Tuning the Molecular Weight Distribution from Atom Transfer Radical Polymerization Using Deep Reinforcement Learning

Journal:	*Molecular Systems Design & Engineering*
Manuscript ID	ME-ART-12-2017-000131.R1
Article Type:	Paper
Date Submitted by the Author:	27-Feb-2018
Complete List of Authors:	Li, Haichen; Carnegie Mellon University, Department of Chemistry Collins, Christopher; Carnegie Mellon University, Department of Chemistry Ribelli, Thomas; Carnegie Mellon University, Department of Chemistry Matyjaszewski, Krzysztof; Carnegie Mellon University, Department of Chemistry Gordon, Geoffrey; Carnegie Mellon University, School of Computer Science Kowalewski, Tomasz; Carnegie Mellon University, Department of Chemistry Yaron, David; Carnegie Mellon University, Department of Chemistry

Design, System, Application Statement

For manuscript "Tuning the Molecular Weight Distribution from Atom Transfer Radical Polymerization Using Deep Reinforcement Learning"

Haichen Li, Christopher R. Collins, Thomas G. Ribelli, Krzysztof Matyjaszewski, Geoffrey J. Gordon, Tomasz Kowalewski, and David J. Yaron

The molecular weight distribution (MWD) of polymer chains can have substantial impact on the mechanical and other properties of the resulting material. Atom transfer radical polymerization (ATRP) provides a means to control the MWD, but most work has focused on creating samples with narrow MWDs. Here, we consider synthetic strategies to achieve more flexible control of the MWD. The design and optimization strategy uses recent advances in reinforcement learning (RL) to train a controller that, by adding reagents at multiple points throughout the reaction, guides the reaction to a target MWD. The target MWDs include Gaussian distributions with varying widths and a number of more complex shapes.

The current work investigates the use of RL to control MWD within a simulation of the ATRP reaction. The future application potential lies in transferring a controller trained in this way from a simulation environment to the real laboratory. The current manuscript evaluates this potential by investigating the degree to which the controller can achieve MWD near the target distribution when noise is added to both the kinetic parameters used in the simulation and to the observations passed from the simulation to the controller.

Journal Name

ARTICLE TYPE

Cite this: DOI: 10.1039/xxxxxxxxxx

Tuning the Molecular Weight Distribution from Atom Transfer Radical Polymerization Using Deep Reinforcement Learning

Haichen Li,[a,b] Christopher R. Collins,[a] Thomas G. Ribelli,[a] Krzysztof Matyjaszewski,[a] Geoffrey J. Gordon,[b] Tomasz Kowalewski,[a] and David J. Yaron*[a]

Received Date
Accepted Date

DOI: 10.1039/xxxxxxxxxx

www.rsc.org/journalname

We devise a novel technique to control the shape of polymer molecular weight distributions (MWDs) in atom transfer radical polymerization (ATRP). This technique makes use of recent advances in both simulation-based, model-free reinforcement learning (RL) and the numerical simulation of ATRP. A simulation of ATRP is built that allows an RL controller to add chemical reagents throughout the course of the reaction. The RL controller incorporates fully-connected and convolutional neural network architectures and bases its decision upon the current status of the ATRP reaction. The initial, untrained, controller leads to ending MWDs with large variability, allowing the RL algorithm to explore a large search space. When trained using an actor-critic algorithm, the RL controller is able to discover and optimize control policies that lead to a variety of target MWDs. The target MWDs include Gaussians of various width, and more diverse shapes such as bimodal distributions. The learned control policies are robust and transfer to similar but not identical ATRP reaction settings, even under the presence of simulated noise. We believe this work is a proof-of-concept for employing modern artificial intelligence techniques in the synthesis of new functional polymer materials.

1 Introduction

Most current approaches to development of new materials follow a sequential, iterative process that requires extensive human labor to synthesize new materials and elucidate their properties and functions. Over the next decades, it seems likely that this inherently slow and labor intensive approach to chemical research will be transformed through the incorporation of new technologies originating from computer science, robotics, and advanced manufacturing.[1,2] A central challenge is finding ways to use these powerful new technologies to guide chemical processes to desired outcomes.[3] Recent advances in reinforcement learning (RL) have enabled computing systems to guide vehicles through complex simulation environments,[4] and select moves that guide games such as Go and chess to winning conclusions.[5–8] For chemical problems, RL has been used to generate candidate drug molecules in a *de novo* manner,[9,10] and to optimize reaction conditions for organic synthesis.[11] This work investigates the benefits and challenges of using RL to guide chemical reactions towards specific synthetic targets. The investigation is done through computational experiments that use RL to control a simulated reaction system, where the simulation models the chemical kinetics present in the system.

In this work, the simulated reaction system is that of atom transfer radical polymerization (ATRP).[12–15] ATRP is among the mostly widely used and effective means to control the polymerization of a wide variety of vinyl monomers. ATRP allows the synthesis of polymers with predetermined molecular weights, narrow molecular weight distributions (MWDs),[16] and adjustable polydispersity.[17–23] The high degree of control allows the synthesis of various polymeric architectures[24] such as block copolymers,[25–28] star polymers,[29–31] and molecular brushes.[32] Temporal and spatial control has also been applied in ATRP to further increase the level of control over the polymerization.[33–36] More recently, chemists have been working on ways to achieve MWDs with more flexible forms,[23,37] as this may provide a means to tailor mechanical and processability of the resulting plastics.[38]

In addition to its importance, ATRP is well suited to the computational experiments carried out here. The chemical kinetics of ATRP are shown schematically in Fig. 1. Control of the polymerization process is related to the activation, k_a, and deactivation, k_d, reactions which inter-convert dormant chains, P_nBr, and active, free radical chains, $P_n^\bullet$. The active chains grow in length through propagation reactions, k_p. The equilibrium between dormant and active chains can be used to maintain a low concen-

[a] *Department of Chemistry, Carnegie Mellon University, Pittsburgh, PA 15213, USA*
[b] *School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA*

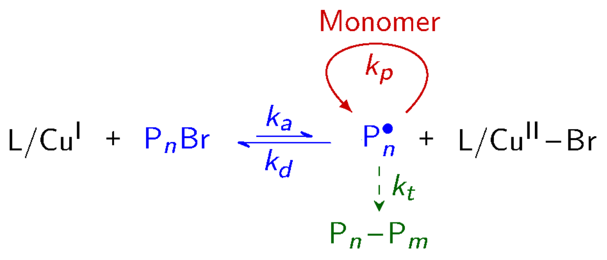


Fig. 1 Reaction mechanism of ATRP. Polymer species include radical chains $P_n^\bullet$ and dormant chains P_nBr with reduced chain length n and chains that terminated through recombination P_n-P_m. L/Cu^I and L/Cu^{II}−Br are ATRP catalysts, where L represents the ligand. k_p, k_a, k_d, and k_t are kinetic rate constants for chain propagation, activation, deactivation, and termination, respectively.

tration of active chains, leading to more controlled growth and a reduction in termination reactions, k_t, that broaden the final MWD. These kinetics are sufficiently well understood[39,40] that simulations provide reliable results.[41–49] It is also computationally feasible to carry out a large number of simulated reactions. Fig. 2 shows how the MWD evolves in a single reaction simulation, which finishes in about 1 minute on a 2.4 GHz CPU core. MWDs will be shown as the fraction of polymer chains (vertical axis) with a specific reduced chain length (horizontal axis), where the reduced chain length refers to the number of monomers incorporated into the chain.

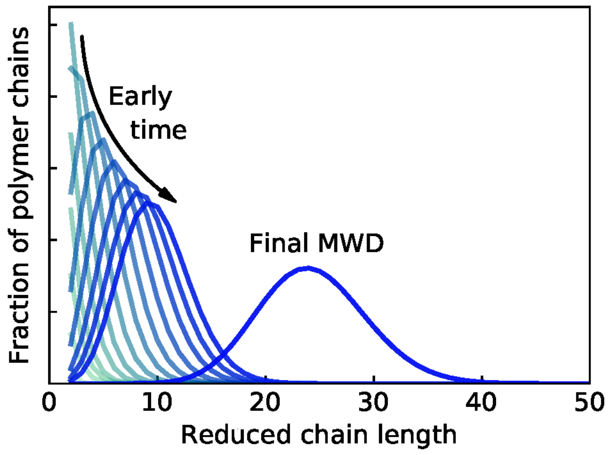


Fig. 2 Evolution of polymer MWD in a simulated ATRP reaction.

ATRP reactions can also be manipulated in a large variety of ways because of the multiple interacting chemical reactions, and the shape of the MWD provides a diverse set of targets. This makes the system a good choice for evaluating the degree to which RL can guide a chemical process to a desired synthetic target. ATRP reactions are typically carried out by creating an initial mixture of chemical reagents and keeping the temperature and other reaction conditions steady. However, a greater diversity of MWDs can be obtained by taking actions, such as adding chemical reagents, throughout the polymerization process.[23] Here, we use RL to decide which actions to take, based on the current state of the reaction system. In this manner, it is analogous to having a human continuously monitor the reaction and take actions that guide the system towards the target MWD. This use of a state-dependent decision process is a potential advantage of using RL. Consider an alternative approach in which the simulation is used to develop a protocol that specifies the times at which to perform various actions. Such a protocol is likely to be quite sensitive to the specific kinetic parameters used in the simulation. The RL controller may lower this sensitivity by basing its decisions on the current state of the reaction system. Below, the current state upon which the RL controller makes its decisions includes the current MWD. The controller is then expected to succeed provided the correct action to take at a given time depends primarily on the difference between the current MWD and the target MWD (Fig. 2), as opposed to the specific kinetic parameters. Ideally, an RL algorithm trained on a simulated reaction may be able to succeed in the real laboratory with limited additional training, provided the simulated reaction behaves like the actual one. Such transfer from simulated to real-world reactions is especially important given the potentially large number of reaction trials needed for training, and the inherent cost of carrying out chemical experiments. In our computational experiments, we assess the sensitivity to the simulation parameters by including noise in both the kinetic parameters used in the simulation and in the states of the current reaction system.

