OBSERVATIONAL OVERFITTING IN REINFORCEMENT LEARNING

Xingyou Song[*], **Yiding Jiang**[†], **Stephen Tu**, **Behnam Neyshabur**
Google
{xingyousong,ydjiang,stephentu,neyshabur}@google.com

Yilun Du[*]
MIT
yilundu@mit.edu

ABSTRACT

A major component of overfitting in model-free reinforcement learning (RL) involves the case where the agent may mistakenly correlate reward with certain spurious features from the observations generated by the Markov Decision Process (MDP). We provide a general framework for analyzing this scenario, which we use to design multiple synthetic benchmarks from only modifying the observation space of an MDP. When an agent overfits to different observation spaces even if the underlying MDP dynamics is fixed, we term this *observational overfitting*. Our experiments expose intriguing properties especially with regards to *implicit regularization*, and also corroborate results from previous works in RL generalization and supervised learning (SL).

1 INTRODUCTION

Generalization for RL has recently grown to be an important topic for agents to perform well in unseen environments. Complication arises when the dynamics of the environments entangle with the observation, which is often a high-dimensional projection of the true latent state. One particular framework, which we denote by *zero-shot supervised framework* (Zhang et al., 2018a;c; Nichol et al., 2018; Justesen et al., 2018) and is used to study RL generalization, is to treat it analogous to a classical supervised learning (SL) problem – i.e. assume there exists a distribution of MDP's, train jointly on a finite "training set" sampled from this distribution, and check expected performance on the entire distribution, with the fixed trained policy. In this framework, there is a spectrum of analysis, ranging from almost purely theoretical analysis (Wang et al., 2019; Asadi et al., 2018) to full empirical results on diverse environments (Zhang et al., 2018c; Packer et al., 2018).

However, there is a lack of analysis in the middle of this spectrum. On the theoretical side, previous work do not provide analysis for the case when the underlying MDP is relatively complex and requires the policy to be a non-linear function approximator such as a neural network. On the empirical side, there is no common ground between recently proposed empirical benchmarks. This is partially caused by multiple confounding factors for RL generalization that can be hard to identify and separate. For instance, an agent can overfit to the MDP dynamics of the training set, such as for control in Mujoco (Pinto et al., 2017; Rajeswaran et al., 2017). In other cases, an RNN-based policy can overfit to maze-like tasks in exploration (Zhang et al., 2018c), or even exploit determinism and avoid using observations (Bellemare et al., 2012; Machado et al., 2018). Furthermore, various hyperparameters such as the batch-size in SGD (Smith et al., 2018), choice of optimizer (Kingma & Ba, 2014), discount factor γ (Jiang et al., 2015) and regularizations such as entropy (Ahmed et al., 2018) and weight norms (Cobbe et al., 2018) can also affect generalization.

[*] Work partially performed as an OpenAI Fellow.

[†] Work performed during the Google AI Residency Program. http://g.co/airesidency

Due to these confounding factors, it can be unclear what parts of the MDP or policy are actually contributing to overfitting or generalization in a principled manner, especially in empirical studies with newly proposed benchmarks. In order to isolate these factors, we study one broad factor affecting generalization that is most correlated with themes in SL, specifically *observational overfitting*, where an agent overfits due to properties of the observation which are irrelevant to the latent dynamics of the MDP family. To study this factor, we fix a single underlying MDP's dynamics and generate a distribution of MDP's by only modifying the observational outputs.

Our contributions in this paper are the following:

1. We discuss realistic instances where observational overfitting may occur and its difference from other confounding factors, and design a parametric theoretical framework to induce observational overfitting that can be applied to *any* underlying MDP.
2. We study observational overfitting with linear quadratic regulators (LQR) in a synthetic environment and neural networks such as multi-layer perceptrons (MLPs) and convolutions in classic Gym environments. A primary novel result we demonstrate for all cases is that *implicit regularization* occurs in this setting in RL. We further test the implicit regularization hypothesis on the benchmark CoinRun from using MLPs, even when the underlying MDP dynamics are changing per level.
3. In the Appendix, we expand upon previous experiments by including full training curve and hyperparamters. We also provide an extensive analysis of the convex one-step LQR case under the observational overfitting regime, showing that under Gaussian initialization of the policy and using gradient descent on the training cost, a generalization gap must necessarily exist.

The structure of this paper is outlined as follows: Section 2 discusses the motivation behind this work and the synthetic construction to abstract certain observation effects. Section 3 demonstrates numerous experiments using this synthetic construction that suggest implicit regularization is at work. Finally, Section 3.4 tests the implicit regularization hypothesis on CoinRun, as well as ablates various ImageNet architectures and margin metrics in the Appendix.

