Computational discovery of human reinforcement learning dynamics from choice behavior

Daniel Weinhardt
Osnabrück University
dweinhardt@uos.de

Maria Eckstein
Google Deepmind
mariaeckstein@deepmind.com

Sebastian Musslick
Osnabrück University,
Brown University
smusslick@uos.de

1 Introduction

One of the central goals of behavioral science is to understand the cognitive processes that occur within individuals. A significant challenge lies in the fact that that these processes are not directly observable and must be inferred from noisy behavioral data. Human reinforcement learning (HRL) is a prime example, positing that individuals adjust the values they assign to actions based on the rewards they receive. However, uncovering the learning dynamics underlying these action values is difficult because the action values are latent and cannot be directly measured. Instead, they must be inferred from observable behaviors, such as action choices. In this work, we introduce a novel machine learning approach for inferring latent HRL dynamics from human behavior leveraging recurrent neural networks (RNN) and sparse identification of non-linear dynamics (SINDy). This approach automates the discovery of interpretable models that explain HRL.

Our approach recovers HRL dynamics using a two-step process. First, drawing inspiration from previous work [1, 2, 3], we train an RNN to the behavior of humans performing a reinforcement learning task. Critically, we disentangle the RNN into multiple components, each representing a distinct cognitive mechanism. Each RNN component is constrained to a low-dimensional memory state representing the latent dynamical system variables for that cognitive mechanism. RNNs are a popular approach to fit human behavior due to their flexibility in capturing cognitive computations [4, 5, 6]. Once fitted, we simulate the RNN to generate time series data for each component's memory state. Next, we apply SINDy [7], a data-driven algorithm developed for identifying dynamical systems from time series data by representing it as a linear combination of predefined functions, often incorporating non-linear expressions like polynomials of the input variables. Here, we apply SINDy to the time series generated by each RNN component's memory state. This enables us to characterize the cognitive mechanism implemented by each component by identifying the underlying dynamical system that governs how it updates memorized action values based on previous rewards and choices. Together, these extracted dynamical systems yield an interpretable model of HRL.

In this work, we evaluate the algorithm's ability to recover various ground truth HRL models from simulated behavioral data in the context of a two-armed bandit task. These ground truth models (GT) incorporate a range of cognitive mechanisms, including Q-learning [8], value forgetting over time [9], choice perseverance [10], asymmetrical learning rates for positive and negative outcomes [11], and confirmation bias [12]. Critically, we attempted to infer these mechanisms from the choice behavior of the GT alone and under decision noise. Our results indicate that our approach is capable of recovering underlying cognitive mechanisms robustly and with a high accuracy, enabling the automated discovery of expressive yet interpretable HRL models.

NeurIPS 2024 Workshop on Behavioral Machine Learning.

2 Methods

2.1 Experimental paradigm

The two-armed bandit task is a paradigm for studying HRL where a participant repeatedly chooses between two options (or "arms"), each offering a potential reward $r = 1$ with some probability $P(r_\mathrm{i})$. Participants are tasked to maximize the rewards across a series of trials by selecting one the two arms a_i, $i \in 1, 2$, on each trial. In this study, $P(r_\mathrm{i})$ is unknown to the participant and fluctuates over the trials t with a drift rate of σ according to

$$P_{\mathrm{t+1}}(r_\mathrm{i}) = P_\mathrm{t}(r_\mathrm{i}) + \mathcal{N}(0, \sigma). \tag{1}$$

Maximizing rewards in this task requires participants to balance exploration (trying new options) and exploitation (sticking with known good ones) to adapt to the changing environment. Critically, participants must learn, based on experience, which of the two options is currently more rewarding.

2.2 Ground truth models

Task representation and action selection. Each GT is an agent that chooses between two actions and learns from the respective outcomes similarly to a human participant. That is, after selecting an action $\vec{A}$ and receiving a reward r, the model applies a Q-learning algorithm to update its estimate about the value of each action $\vec{V}$. Action selection is a stochastic procedure with the action probabilities $P(\vec{A})$ given by passing the current action values $\vec{V} = (v_1, v_2)$ multiplied by the inverse noise temperature β to the Softmax function

$$P(\vec{A}) = \mathrm{Softmax}(\beta \vec{V}), \tag{2}$$

where the single action values are computed by

$$v_\mathrm{i} = q_\mathrm{i} + c_\mathrm{i}, \tag{3}$$

with $i \in 1, 2$ indicating the action, q_i being the reward-history based part of the action value, and c_i being the choice-history based part.