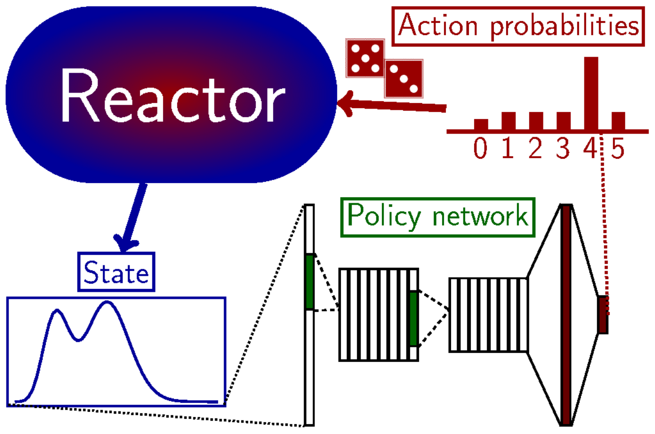


Fig. 3 Flow chart showing how the policy network of the RL controller selects actions to apply to the simulated ATRP reactor.

Fig. 3 provides a schematic view of the RL controller. The current state is fed into the RL controller (policy network), which produces a probability distribution for each of the available actions. An action is then drawn from this probability distribution, and performed on the reactor. The design of the RL controller is inspired by recent advances in deep reinforcement learning,[50–52] which use neural networks for the policy network and other components. The combination of modern deep learning models, represented by convolutional neural networks,[53–56] and efficient RL algorithms[57,58] such as deep Q-learning,[5,59,60] proximal policy methods,[61] and asynchronous advantage actor-critic (A3C)[62,63] has lead to numerous successful applications in control tasks with large state spaces.[1,64,65] The computational experiments presented here examine the use of modern deep reinforcement learning techniques to guide chemical synthesis of new materials.

2 Related works

There have been many studies that control the state and dynamics of chemical reactors based on classical control theory.[66] Model-based controllers,[67] some of which employ neural networks,[68] have been developed for a number of control tasks involving continuous stirred tank reactors,[69–74] batch processes,[75–77] hydrolyzers,[78] bioreactors,[79–81] pH neutralization processes,[82–85] strip thickness in steel-rolling mills,[86] and system pressure.[87] Model-free controllers trained through RL also exist for controlling chemical processes such as neutralization[88] and wastewater treatment[89] or chemical reactor valves.[90]

Due to its industrial importance, polymer synthesis has been a primary target for the development of chemical engineering controllers.[91] Some of these make use of neural networks to control the reactor temperature in the free radical polymerization of styrene.[92] McAfee et al. developed an automatic polymer molecular weight controller[93] for free radical polymerization. This controller is based on online molar mass monitoring techniques[94] and is able to follow a specific chain growth trajectory with respect to time by controlling the monomer flow rate in a continuous flow reactor. Similar online monitoring techniques have recently enabled controlling the modality of free radical polymerization products,[95] providing optimal feedback control to acrylamide-water-potassium persulfate polymerization reactors,[96] and monitoring multiple ionic strengths during the synthesis of copolymeric polyelectrolytes.[97] However, none of these works attempted to control the precise shape of polymer MWD shapes, nor did they use an artificial intelligence (AI) driven approach to design new materials. The significance of this work lies in that it is a first trial of building an AI agent that is trained *tabula rasa* to discover and optimize synthetic routes for human-specified, arbitrary polymer products with specific MWD shapes. Another novel aspect of the current work is the use of a simulation to train a highly-flexible controller, although the transfer of this controller to actual reaction processes, possibly achievable with modern transfer learning[98–102] and imitation learning techniques,[103,104] is left to future work.

3 Methodology

3.1 Simulating ATRP

We select styrene ATRP as our simulation system. Simulation of styrene ATRP may be done by solving the ATRP chemical kinetics ordinary differential equations (ODEs) in Table 1,[41,42,105,106] by method of moments,[107] or by Monte Carlo methods.[108–113] This work directly solves the ODEs because this allows accurate tracking of the concentration of individual polymer chains while being more computationally efficient than Monte Carlo methods.

In the ODEs of Table 1, M is monomer; $\mathrm{P_n^\bullet}$, $\mathrm{P_nBr}$, and $\mathrm{T_n}$ represent length-n radical chain, dormant chain, and terminated chain, respectively. $\mathrm{P_1Br}$ is also the initiator of radical polymerization. k_p, k_a, k_d, and k_t are propagation, activation, deactivation, and termination rate constants, respectively. N is the maximum allowed dormant/radical chain length in the numerical simulation. Consequently, the maximum allowed terminated chain length is $2N$, assuming styrene radicals terminate via combination.[114] We set $N = 100$ in all ATRP simulations in this work. This number is sufficiently large for our purpose as the lengths of dormant or terminated chains do not exceed 75 or 150, respectively, in any of the simulations. We used a set of well-established rate constants based on experimental results of the ATRP of bulk styrene at 110 °C (383.15 K) using dNbpy as the ligand[13,115–117]: $k_p = 1.6 \times 10^3$, $k_a = 0.45$, $k_d = 1.1 \times 10^7$, and $k_t = 10^8$ (units are $\mathrm{M^{-1}s^{-1}}$). It was assumed the reactor remained at this temperature for the duration of the polymerization. Although the rate constants depend on the degree of polymerization,[118] we assumed the same rate constants for polymer chains with different lengths. This assumption does not bias the nature of ATRP qualitatively and has been practiced in almost all previous ATRP simulation research.[41,42,105,106,119] In some of our simulations, we altered the rate constants by up to ±30% to account for possible inaccuracies in the measurement of these values and other unpredictable situations such as turbulence in the reactor temperature. We employed the VODE[120–124] integrator implemented in SciPy 0.19 using a maximum internal integration step of 5000, which is sufficient to achieve final MWDs with high accuracy. We chose the "backward differentiation formulas" integration method because the ODEs are stiff.

In practice, styrene ATRP is close to an ideal living polymerization,[116,117] with termination playing only a small role in establishing the final MWD. Excluding termination from the simula-

Table 1 ATRP kinetics equations. $\mathrm{Cu^I}$ and $\mathrm{Cu^{II}}$ stand for the ATRP activator and deactivator $\mathrm{L/Cu^I}$ and $\mathrm{L/Cu^{II}{-}Br}$, respectively.

Monomer	$[\mathrm{M}]' = -k_p[\mathrm{M}]\sum_{i=1}^{N}[\mathrm{P_i^\bullet}]$
Activator	$[\mathrm{Cu^I}]' = k_d[\mathrm{Cu^{II}}]\sum_{i=1}^{N}[\mathrm{P_i^\bullet}] - k_a[\mathrm{Cu^I}]\sum_{i=1}^{N}[\mathrm{P_iBr}]$
Deactivator	$[\mathrm{Cu^{II}}]' = k_a[\mathrm{Cu^I}]\sum_{i=1}^{N}[\mathrm{P_iBr}] - k_d[\mathrm{Cu^{II}}]\sum_{i=1}^{N}[\mathrm{P_i^\bullet}]$
Dormant chains	$[\mathrm{P_nBr}]' = k_d[\mathrm{Cu^{II}}][\mathrm{P_n^\bullet}] - k_a[\mathrm{Cu^I}][\mathrm{P_nBr}],\ 1 \le n \le N$
Smallest radical	$[\mathrm{P_1^\bullet}]' = -k_p[\mathrm{M}][\mathrm{P_1^\bullet}] + k_a[\mathrm{Cu^I}][\mathrm{P_1Br}] - k_d[\mathrm{Cu^{II}}][\mathrm{P_1^\bullet}] - 2k_t[\mathrm{P_1^\bullet}]\sum_{i=1}^{N}[\mathrm{P_i^\bullet}]$
Other radicals	$[\mathrm{P_n^\bullet}]' = k_p[\mathrm{M}]([\mathrm{P_{n-1}^\bullet}] - [\mathrm{P_n^\bullet}]) + k_a[\mathrm{Cu^I}][\mathrm{P_nBr}] - k_d[\mathrm{Cu^{II}}][\mathrm{P_n^\bullet}] - 2k_t[\mathrm{P_n^\bullet}]\sum_{i=1}^{N}[\mathrm{P_i^\bullet}],\ 2 \le n \le N$
Terminated chains	$[\mathrm{T_n}]' = \sum_{i=1}^{n-1} k_t[\mathrm{P_i^\bullet}][\mathrm{P_{n-i}^\bullet}],\ 2 \le n \le 2N$

tion reduces the the total number of ODEs by about 2/3 and substantially reduces the computer time needed for the simulation. Therefore, in most of the cases, we train the RL agents on *no-termination* environments to save computational cost. Note that we still evaluate their performance on *with-termination* environments. Moreover, this strategy allows us to test the transferability of control policies learned by the RL agent onto similar but not identical environments, which could be of great importance in later works where we need to apply control policies learned with simulated environments to real, physically built reactors.

We assume that the volume of the system is completely determined by the amount of solvent and the number of monomer equivalents (including monomers incorporated in polymer chains). To calculate the system volume, we use a bulk styrene density of 8.73 mol/L as reported in early works[41] and a solvent density of 1.00 mol/L.