2 Motivation and Related Work

We start by showing an example of observational overfitting in Figure 1. This example highlights the issues surrounding MDP's with rich, textured observations - specifically, the agent can use any features that are correlated with progress, even those which may not generalize across levels. This is an important issue for vision-based policies, as many times it is not obvious what part of the observation causes an agent to act or generalize.

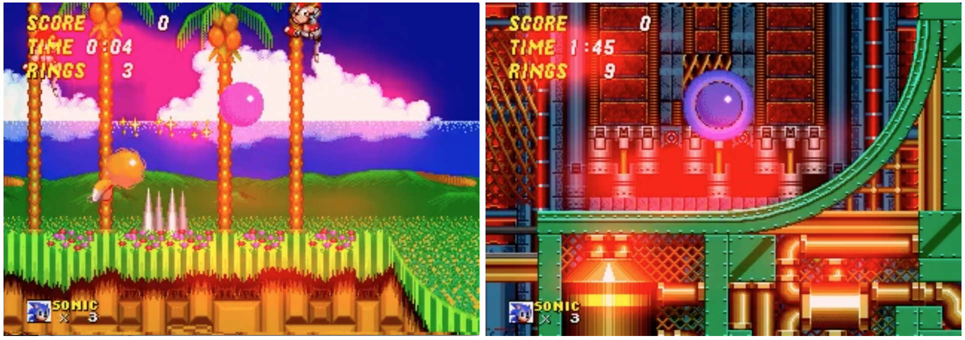


Figure 1: Example of observational overfitting in Sonic from Gym Retro (Nichol et al., 2018). Saliency maps (Greydanus et al., 2018) highlight (in red) the top-left timer and background objects such as clouds and textures because they are correlated with progress, as they move backwards while agent is moving forwards. The agent could memorize optimal actions for training levels even if its observation was *only from the timer*, and "blacking-out" the timer consistently improved generalization performance (see Appendix A.2.3).

Currently most architectures used in model-free RL are simple (with fewer than one million parameters) compared to the much larger and more complex ImageNet architectures used for classification.

This is due to the fact that most RL environments studied either have relatively simple and highly structured images (e.g. Atari) compared to real world images, or conveniently do not directly force the agent to observe highly detailed images. For instance in large scale RL such as DOTA2 (OpenAI, 2018) or Starcraft 2 (Vinyals et al., 2017), the agent observations are internal minimaps pertaining to object xy-locations, rather than human-rendered observations.

2.1 What Happens in Observation Space?

Several artificial benchmarks (Zhang et al., 2018b; Gamrian & Goldberg, 2019) have been proposed before to portray this notion of overfitting, where an agent must deal with a changing background - however, a key difference in our work is that we explicitly require the "background" to be **correlated with the progress** rather than loosely correlated (e.g. through determinism between the background and the game avatar) or not at all. This makes a more explicit connection to *causal inference* (Arjovsky et al., 2019; Heinze-Deml & Meinshausen, 2019; Heinze-Deml et al., 2019) where *spurious correlations* between ungeneralizable features and progress may make training easy, but are detrimental to test performance because they induce false attributions.

Previously, many works interpret the decision-making of an agent through saliency and other network visualizations (Greydanus et al., 2018; Such et al., 2018) on common benchmarks such as Atari. Other recent works such as (Igl et al., 2019) analyze the interactions between noise-injecting explicit regularizations and the information bottleneck. However, our work is motivated by learning theoretic frameworks to capture this phenomena, as there is vast literature on understanding the generalization properties of SL classifiers (Vapnik & Chervonenkis, 1971; McAllester, 1999; Bartlett & Mendelson, 2002) and in particular neural networks (Neyshabur et al., 2015b; Dziugaite & Roy, 2017; Neyshabur et al., 2017; Bartlett et al., 2017; Arora et al., 2018c). For an RL policy with high-dimensional observations, we hypothesize its overfitting can come from more theoretically principled reasons, as opposed to purely good inductive biases on game images.