Cognitive mechanisms. The model updates the chosen and the non-chosen action values v_ch and v_nch separately. Moreover, these updates are based on multiple cognitive mechanisms. The chosen reward value q_ch is updated by using the reward prediction error (RPE) $e_\mathrm{rp} = (r - q_\mathrm{ch})$ multiplied by an adaptive learning rate α [8]. This learning rate incorporates two further cognitive mechanisms, i.e. asymmetric learning rates for positive and negative outcomes [11] and a confirmation bias which enhances learning for confirming outcomes (e.g. high estimate q_ch and positive outcome) and reduces learning for contradicting outcomes (e.g. high estimate q_ch and negative outcome) [12]. The chosen choice value c_ch is updated by adding a choice persistence bias which lets the model favor previously chosen actions [10]. The non-chosen reward value q_nch is updated with value forgetting over time [9], while the non-chosen choice value $c_\mathrm{nch} = 0$. By implementing these different update architectures, we ensure the recoverability of a diverse array of mathematical functions through our approach. Summarized, these mechanisms influence the update equations according to

$$q_{\mathrm{ch,t+1}} = q_{\mathrm{ch,t}} + (\alpha_\mathrm{r} r + \alpha_\mathrm{p} \overline{r} + b_\mathrm{cb}(q_{\mathrm{ch,t}} - q_0)(r - q_0))e_\mathrm{rp}, \tag{4}$$

$$q_{\mathrm{nch,t+1}} = q_{\mathrm{nch,t}} + b_\mathrm{f}(q_{\mathrm{nch,t}} - q_0), \tag{5}$$

$$c_{\mathrm{ch,t+1}} = b_\mathrm{cp} a_\mathrm{repeat}, \tag{6}$$

$$c_{\mathrm{nch,t+1}} = 0, \tag{7}$$

with the current time step t, the learning rate α_r for positive outcomes, the learning rate α_p for negative outcomes, the variable $\overline{r}$ being the neglected version of the binary reward signal r (i.e. penalty/non-rewarded), the confirmation bias weight b_cb, the initial reward value q_0, the forget rate b_f, the choice persistence bias b_cp, and the binary signal a_repeat identifying whether a choice is repeated.

2.3 Framework for the automated discovery of cognitive models

Recurrent neural network. The RNN is fitted to emulate the given behavioral data, i.e. selecting an action $\vec{A}$, learning from the outcome r and selecting the next action based on the gained task-specific knowledge like the participant or agent would do. To achieve this objective, the RNN has to infer the underlying mental processes from mere choice data.

To facilitate the discovery of separate cognitive mechanisms, we disentangle the RNN into several smaller sub-networks and apply information bottlenecks for each of them (for similar approaches, see [2, 13]). These information bottlenecks control the inputs and outputs each sub-network is processing. The sub-networks represent the separated action value updates. Here, the sub-networks $\mathrm{f}_{\mathrm{q,ch}}$ and $\mathrm{f}_{\mathrm{c,ch}}$ update the reward-based and choice-based parts of the chosen action value v_{ch}, respectively. The reward-based and choice-based parts of the non-chosen action value v_{nch} are updated by the sub-networks $\mathrm{f}_{\mathrm{q,nch}}$ and $\mathrm{f}_{\mathrm{c,nch}}$, respectively. These updates have the structure

$$q_{\mathrm{ch,t+1}} = q_{\mathrm{ch,t}} + \mathrm{Sigmoid}(\mathrm{f}_{\mathrm{q,ch}}(q_{\mathrm{ch,t}}, r))e_{\mathrm{rp}}, \tag{8}$$

$$q_{\mathrm{nch,t+1}} = q_{\mathrm{nch,t}} + \mathrm{Tanhshrink}(\mathrm{f}_{\mathrm{q,nch}}(q_{\mathrm{nch,t}})), \tag{9}$$