3.2 Using RL to control the ATRP reactor simulation

A reinforcement learning problem is usually phrased as an agent interacting with an environment (Fig. 4). In our case, the agent is an RL controller and the environment is the ATRP reactor simulator. The agent interacts with the simulation at times separated by constant intervals, t_{step}. The interaction between the agent and the environment consists of three elements, each of which is indexed by the timestep (shown as a subscript t):

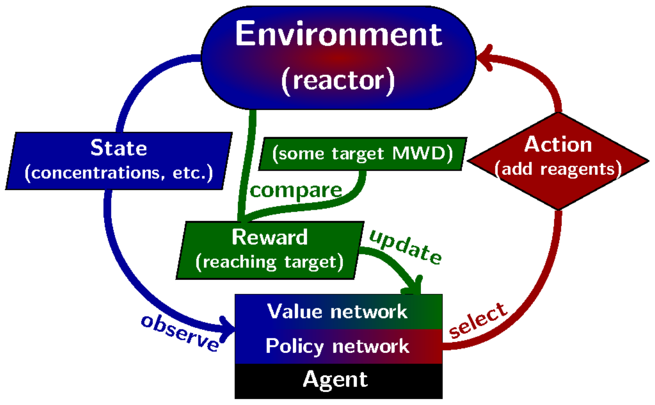


Fig. 4 A schematic diagram of applying deep reinforcement learning in the ATRP reactor control setting.

State (s_t) At each timestep, the agent is given a vector, s_t, that is interpreted as the current state of the reaction system. The state vector is used by the agent to select actions. Here, s_t includes: (i) the concentrations of the non-trace species: monomer, dormant chains ($P_1Br, \cdots, P_NBr$), and Cu-based ATRP catalysts, (ii) the volume of the solution, and (iii) binary indicators of whether each of the addable reagents has reached its budget. Note that we include the monomer quantity into the state vector by adding it onto the quantity of the initiator, or the shortest dormant chain.

Action (a_t) The agent is given a set of actions, $\mathscr{A}$, from which to select an action, a_t, to apply at timestep t. The set of actions is fixed and does not change throughout the simulation. Here, the actions correspond to the addition of a fixed amount of a chemical reagent. The set of actions, $\mathscr{A}$, also includes a *no-op*, selection of which means that no action is taken on the reaction simulation environment. The addable reagents are listed in Table 2, along with the amount that is added when the action is selected and the budget. When a reagent reaches its budget, the agent may still select the corresponding action, but this action becomes a no-op and does not alter the reaction simulation environment. Although the simulation allows addition of solvent, the effects of this action are not examined here. A very small amount of solvent is, however, used to initialize the simulation with a non-zero volume of a non-reactive species. Inclusion of other actions, such as changes in temperature, are possible but these are also not examined here.

Reward (r_t) At each timestep, the agent is given a reward, r_t, that indicates the degree to which the agent is succeeding at its task. In many RL problems, rewards may accrue at any time point. Here, however, the reward is based on the final MWD and so the agent receives a reward only when the reaction has run to completion. In practice, we allow the agent to interact with the simulation until all addable reagents have reached their budgets. The simulation then continues for a terminal simulation time of $t_{terminal} = 10^5$ seconds. The simulation environment then provides a reward to the agent based on the difference between the ending dormant chain MWD and the target MWD. This reward is defined in a two-level manner: when the maximum absolute difference between the normalized ending MWD and target MWD is less than 1×10^{-2} the agent obtains a reward of 0.1, and when this difference is less than 3×10^{-3}, the agent obtains a reward of 1.0. This two-level reward structure was determined empirically, with the lower first-level reward helping guide the agent in the early stages of training.

Table 2 The initial amounts, addition unit amounts, and budget limits used for simulating styrene ATRP in this work. All quantities are in units of *mol*.

Addable reagents	Initial	Addition unit	Budget limit
Monomer	0	0.1	10.0
Activator	0	0.004	0.2
Deactivator	0	0.004	0.2
Initiator	0	0.008	0.4
Solvent	0.01	0	0

A single simulated ATRP reaction corresponds, in RL, to a single *episode*. Each episode begins with a small amount of solvent (Table 2) and iterates through steps in which the agent is given the current state, s_t, the agent selects an action a_t that is applied to the simulation, and the simulation then runs for a time t_{step}. When all addable reagents have reached their budgets, the simulation continues for $t_{terminal} = 10^5$ seconds and returns a reward based on the difference between the ending dormant chain MWD and the target MWD.

To train the agent, we use the A3C algorithm, a recent advance in actor-critic methods[125] that achieved state-of-the-art performance on many discrete-action control tasks.[126] Actor-critic[127] algorithms are a subclass of RL algorithms based on simultaneous training of two functions:

Policy ($\pi_{\theta_p}(s_t)$) The policy is used to select actions, e.g., which chemical reagent to add at time t. As shown schematically in Fig. 3, actions are drawn from a probability distribution. The policy function generates this probability distribution, $\pi_{\theta_p}(a_t|s_t)$, which specifies, given the state of the ATRP reactor s_t, the probability that action a_t should be selected. The subscript θ_p represents the set of parameters that parameterize the policy function. In A3C, where a neural network is used for the policy, θ_p represents the parameters in this neural network.[128,129]

Value ($V_{\theta_v}(s_t)$) Although the policy function is sufficient for use of the RL controller, training also involves a value function, $V_{\theta_v}(s_t)$. Qualitatively, this function is a measure of whether the reaction is on track to generate rewards. More precisely, we define a *return* $R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$ which includes not only the reward at the current state, but also future states up to timestep T. This is especially relevant here, as rewards are based on the final MWD and so are given only at the end of a reaction. A factor γ, which is greater than 0 and less than 1, discounts the reward for each step into the future, and is included to guarantee convergence of RL algorithms. The value function, $V_{\theta_v}(s_t)$, approximates the expected return, $\mathbb{E}[R_t|s_t]$, from state s_t. A3C uses a neural network for the value function, and θ_v represents the parameters in this network.

Below, we compare results from two different neural network architectures, labeled FCNN and 1D-CNN (see Section 3.3).

During training, A3C updates the parameters, θ_p and θ_v, of the policy and value functions. The actor-critic aspect of A3C refers to the use of the value function to critique the policy's ability to select valuable actions. To update θ_p, policy gradient steps are taken according to the direction given by $\nabla_{\theta_p} \log \pi_{\theta_p}(a_t|s_t)\big(R_t - V_{\theta_v}(s_t)\big)$. Note that the current value function, $V_{\theta_v}(s_t)$, is used to update the policy, with the policy gradient step being in a direction that will cause the policy to favor actions that maximize the expected return. This may be viewed as using the value function to critique actions being selected by the policy. Moreover, the policy gradient becomes more reliable when the value function estimates the expected return more accurately. To improve the value function, the parameters θ_v are updated to minimize the ℓ^2 error $\mathbb{E}\big(R_t - V_{\theta_v}(s_t)\big)^2$ between the value function, $V_{\theta_v}(s_t)$, and the observed return, R_t. The observed return is obtained by using the current policy to select actions to apply to the reaction simulation environment.

The training therefore proceeds iteratively, with the current value function being used to update the policy and the current policy being used to update the value function. The parameter updates occur periodically throughout the course of an episode, or single polymerization reaction. The current policy is first used to generate a length-L sequence of state transitions $\{s_t, a_t, r_t, s_{t+1}, a_{t+1}, r_{t+1}, \cdots, s_{t+L}\}$. This length-$L$ sequence is referred to as a *rollout*. At the end of each rollout, the information generated during the rollout is used to update θ_p and θ_v. To take advantage of multi-core computing architectures, the training process is distributed to multiple asynchronous parallel learners. A3C keeps a global version of θ_p and θ_v. Each learner has access to a separate copy of the reaction simulation environment and a local version of θ_p and θ_v. After a learner performs a rollout, it generates updates to θ_p and θ_v. These updates are then applied to the global versions of θ_p and θ_v, and the learner replaces its local version with the global version. In this manner, each learner periodically incorporates updates generated by all learners.

3.3 Additional implementation details

The neural networks used for the policy and value functions share a common stack of hidden layers, but use separate final output layers. We compare results from two different network architectures for the hidden layers. The first architecture, FCNN, is a simple fully-connected neural network with two hidden layers containing 200 and 100 hidden units, respectively. The second architecture, 1D-CNN, is convolutional. In 1D-CNN, the input feature vector is fed into a first 1D convolutional layer having 8 filters of length 32 with stride 2, followed by a second 1D convolutional layer having 8 filters of length 32 with stride 1. The output of the second 1D convolutional layer is then fed into a fully-connected layer with 100 units. All hidden layers use rectifier activation. The final layer of the value network produces a single scalar output that is linear in the 100 units of the last hidden layer. The final layer of the policy network is a softmax layer of the same 100 hidden units, with a length-6 output representing a probability distribution over the 6 actions. For a crude estimate of model complexity, FCNN and 1D-CNN contain 42607 and 9527 trainable parameters, respectively.