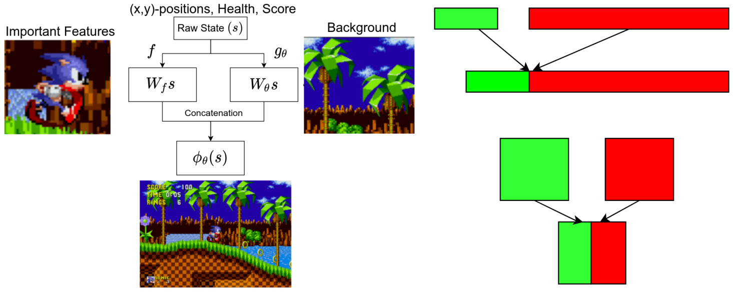
As an example of what may happen in high dimensional observation space, consider linear least squares regression task where given the set $X \in \mathbb{R}^{m \times d}$ and $Y \in \mathbb{R}^m$, we want to find $w \in \mathbb{R}^d$ that minimizes $\ell_{X,Y}(w) = \|Y - Xw\|^2$ where m is the number of samples and d is the input dimension. We know that if $X^\top X$ is full rank (hence $d \leq m$), $\ell_{X,Y}(.)$ has a unique global minimum $w^* = (X^\top X)^{-1} X^\top Y$. On the other hand if $X^\top X$ is not full rank (eg. when $m < d$), then there are many global minima w^* such that $Y = Xw^*$ [1]. Luckily, if we use any gradient based optimization to minimize the loss and initialize with $w = 0$, the solution will only span column spaces of X and converges to minimum ℓ_2 norm solution among all global minima due to implicit regularization (Gunasekar et al., 2017). Thus a high dimensional observation space with a low dimensional state space can induce multiple solutions, some of which are not generalizable to other functions or MDP's but one could hope that implicit regularization would help avoiding this issue. We analyze this case in further detail for the convex one-step LQR case in Section 3.1 and Appendix A.4.3.

2.2 Notation

In the zero-shot framework for RL generalization, we assume there exists a distribution $\mathcal{D}$ over MDP's $\mathcal{M}$ for which there exists a fixed policy π^{opt} that can achieve maximal return on expectation over MDP's generated from the distribution. An appropriate finite training set $\widehat{\mathcal{M}}_{train} = \{\mathcal{M}_1, \dots, \mathcal{M}_n\}$ can then be created by repeatedly randomly sampling $\mathcal{M} \sim \mathcal{D}$. Thus for a MDP $\mathcal{M}$ and any policy π, expected episodic reward is defined as $R_{\mathcal{M}}(\pi)$.

In many empirical cases, the support of the distribution $\mathcal{D}$ is made by parametrized MDP's where some process, given a parameter θ, creates a mapping $\theta \rightarrow \mathcal{M}_\theta$ (e.g. through procedural generation), and thus we may simplify notation and instead define a distribution Θ that induces $\mathcal{D}$, which implies a set of samples $\widehat{\Theta}_{train} = \{\theta_1, \dots, \theta_n\}$ also induces a $\widehat{\mathcal{M}}_{train} = \{\mathcal{M}_1, \dots, \mathcal{M}_n\}$, and we may redefine reward as $R_{\mathcal{M}_\theta}(\pi) = R_\theta(\pi)$.

[1] Given any X with full rank $X^\top X$, it is possible to create many global minima by projecting the data onto high dimensions using a semi-orthogonal matrix $Z \in \mathbb{R}^{d \times d'}$ where $d' > m \geq d$ and $ZZ^\top = I_d$. Therefore, we the loss $\ell_{XZ,Y}(w) = \|Y - XZw\|^2$ will have many global optima w^* with $Y = XZw^*$.



(a)

(b)

Figure 2: (a) Visual Analogy of the Observation Function. (b) Our combinations for 1-D (top) and 2-D (bottom) images for synthetic tasks.

As a simplified model of the observational problem from Sonic, we can construct a mapping $\theta \rightarrow \mathcal{M}_\theta$ by first fixing a base MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, \mathcal{T})$, which corresponds to state space, action space, reward, and transition. The only effect of θ is to introduce an additional *observation function* $\phi_\theta : \mathcal{S} \rightarrow \mathcal{O}$, where the agent receives input from the high dimensional observation space $\mathcal{O}$ rather than from the state space $\mathcal{S}$. Thus, for our setting, θ actually parameterizes a POMDP family which can be thought of as simply a combination of a base MDP $\mathcal{M}$ and an observational function ϕ_θ, hence $\mathcal{M}_\theta = (\mathcal{M}, \phi_\theta)$.

Let $\widehat{\Theta}_{train} = \{\theta_1, \dots, \theta_n\}$ be a set of n i.i.d. samples from Θ, and suppose we train π to optimize reward against $\{\mathcal{M}_\theta : \theta \sim \widehat{\Theta}_{train}\}$. The objective $J_{\widehat{\Theta}}(\pi) = \frac{1}{|\widehat{\Theta}_{train}|} \sum_{\theta_i \in \widehat{\Theta}_{train}} R_{\theta_i}(\pi)$ is the average reward over this empirical sample. We want to generalize to the distribution Θ, which can be expressed as the average episode reward R over the full distribution, i.e. $J_\Theta(\pi) = \mathbb{E}_{\theta \sim \Theta}\left[R_\theta(\pi)\right]$. Thus we define the generalization gap as $J_{\widehat{\Theta}}(\pi) - J_\Theta(\pi)$.

