Finding optimal Learning rate schedules for Non-negative Matrix Factorization using Reinforcement Learning

Anonymous authors
Paper under double-blind review

Abstract

Learning rate schedules are often picked out of experience or through trial and error, even though there are no real guarantees that the choice of schedule may lead to convergence across a wide range of tasks when training a complex Deep Learning network. In this paper we calculate a theoretical bound for the learning rate (LR) schedule that can be used along with SGD for the Non-negative Matrix Factorization (NMF) problem for guaranteed convergence, and also use a Reinforcement Learning agent to try and recover, or beat the theoretical bounds by finding efficient LR schedules for the NMF problem.

1 Introduction

Non-negative Matrix Factorization (NMF) is the problem of decomposing a matrix while guaranteeing positive factors. For a matrix $V \in \mathbb{R}_+^{m \times n}$, NMF finds two matrices $W \in \mathbb{R}_+^{m \times r}$ and $H \in \mathbb{R}_+^{r \times n}$ such that $V \approx WH$. NMF has been applied to recommendation systems Luo et al. (2014); Khan et al. (2020); Li et al. (2009a;b); Luo et al. (2015), image decomposition Tang et al. (2013); Zhao et al. (2008); Kalayeh et al. (2014); Padilla et al. (2011); Zhang et al. (2004), acoustic signals Kameoka et al. (2009); Févotte et al. (2018); Hennequin et al. (2010); Yoshii et al. (2013) and many other applications.

Several heuristic methods have been studied in the past for finding NMF factors Wang & Zhang (2012); Lin (2007), but more recently, gradient based methods have also shown to find good solutions. Typically this decomposition is found by using gradient descent to minimize a distance measure like $\|V - WH\|_F^2$ where $\|\cdot\|_F$ is the Frobenius norm. Each factor is alternatively updated using the gradient and a predetermined learning rate α (or schedule). For example, for W updates and loss function F, we can find $W^{t+1} = P \cdot (W^t - \alpha \nabla_W F(W^t, H^t))$. Using the Frobenius norm (1) above, Guan et al. (2012) proposed a Fast Gradient Descent method that uses a modified objective function to retain class information, and the Newton method to calculate the optimal step size, and proved that it converged faster than standard multiplicative rule based algorithms. Here, the step length (or learning rate) is calculated to ensure H^{t+1} (or H at time step/iteration t) is as close as possible to the minimum along ∇_H. A similar, commonly used rule to select step length (see Guan et al. (2011) for example) is the Armijo rule, where the step size α is iteratively calculated as a result of the Armijo line search Zhang et al. (2006). At each iteration t we aim to find an α^t that satisfies (for the W update) $F(W^{t+1}, H) \leq F(W^t, H) - c \cdot \alpha^t \|\nabla F(W^t, H)\|^2$.

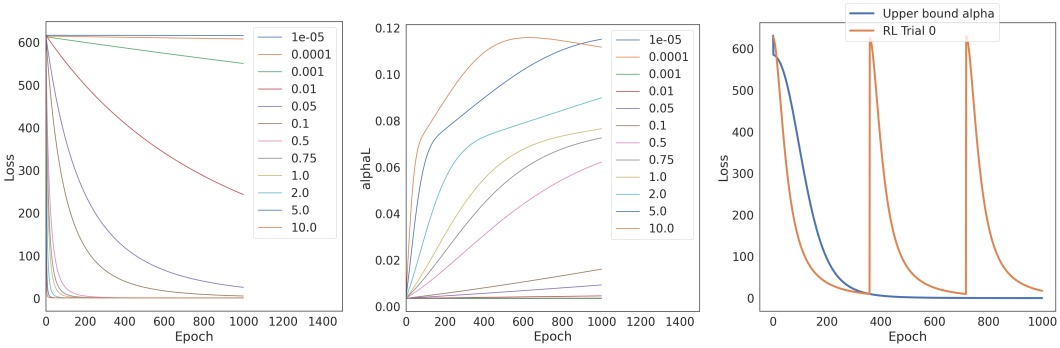
This is similar to schedules that are used in practice today for larger Deep Learning models, and are also included in popular packages such as TensorFlow and PyTorch. These updates to learning rate (LR) are not guaranteed to converge for complex non-convex functions, but are still used widely in practice today. Therefore, finding optimal LR schedules, beginning with well-behaved problems and followed by more complex deep learning training is beneficial to the larger Deep Learning community. In this paper, we propose that 1) Learning rate schedules are problem specific, and 2) Learning rate schedules can be learned using RL. In section 2, we present theoretical upper bounds for the LR schedule for the NMF problem. In section 3, we run experiments related to fixed, and theoretically optimal schedules for a synthetic NMF problem that is representative of real-world data. In section ?? we find optimal schedules for the NMF problem using an RL agent.