We implemented the A3C algorithm with 12 parallel CPU learners.[62] The discount factor in the return is $\gamma = 0.99$, and the maximum rollout length is 20. The length of a rollout may be shorter than 20 when the last state in the sequence is a terminal state. After a learner collects a length-L rollout, $\{s_t, a_t, r_t, s_{t+1}, a_{t+1}, r_{t+1}, \cdots, s_{t+L}\}$, it generates updates for θ_p and θ_v by performing stochastic gradient descent steps for each $t' \in \{t, \cdots, t+L-1\}$. Define the bootstrapped multi-step return $R'_{t'} = I_{t+L}\gamma^{t+L-t'} V_{\theta'_v}(s_{t+L}) + \sum_{i=t'}^{t+L} \gamma^{i-t'} r_i$ where $I_{t+L} = 0$ if s_{t+L} is the terminal state and 1 otherwise. The prime on θ'_v in $V_{\theta'_v}(s_{t+L})$ indicates that the value function is evaluated using the local copy of the network parameters. The update direction of θ_p is set according to

$$d\theta_p = -\nabla_{\theta'_p} \log \pi_{\theta'_p}(a_{t'}|s_{t'})\big(R'_{t'} - V_{\theta'_v}(s_{t'})\big) + \beta \nabla_{\theta'_p} H\big(\pi_{\theta'_p}(s_{t'})\big).$$

$H\big(\pi_{\theta'_p}(s_{t'})\big)$ is the entropy of $\pi_{\theta'_p}(s_{t'})$ and acts as a regularization term that helps prevent $\pi_{\theta'_p}(s_{t'})$ from converging to sub-optimal solutions. β is the regularization hyperparameter, for which we use $\beta = 0.01$. θ_v is updated according to the direction of

$$d\theta_v = \nabla_{\theta'_v}\big(R'_{t'} - V_{\theta'_v}(s_{t'})\big)^2.$$

Updates of the network parameters are done using the ADAM optimizer[130] with a learning rate of 1×10^{-4}.

Additionally, after each action is drawn from the probability distribution generated by the policy, the agent repeats the action for 4 times before selecting the next action. This repetition shortens the length of a full episode by a factor of 4 from the RL agent's perspective and so prevents the value function from exponential vanishing.[131]

4 Results and discussion

4.1 Targeting Gaussian MWDs with different variance

Our first goal is to train the RL controller against some MWDs with simple analytic forms, for which Gaussian distributions with different variances seem a natural choice. Seemingly simple, Gaussian MWDs exemplify the set of symmetric MWDs the synthesis of which requires advanced ATRP techniques such as activators regenerated by electron transfer (ARGET).[22] Living polymerization produces a Poisson distribution with a variance that depends only on the average chain length, which is set by the monomer-to-initiator ratio. The variance from the ideal living polymerization provides a lower limit to the variance of the MWD. Here, we choose Gaussian distributions with variances ranging from near this lower limit to about twice that limit. Increasing the variance of the MWD can have substantial effects on the properties of the resulting material.[132]

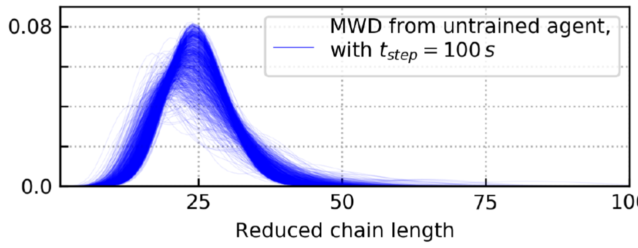


Fig. 5 Superposition of 1000 ending MWDs from untrained agents when the time interval between actions is 100 seconds. Vertical axis is fraction of polymer chains.

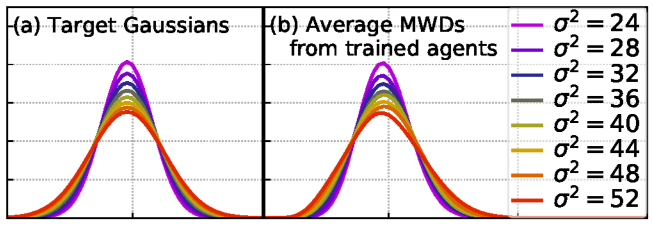


Fig. 6 Comparison of the human-specified target Gaussian MWDs with the average ending MWDs given by trained 1D-CNN agents, with averaging being over 100 episodes. The horizontal and vertical spacings between dotted line grids are 25 and 0.02, respectively.

For this task, we set the time interval between two actions to 100 seconds. This setting was chosen for two main reasons. First, due to the choice of the addition unit amounts and budget limits of addable reagents, it typically takes 300∼400 simulator steps to finish one episode, and so this choice of time interval corresponds to ∼10 hours of real reaction time before the terminal step. More importantly, it allows an untrained RL controller to produce a widely variable ending MWD, as illustrated by the 1000 MWDs of Fig. 5. A widely variable ending MWD is necessary for RL agents to discover strategies for target MWDs through self-exploration.[133,134]

As specific training targets, we select Gaussian MWDs with variances (σ^2's) ranging from 24 to 52, which covers the theoretical lower limit of the variance to a variance of more than twice this limit. Fig. 6(a) shows the span of these target MWDs. A summary of the trained 1D-CNN agents' performance on this task is shown in Fig. 6(b). Each ending MWD is an average over 100 episodes, generated using the trained 1D-CNN controller. Note that this MWD averaging is equivalent to blending polymer products generated in different reactions,[95] a common practice in both laboratory and industrial polymerization.[135–138] The trained 1D-CNN agent used in these test runs is that which gave the best performance in the training process, i.e., the neural network weights are those that generated the highest reward during the training process. During training, termination reactions are not included

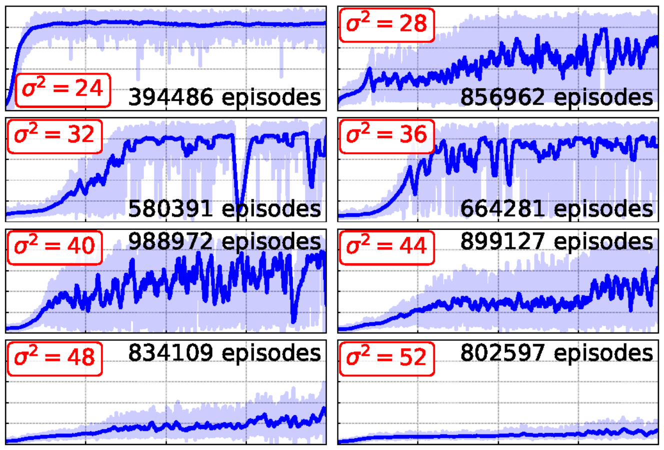


Fig. 7 Learning curves for training FCNN agents on the target Gaussian MWDs of Fig. 6. Horizontal axis is number of episodes, or simulated reactions, with total number of episodes shown in legend. The vertical axis is the instantaneous (light blue) or averaged (dark blue) reward, as defined in the main text, on a 0 to 1 scale.

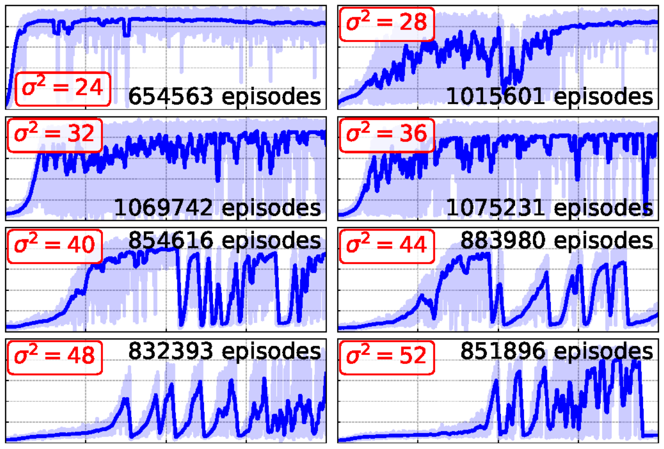


Fig. 8 Learning curves for training 1D-CNN agents on the target Gaussian MWDs of Fig. 6. Convention is as in Fig. 7.

in the simulation, but during testing, these reactions are included. For all 8 target Gaussian MWDs, the average ending MWDs are remarkably close to the corresponding targets. The maximum absolute deviation from the target MWD is an order of magnitude less than the peak value of the distribution function. These results show that control policies learned on simulation environments that exclude termination transfer well to environments that include termination. This is perhaps not surprising because ATRP of styrene is close to an ideal living polymerization, with less than 1% of monomers residing in chains that underwent a termination reaction. Tests on changing other aspects of the polymerization simulation are given in the following sections.

4.1.1 Training process and learning curves

Fig. 7 and 8 compare the learning curves of FCNN and 1D-CNN agents. The horizontal axis shows the number of ATRP experiments (episodes) run by the agent during the training process. The vertical axis shows the reward received by the agent, which runs from 0.0 to the maximum possible reward of 1.0. The dark blue lines average over a window of length 10000 and so reflects the agents' average performance during training. The light blue regions average over a window of length 100 and so may be interpreted as the agents' instantaneous performance. The transition from low to high average reward partially reflects the two-level reward structure, in which the reward is 0.1 for loose agreement with the target MWD and 1.0 for tight agreement.

Two general trends emerge from these learning curves. The first is that broader target MWDs require strategies that are harder for the RL agents to learn. When the target MWD is a Gaussian with variance 24, both FCNN and 1D-CNN can learn a strategy in less than 10^5 training episodes. As the variance of the target distribution increases, the number of required training episodes increases substantially. The second general trend is that the 1D-CNN outperforms the FCNN. For the narrower target distributions, both architectures obtain similar peak performance, but the 1D-CNN trains faster and the performance is more steady. For broader target distributions, only the 1D-CNN could achieve the 1.0 tight-threshold reward consistently.

4.1.2 Transferability tests on noisy environments

To test the robustness of the learned control policies, the trained 1D-CNN agents were evaluated on simulation environments that include both termination reactions and simulated noise.[139–141] We introduce noise on the states as well as actions. On states, we apply Gaussian noise with standard deviation 1×10^{-3} on every observable quantity. (The magnitude of the observable quantities range from 0.01 to 0.1.) In the simulation, we introduce three types of noise. First, the time interval between consecutive actions is subject to a Gaussian noise, whose standard deviation is 1% of the mean time interval. Gaussian noise is also applied to the amount of chemical reagent added for an action, again with a standard deviation that is 1% of the addition amount. Lastly, every kinetics rate constant used in non-terminal steps is subject to Gaussian noise, with the standard deviation being 10% of the mean value. Note that we crop the Gaussian noise in the simulation at $\pm 3\sigma$ to avoid unrealistic physics, such as negative time intervals, addition of negative amounts, or negative kinetic rate constants. Once all budgets have been met, the simulation enters its terminal step and the RL agent no longer has control over the process. During this terminal step, we do not apply noise.

