alpha = \mathrm{H}, \mathrm{C}$ (see Refs. (49, 63) for details). As in Ref. (63), we choose $\omega_{\mathrm{C}} = 2\mathrm{E}_0\Delta$ and $\omega_{\mathrm{H}} = 2\mathrm{E}_0\sqrt{\Delta^2 + 1/4}$, such that the C (H) bath is in resonance with the qubit when $u = 0$ ($u = 1/2$). The width of the resonance is governed by Q_{α}. The total coupling strength to bath α, plotted in Fig. 3F, is quantified by

$$\gamma^{(\alpha)}_{u(t)} \equiv \gamma^{(\alpha)}_{+,u(t)} + \gamma^{(\alpha)}_{-,u(t)}. \tag{61}$$

In the quantum harmonic oscillator-based heat engine, following Ref. (43), we describe the coupling to the baths through the

Lindblad operators $\hat{A}^{(\alpha)}_{+,u(t)} = \hat{a}^{\dagger}_{u(t)}$, $\hat{A}^{(\alpha)}_{-,u(t)} = \hat{a}_{u(t)}$ and corresponding rates $\gamma^{(\alpha)}_{+,u(t)} = \Gamma_\alpha\, n(\beta_\alpha u(t)\omega_0)$ and $\gamma^{(\alpha)}_{-,u(t)} = \Gamma_\alpha[1 + n(\beta_\alpha u(t)\omega_0)]$, where we identify $k = \pm$. $\hat{a}_{u(t)} = (1/\sqrt{2})\sqrt{m\omega_0 u(t)}\,\hat{q} + i/\sqrt{m\omega_0 u(t)}\,\hat{p}$ and $\hat{a}^{\dagger}_{u(t)}$ are, respectively, the (control dependent) lowering and raising operators, Γ_α is a constant rate setting the thermalization timescale of the system coupled to bath α, and $n(x) = [\exp(x) - 1]^{-1}$ is the Bose-Einstein distribution.

Training details

We now provide additional practical details and the hyper parameters used to produce the results of this manuscript.

In order to enforce sufficient exploration in the early stage of training, we do the following. As in Ref. (111), for a fixed number of initial steps, we choose random actions sampling them uniformly withing their range. Furthermore, for another fixed number of initial steps, we do not update the parameters to allow the replay buffer to have enough transitions. $\mathcal{B}$ is a first-in-first-out buffer, of fixed dimension, from which batches of transitions $(s_k, a_k, r_{k+1}, s_{k+1}, a_{k+1})$ are randomly sampled to update the NN parameters. After this initial phase, we repeat a policy evaluation, a policy improvement step and a temperature update step n_{updates} times every n_{updates} steps. This way, the overall number of updates coincides with the number of actions performed on the QTM. The optimization steps for the value function and the policy are performed using the ADAM optimizer with the standard values of β_1 and β_2. The temperature parameters α_{D} and α_{C} instead are determined using stochastic gradient descent with learning rate 0.001. To favor an exploratory behavior early in the training, and at the same time to end up with a policy that is approximately deterministic, we schedule the target entropies $\bar{H}_{\text{C}}$ and $\bar{H}_{\text{D}}$. In particular, we vary them exponentially during each step according to

$$\bar{H}_a(n_{\text{steps}}) = \bar{H}_{a,\text{end}} + (\bar{H}_{a,\text{start}} - \bar{H}_{a,\text{end}})\exp(-n_{\text{steps}}/\bar{H}_{a,\text{decay}}), \tag{62}$$

where $a = \text{C}, \text{D}$, n_{steps} is the current step number, and $\bar{H}_{a,\text{start}}$, $\bar{H}_{a,\text{end}}$ and $\bar{H}_{a,\text{decay}}$ are hyperparameters. In the superconducting qubit refrigerator case, we schedule the parameter c according to a Fermi distribution, that is

$$c(n_{\text{step}}) = c_{\text{end}} + (c_{\text{start}} - c_{\text{end}})\left[1 + \exp\left(\frac{n_{\text{step}} - c_{\text{mean}}}{c_{\text{decay}}}\right)\right]^{-1}. \tag{63}$$

In the harmonic oscillator engine case, to improve stability while training for lower values of c, we do not vary c during training, as we do in the superconducting qubit refrigerator case. Instead, we discourage the agent from never utilizing one of the two thermal baths by adding a negative reward if, withing the last $N = 128$ actions describing the state, less than 25 describe a coupling to either bath. In particular, if the number of actions N_α where $d = \alpha$, with $\alpha = \text{Hot}, \text{Cold}$ is less than 25 in the state time-series, we sum to the reward the following penalty

$$r_{\text{penalty}} = -1.4\,\frac{25 - N_\alpha}{25}. \tag{64}$$

This penalty has no impact on the final cycles where N_α is much larger than 25.