2 Theoretical bounds on optimal LR schedules for NMF

We first find theoretical bounds for optimal LR schedules that can be used for NMF with SGD. Theorem A.3 states that the optimal LR schedule (α^t) for NMF with SGD is bounded between $0 < \alpha^t < \frac{2}{L}$, where $L = \|W^t\|^2$. Any LR schedule within this bound will lead to convergence as we show in the experiments below. Detailed assumptions and proofs are included in Appendix A.

3 Results: NMF with SGD using fixed and RL schedules

We ran experiments for NMF with a sparsely initialized matrix of size 10000×1000 with a latent dimension of 100 for SGD and several fixed LRs. Figure 1 a) shows loss curves for fixed LRs from $1e-5$ to 10. Corresponding theoretical upper bound LRs are calculated based on Theorem A.3 as the problem converges (shown in Figure 1 b). We conducted 25 trials where the LR for the current epoch is calculated as a random value less than the upper bound for that LR; all these random LR schedules under the upper-bound still converge. Interestingly, we also observe that the loss curves from these random schedules closely follow the loss curves corresponding to the mean LR (see loss curves in Appendix B for two different random walk schedules).



(a) (b) (c)

Figure 1: a) Fixed LR loss curves b) Corresponding Upper bound (Theorem A.3) c) Learned schedule vs. a fixed upper-bound loss

We trained an RL agent using Proximal Policy Optimization (PPO) Schulman et al. (2017) to find LR schedules within bounds of $[0, 1]$ for NMF with SGD with observations of the current loss value, epoch number, and gradient norm.[1] The reward function used was the final loss value plus the number of epochs remaining. The agent is penalized with a high artificial loss on divergence and the episode is reset. As we can see in Fig. 1 c), the RL agent finds a schedule with a very high convergence rate, and is seen resetting and converging thrice within the 1000 epoch run, compared to the run with theoretical upper bound loss curve (more experiments and action histories in Appendix C).

4 Conclusion

In this paper we showed theoretical upper bound LR schedules for NMF convergence with SGD, and ran experiments with fixed, random and learned RL agent schedules showing that it is possible to learn problem-specific schedules for typical gradient-based learning problems. Our followup work with Deep Learning models also show promising results with the same method presented here.

[1]The upper bound of 1 used here is 10 times the theoretical upper bound shown in Fig 1 b) - However in practice we see the optimal agent suggesting low values of LR

URM Statement

The authors acknowledge that at least one key author of this work meets the URM criteria of ICLR 2023 Tiny Papers Track.

References

Cédric Févotte, Emmanuel Vincent, and Alexey Ozerov. Single-channel audio source separation with nmf: divergences, constraints and algorithms. Audio Source Separation, pp. 1–24, 2018.

Naiyang Guan, Dacheng Tao, Zhigang Luo, and Bo Yuan. Manifold regularized discriminative nonnegative matrix factorization with fast gradient descent. IEEE Transactions on Image Processing, 20(7):2030–2048, 2011.

Naiyang Guan, Dacheng Tao, Zhigang Luo, and Bo Yuan. Nenmf: An optimal gradient method for nonnegative matrix factorization. IEEE Transactions on Signal Processing, 60(6):2882–2898, 2012.

Romain Hennequin, Roland Badeau, and Bertrand David. Nmf with time–frequency activations to model nonstationary audio events. IEEE Transactions on Audio, Speech, and Language Processing, 19(4):744–753, 2010.

Mahdi M Kalayeh, Haroon Idrees, and Mubarak Shah. Nmf-knn: Image annotation using weighted multi-view non-negative matrix factorization. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 184–191, 2014.

Hirokazu Kameoka, Nobutaka Ono, Kunio Kashino, and Shigeki Sagayama. Complex nmf: A new sparse representation for acoustic signals. In 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 3437–3440. IEEE, 2009.

Zafran Khan, Naima Iltaf, Hammad Afzal, and Haider Abbas. Enriching non-negative matrix factorization with contextual embeddings for recommender systems. Neurocomputing, 380:246–258, 2020.