$$c_{\mathrm{ch,t+1}} = c_{\mathrm{ch,t}} + \mathrm{Tanhshrink}(\mathrm{f}_{\mathrm{c,ch}}(c_{\mathrm{ch,t}}, a_{\mathrm{repeat}})), \tag{10}$$

$$c_{\mathrm{nch,t+1}} = c_{\mathrm{nch,t}} + \mathrm{Tanhshrink}(\mathrm{f}_{\mathrm{c,nch}}(c_{\mathrm{nch,t}})), \tag{11}$$

$$\vec{V} = \mathrm{ReLU}(\beta)(\vec{A}(q_{\mathrm{ch}} + c_{\mathrm{ch}}) + (1 - \vec{A})(q_{\mathrm{nch}} + c_{\mathrm{nch}})), \tag{12}$$

where Sigmoid is the sigmoid function imposing a value range of $0 < \mathrm{f}(x) < 1$ on the computed learning rate and $\mathrm{Tanhshrink}(x) = x - \tanh(x)$ imposing a quasi-zero-plateau on an otherwise quasi-linear function enhancing the RNN's ability to sparsify the value representation and therefore omit non-existing hypothesized cognitive mechanisms, and ReLU is the rectified linear unit function imposing a value range of $0 \leq x < \inf$ on the inverse noise temperature β. In Equation 8 the RPE e_{rp} is included as a hard-coded part due its big overlap with the functional inputs of $\mathrm{f}_{\mathrm{q,ch}}$ which encodes the update characteristics of the adaptive learning rate leading us the the biggest drawback—the separability of hypothesized cognitive mechanisms. This constraint is not only specific to our approach but is omnipresent in the behavioral sciences due to the extremely ill-posed setting. While the learning rate function $\mathrm{f}_{\mathrm{q,ch}}$ can approximate multiple mechanisms at once (e.g. asymmetric learning rates and confirmation bias), the RPE could not be separated from this function due to their identical inputs leading to non-identifiable influences on the respective update. We can mitigate this problem by making use of the consensus in the HRL and neuroscience literature about the RPE. Further, the inverse noise temperature β is a trainable parameter.

The RNN is trained by minimizing the cross entropy loss between the actual next action $\vec{A}_{\mathrm{t+1}}$ and the predicted probabilities $P(\vec{A}_{\mathrm{t+1}})$ through sole observation of the given current action $\vec{A}_{\mathrm{t}}$, the current reward r_{t} and its own latent state space $\vec{V}_{\mathrm{t}}$.

Sparse identification of non-linear dynamics. SINDy is a data-driven method for the discovery of dynamical systems that underlie given data. One reason why SINDy—and symbolic regression algorithms in general—were not applied yet to a significant extend to describe the dynamics of cognitive systems are the non-observable variables. But after fitting the RNN, we can uncover these latent variables over the trials t. Now we can apply SINDy on top of each of the fitted sub-networks to discover the learned cognitive mechanisms as closed-form solutions leading to a fully interpretable model of HRL. SINDy represents the cognitive mechanisms as sparse linear combinations of non-linear terms according to

$$\Delta q_{\mathrm{ch}} = \Theta([1, q_{\mathrm{ch}}, r, \overline{r}])\vec{W}_{\mathrm{q,ch}}, \tag{13}$$

$$\Delta q_{\mathrm{nch}} = \Theta([1, q_{\mathrm{nch}}])\vec{W}_{\mathrm{q,nch}}, \tag{14}$$

$$\Delta c_{\mathrm{ch}} = \Theta([1, c_{\mathrm{ch}}, a_{\mathrm{repeat}}])\vec{W}_{\mathrm{c,ch}}, \tag{15}$$

$$\Delta c_{\mathrm{nch}} = \Theta([1, c_{\mathrm{nch}}])\vec{W}_{\mathrm{c,nch}}, \tag{16}$$

where Θ is a library of polynomial combinations up to the second degree of the given candidate terms and $\vec{W}$ is the array of corresponding fitted weights to each of the RNN's sub-networks. We used two optimizers (STLSQ [7] and SR3 [14]) in our simulation study due to their different performance in different contexts. The SINDy weight arrays $\vec{W}$ are trained by minimizing the mean-squared error between the observed variables and the predicted ones. Additionally, the weights are regularized (STLSQ: $L0$ and $L2$-norm; SR3: $L1$-norm). This way SINDy fits a predictive but sparse and thus interpretable equation by setting most weights $w_{\mathrm{i}} \in \vec{W}$ to $w_{\mathrm{i}} = 0$.