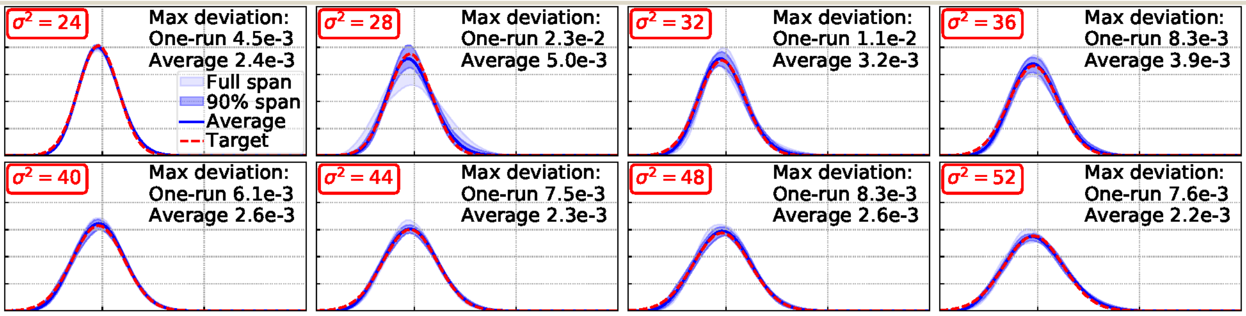


Fig. 9 Performance of 1D-CNN agents trained on the target Gaussian MWDs of Fig. 6 on simulation environments that include both termination reactions and noise. In each subplot, the horizontal axis represents the reduced chain length and runs from 1 to 75, and the vertical axis represents fraction of polymer chains and runs from 0.0 to 0.11.

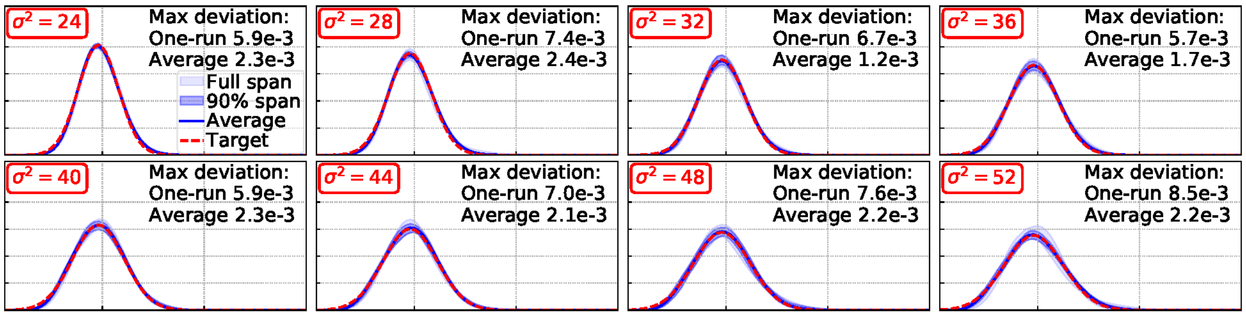


Fig. 10 Performance of 1D-CNN agents trained on noisy, with-termination environments targeting Gaussian MWDs of Fig. 6. Convention is as in Fig. 9.

Performance of the 1D-CNN agents, trained against the target Gaussian MWDs of Fig. 6, on noisy environments is shown in Fig. 9. The trained agent is used to generate 100 episodes and the statistics of final MWDs are reported in a variety of ways. The average MWD from the episodes is shown as a solid dark blue line. The light blue band shows the full range of the 100 MWDs and the blue band shows, at each degree of polymerization, the range within which 90 of the MWDs reside. The control policies learned by 1D-CNN agents seem to be robust. The deviation of the average MWD is an order of magnitude less than the peak value of the MWD. Deviations of the MWD from a single episode can vary more substantially from the target MWD, but the resulting MWDs are still reasonably close to the target MWD. On average, the maximum absolute deviation between a one-run MWD and the target is still less than 5% of the peak MWD value.

4.1.3 Training directly on noisy environments

Training the RL agents on noisy environments can significantly reduce the deviations of the single-run MWDs from the target MWD, as shown in Fig. 10. Noticeably, on the $\sigma^2 = 28$ environment, the "one-run" maximum absolute deviation is reduced from 2.3×10^{-2} to 7.4×10^{-3}, a reduction of over a factor of 3. These results are consistent with an expected advantage of state-dependent control policies, where the agents can respond to the real-time status of the reactor and autonomously choose the proper action to perform. Even though the states may be noisy, the RL agents are still able to detect patterns and use them to form a probability distribution over actions that maximizes the chance of reaching the target MWDs.

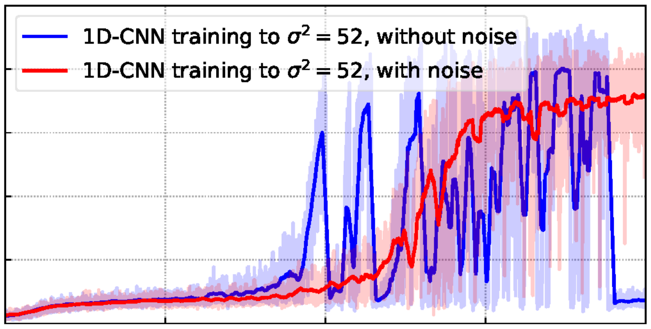


Fig. 11 Learning curves of the 1D-CNN agent targeting Gaussian MWD with $\sigma^2 = 52$, trained on environments with and without simulated noises. Convention is as in Fig. 7, with the horizontal axis having a range of 851896 episodes.

Another interesting finding is that performance collapses during the training process may be alleviated by introducing noise to the training environment. Fig. 11 compares the learning curve of the 1D-CNN agent on the non-noisy environment with that on the noisy environment, both targeting a Gaussian MWD with $\sigma^2 = 52$. Although, on the non-noisy environment, the agent can learn a high-reward strategy more quickly and achieve a slightly higher peak performance, the learning curve on the noisy environment is much steadier. Intuitively, exposing the agent to noisy states increases its tolerance to abrupt changes in concentrations of observables and so may improve the generalization of the learned network.[142] Moreover, introducing noise may also be regarded as a stochastic regularization technique.[143,144] Overall, introducing certain types of noise on the states and actions seems to have little adverse effect on the training while helping the agents achieve better generalization.

4.2 Targeting MWDs with diverse shapes

Beyond Gaussian MWDs, we also trained the 1D-CNN agent against a series of diverse MWD shapes. We have chosen bimodal distributions as a challenging MWD to achieve in a single batch process. Such bimodal distributions have been previously studied as a means to controlling the microstructure of a polymeric material.[145–147]

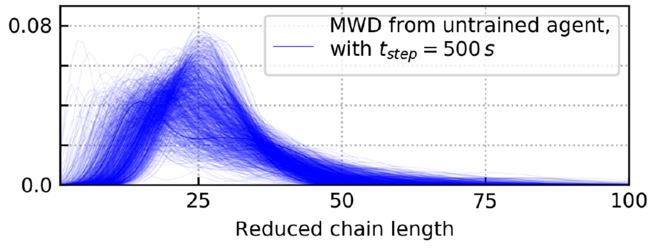


Fig. 12 Superposition of 1000 ending MWDs from untrained agents when the time interval between actions is 500 seconds. Vertical axis is fraction of polymer chains.

To enable automatic discovery of control policies that lead to diverse MWD shapes, it is necessary to enlarge the search space of the RL agent, which is related to the variability in the ending MWDs generated by an untrained agent. We found empirically that a larger time interval between actions leads to wider variation in the MWDs obtained with an untrained agent. Throughout this section, the time interval between actions t_{step} is set to 500 seconds. Fig. 12 shows 1000 superimposed ending MWDs given by the untrained agent with this new time interval setting, and the span is much greater than in Fig. 5 where $t_{step} = 100$ seconds.

The target MWDs with diverse shapes are manually picked from 1000 random ATRP simulation runs (i.e., episodes under the control of an untrained agent). Agents trained on these targets have satisfactory performance. The average MWDs over 100 batch runs match the targets nearly perfectly. In addition, there is a large probability (90%) that a one-run ending MWD controlled by a trained agent falls into a thin band whose deviation from the target is less than 1×10^{-2} (Fig. 13). All these agents are trained on noisy, no-termination environments and evaluated on noisy, with-termination environments. The parameters specifying the noise are identical to those used in the earlier sections. The results indicate that a simple convolutional neural network with less than 10^4 parameters can encode control policies that lead to complicated MWD shapes with surprisingly high accuracy. Again, adding noise to the states, actions, and simulation parameters does not degrade the performance of the RL agents significantly. This tolerance to noise may allow transfer of control policies, learned on simulated reactors, to actual reactors.

To further investigate the potential transferability of the state-dependent control policies, we also evaluate the agents trained

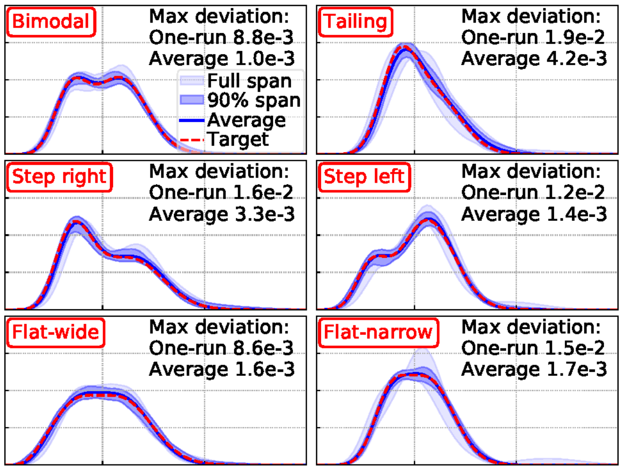


Fig. 13 Performance of trained 1D-CNN agents on noisy, with-termination environments targeting diverse MWD shapes. In each subplot, the horizontal axis represents the reduced chain length and runs from 1 to 75, and the vertical axis is fraction of polymer chains and runs from 0.0 to 0.08.