Tao Li, Chao Gao, and Jinglin Du. A nmf-based privacy-preserving recommendation algorithm. In 2009 First International Conference on Information Science and Engineering, pp. 754–757. IEEE, 2009a.

Tao Li, Chao Gao, and Jinglin Du. A nmf-based privacy-preserving recommendation algorithm. In 2009 First International Conference on Information Science and Engineering, pp. 754–757. IEEE, 2009b.

Chih-Jen Lin. On the convergence of multiplicative update algorithms for nonnegative matrix factorization. IEEE Transactions on Neural Networks, 18(6):1589–1596, 2007.

Xin Luo, Mengchu Zhou, Yunni Xia, and Qingsheng Zhu. An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems. IEEE Transactions on Industrial Informatics, 10(2):1273–1284, 2014.

Xin Luo, MengChu Zhou, Shuai Li, Zhuhong You, Yunni Xia, and Qingsheng Zhu. A nonnegative latent factor model for large-scale sparse matrices in recommender systems via alternating direction method. IEEE transactions on neural networks and learning systems, 27(3):579–592, 2015.

Pablo Padilla, Miriam López, Juan Manuel Górriz, Javier Ramirez, Diego Salas-Gonzalez, and I Alvarez. Nmf-svm based cad tool applied to functional brain images for the diagnosis of alzheimer's disease. IEEE Transactions on medical imaging, 31(2):207–216, 2011.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017.

Zhenjun Tang, Xianquan Zhang, and Shichao Zhang. Robust perceptual image hashing based on ring partition and nmf. IEEE transactions on knowledge and data engineering, 26(3):711–724, 2013.

Yu-Xiong Wang and Yu-Jin Zhang. Nonnegative matrix factorization: A comprehensive review. IEEE Transactions on knowledge and data engineering, 25(6):1336–1353, 2012.

Kazuyoshi Yoshii, Ryota Tomioka, Daichi Mochihashi, and Masataka Goto. Beyond nmf: Time-domain audio source separation without phase reconstruction. In ISMIR, pp. 369–374. Citeseer, 2013.

Junying Zhang, Le Wei, Qiguang Miao, and Yue Wang. Image fusion based on nonnegative matrix factorization. In 2004 International Conference on Image Processing, 2004. ICIP'04., volume 2, pp. 973–976. IEEE, 2004.

Li Zhang, Weijun Zhou, and Donghui Li. Global convergence of a modified fletcher–reeves conjugate gradient method with armijo-type line search. Numerische mathematik, 104 (4):561–572, 2006.

Lihong Zhao, Guibin Zhuang, and Xinhe Xu. Facial expression recognition based on pca and nmf. In 2008 7th World Congress on Intelligent Control and Automation, pp. 6826–6829. IEEE, 2008.

A Appendix A - Theoretical bounds on optimal LR schedules

First, we show that F is convex in W or H alone. Since F is symmetric and the order of updates don't matter, we can show that

Lemma A.1. $F(W^t, H)$ is convex

Proof. For three matrices H_1, H_2 and $H_\lambda \in \mathbb{R}_+^{r\times n}$, and a positive number $\lambda \in (0,1)$, let:

$$H_\lambda = \lambda H_1 + (1-\lambda)H_2$$

$$\begin{aligned}
F(W^t, H_\lambda) &= F(W^t, \lambda H_1 + (1-\lambda)H_2) \\
&= \frac{1}{2}\|V - W^t(\lambda H_1 + (1-\lambda)H_2)\|_F^2 \\
&= \frac{1}{2}\operatorname{tr}\left(V - W^t(\lambda H_1 + (1-\lambda)H_2)\right)^\top \left(V - W^t(\lambda H_1 + (1-\lambda)H_2)\right) \\
&= \frac{1}{2}\sum_{ij}\left(V_{ij} - W^t_{ij}\left(\lambda H_{1,ij} + (1-\lambda)H_{2,ij}\right)\right)^2 \\
&= \frac{1}{2}\sum_{ij}\left(V_{ij} - \lambda W^t_{ij}H_{1,ij} - (1-\lambda)W^t_{ij}H_{2,ij}\right)^2 \\
&= \frac{1}{2}\sum_{ij}\left(V_{ij} + \lambda V_{ij} - \lambda W^t_{ij}H_{1,ij} - (1-\lambda)W^t_{ij}H_{2,ij} - \lambda V_{ij}\right)^2 \text{(Adding and subtracting } \lambda V_{ij}) \\
&= \frac{1}{2}\sum_{ij}\left(\lambda\left(V_{ij} - W^tH_{1,ij}\right) + (1-\lambda)\left(V_{ij} - W^tH_{2,ij}\right)\right)^2 \\
&\leq \frac{\lambda}{2}\sum_{ij}\left(V_{ij} - W^tH_{1,ij}\right)^2 + \frac{(1-\lambda)}{2}\sum_{ij}\left(V_{ij} - W^tH_{2,ij}\right)^2 \\
&\leq \lambda \cdot F(W^t, H_1) + (1-\lambda)\cdot F(W^t, H_2)
\end{aligned}$$