2.4 Simulation experiments

Tested ground truth models. We tested our approach within a simulation study with two ground truth (GT) models. We initialized one GT θ_{full} with all described cognitive mechanisms (i.e. $\beta = 3.0$,

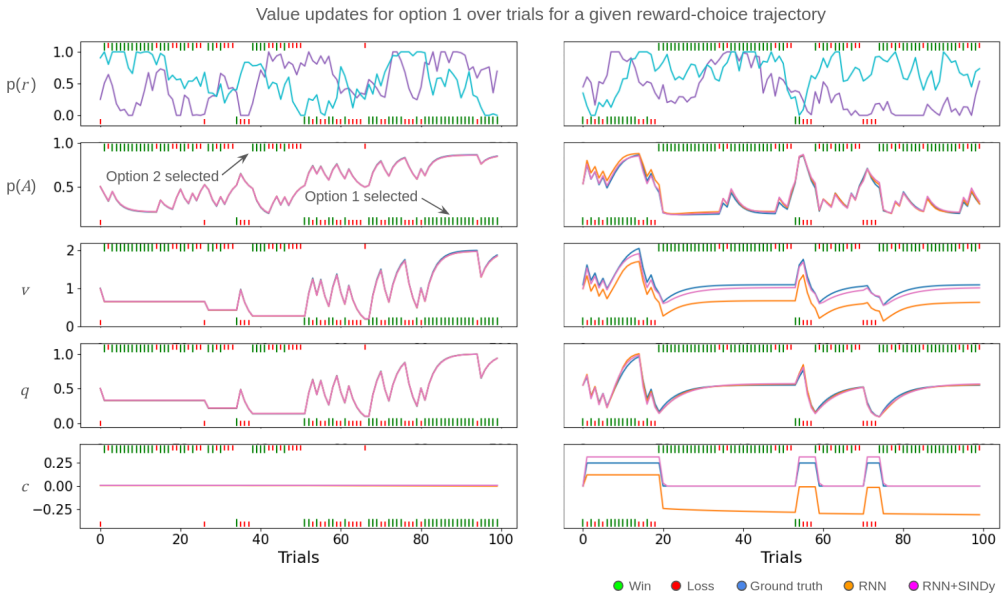

Figure 1: Reward probabilities $p(r)$, action selection probabilities $p(A)$, and value updates for option 1 over a given reward-choice trajectory for the simple model (left) and the full model (right) computed by the respective GT, the fitted RNN and the SINDy model.

$\alpha_\text{r} = 0.25$, $\alpha_\text{p} = 0.5$, $b_\text{cb} = 0.5$, $q_0 = 0.5$, $b_\text{cp} = 0.25$, $b_\text{f} = 0.2$) and a second GT θ_simple with Q-learning only (i.e. $\beta = 2.0$, $\alpha_\text{r} = \alpha_\text{p} = 0.35$, $q_0 = 0.5$, and the rest set to 0).

Dataset. Each GT performed the two-armed bandit task to generate training datasets of $n_\text{training} = 4096$ sessions and test datasets of $n_\text{test} = 1024$ sessions with $n_\text{trials} = 64$ trials per session. The two-armed bandit had drifting reward probabilities with a drift rate of $\sigma = 0.1$ and yielded binary rewards r.