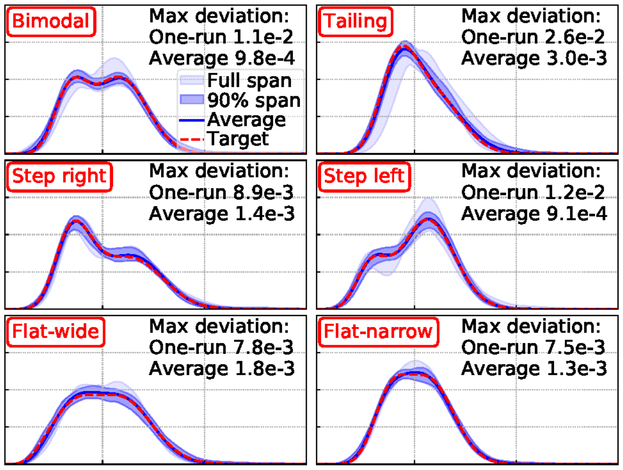


Fig. 14 Performance of trained 1D-CNN agents on noisy, with-termination environments targeting diverse MWD shapes, where the chain propagation rate constant is increased by 100% relative to the environments on which the agents were trained. Convention is as in Fig. 13.

above on environments where the propagation rate constant k_p is increased by 100%. The other rate constants (k_a, k_d, and k_t) were held fixed, such that we are varying the relative time scales of the two interacting chemistries, propagation versus activation/deactivation, in ATRP (Fig. 1). The increase in chain propagation alters, for example, the average number of monomers added to an active chain before it is converted back to a dormant chain. In applying the agents, the time intervals t_{step} and $t_{terminal}$ are reduced by 50% so that the reactions have a similar monomer conversion rate before and after the change to k_p. As shown by Fig. 14, this significant change in the ATRP reaction kinetics only slightly downgrades the agents' performance, with the average ending MWD remaining close to the target. The successful transfer of agents trained on one set of kinetic parameters to a simulation with a different set of kinetic parameters suggests that having the agents base their decisions on the current state of the reaction leads to control policies that can transfer between chemical systems.

5 Conclusion

This paper introduces a general methodology for using deep reinforcement learning techniques to control a chemical process in which the product evolves throughout the progress of the reaction. A proof-of-concept for the utility of this approach is obtained by using the controller to guide growth of polymer chains in a simulation of ATRP. ATRP was chosen because this reaction system allows detailed control of a complex reaction process. The resulting controllers are tolerant to noise in the kinetic rate constants used in the simulation, noise in the states on which the controller bases its decisions, and noise in the actions taken by the controller. This tolerance to noise may allow agents trained on simulations of the reaction to be transferred to the actual laboratory without extensive retraining, although evaluation of this aspect is left to future work. This approach, of carrying out initial training of a controller on a simulation, has been successfully applied in other domains such as robotics and vision-based RL.[101,126,148] Additional work is also needed to better understand the extent to which the controller can achieve synthetic targets when decisions are based on less detailed information regarding the state of the reactor. The ability of the approach to target multiple properties,[149,150] such as targeting MWD and viscosity simultaneously, or targeting more complex architectures, such as gradient or brush polymers, also remains to be explored. Our efforts to optimize the reinforcement learning methodology is still ongoing, and we hope to apply similar approaches to guide other chemical reactions.

A developmental open-source implementation of our approach is freely available on GitHub (`https://github.com/spring01/reinforcement_learning_atrp`) under the GPL-v3 license.

Conflict of interest

There are no conflicts to declare.

References

1 A. D. Dragan, G. J. Gordon and S. S. Srinivasa, in *Robotics Research*, Springer, 2017, pp. 309–326.
2 B. Boots, S. M. Siddiqi and G. J. Gordon, *The International Journal of Robotics Research*, 2011, **30**, 954–966.
3 S. V. Ley, D. E. Fitzpatrick, R. Ingham and R. M. Myers, *Angewandte Chemie International Edition*, 2015, **54**, 3449–3464.
4 J. Koutník, J. Schmidhuber and F. Gomez, Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation, 2014, pp. 541–548.
5 V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg and D. Hassabis, *Nature*, 2015, **518**, 529–533.
6 D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneer-

shelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel and D. Hassabis, *Nature*, 2016, **529**, 484–489.

7 D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel and D. Hassabis, *Nature*, 2017, **550**, 354–359.

8 D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan and D. Hassabis, *arXiv preprint*, 2017, arXiv:1712.01815.

9 M. Popova, O. Isayev and A. Tropsha, *arXiv preprint*, 2017, arXiv:1711.10907.

10 M. Olivecrona, T. Blaschke, O. Engkvist and H. Chen, *Journal of Cheminformatics*, 2017, **9**, 48.

11 Z. Zhou, X. Li and R. N. Zare, *ACS Central Science*, 2017, **3**, 1337.

12 K. Matyjaszewski, *Macromolecules*, 2012, **45**, 4015–4039.

13 K. Matyjaszewski and J. Xia, *Chemical Reviews*, 2001, **101**, 2921–2990.

14 K. Matyjaszewski and N. V. Tsarevsky, *Journal of the American Chemical Society*, 2014, **136**, 6513–6533.

15 C. J. Hawker, *Journal of the American Chemical Society*, 1994, **116**, 11185–11186.

16 F. di Lena and K. Matyjaszewski, *Progress in Polymer Science*, 2010, **35**, 959–1021.

17 A. Plichta, M. Zhong, W. Li, A. M. Elsen and K. Matyjaszewski, *Macromolecular Chemistry and Physics*, 2012, **213**, 2659–2668.

18 N. A. Lynd and M. A. Hillmyer, *Macromolecules*, 2005, **38**, 8803–8810.

19 N. A. Lynd and M. A. Hillmyer, *Macromolecules*, 2007, **40**, 8050–8055.

20 N. A. Lynd, B. D. Hamilton and M. A. Hillmyer, *Journal of Polymer Science Part B: Polymer Physics*, 2007, **45**, 3386–3393.

21 N. A. Lynd, M. A. Hillmyer and M. W. Matsen, *Macromolecules*, 2008, **41**, 4531–4533.

22 J. Listak, W. Jakubowski, L. Mueller, A. Plichta, K. Matyjaszewski and M. R. Bockstaller, *Macromolecules*, 2008, **41**, 5919–5927.

23 D. T. Gentekos, L. N. Dupuis and B. P. Fors, *Journal of the American Chemical Society*, 2016, **138**, 1848–1851.

24 K. Matyjaszewski and J. Spanswick, *Materials Today*, 2005, **8**, 26–33.

25 K. Min, H. Gao and K. Matyjaszewski, *Journal of the American Chemical Society*, 2005, **127**, 3825–3830.

26 A. Carlmark and E. E. Malmström, *Biomacromolecules*, 2003, **4**, 1740–1745.

27 P. W. Majewski and K. G. Yager, *ACS Nano*, 2015, **9**, 3896–3906.

28 P. W. Majewski, A. Rahman, C. T. Black and K. G. Yager, *Nature Communications*, 2015, **6**, 7448.

29 Y. Miura, A. Narumi, S. Matsuya, T. Satoh, Q. Duan, H. Kaga and T. Kakuchi, *Journal of Polymer Science Part A: Polymer Chemistry*, 2005, **43**, 4271–4279.

30 H. Gao and K. Matyjaszewski, *Macromolecules*, 2006, **39**, 4960–4965.

31 Z. Li, E. Kesselman, Y. Talmon, M. A. Hillmyer and T. P. Lodge, *Science*, 2004, **306**, 98–101.

32 H. Gao and K. Matyjaszewski, *Journal of the American Chemical Society*, 2007, **129**, 6633–6639.

33 Z. Wang, X. Pan, L. Li, M. Fantin, J. Yan, Z. Wang, Z. Wang, H. Xia and K. Matyjaszewski, *Macromolecules*, 2017, **50**, 7940–7948.

34 Z. Wang, X. Pan, J. Yan, S. Dadashi-Silab, G. Xie, J. Zhang, Z. Wang, H. Xia and K. Matyjaszewski, *ACS Macro Letters*, 2017, **6**, 546–549.

35 T. G. Ribelli, D. Konkolewicz, S. Bernhard and K. Matyjaszewski, *Journal of the American Chemical Society*, 2014, **136**, 13303–13312.

36 S. Dadashi-Silab, X. Pan and K. Matyjaszewski, *Macromolecules*, 2017, **50**, 7967–7977.

37 R. N. Carmean, T. E. Becker, M. B. Sims and B. S. Sumerlin, *Chem*, 2017, **2**, 93–101.

38 V. Kottisch, D. T. Gentekos and B. P. Fors, *ACS Macro Letters*, 2016, **5**, 796–800.

39 A. Goto and T. Fukuda, *Progress in Polymer Science*, 2004, **29**, 329–385.

40 W. Tang and K. Matyjaszewski, *Macromolecules*, 2006, **39**, 4953–4959.

41 E. D. Weiss, R. Jemison, K. J. Noonan, R. D. McCullough and T. Kowalewski, *Polymer*, 2015, **72**, 226–237.

42 J. G. Preturlan, R. P. Vieira and L. M. Lona, *Computational Materials Science*, 2016, **124**, 211–219.

43 M. Drache and G. Drache, *Polymers*, 2012, **4**, 1416–1442.

44 R. P. Vieira and L. M. Lona, *Polymer Bulletin*, 2016, **73**, 1795–1810.

45 P. H. Van Steenberge, D. R. DâĂŹhooge, Y. Wang, M. Zhong, M.-F. Reyniers, D. Konkolewicz, K. Matyjaszewski and G. B. Marin, *Macromolecules*, 2012, **45**, 8519–8531.