Therefore, F is convex when updating W^t. Similarly we can prove that it is convex when updating H^t

□

Lemma A.2. The gradient of $F(W^t, H)$ is Lipschitz continuous

Proof. We know that the gradient of F

$$\nabla F(W^t, H) = (W^t)^\top(W^tH - V)$$

Therefore,

$$\begin{aligned}
\|\nabla F(W^t, H) - \nabla F(W^{t+1}, H)\| &= \|(W^t)^\top W^t(W^t - W_{t+1})\| \\
&\leq L\|(W^t - W_{t+1})\|
\end{aligned}$$

where $L = \|(W^t)^\top W^t\|$

□

Assumption A.1. In the stochastic setting, assume that we only have access to noisy gradient estimates and not the actual gradient, and that $E[\nabla F_i(W^t, H)] = \nabla F(W^t, H)\ \forall$ samples i

Assumption A.2. Assume that the simplest form of Stochastic Gradient Descent (SGD) is used to find subsequent iterates of W and H. Focusing on W, we have the following update formula:

$$W^{t+1} = W^t - \alpha^t \cdot \nabla F(W^t, H)$$

where t is the iteration counter.

Typically, step size selection is through heuristics or experimentation. The most common way to select step size is to use a constant throughout the training process, i.e.:

$$\alpha^t = \alpha_0$$

For NMF problems, another typical way to select the step size is through Armijo line search. At each iteration t we aim to find an α^t that satisfies:

$$F(W^{t+1}, H) \leq F(W^t, H) - c \cdot \alpha^t \|\nabla F(W^t, H)\|^2 \tag{1}$$

That is, at each iteration t, we start with a large α_{max} and successively try step sizes $\alpha_{max} > \beta\alpha_{max} > \beta^2\alpha_{max} \ldots$ until we find the largest α that satisfies the above equation. Additionally, by definition, 1 satisfies the Polyak-Lojasiewicz (PL) inequality at convergence, where for $\mu > 0$ and function F:

$$\frac{1}{2}\|\nabla F(W^t, H)\|^2 \geq \mu(F(W^t, H) - F^*)$$

From 1, we can expect a sequence of μ^t where $\mu^t = \frac{1}{2c\alpha^t}$. We can therefore expect that F decreases towards F^* with the appropriate selection of α. This leads us to Theorem 1:

Theorem A.3. For SGD to converge for the NMF problem, we select α^t such that $0 < \alpha^t < \frac{2}{L}$ where $L = \|W^t\|^2$

Proof. From Lemma 1.1 and 1.2, we know that F is convex and ∇F is Lipschitz continuous.

Dropping H notation in F, and using the Lipschitz condition and SGD update rule, we have:

$$\begin{aligned}
F(W^{t+1}) &\leq F(W^t) + \left(\nabla F(W^t)\right)^\top (W^{t+1} - W^t) + \frac{L}{2}\|W^{t+1} - W^t\|^2 \\
&\leq F(W^t) + \left(\nabla F(W^t)\right)^\top (-\alpha^t \nabla F(W^t)) + \frac{L}{2}\|-\alpha^t \nabla F(W^t)\|^2 \\
&\leq F(W^t) - \|F(W^t)\|^2 \left(\alpha^t - {\alpha^t}^2 \frac{L}{2}\right) \\
\implies F(W^t) &\geq F(W^{t+1}) + \|F(W^t)\|^2 \left(\alpha^t - {\alpha^t}^2 \frac{L}{2}\right)
\end{aligned}$$

For F to decrease towards F^*, we need:

$$\begin{aligned}
&\alpha^t - {\alpha^t}^2 \frac{L}{2} > 0 \\
\implies &\alpha^t(2 - \alpha^t L) > 0 \\
\implies &0 < \alpha^t < \frac{2}{L}
\end{aligned}$$