All hyperparameters used to produce the results of the superconducting qubit refrigerator and of the harmonic oscillator heat engine are provided respectively in Tables 1 and 2, where c refers to the weight at which we are optimizing the return.

Table 1. Hyperparameters used in numerical calculations relative to the superconducting qubit refrigerator that are not reported in the caption of Fig. 3.

Hyperparameter	Qubit refrigerator
Batch size	512
Training steps	500 k
Learning rate	0.0003
$\mathcal{B}$ size	280 k
ρ_{polyak}	0.995
Channels per conv. block	(64, 64, 64, 128, 128, 128, 128)
Units per f.c. layer in π	(256)
Units per f.c. layer in Q^π	(256, 256)
Initial random steps	5 k
First update at step	1 k
n_{updates}	50
$\bar{H}_{\text{C,start}}$	0
$\bar{H}_{\text{C,end}}$	−3.5
$\bar{H}_{\text{C,decay}}$	440 k
c_{start}	1
c_{end}	c
c_{mean}	170 k
c_{decay}	20 k

Convergence of the RL approach

The training process presents some degree of stochasticity, such as the initial random steps, the stochastic sampling of actions from the policy function, and the random sampling of a batch of experience from the replay buffer to compute an approximate gradient of the loss functions. We thus need to evaluate the reliability of our approach.

As shown in the main text, specifically in Figs. 4 and 5, we ran the full optimization five times. Out of 65 trainings in the superconducting qubit refrigerator case, only 4 failed, and out of the 55 in the harmonic oscillator engine, only 2 failed, where by failed we mean that the final return was negative. In such cases, we ran the training an additional time.

Figs. 4A and 5B display an error bar corresponding to the standard deviation, at each value of c, computed over the five repetitions. Instead, in Figs. 4B and 5C we display one black dot for each individual training. As we can see, the overall performance is quite stable and reliable.

At last, we discuss the variability of the discovered cycles. The cycles shown in Figs. 4C–F and 5D–E were chosen by selecting the largest return among the five repetitions. In Figs. 8 and 9, we display cycles discovered in the last of the five repetition, i.e. chosen without any post-selection. They correspond to the same setups and parameters displayed in Figs. 4C–F and 5D–E. As we can see, 5 out of the 6 displayed cycles are very similar to the ones displayed in Figs. 4C–F and 5D–E, with a very slight variability. The only exception is Fig. 8B, where the cycle has a visibly shorter period and amplitude than the one shown in Fig. 4D. Despite this visible difference in the cycle shape, the return of the cycle shown in Fig. 8B is 0.382 compared to 0.385 of the cycle shown in Fig. 4B.

We therefore conclude that, up to minor changes, the cycles are generally quite stable across multiple trainings.

Comparing with other methods

In Figs. 4 and 5, we compare the performance of our method respectively against optimized trapezoidal cycles, and optimized Otto cycles. In both cases, we also maximize the power using

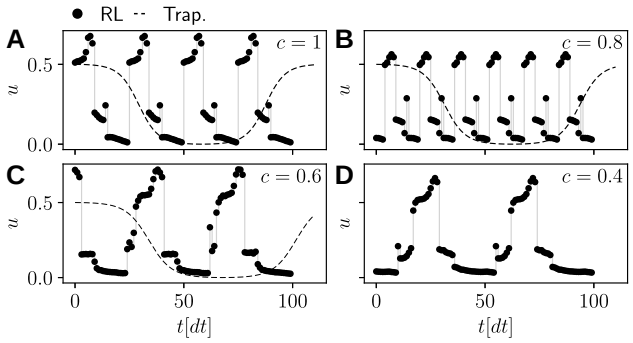

Fig. 8. Final deterministic cycle, identified in the superconducting qubit refrigerator, at the fifth training. Same parameters and quantities are shown as in Fig. 4C to F.