46 D. R. D'hooge, D. Konkolewicz, M.-F. Reyniers, G. B. Marin and K. Matyjaszewski, *Macromolecular Theory and Simulations*, 2012, **21**, 52–69.

47 P. Krys, M. Fantin, P. V. Mendonça, C. M. Abreu, T. Guliashvili, J. Rosa, L. O. Santos, A. C. Serra, K. Matyjaszewski and J. F. Coelho, *Polymer Chemistry*, 2017, **8**, 6506–6519.

48 P. Krys and K. Matyjaszewski, *European Polymer Journal*, 2017, **89**, 482–523.

49 M. Zhong, Y. Wang, P. Krys, D. Konkolewicz and K. Matyjaszewski, *Macromolecules*, 2013, **46**, 3816–3827.

50 Y. Li, *arXiv preprint*, 2017, arXiv:1701.07274.

51 K. Arulkumaran, M. P. Deisenroth, M. Brundage and A. A. Bharath, *arXiv preprint*, 2017, arXiv:1708.05866.

52 P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup and D. Meger, *arXiv preprint*, 2017, arXiv:1709.06560.

53 A. Krizhevsky, I. Sutskever and G. E. Hinton, Advances in

Neural Information Processing Systems, 2012, pp. 1097–1105.

54 L. Deng, G. Hinton and B. Kingsbury, Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, 2013, pp. 8599–8603.

55 Y. LeCun, Y. Bengio and G. Hinton, *Nature*, 2015, **521**, 436–444.

56 J. Schmidhuber, *Neural Networks*, 2015, **61**, 85–117.

57 G. J. Gordon, Advances in Neural Information Processing Systems, 2001, pp. 1040–1046.

58 G. J. Gordon, Proceedings of the Twelfth International Conference on Machine Learning, 1995, pp. 261–268.

59 H. Van Hasselt, A. Guez and D. Silver, AAAI, 2016, pp. 2094–2100.

60 Y. Liang, M. C. Machado, E. Talvitie and M. Bowling, Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, 2016, pp. 485–493.

61 J. Schulman, F. Wolski, P. Dhariwal, A. Radford and O. Klimov, *arXiv preprint*, 2017, arXiv:1707.06347.

62 V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver and K. Kavukcuoglu, International Conference on Machine Learning, 2016, pp. 1928–1937.

63 M. Fortunato, M. G. Azar, B. Piot, J. Menick, I. Osband, A. Graves, V. Mnih, R. Munos, D. Hassabis, O. Pietquin, C. Blundell and S. Legg, *arXiv preprint*, 2017, arXiv:1706.10295.

64 T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver and D. Wierstra, *arXiv preprint*, 2015, arXiv:1509.02971.

65 N. Roy and G. J. Gordon, Advances in Neural Information Processing Systems, 2003, pp. 1667–1674.

66 P. Nomikos and J. F. MacGregor, *AIChE Journal*, 1994, **40**, 1361–1375.

67 T. Binder, L. Blank, H. G. Bock, R. Bulirsch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J. P. Schlöder and O. von Stryk, in *Online Optimization of Large Scale Systems*, Springer, 2001, pp. 295–339.

68 M. A. Hussain, *Artificial Intelligence in Engineering*, 1999, **13**, 55–68.

69 B. Ydstie, *Computers & Chemical Engineering*, 1990, **14**, 583–599.

70 G. Lightbody and G. Irwin, *IEE Proceedings-Control Theory and Applications*, 1995, **142**, 31–43.

71 Y. Yang and D. Linkens, *IEE Proceedings-Control Theory and Applications*, 1994, **141**, 341–349.

72 N. Watanabe, *IFAC Proceedings Volumes*, 1994, **27**, 377–382.

73 M. Bahita and K. Belarbi, *IFAC-PapersOnLine*, 2016, **49**, 158–162.

74 M. Galluzzo and B. Cosenza, *Information Sciences*, 2011, **181**, 3535–3550.

75 B. Srinivasan, S. Palanki and D. Bonvin, *Computers & Chemical Engineering*, 2003, **27**, 1–26.

76 B. Srinivasan, D. Bonvin, E. Visser and S. Palanki, *Computers & Chemical Engineering*, 2003, **27**, 27–44.

77 Y. Nie, L. T. Biegler and J. M. Wassick, *AIChE Journal*, 2012, **58**, 3416–3432.

78 J. Lim, M. A. Hussain and M. Aroua, *Neurocomputing*, 2010, **73**, 3242–3255.

79 J. D. Bošković and K. S. Narendra, *Automatica*, 1995, **31**, 817–840.

80 T. Chovan, T. Catfolis and K. Meert, *AIChE Journal*, 1996, **42**, 493–502.

81 J. F. de Canete, P. del Saz-Orozco, R. Baratti, M. Mulas, A. Ruano and A. Garcia-Cerezo, *Expert Systems with Applications*, 2016, **63**, 8–19.

82 E. Nahas, M. Henson and D. Seborg, *Computers & Chemical Engineering*, 1992, **16**, 1039–1057.

83 S. Mahmoodi, J. Poshtan, M. R. Jahed-Motlagh and A. Montazeri, *Chemical Engineering Journal*, 2009, **146**, 328–337.

84 A. Hermansson and S. Syafiie, *Control Engineering Practice*, 2015, **45**, 98–109.

85 A. Nejati, M. Shahrokhi and A. Mehrabani, *Journal of Process Control*, 2012, **22**, 263–271.

86 D. Sbarbaro-Hofer, D. Neumerkel and K. Hunt, *IEEE Control Systems*, 1993, **13**, 69–75.

87 P. Turner, G. Montague and J. Morris, *4th International Conference on Artificial Neural Networks*, 1995, 284–289.

88 S. Syafiie, F. Tadeo and E. Martinez, *Engineering Applications of Artificial Intelligence*, 2007, **20**, 767–782.

89 S. Syafiie, F. Tadeo, E. Martinez and T. Alvarez, *Applied Soft Computing*, 2011, **11**, 73–82.

90 M. A. de Souza L. Cuadros, C. J. Munaro and S. Munareto, *Industrial & Engineering Chemistry Research*, 2012, **51**, 8465–8476.

91 C. Chatzidoukas, J. Perkins, E. Pistikopoulos and C. Kiparissides, *Chemical Engineering Science*, 2003, **58**, 3643–3658.

92 M. A. Hosen, M. A. Hussain and F. S. Mjalli, *Control Engineering Practice*, 2011, **19**, 454–467.

93 T. McAfee, N. Leonardi, R. Montgomery, J. Siqueira, T. Zekoski, M. F. Drenski and W. F. Reed, *Macromolecules*, 2016, **49**, 7170–7183.

94 F. H. Florenzano, R. Strelitzki and W. F. Reed, *Macromolecules*, 1998, **31**, 7226–7238.

95 R. Leonardi, C. Natalie, R. D. Montgomery, J. Siqueira, T. McAfee, M. F. Drenski and W. F. Reed, *Macromolecular Reaction Engineering*, 2017, 1600072.

96 N. Ghadipasha, W. Zhu, J. A. Romagnoli, T. McAfee, T. Zekoski and W. F. Reed, *Industrial & Engineering Chemistry Research*, 2017, **56**, 7322–7335.

97 A. Wu, Z. Zhu, M. F. Drenski and W. F. Reed, *Processes*, 2017, **5**, 17.

98 M. E. Taylor and P. Stone, *Journal of Machine Learning Research*, 2009, **10**, 1633–1685.

99 S. J. Pan and Q. Yang, *IEEE Transactions on Knowledge and Data Engineering*, 2010, **22**, 1345–1359.

100 X. Wang and J. Schneider, Advances in Neural Information Processing Systems, 2014, pp. 1898–1906.

101 P. Christiano, Z. Shah, I. Mordatch, J. Schneider, T. Black-

well, J. Tobin, P. Abbeel and W. Zaremba, *arXiv preprint*, 2016, arXiv:1610.03518.

102 S. Barrett, M. E. Taylor and P. Stone, Ninth International Conference on Autonomous Agents and Multiagent Systems-Adaptive Learning Agents Workshop (AAMAS-ALA), 2010.

103 S. Ross, G. J. Gordon and D. Bagnell, International Conference on Artificial Intelligence and Statistics, 2011, pp. 627–635.

104 W. Sun, A. Venkatraman, G. J. Gordon, B. Boots and J. A. Bagnell, *arXiv preprint*, 2017, arXiv:1703.01030.

105 R. P. Vieira and L. M. F. Lona, *Polímeros*, 2016, **26**, 313–319.

106 X. Li, W.-J. Wang, B.-G. Li and S. Zhu, *Macromolecular Reaction Engineering*, 2011, **5**, 467–478.

107 S. Zhu, *Macromolecular Theory and Simulations*, 1999, **8**, 29–37.

108 M. Al-Harthi, J. B. Soares and L. C. Simon, *Macromolecular Materials and Engineering*, 2006, **291**, 993–1003.

109 M. Najafi, H. Roghani-Mamaqani, M. Salami-Kalajahi and V. Haddadi-Asl, *Chinese Journal of Polymer Science*, 2010, **28**, 483–497.

110 M. Najafi, H. Roghani-Mamaqani, V. Haddadi-Asl and M. Salami-Kalajahi, *Advances in Polymer Technology*, 2011, **30**, 257–268.