3 Results

We verified our method's ability to recover different GTs both qualitatively and quantitatively. First qualitatively, we visually compared the trajectories of the single value updates for q, c and v, and the action selection probability $p(\vec{A})$ of the GTs against the corresponding fitted models over the trials t for given choice and reward trajectories. As shown in Figure 1, the value updates and the action selection probability were matched with a high accuracy for both GTs. In the case of the full model, the RNN established an offset for the choice value. However, this offset does not alter the action selection probabilities $p(\vec{A})$ because this offset is applied to both options equally, and therefore does not influence the difference between v_1 and v_2. In the case of RNN+SINDy, this minor problem is solved by simply eliminating this offset in the generated training data. Another important note is the already mentioned differing performance of the used SINDy optimizers STLSQ ($L0$ and $L2$ regularization) and SR3 ($L1$ regularization) which were used to recover θ_full and θ_simple, respectively. Thanks to the circumstance that SINDy is finding an equation for the approximated functions of the RNN, we can tell which optimizer works better by validating the results visually. The selection of the best-suited optimizer can be seen as a mere hyperparameter optimization known from classic machine learning approaches.

After confirming the models fit well on a couple of trained models, we analyzed the results of the actual simulation study quantitatively. Here we trained in total 16 models on 8 datasets from the simple GT and on 8 datasets from the full GT to verify our approach's stability. The mean test losses and standard deviations were $\mathcal{L}_\text{simple} = 0.5970 \pm 0.0011$ and $\mathcal{L}_\text{full} = 0.5265 \pm 0.0024$ for the simple and full GTs, respectively. These results verify our approach's stability. The lower loss $\mathcal{L}_\text{full}$ compared

to $\mathcal{L}_{\text{simple}}$ is explained by the higher β-parameter leading to less noise and also to probabilities closer to 0 and 1, thus reducing the overall distance to the actual binary targets in $\vec{A}_{\text{t+1}}$.

Further, we compared the mean weights $\vec{W}$ and the standard deviations of the fitted RNN+SINDy model with the respective GTs. As shown in Table 1, the respective weights $\vec{W}$ of the two fitted models ϕ_{full} and ϕ_{simple} were matched overall with a high accuracy, by not only identifying the active terms but also omitting the non-present ones. Recovering the simple GT works hereby better than recovering the full GT - even though the simple GT exhibits more noise. For the full GT, our approach struggles mostly with ambiguity—mostly for the simpler choice functions (columns $C_{\text{c,ch}}$ until c^2_{nch}). Here, the standard deviations are much bigger than in other cases, and even surpassing the respective mean weights. In the light of the stable test losses $\mathcal{L}_{\text{full}}$, this shows that our approach simply finds multiple solutions for the given observations. This can be further optimized by e.g. optimal experimental design taking uncertainties into account, thus collecting more informative samples.

Table 1: Weights of each candidate term of the full and the simple GTs θ_{full} and θ_{simple}, and the mean weights and standard deviations (σ) of the corresponding fitted models ϕ_{full} and ϕ_{simple}. The variables $C_{\text{q,ch}}$, $C_{\text{q,nch}}$, $C_{\text{c,ch}}$, and $C_{\text{c,nch}}$ are constants. The weights of each model are spanned over multiple rows. The mean weights and standard deviations were computed over 8 different runs.

Model	β	$C_{\text{q,ch}}$	α	q_{ch}	r	$\overline{r}$	α^2	αq_{ch}	αr	$\alpha\overline{r}$
θ_{full}	3.00	0.13	0.00	-0.25	0.00	0.50	0.00	0.00	0.00	0.00
ϕ_{full}	2.58	0.17	0.00	0.08	0.00	0.22	0.00	0.00	0.00	0.00
(σ)	0.07	0.01	0.00	0.05	0.00	0.01	0.00	0.00	0.00	0.00
θ_{simple}	2.00	0.35	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ϕ_{simple}	2.00	0.35	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
(σ)	(0.02)	(0.01)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)
Model	q^2_{ch}	$q_{\text{ch}}r$	$q_{\text{ch}}\overline{r}$	r^2	$r\overline{r}$	$\overline{r}^2$	$C_{\text{q,nch}}$	q_{nch}	q^2_{nch}	$C_{\text{c,ch}}$
θ_{full}	0.00	0.5	0.00	0.00	0.00	0.00	0.10	0.80	0.00	0.00
ϕ_{full}	-0.08	0.21	-0.11	0.00	0.00	0.22	0.08	0.81	0.00	0.32
(σ)	0.05	0.02	0.02	0.00	0.00	0.01	0.03	0.04	0.00	0.02
θ_{simple}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
ϕ_{simple}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
(σ)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)
Model	c_{ch}	a_{repeat}	c^2_{ch}	$c_{\text{ch}}a_{\text{repeat}}$	a^2_{repeat}	$C_{\text{c,nch}}$	c_{nch}	c^2_{nch}		
θ_{full}	0.00	0.25	0.00	0.00	0.00	0.00	0.00	0.00		
ϕ_{full}	0.14	-0.06	0.04	0.19	-0.06	0.00	0.49	-0.97		
(σ)	0.22	0.08	0.39	0.30	0.08	0.00	0.67	1.48		
θ_{simple}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		
ϕ_{simple}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		
(σ)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)		