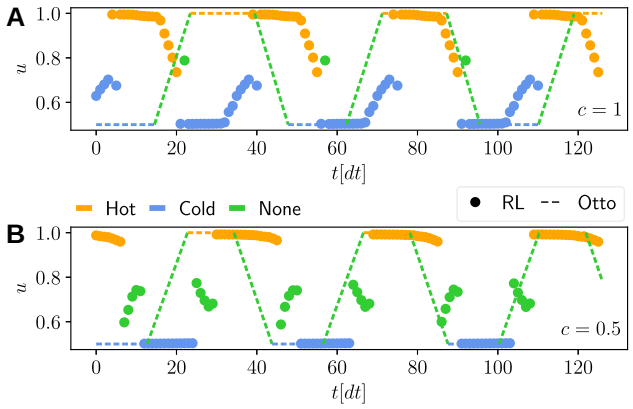

Fig. 9. Final deterministic cycle, identified in the harmonic oscillator engine, at the fifth training. Same parameters and quantities are shown as in Fig. 5D and E.

Table 2. Hyperparameters used in numerical calculations relative to the harmonic oscillator heat engine that are not reported in the caption of Fig. 5.

Hyperparameter	Harmonic engine
Batch size	512
Training steps	500 k
Learning rate	0.0003
$\mathcal{B}$ size	160 k
ρ_{polyak}	0.995
Channels per conv. block	(64, 64, 64, 128, 128, 128, 128)
Units per f.c. layer in π	(256)
Units per f.c. layer in Q^π	(256, 128)
Initial random steps	5 k
First update at step	1 k
n_{updates}	50
$\tilde{H}_{\text{C,start}}$	−0.72
$\tilde{H}_{\text{C,end}}$	−3.5
$\tilde{H}_{\text{C,decay}}$	144 k
$\tilde{H}_{\text{D,start}}$	ln 3
$\tilde{H}_{\text{D,end}}$	0.01
$\tilde{H}_{\text{D,decay}}$	144 k

the RL method of Ref. (69). We now detail how we perform such comparison.

In the refrigerator based on a superconducting qubit, we consider the trapezoidal cycle proposed in Ref. (49, 63), i.e. we fix

$$u(t) = \frac{1}{4}\left(1 + \frac{\tan(a\cos\Omega t)}{\tanh(a)}\right) \tag{65}$$

Table 3. Coherence generated by the final deterministic cycles identified by the RL method (RL column) and generated by a trapezoidal cycle operated at the same speed (Trapez. column) at the values of c shown in the first column. These values correspond to the cycles shown in Fig. 4C to F.

c	RL	Trapez.
1	0.068	0.13
0.8	0.050	0.12
0.6	0.054	0.092
0.4	0.035	0.090

with $a = 2$, and we optimize $\langle r_c \rangle$ with respect to frequency Ω. In the heat engine case based on a quantum harmonic oscillator, we fix an Otto cycle as described in Ref. (43), i.e. a trapezoidal cycle consisting of the 4 strokes shown in Fig. 5D and E as a dashed line, and we optimize over the duration of each of the 4 strokes. In particular, we first performed a grid search in the space of these four durations for $c = 1$. After identifying the largest power, we ran the Newton algorithm to further maximize the return. We then ran the Newton algorithm for all other values of c.

The comparison with Ref. (69) was done using the source code provided in Ref. (69), and using the same exact hyperparameters that were used in Ref. (69).

In particular, in the case of the refrigerator based on a superconducting qubit, we re-ran the code using the hyperparameters reported in Table 1, column "Figs. 3 and 4," of the Methods section of Ref. (69), and we trained for the same number of steps (500k). We then evaluated its power and coefficient of performance evaluating the deterministic policy (which typically has a better performance). In the heat engine case based on a quantum harmonic oscillator, we evaluated the performance of the cycle reported in Fig. 5a,c of Ref. (69), whose training hyperparameters are reported in Table 1, column "Fig. 5A," of the Methods section of Ref. (69).