111 S. Turgman-Cohen and J. Genzer, *Macromolecules*, 2012, **45**, 2128–2137.

112 K. A. Payne, D. R. DâĂŹhooge, P. H. Van Steenberge, M.-F. Reyniers, M. F. Cunningham, R. A. Hutchinson and G. B. Marin, *Macromolecules*, 2013, **46**, 3828–3840.

113 C. Toloza Porras, D. R. D'hooge, P. H. Van Steenberge, M.-F. Reyniers and G. B. Marin, *Macromolecular Reaction Engineering*, 2013, **7**, 311–326.

114 Y. Nakamura, T. Ogihara and S. Yamago, *ACS Macro Letters*, 2016, **5**, 248–252.

115 J.-S. Wang and K. Matyjaszewski, *Journal of the American Chemical Society*, 1995, **117**, 5614–5615.

116 T. E. Patten, J. Xia, T. Abernathy and K. Matyjaszewski, *Science*, 1996, **272**, 866.

117 K. Matyjaszewski, T. E. Patten and J. Xia, *Journal of the American Chemical Society*, 1997, **119**, 674–680.

118 A. Gridnev and S. Ittel, *Macromolecules*, 1996, **29**, 5864–5874.

119 R. P. Vieira, A. Ossig, J. M. Perez, V. G. Grassi, C. L. Petzhold, A. C. Peres, J. M. Costa and L. M. Lona, *Polymer Engineering & Science*, 2015, **55**, 2270–2276.

120 P. N. Brown, G. D. Byrne and A. C. Hindmarsh, *SIAM journal on scientific and statistical computing*, 1989, **10**, 1038–1051.

121 G. D. Byrne and A. C. Hindmarsh, *ACM Transactions on Mathematical Software (TOMS)*, 1975, **1**, 71–96.

122 A. Hindmarsh and G. Byrne, *EPISODE: an effective package for the integration of systems of ordinary differential equations.[For stiff or non-stiff problems, in FORTRAN for CDC or IBM computers; TSTEP, core integrator routine; CONVRT, to change between single and double precision coding]*, California univ., livermore (usa). lawrence livermore lab. technical report, 1977.

123 A. C. Hindmarsh, *IMACS Transactions on Scientific Computation*, 1983, **1**, 55–64.

124 K. R. Jackson and R. Sacks-Davis, *ACM Transactions on Mathematical Software (TOMS)*, 1980, **6**, 295–318.

125 T. Degris, P. M. Pilarski and R. S. Sutton, American Control Conference (ACC), 2012, 2012, pp. 2177–2182.

126 A. A. Rusu, M. Vecerik, T. Rothörl, N. Heess, R. Pascanu and R. Hadsell, *arXiv preprint*, 2016, arXiv:1610.04286.

127 V. R. Konda and J. N. Tsitsiklis, Advances in Neural Information Processing Systems, 2000, pp. 1008–1014.

128 R. S. Sutton, D. A. McAllester, S. P. Singh and Y. Mansour, Advances in Neural Information Processing Systems, 2000, pp. 1057–1063.

129 E. Greensmith, P. L. Bartlett and J. Baxter, *Journal of Machine Learning Research*, 2004, **5**, 1471–1530.

130 D. Kingma and J. Ba, *arXiv preprint*, 2014, arXiv:1412.6980.

131 V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra and M. Riedmiller, *arXiv preprint*, 2013, arXiv:1312.5602.

132 N. A. Lynd, A. J. Meuler and M. A. Hillmyer, *Progress in Polymer Science*, 2008, **33**, 875–893.

133 T. Jaksch, R. Ortner and P. Auer, *Journal of Machine Learning Research*, 2010, **11**, 1563–1600.

134 M. Kearns and S. Singh, *Machine Learning*, 2002, **49**, 209–232.

135 R. Jovanović, K. Ouzineb, T. F. McKenna and M. A. Dubé, Macromolecular Symposia, 2004, pp. 43–56.

136 M. K. Lenzi, M. F. Cunningham, E. L. Lima and J. C. Pinto, *Industrial & Engineering Chemistry Research*, 2005, **44**, 2568–2578.

137 P. J. DesLauriers, M. P. McDaniel, D. C. Rohlfing, R. K. Krishnaswamy, S. J. Secora, E. A. Benham, P. L. Maeger, A. Wolfe, A. M. Sukhadia and B. B. Beaulieu, *Polymer Engineering & Science*, 2005, **45**, 1203–1213.

138 M. Zhang and W. H. Ray, *Journal of Applied Polymer Science*, 2002, **86**, 1047–1056.

139 Y. Duan, X. Chen, R. Houthooft, J. Schulman and P. Abbeel, International Conference on Machine Learning, 2016, pp. 1329–1338.

140 T. Hester and P. Stone, Robot Soccer World Cup, 2013, pp. 536–543.

141 B. Bakker, Advances in Neural Information Processing Systems, 2002, pp. 1475–1482.

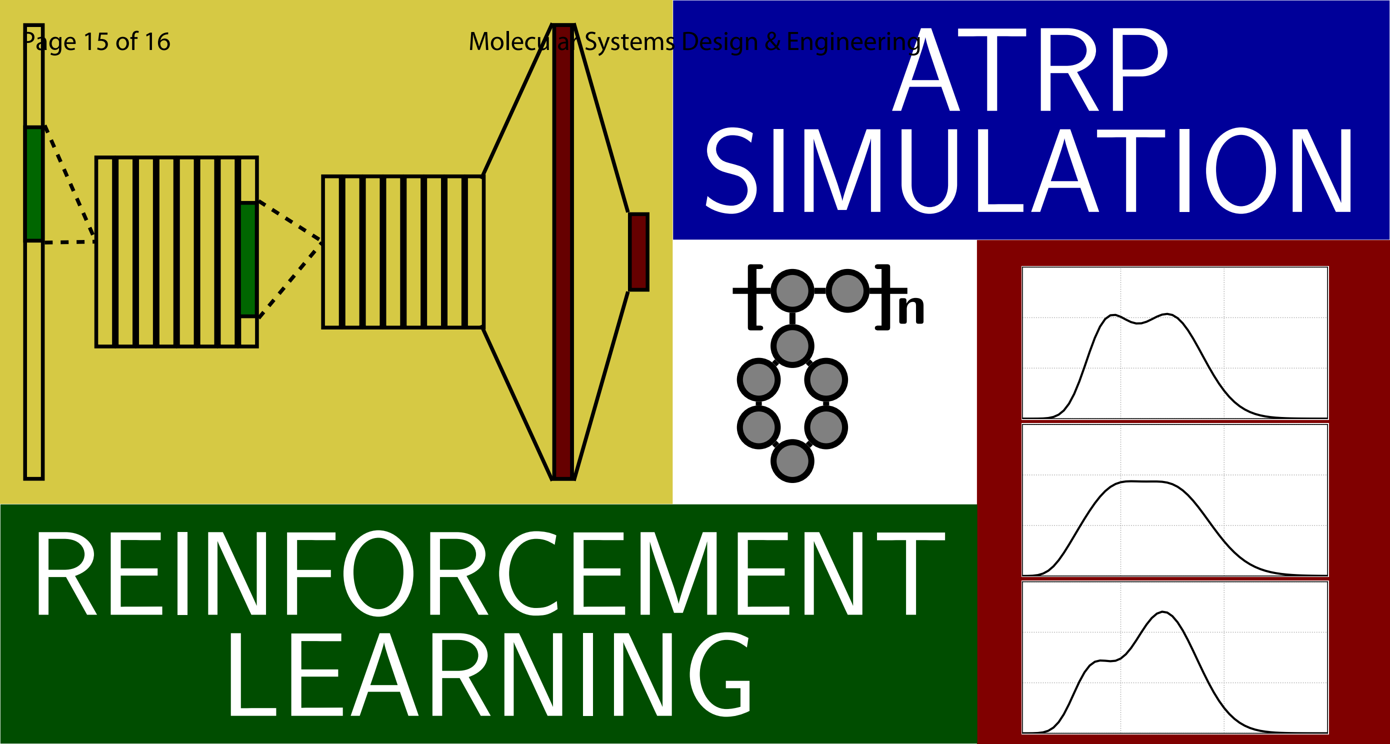
142 K. Jim, B. G. Horne and C. L. Giles, Advances in Neural Information Processing Systems, 1995, pp. 649–656.

143 N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, *Journal of Machine Learning Research*, 2014, **15**, 1929–1958.

144 S. Wager, S. Wang and P. S. Liang, Advances in Neural Information Processing Systems, 2013, pp. 351–359.

145 J. Yan, T. Kristufek, M. Schmitt, Z. Wang, G. Xie, A. Dang, C. M. Hui, J. Pietrasik, M. R. Bockstaller and K. Matyjaszewski, *Macromolecules*, 2015, **48**, 8208–8218.

146 Y. Zheng, Y. Huang and B. C. Benicewicz, *Macromolecular Rapid Communications*, 2017, **38**, 1700300.
147 T. Sarbu, K.-Y. Lin, J. Ell, D. J. Siegwart, J. Spanswick and K. Matyjaszewski, *Macromolecules*, 2004, **37**, 3120–3127.
148 S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz and D. Quillen, *The International Journal of Robotics Research*, 2016, 0278364917710318.
149 N. Sprague and D. Ballard, IJCAI, 2003, pp. 1445–1447.
150 K. Van Moffaert and A. Nowé, *The Journal of Machine Learning Research*, 2014, **15**, 3483–3512.


ATRP SIMULATION
n
REINFORCEMENT LEARNING

Combination of deep reinforcement learning and atom transfer radical polymerization gives precise in silico control on polymer molecular weight distributions.