With $L = \|(W^t)^\top W^t\|$ □

Lastly, when using Armijo line search in 1, we can see that :

$$
\begin{aligned}
c\alpha^t &\leq \alpha^t - {\alpha^t}^2 \frac{L}{2} \\
\implies c &\leq 1 - \alpha^t \frac{L}{2} \\
\implies \alpha^t &\leq \frac{2(1-c)}{L}
\end{aligned}
$$

With $0 < \alpha^t < \frac{2}{L}$, we have:

$$0 < c < 1 \tag{2}$$

B Appendix B - Loss curves for LR schedules

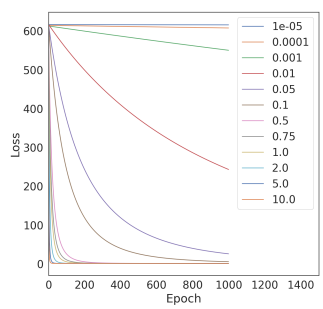
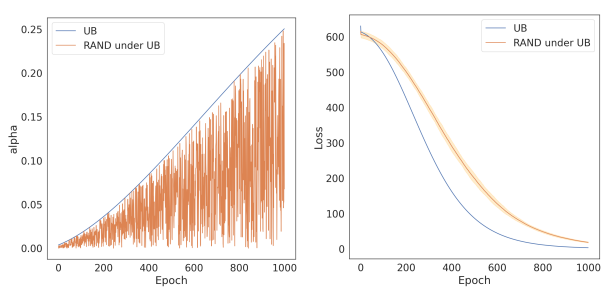
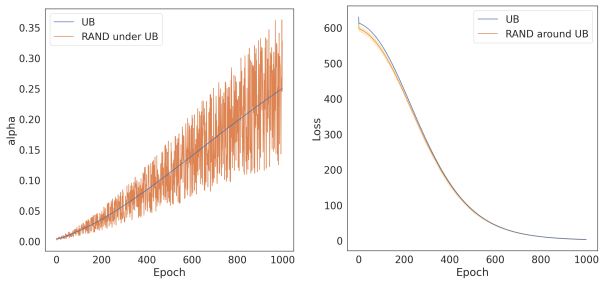


Figure 2: Loss histories corresponding to fixed LRs for NMF with SGD



(a) (b)

Figure 3: a) Example random walk LR schedule under the upper bound LR b) Mean loss curve with variance corresponding to 25 experiment trials, and compared to the the upper bound loss curve



(a) (b)

Figure 4: a) Example random walk LR schedule around the upper bound LR, where magnitude is $+/-50\%$ of the value on the upper bound curve b) Mean loss curve with variance corresponding to 25 experiment trials, and compared to the the upper bound loss curve

C Appendix B - Loss curves for LR schedules

For the NMF problem, we set the reward as the sum of

1. percentage decrease over the best loss so far
2. epochs remaining

The agent is therefore incentivized to both reduce over the best loss so far as well as converge faster. We see that the policy converges to several local optima based on the number of RL iterations. We set the episode to end when a divergence is detected, or if loss is <10; this is a a low enough value after which no LR changes affect the low converge rate close to an optima. This is done since we observe that even arbitrarily high LRs slowly converge over 1000 epochs beyond a point. Below we include loss curves and action histories of several RL experiments based on different number of RL learning iterations and RL agent Learning rates. Note that in all cases below, the PPO agent has its own LR that is fixed to some value; the action history is the LR schedule that is applied to the underlying learning problem, here - NMF.

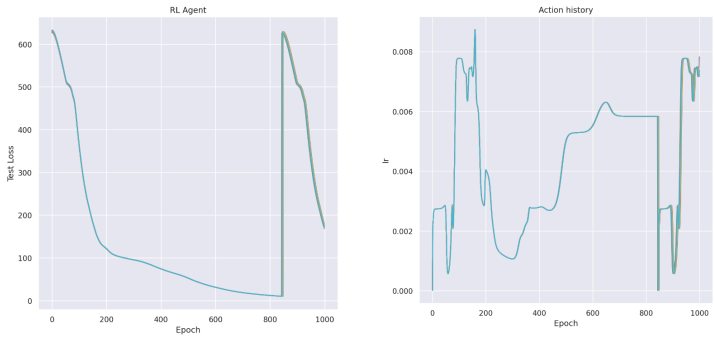


Figure 5: RL agent loss curve and action history for a PPO agent LR of 0.01 and 10K iterations. We see an interesting action history that is not a clean parametric curve, and even though the magnitude of the action is low overall, convergence rate is higher than much higher fixed LR schedules. The model converges once, resets the environment (reinitializes the problem) and starts again at around 850 epochs