2.3 Setup

We can model the effects of Figure 1 more generally, not specific to sidescroller games. We assume that there is an underlying *state* s (e.g. xy-locations of objects in a game), whose features may be very well structured, but that this state has been projected to a high dimensional observation space by ϕ_θ. To abstract the notion of generalizable and non-generalizable features, we construct a simple and natural candidate class of functions, where

$$\phi_\theta(s) = h(f(s), g_\theta(s)) \tag{1}$$

In this setup, $f(\cdot)$ is a function invariant for the entire MDP population Θ, while $g_\theta(\cdot)$ is a function dependent on the sampled parameter θ. h is a "combination" function which combines the two outputs of f and g to produce a final observation. While f projects this latent data into salient and important, *invariant* features such as the avatar, monsters, and items, g_θ projects the latent data to unimportant features that do not contribute to extra generalizable information, and can cause overfitting, such as the changing background or textures. A visual representation is shown in Figure 2. This is a simplified but still insightful model relevant in more realistic settings. For instance, in settings where g_θ does matter, learning this separation and task-identification (Yu et al., 2017; Peng et al., 2018) could potentially help fast adaptation in meta-learning (Finn et al., 2017). From now on, we denote this setup as the (f, g)-*scheme*.

This setting also leads to more interpretable generalization bounds - Lemma 2 of (Wang et al., 2019) provides a high probability $(1 - \delta)$ bound for the "intrinsic" generalization gap when m levels are sampled: $gap \leq Rad_m(R_\Pi) + \mathcal{O}\left(\sqrt{\frac{\log(1/\delta)}{m}}\right)$, where

$$Rad_m(R_\Pi) = \mathbb{E}_{(\theta_1,\dots,\theta_m)\sim\Theta^m}\left[\mathbb{E}_{\sigma\in\{-1,+1\}}\left[\sup_{\pi\in\Pi}\frac{1}{m}\sum_{i=1}^{m}\sigma_i R_{\theta_i}(\pi)\right]\right] \tag{2}$$

is the Rademacher Complexity under the MDP, where θ_i are the ζ_i parameters used in the original work, and the transition $\mathcal{T}$ and initialization $\mathcal{I}$ are fixed, therefore omitted, to accommodate our setting.

The Rademacher Complexity term captures how invariant policies in the set Π with respect to θ. For most RL benchmarks, this is not interpretable due to multiple confounding factors such as the varying level dynamics. For instance, it is difficult to imagine what behaviors or network weights a policy would possess in order to produce the same total rewards, *regardless of changing dynamics.*

However, in our case, because the environment parameters θ are only from g_θ, the Rademacher Complexity is directly based on how much the policy "looks at" g_θ. More formally, let Π^* be the set of policies π^* which are not be affected by changes in g_θ; i.e. $\nabla_\theta \pi^*(\phi_\theta(s)) = 0 \;\; \forall s$ and thus $R_\theta(\pi^*) = R_{const} \;\; \forall \theta$, which implies that the environment parameter θ has no effect on the reward; hence $Rad_m(R_{\Pi^*}) = \mathbb{E}_{\sigma \in \{-1,+1\}} \left[\sup_{\pi^* \in \Pi^*} \frac{1}{m} \sum_{i=1}^{m} \sigma_i R_{const} \right] = 0$.

2.4 Architecture and Implicit Regularization

Normally in a MDP such as a game, the concatenation operation may be dependent on time (e.g. textures move around in the frame). In the scope of this work, we simplify the concatenation effect and assume $h(\cdot)$ is a static concatenation, but still are able to demonstrate insightful properties. We note that this inductive bias on h allows *explicit regularization* to trivially solve this problem, by penalizing a policy's first layer that is used to "view" $g_\theta(s)$ (Appendix A.1.1), hence we only focus on implicit regularizations.

This setting is naturally attractive to analyzing architectural differences, as it is more closely related in spirit to image classifiers and SL. One particular line of work to explain the effects of certain architectural modifications in SL such as overparametrization and residual connections is *implicit regularization* (Neyshabur et al., 2015a; Gunasekar et al., 2017; Neyshabur, 2017), as overparametrization through more layer depth and wider layers has proven to have no ℓ_p-regularization equivalent (Arora et al., 2019), but rather precondition the dynamics during training. Thus, in order to fairly experimentally measure this effect, we always use fixed hyperparameters and only vary based on architecture. In this work, we only refer to *architectural* implicit regularization techniques, which do not have a explicit regularization equivalent. Some techniques e.g. coordinate descent (Bradley et al., 2011) are equivalent to explicit ℓ_1-regularization.