Generation of coherence

In order to quantify the coherence generated in the instantaneous eigenbasis of the Hamiltonian in the refrigerator based on a superconducting qubit, we evaluated the time average of *relative entropy of coherence* (115), defined as

$$C(\hat{\rho}(t)) = S(\hat{\rho}_{\text{diag.}}(t)) - S(\hat{\rho}(t)), \tag{66}$$

where $S(\hat{\rho}) = -\text{Tr}[\hat{\rho}\ln\hat{\rho}]$ is the Von Neumann entropy, and

$$\begin{aligned}\hat{\rho}_{\text{diag.}}(t) = &\langle g_{u(t)}|\hat{\rho}(t)|g_{u(t)}\rangle \cdot |g_{u(t)}\rangle\langle g_{u(t)}| \\ &+ \langle e_{u(t)}|\hat{\rho}(t)|e_{u(t)}\rangle \cdot |e_{u(t)}\rangle\langle e_{u(t)}|\end{aligned} \tag{67}$$

is the density matrix, in the instantaneous eigenbasis $|g_{u(t)}\rangle$ and $|e_{u(t)}\rangle$, with the off-diagonal terms canceled out.

We compute the time-average of the relative entropy of coherence generated by the final deterministic cycle found by the RL agent, and compare it to the coherence generated by a trapezoidal cycle operated at the same speed, i.e. with the same period. As we can see in Table 3, the trapezoidal cycles generate twice as much coherence as the RL cycles shown in Fig. 4C to F, i.e. corresponding to $c = 1, 0.8, 0.6, 0.4$.

Acknowledgments

We are greatly thankful to Martí Perarnau-Llobet, Paolo Abiuso and Alberto Rolandi for useful discussions and for suggesting to include the entropy production in the return. We gratefully

acknowledge funding by the BMBF (Berlin Institute for the Foundations of Learning and Data – BIFOLD), the European Research Commission (ERC CoG 772230) and the Berlin Mathematics Center MATH+ (AA1-6, AA2-8, AA2-18). This manuscript was posted on a preprint: https://doi.org/10.48550/arXiv.2204.04785.

Author contributions

P.A.E. and F.N. designed the research and method. P.A.E. wrote the computer code and carried out the numerical calculations. P.A.E. and F.N. analyzed the data and wrote the manuscript.

Code and data availability

The code used to generate all results is available on GitHub (https://github.com/PaoloAE/paper_rl_blackbox_thermal_machines). All raw data that was generated with the accompanying code and that was used to produce the results in the manuscript is available on Figshare (https://doi.org/10.6084/m9.figshare.19180907).