4 Conclusion

Combining RNNs with SINDy enables the extraction of complex yet interpretable cognitive mechanisms underlying HRL from noisy behavioral data, while disregarding irrelevant mechanisms, as shown through the recovery of both simple and complex models. This data-driven approach offers a promising alternative to theory-driven HRL models that often rely on standard ML algorithms [15].

The inclusion of automated architecture search algorithms can advance model discovery, optimizing cognitive model structures based on empirical data and potentially uncovering novel mechanisms. While further refinement is needed, particularly around separability constraints, optimizer selection, and a more efficient experimental design, testing this framework on real-world data could establish automated architecture search as a key tool for personalized cognitive modeling in research and applied settings.

References

[1] Maria K. Eckstein et al. *Predictive and Interpretable: Combining Artificial Neural Networks and Classic Cognitive Models to Understand Human Learning and Decision Making*. Pages: 2023.05.17.541226 Section: New Results. May 17, 2023.

[2] Kevin J. Miller et al. *Cognitive Model Discovery via Disentangled RNNs*. Pages: 2023.06.23.546250 Section: New Results. June 26, 2023.

[3] Li Ji-An, Marcus K Benna, and Marcelo G Mattar. "Automatic discovery of cognitive strategies with tiny recurrent neural networks". In: *bioRxiv* (2023), pp. 2023–04.

[4] Matthew Botvinick and David C Plaut. "Doing without schema hierarchies: a recurrent connectionist approach to normal and impaired routine sequential action." In: *Psychological review* 111.2 (2004), p. 395.

[5] Amir Dezfouli et al. "Integrated accounts of behavioral and neuroimaging data using flexible recurrent neural network models". In: *Advances in neural information processing systems* 31 (2018).

[6] Baihan Lin, Djallel Bouneffouf, and Guillermo Cecchi. "Predicting human decision making in psychological tasks with recurrent neural networks". In: *PloS one* 17.5 (2022), e0267907.

[7] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. "Discovering governing equations from data by sparse identification of nonlinear dynamical systems". In: *Proceedings of the National Academy of Sciences* 113.15 (Apr. 12, 2016). Publisher: Proceedings of the National Academy of Sciences, pp. 3932–3937.

[8] Roy A Wise. "Dopamine, learning and motivation". In: *Nature reviews neuroscience* 5.6 (2004), pp. 483–494.

[9] Asako Toyama, Kentaro Katahira, and Hideki Ohira. "Reinforcement learning with parsimonious computation and a forgetting process". In: *Frontiers in human neuroscience* 13 (2019), p. 153.

[10] Samuel J Gershman. "Origin of perseveration in the trade-off between reward and complexity". In: *Cognition* 204 (2020), p. 104394.

[11] Christopher Mark Hill. *Reward and punishment: the neural correlates of reinforcement feedback during motor learning*. 2019.

[12] Max Rollwage et al. "Confidence drives a neural confirmation bias". In: *Nature communications* 11.1 (2020), p. 2634.

[13] Amir Dezfouli et al. "Disentangled behavioural representations". In: *Advances in neural information processing systems* 32 (2019).

[14] Peng Zheng et al. "A unified framework for sparse relaxed regularized regression: SR3". In: *IEEE Access* 7 (2018), pp. 1404–1423.

[15] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 2018.