3 Experiments

3.1 Overparamterized LQR

We first analyze the case of the LQR as a surrogate for what may occur in deep RL, which has been done before for various topics such as sample complexity (Dean et al., 2019) and model-based RL (Tu & Recht, 2019). This is analogous to analyzing linear/logistic regression (Kakade et al., 2008; McAllester, 2003) as a surrogate to understanding extensions to deep SL techniques (Neyshabur et al., 2018a; Bartlett et al., 2017). In particular, this has numerous benefits - the cost (negative of reward) function is deterministic, and allows exact gradient descent (i.e. the policy can differentiate through the cost function) as opposed to necessarily using stochastic gradients in normal RL, and thus can cleanly provide evidence of implicit regularization. Furthermore, in terms of gradient dynamics and optimization, LQR readily possesses nontrivial qualities compared to linear regression, as the LQR cost is a non-convex function but all of its minima are *global minima* (Fazel et al., 2018).

To show that overparametrization alone is an important implicit regularizer in RL, LQR allows the use of linear policies (and consequently also allows stacking linear layers) without requiring a stochastic output such as discrete Gumbel-softmax or for the continuous case, a parametrized Gaussian. This is setting able to show that overparametrization alone can affect gradient dynamics, and is not a consequence of extra representation power due to additional non-linearities in the policy. There have been multiple recent works on this linear-layer stacking in SL and other theoretical problems such as matrix factorization and matrix completion (Arora et al., 2018b;a; Gunasekar et al., 2017), but to our knowledge, we are the first to analyze this case in the context of RL generalization.

We explicitly describe setup as follows: for a given θ, we let $f(s) = W_c \cdot s$, while $g_\theta(s) = W_\theta \cdot s$ where W_c, W_θ are semi-orthogonal matrices, to prevent information loss relevant to outputting the optimal action, as the state is transformed into the observation. Hence, if s_t is the underlying state at time t, then the observation is $o_t = \begin{bmatrix} W_c \\ W_\theta \end{bmatrix} s_t$ and thus the action is $a_t = K o_t$, where K is the policy matrix. While W_c remains a constant matrix, we sample W_θ randomly, using the "level ID" integer θ as the seed for random generation. In terms of dimensions, if s is of shape d_{state}, then f also projects to a shape of d_{state}, while g_θ projects to a much larger shape d_{noise}, implying that the observation to the agent is of dimension $d_{signal} + d_{noise}$. In our experiments, we set as default $(d_{signal}, d_{noise}) = (100, 1000)$.

If $P_\star$ is the unique minimizer of the original cost function, then the unique minimizer of the population cost is $K_\star = \begin{bmatrix} W_c P_\star^\intercal \\ 0 \end{bmatrix}^\intercal$. However, if we have a single level, then there exist multiple solutions, for instance $\begin{bmatrix} \alpha W_c P_\star^\intercal \\ (1-\alpha) W_\theta P_\star^\intercal \end{bmatrix}^\intercal$ $\forall \alpha$. This extra bottom component $W_\theta P_\star^\intercal$ causes overfitting. In Appendix A.4.3, we show that in the 1-step LQR case (which can be extended to convex losses whose gradients are linear in the input), gradient descent cannot remove this component, and thus overfitting necessarily occurs.

Furthermore, we find that increasing d_{noise} increases the generalization gap in the LQR setting. This is empirically verified in Figure 3 using an actual non-convex LQR loss, and the results suggest that the gap scales by $\mathcal{O}(\sqrt{d_{noise}})$. In terms of overparametrization, we experimentally added more (100×100) linear layers $K = K_0 K_1, ..., K_j$ and increased widths for a 2-layer case (Figure 3), and observe that both settings reduce the generalization gap, and also reduce the norms (spectral, nuclear, Frobenius) of the final end-to-end policy K, without changing its expressiveness. This suggests that gradient descent under overparametrization implicitly biases the policy towards a "simpler" model in the LQR case.