References

1 Giazotto F, Heikkilä TT, Luukanen A, Savin AM, Pekola JP. 2006. Opportunities for mesoscopics in thermometry and refrigeration: physics and applications. *Rev Mod Phys*. 78:217.
2 Pekola JP. 2015. Towards quantum thermodynamics in electronic circuits. *Nat Phys*. 11:118.
3 Vinjanampathy S, Anders J. 2016. Quantum thermodynamics. *Contemp Phys*. 57:545.
4 Benenti G, Casati G, Saito K, Whitney RS. 2017. Fundamental aspects of steady-state conversion of heat to work at the nanoscale. *Phys Rep*. 694:1.
5 Binder F, Correa LA, Gogolin C, Anders J, Adesso G. 2019: *Thermodynamics in the quantum regime: fundamental aspects and new directions*. Fundamental Theories of Physics. Springer International Publishing.
6 Ronzani A, *et al.* 2018. Tunable photonic heat transport in a quantum heat valve. *Nat Phys*. 14:991.
7 Dutta B, *et al.* 2019. Direct probe of the seebeck coefficient in a Kondo-correlated single-quantum-dot transistor. *Nano Lett*. 19:506.
8 Senior J, *et al.* 2020. Heat rectification via a superconducting artificial atom. *Commun Phys*. 3:40.
9 Maillet O, Subero D, Peltonen JT, Golubev DS, Pekola JP. 2020. Electric field control of radiative heat transfer in a superconducting circuit. *Nat Commun*. 11:4326.
10 Roßnagel J, *et al.* 2016. A single-atom heat engine. *Science*. 352:325.
11 Josefsson M, *et al.* 2018. A quantum-dot heat engine operating close to the thermodynamic efficiency limits. *Nat Nanotechnol*. 13:920.
12 Klatzow J, *et al.* 2019. Experimental demonstration of quantum effects in the operation of microscopic heat engines. *Phys Rev Lett*. 122:110601.
13 von Lindenfels D, *et al.* 2019. Spin heat engine coupled to a harmonic-oscillator flywheel. *Phys Rev Lett*. 123:080602.
14 Maslennikov G, *et al.* 2019. Quantum absorption refrigerator with trapped ions. *Nat Commun*. 10:202.
15 Peterson JPS, *et al.* 2019. Experimental characterization of a spin quantum heat engine. *Phys Rev Lett*. 123:240601.
16 Prete D, *et al.* 2019. Thermoelectric conversion at 30K in InAs/InP nanowire quantum dots. *Nano Lett*. 19:3033.
17 Van Horne N, *et al.* 2020. Single-atom energy-conversion device with a quantum load. *NPJ Quantum Inf*. 6:37.
18 Krantz P, *et al.* 2019. A quantum engineer's guide to superconducting qubits. *Appl Phys Rev*. 6:021318.
19 Huang K. 1987. *Statistical mechanics*. 2nd ed. New York: Wiley.
20 Esposito M, Kawai R, Lindenberg K, Van den Broeck C. 2010. Efficiency at maximum power of low-dissipation Carnot engines. *Phys Rev Lett*. 105:150603.
21 Wang J, He J, He X. 2011. Performance analysis of a two-state quantum heat engine working with a single-mode radiation field in a cavity. *Phys Rev E*. 84:041127.
22 Avron JE, Fraas M, Graf GM, Grech P. 2012. Adiabatic theorems for generators of contracting evolutions. *Commun Math Phys*. 314:163.
23 Ludovico MF, Battista F, von Oppen F, Arrachea L. 2016. Adiabatic response and quantum thermoelectrics for ac-driven quantum systems. *Phys Rev B*. 93:075136.
24 Cavina V, Mari A, Giovannetti V. 2017. Slow dynamics and thermodynamics of open quantum systems. *Phys Rev Lett*. 119:050601.
25 Abiuso P, Giovannetti V. 2019. Non-Markov enhancement of maximum power for quantum thermal machines. *Phys Rev A*. 99:052106.
26 Scandi M, Perarnau-Llobet M. 2019. Thermodynamic length in open quantum systems. *Quantum*. 3:197.
27 Bhandari B, *et al.* 2020. Geometric properties of adiabatic quantum thermal machines. *Phys Rev B*. 102:155407.
28 Alonso PT, Abiuso P, Perarnau-Llobet M, Arrachea L. 2021. Geometric optimization of non-equilibrium adiabatic thermal machines and implementation in a qubit system. https://journals.aps.org/prxquantum/abstract/10.1103/PRXQuantum.3.010326.
29 Eglinton J, Brandner K. 2022. Geometric bounds on the power of adiabatic thermal machines. *Phys Rev E*. 105:L052102.
30 Abiuso P, Perarnau-Llobet M. 2020. Optimal cycles for low-dissipation heat engines. *Phys Rev Lett*. 124:110606.
31 Abiuso P, Miller HJD, Perarnau-Llobet M, Scandi M. 2020. Geometric optimisation of quantum thermodynamic processes. *Entropy*. 22:1076.
32 Cavina V, Erdman PA, Abiuso P, Tolomeo L, Giovannetti V. 2021. Maximum-power heat engines and refrigerators in the fast-driving regime. *Phys Rev A*. 104:032226.
33 Arrachea L, Moskalets M, Martin-Moreno L. 2007. Heat production and energy balance in nanoscale engines driven by time-dependent fields. *Phys Rev B*. 75:245420.
34 Esposito M, Kawai R, Lindenberg K, Van den Broeck C. 2010. Quantum-dot Carnot engine at maximum power. *Phys Rev E*. 81:041106.
35 Juergens S, Haupt F, Moskalets M, Splettstoesser J. 2013. Thermoelectric performance of a driven double quantum dot. *Phys Rev B*. 87:245423.
36 Campisi M, Pekola J, Fazio R. 2015. Nonequilibrium fluctuations in quantum heat engines: theory, example, and possible solid state experiments. *New J Phys*. 17:035012.
37 Dann R, Kosloff R. 2020. Qunatum signatures in the quantum Carnot cycle. *New J Phys*. 22:013055.
38 Molitor OAD, Landi GT. 2020. Stroboscopic two-stroke quantum heat engines. *Phys Rev A*. 102:042217.
39 Shaghaghi V, Palma GM, Benenti G. 2022. Extracting work from random collisions: a model of a quantum heat engine. *Phys Rev E*. 105:034101.
40 Cavaliere F, *et al.* 2022. Dynamical heat engines with non-Markovian reservoirs. *Phys Rev Res*. 4:033233.