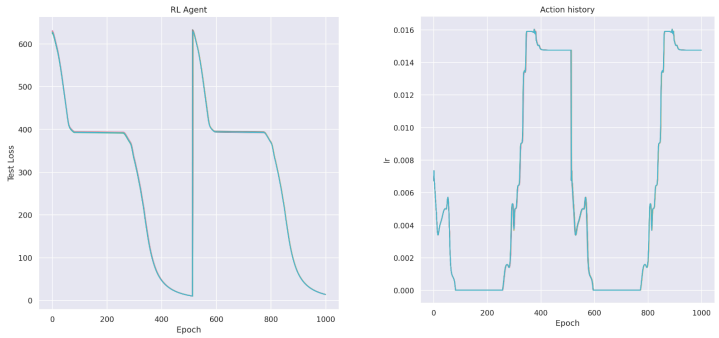


Figure 6: RL agent loss curve and action history for a PPO agent LR of 0.01 and 20K iterations. The agent converges twice and applies the same learned schedule

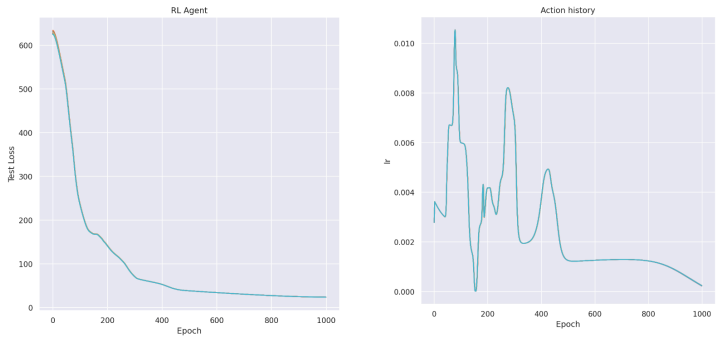


Figure 7: RL agent loss curve and action history for a PPO agent LR of 0.01 and 50K iterations. Convergence rate is lower, but looks like standard loss curves

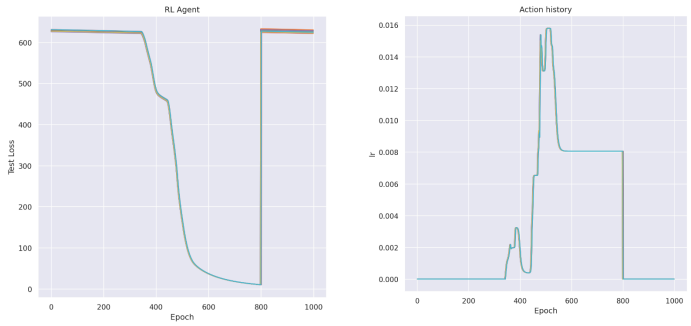


Figure 8: RL agent loss curve and action history for a PPO agent LR of 0.01 and 100K iterations. It is not clear why the agent converged to a policy where the first 500 epochs includes a fixed, lower bound LR

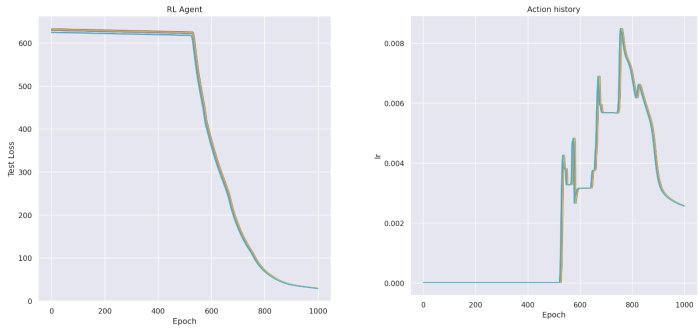


Figure 9: RL agent loss curve and action history for a PPO agent with higher LR of 0.1 and 100K iterations. Higher RL learning rate and higher iterations may still lead to a sub-optimal schedules

What we conclude from these experiments is that:

1. A learned schedule may work better than any predetermined schedule even though the mean value of the LRs applied is lower. This is interesting since most proofs have the highest convergence rate at the highest applied LR
2. The agent may find several local optima based on the reward function used. So reward shaping can find better, more usable schedules