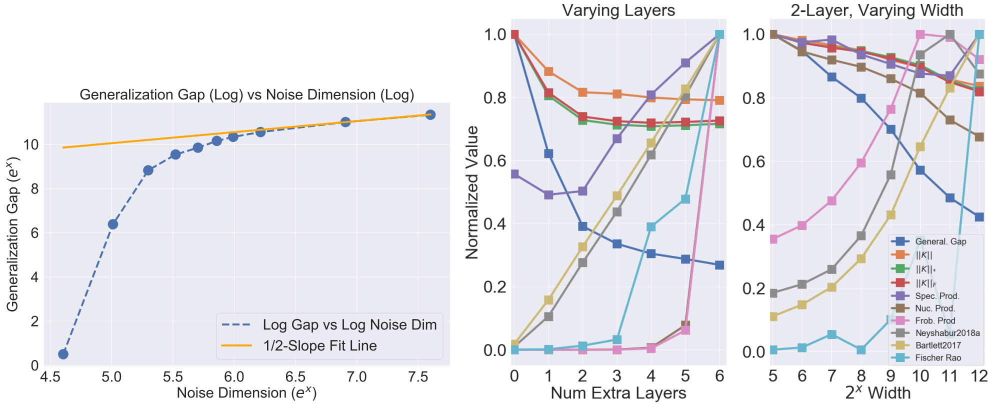


Figure 3: (**Left**) We show that the generalization gap vs noise dimension is tight as the noise dimension increases, showing that this bound is accurate. (**Middle** and **Right**) LQR Generalization Gap vs Number of Intermediate Layers. We plotted different $\Phi = \sum_{i=0}^{j} \frac{\|A\|_*}{\|A\|}$ terms without exponents, as powers of those terms are monotonic transforms since $\frac{\|A\|_*}{\|A\|} \geq 1 \ \forall A$ and $\|A\|_* = \|A\|_F, \|A\|_1$. We see that the naive spectral bound diverges at 2 layers, and the weight-counting sums are too loose.

As a surrogate model for deep RL, one may ask if the generalization gap of the final end-to-end policy K can be predicted by functions of the layers $K_0, ..., K_j$. This is an important question as it is a required base case for predicting generalization when using stochastic policy gradient with nonlinear activations such as ReLU or Tanh. From examining the distribution of singular values on K (Appendix A.1.1), we find that more layers does not bias the policy towards a low rank solution in the nonconvex LQR case, unlike (Arora et al., 2018b) which shows this does occur for matrix completion, and in general, convex losses. Ultimately, we answer in the negative: intriguingly, SL bounds have very little predictive power in the RL domain case.

To understand why SL bounds may be candidates for the LQR case, we note that as a basic smoothness bound $C(K) - C(K') \leq \mathcal{O}(\|K - K'\|)$ (Appendix A.4) can lead to very similar reasoning with SL bounds. Since our setup is similar to SL in that "LQR levels" which may be interpreted as a dataset, we use bounds of the form $\Delta \cdot \Phi$, where Δ is a "macro" product term $\Delta = \prod_{i=0}^{j} \|K_i\| \geq \left\|\prod_{i=0}^{j} K_i\right\|$ derivable from the fact that $\|AB\| \leq \|A\| \, \|B\|$ in the linear case, and Φ is a *weight-counting* term which deals with the overparametrized case, such as $\Phi = \sum_{i=0}^{j} \frac{\|K_i\|_F^2}{\|K_i\|^2}$ (Neyshabur et al., 2018a) or $\Phi = \left(\sum_{i=0}^{j} \left(\frac{\|K_i\|_1}{\|K_i\|}\right)^{2/3}\right)^3$ (Bartlett et al., 2017). However, the Φ terms increase too rapidly as shown in Figure 3. Terms such as Frobenius product (Golowich et al., 2018) and Fischer-Rao (Liang et al., 2019) are effective for the SL depth case, but are both ineffective in the LQR depth case. For width, the only product which is effective is the nuclear norm product.

3.2 Projected Gym Environments

In Section 3.1, we find that observational overfitting exists and overparametrization potentially helps in the linear setting. In order to analyze the case when the underlying dynamics are nonlinear, we let $\mathcal{M}$ be a classic Gym environment and we generate a $\mathcal{M}_\theta = (\mathcal{M}, w_\theta)$ by performing the exact same (f, g)-scheme as the LQR case, i.e. sampling θ to produce an observation function $w_\theta(s) = \begin{bmatrix} W_c \\ W_\theta \end{bmatrix} s$. We again can produce training/test sets of MDPs by repeatedly sampling θ, and for policy optimization, we use Proximal Policy Gradient (Schulman et al., 2017).

Although bounds on the smoothness term $R_\theta(\pi) - R_\theta(\pi')$ affects upper bounds on Rademacher Complexity (and thus generalization bounds), we have no such theoretical guarantees in the Mujoco case as it is difficult to analyze the smoothness term for complicated transitions such as Mujoco's physics simulator. However, in Figure 4, we can observe empirically that the underlying state dynamics has a significant effect on generalization performance as the policy nontrivially increased test performance such as in CartPole-v1 and Swimmer-v2, while it could not for others. This suggests that the Rademacher complexity and smoothness on the reward function vary highly for different environments.