41 Feldmann T, Geva E, Kosloff R, Salamon P. 1996. Heat engines in finite time governed by master equations. *Am J Phys*. 64:485.
42 Feldmann T, Kosloff R. 2000. Performance of discrete heat engines and heat pumps in finite time. *Phys Rev E*. 61:4774.
43 Rezek Y, Kosloff R. 2006. Irreversible performance of a quantum harmonic heat engine. *New J Phys*. 8:83.
44 Quan H, Liu YX, Sun C, Nori F. 2007. Quantum thermodynamic cycles and quantum heat engines. *Phys Rev E*. 76:031105.
45 Abah O, *et al.* 2012 Nov. Single-ion heat engine at maximum power. *Phys Rev Lett*. 109:203006.
46 Allahverdyan AE, Hovhannisyan KV, Melkikh AV, Gevorkian SG. 2013. Carnot cycle at finite power: attainability of maximal efficiency. *Phys Rev Lett*. 111:050601.
47 Zhang K, Bariani F, Meystre P. 2014 Apr. Quantum optomechanical heat engine. *Phys Rev Lett*. 112:150602.
48 Campisi M, Fazio R. 2016. The power of a critical heat engine. *Nat Commun*. 7:11895.
49 Karimi B, Pekola JP. 2016. Otto refrigerator based on a superconducting qubit: classical and quantum performance. *Phys Rev B*. 94:184503.
50 Kosloff R, Rezek Y. 2017. The quantum harmonic Otto cycle. *Entropy*. 19:136.
51 Watanabe G, Venkatesh BP, Talkner P, del Campo A. 2017. Quantum performance of thermal machines over many cycles. *Phys Rev Lett*. 118:050601.
52 Deffner S. 2018. Efficiency of harmonic quantum Otto engines at maximal power. *Entropy*. 20:875.
53 Gelbwaser-Klimovsky D, *et al.* 2018. Single-atom heat machines enabled by energy quantization. *Phys Rev Lett*. 120:170601.
54 Chen J, Sun C, Dong H. 2019. Boosting the performance of quantum Otto heat engines. *Phys Rev E*. 100:032144.
55 Pekola JP, Karimi B, Thomas G, Averin DV. 2019. Supremacy of incoherent sudden cycles. *Phys Rev B*. 100:085405.
56 Das A, Mukherjee V. 2020. Quantum-enhanced finite-time Otto cycle. *Phys Rev B*. 2:033083.
57 Berry MV. 2009. Transitionless quantum driving. *J Phys A: Math Theor*. 42:365303.
58 Deng J, Wang Q-h, Liu Z, Hänggi P, Gong J. 2013. Boosting work characteristics and overall heat-engine performance via shortcuts to adiabaticity: quantum and classical systems. *Phys Rev E*. 88:062122.
59 Torrontegui E, *et al.* 2013. Shortcuts to adiabaticity. *Adv At Mol Opt Phys*. 62:117.
60 del Campo A, Goold J, Paternostro M. 2014. More bang for your buck: super-adiabatic quantum engines. *Sci Rep*. 4:6208.
61 Çakmak B, Müstecaplıoğlu OE. 2019. Spin quantum heat engines with shortcuts to adiabaticity. *Phys Rev E*. 99:032108.
62 Deng S, *et al.* 2018. Superadiabatic quantum friction suppression in finite-time thermodynamics. *Sci Adv*. 18:eaar5909.
63 Funo K, *et al.* 2019. Speeding up a quantum refrigerator via counterdiabatic driving. *Phys Rev B*. 100:035407.
64 Villazon T, Polkovnikov A, Chandran A. 2019. Swift heat transfer by fast-forward driving in open quantum systems. *Phys Rev A*. 100:012126.
65 Khait I, Carrasquilla J, Segal D. 2021. Optimal control of quantum thermal machines using machine learning. https://journals.aps.org/prresearch/abstract/10.1103/PhysRevResearch.4.L012029.
66 Cavina V, Mari A, Carlini A, Giovannetti V. 2018. Optimal thermodynamic control in open quantum systems. *Phys Rev A*. 98:012139.