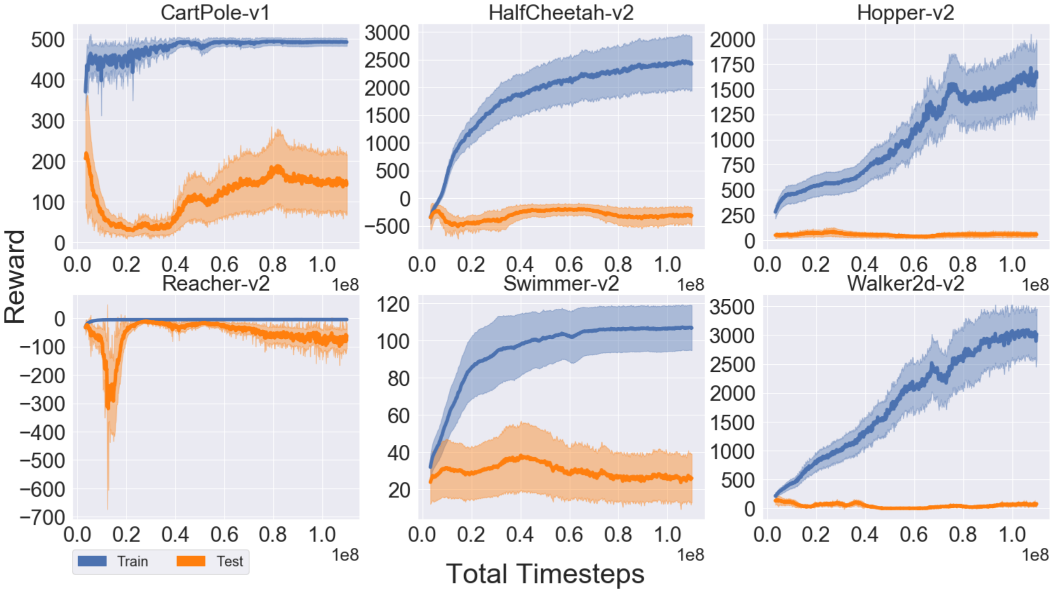


Figure 4: Each Mujoco task is given 10 training levels (randomly sampling g_θ parameters). We used a 2-layer ReLU policy, with 128 hidden units each. Dimensions of outputs of (f, g) were $(30, 100)$ respectively.

Even though it is common practice to use basic (2-layer) MLPs in these classic benchmarks, there are highly nontrivial generalization effects from modifying on this class of architectures. Our results in

Figures 5 and 6 show that increasing width and depth for basic MLPs can increase generalization and is significantly dependent on the choice of activation, and other implicit regularizations such as using residual layers can also improve generalization. Specifically, switching between ReLU and Tanh activations produces different results during overparametrization. For instance, increasing Tanh layers improves generalization on CartPole-v1, and width increase with ReLU helps on Swimmer-v2. Tanh is noted to consistently improve generalization performance. However, stacking Tanh layers comes at a cost of also producing vanishing gradients which can produce subpar training performance, for e.g. HalfCheetah. To allow larger depths, we use ReLU *residual* layers, which also improves generalization and stabilizes training.

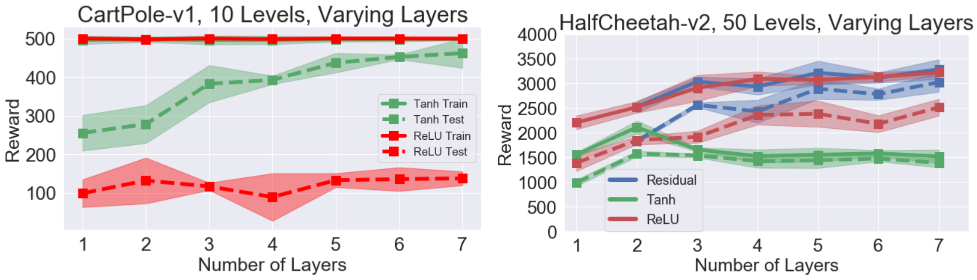


Figure 5: Effects of Depth.

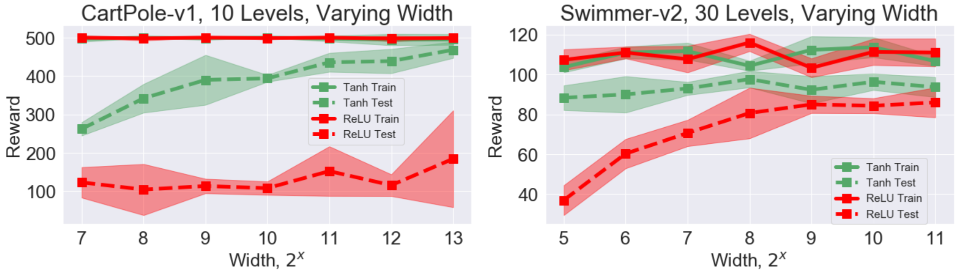


Figure 6: Effects of Width.