67 Suri N, Binder FC, Muralidharan S, Vinjanampathy B. 2018. Speeding up thermalisation via open quantum system variational optimisation. *Eur Phys J Spec Top*. 227:203.
68 Menczel P, Pyhäranta T, Flindt C, Brandner K. 2019. Two-stroke optimization scheme for mesoscopic refrigerators. *Phys Rev B*. 99:224306.
69 Erdman PA, Noé F. 2022. Identifying optimal cycles in quantum thermal machines with reinforcement-learning. *NPJ Quantum Inf*. 8:1.
70 Haarnoja T, Zhou A, Abbeel P, Levine S. 2018. Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: International Conference on Machine Learning. Vol. 80. PMLR, p. 1861.
71 Haarnoja T, *et al.* 2018. Soft actor-critic algorithms and applications. https://arxiv.org/abs/1812.05905.
72 Christodoulou P. 2019. Soft actor-critic for discrete action settings. https://arxiv.org/abs/1910.07207.
73 Delalleau O, Peter M, Alonso E, Logut A. 2019. Discrete and continuous action representation for practical RL in video games. https://arxiv.org/abs/1912.11077.
74 Mnih V, *et al.* 2015. Human-level control through deep reinforcement learning. *Nature*. 518(7540):529.
75 Silver D, *et al.* 2017. Mastering the game of go without human knowledge. *Nature*. 550:354.
76 Vinyals O, *et al.* 2019. Grandmaster level in starcraft II using multi-agent reinforcement learning. *Nature*. 575:350.
77 Haarnoja T, *et al.* 2018. Learning to walk via deep reinforcement learning. https://arxiv.org/abs/1812.11103.
78 Bukov M, *et al.* 2018. Reinforcement learning in different phases of quantum control. *Phys Rev X*. 8:031086.
79 An Z, Zhou DL. 2019. Deep reinforcement learning for quantum gate control. *EPL*. 126:60002.
80 Dalgaard M, Motzoi F, Sørensen JJ, Sherson J. 2020. Global optimization of quantum dynamics with alphazero deep exploration. *NPJ Quantum Inf*. 6:6.
81 Mackeprang J, Dasari DBR, Wrachtrup J. 2020. A reinforcement learning approach for quantum state engineering. *Quantum Mach Intell*. 2:5.
82 Schäfer F, Kloc M, Bruder C, Lörch N. 2020. A differentiable programming method for quantum control. *Mach Learn: Sci Technol*. 1:035009.
83 Schäfer F, Sekatski P, Koppenhöfer M, Bruder C, Kloc M. 2021. Control of stochastic quantum dynamics by differentiable programming. *Mach Learn: Sci Technol*. 2:035004.
84 Porotti R, Essig A, Huard B, Marquardt F. 2022. Deep reinforcement learning for quantum state preparation with weak nonlinear measurements. *Quantum*. 6:747.
85 Marquardt F. 2021. Machine learning and quantum devices. *SciPost Phys Lect Notes*. 29. https://scipost.org/10.21468/SciPostPhysLectNotes.29.
86 Brown J, *et al.* 2021. Reinforcement learning-enhanced protocols for coherent population-transfer in three-level quantum systems. *New J Phys*. 23:093035.
87 Metz F, Bukov M. 2022. Self-correcting quantum many-body control using reinforcement learning with tensor networks. https://www.nature.com/articles/s42256-023-00687-5.
88 Niu MY, Boixo S, Smelyanskiy VN, Neven H. 2019. Universal quantum control through deep reinforcement learning. *NPJ Quantum Inf*. 5:33.
89 Zhang X-M, Wei Z, Asad R, Yang X-C, Wang X. 2019. When does reinforcement learning stand out in quantum control? A comparative study on state preparation. *NPJ Quantum Inf*. 5:85.
90 Fösel T, Tighineanu P, Weiss T, Marquardt F. 2018. Reinforcement learning with neural networks for quantum feedback. *Phys Rev X*. 8:031084.

91 Sweke R, Kesselring MS, van Nieuwenburg EPL, Eisert J. 2020. Reinforcement learning decoders for fault-tolerant quantum computation. *Mach Learn: Sci Technol*. 2:025005.

92 Sgroi P, Palma GM, Paternostro M. 2021. Reinforcement learning approach to nonequilibrium quantum thermodynamics. *Phys Rev Lett*. 126:020601.

93 Kosloff R, Feldmann T. 2002. Discrete four-stroke quantum heat engine exploring the origin of friction. *Phys Rev E*. 65:055102.

94 Friedenberger A, Lutz E. 2017. When is a quantum heat engine quantum? *EPL*. 120:10002.

95 Brandner K, Bauer M, Seifer U. 2017. Universal coherence-induced power losses of quantum heat engines in linear response. *Phys Rev Lett*. 119:170602.

96 Lekscha J, Wilming H, Eisert J, Gallego R. 2018. Quantum thermodynamics with local control. *Phys Rev E*. 97:022142.

97 Strasberg P, Schaller G, Lambert N, Brandes T. 2016. Nonequilibrium thermodynamics in the strong coupling and non-Markovian regime based on a reaction coordinate mapping. *New J Phys*. 18:073007.

98 Seoane LF, Solé R. 2016. Multiobjective optimization and phase transitions. In: Battiston S, De Pellegrini F, Caldarelli G, Merelli E, editors. *Proceedings of ECCS 2014*. Cham: Springer International Publishing. p. 259–270.

99 Miller HJD, Scandi M, Anders J, Perarnau-Llobet M. 2019. Work fluctuations in slow processes: quantum signatures and optimal control. *Phys Rev Lett*. 123:230603.

100 Solon AP, Horowitz JM. 2018. Phase transition in protocols minimizing work fluctuations. *Phys Rev Lett*. 120:180605.

101 van den Oord A, *et al*. 2016. Wavenet: a generative model for raw audio. https://arxiv.org/abs/1609.03499.

102 He K, Zhang X, Ren S, Sun J. 2015. Deep residual learning for image recognition. https://arxiv.org/abs/1512.03385.

103 Kingma DP, Ba J. 2014. Adam: a method for stochastic optimization. https://arxiv.org/abs/1412.6980.

104 Gorini V, Kossakowski A, Sudarshan ECG. 1976. Completely positive dynamical semigroups of N-level systems. *J Math Phys*. 17:821.

105 Lindblad G. 1976. On the generators of quantum dynamical semigroups. *Commun Math Phys*. 48:119–130.

106 Breuer H, Petruccione F. 2002. *The theory of open quantum systems*. New York: Oxford University Press.

107 Yamaguchi M, Yuge T, Ogawa T. 2017. Markovian quantum master equation beyond adiabatic regime. *Phys Rev E*. 95:012136.

108 Barato AC, Seifert U. 2015. Thermodynamic uncertainty relation for biomolecular processes. *Phys Rev Lett*. 114:158101.

109 Guarnieri G, Landi GT, Clark SR, Goold J. 2019. Thermodynamics of precision in quantum nonequilibrium steady states. *Phys Rev Res*. 1:033021.

110 Miller HJD, Mohammady MH, Perarnau-Llobet M, Guarnieri G. 2021. Thermodynamic uncertainty relation in slowly driven quantum heat engines. *Phys Rev Lett*. 126:210603.

111 Achiam J. 2018. Spinning up in deep reinforcement learning. https://github.com/openai/spinningup.

112 Sutton RS, Barto AG. 2018. *Reinforcement learning: an introduction*. Cambridge (MA): MIT Press.

113 Dann R, Levy A, Kosloff R. 2018. Time-dependent markovian quantum master equation. *Phys Rev A*. 98:052129.

114 Alicki R. 1979. The quantum open system as a model of the heat engine. *J Phys A: Math Gen*. 12:L103.

115 Baumgratz T, Cramer M, Plenio MB. 2014. Quantifying coherence. *Phys Rev Lett*. 113:140401.