Previous work (Zhang et al., 2018c) did not find such an architectural pattern for GridWorld environments, suggesting that this effect may exist primarily for observational overfitting cases. While there have been numerous works which avoid overparametrization on simplifying policies (Rajeswaran et al., 2017; Mania et al., 2018) or compactifying networks (Choromanski et al., 2018; Gaier & Ha, 2019), we instead find that there are generalization benefits to overparametrization even in the nonlinear control case.

3.3 Deconvolutional Projections

From the above results with MLPs, one may wonder if similar results may carry to convolutional networks, as they are widely used for vision-based RL tasks. As a ground truth reference for our experiment, we the canonical networks proven to generalize well in the dataset CoinRun, which are from worst to best, NatureCNN Mnih et al. (2013), IMPALA Espeholt et al. (2018), and IMPALA-LARGE (IMPALA with more residual blocks and higher convolution depths), which have respective parameter numbers (600K, 622K, 823K).

We setup a similar (f, g)-scheme appropriate for the inductive bias of convolutions, by passing the vanilla Gym 1D state corresponding to joint locations and velocities, through multiple deconvolutions. We do so rather than using the RGB image from `env.render()` to enforce that the actual state is indeed low dimensional and minimize complications in experimentation, as e.g. inference of velocity information would require frame-stacking.

Specifically in our setup, we project the actual state to a fixed length, reshaping it into a square, and replacing f and g_θ both with the same orthogonally-initialized *deconvolution* architecture to each produce a 84×84 image (but g_θ's network weights are still generated by $\theta_1, ..., \theta_m$ similar to before). We combine the two outputs by using one half of the "image" from f, and one half from g_θ, as shown back in Figure 2.

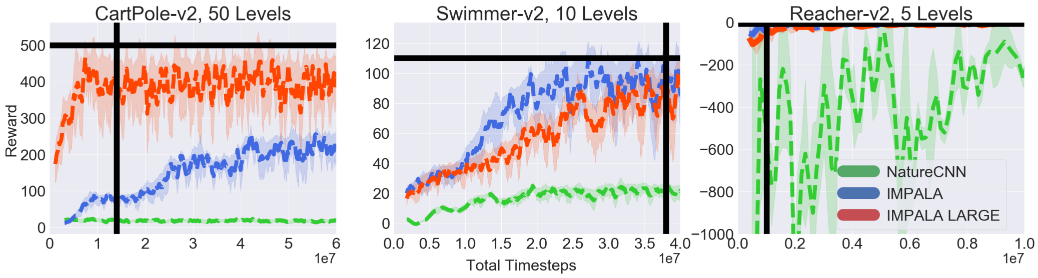


Figure 7: Performance of architectures in the synthetic Gym-Deconv dataset. To cleanly depict test performance, training curves are replaced with horizontal (max env. reward) and vertical black lines (avg. timestep when all networks reach max reward).

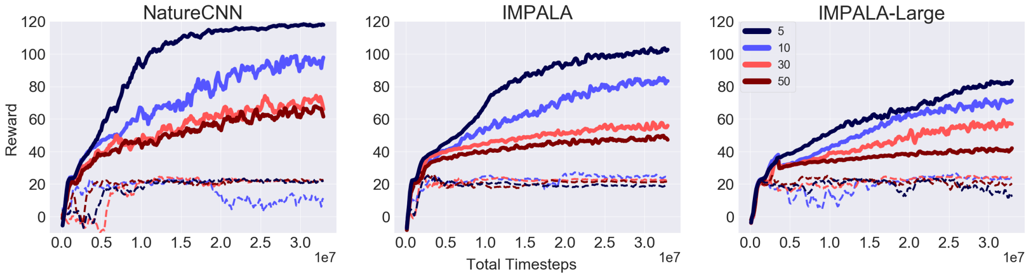


Figure 8: We only show the observation from $g_\theta(s)$, which tests memorization capacity on Swimmer-v2.

Figure 7 shows that the same ranking between the three architectures exists as well on the Gym-Deconv dataset. We show that generalization ranking among NatureCNN/IMPALA/IMPALA-LARGE remains the same regardless of whether we use our synthetic constructions or CoinRun. This suggests that the RL generalization quality of a convolutional architecture is not limited to *real world data*, as our test purely uses numeric observations, which are not based on a human-prior. From these findings, one may conjecture that these RL generalization performances are highly correlated and may be due to common factors.

One of these factors we suggest is due to implicit regularization. In order to support this claim, we perform a memorization test by only showing g_θ's output to the policy. This makes the dataset impossible to generalize to, as the policy network cannot invert every single observation function $\{g_{\theta_1}(\cdot), g_{\theta_2}(\cdot), ..., g_{\theta_n}(\cdot)\}$ simultaneously. Zhang et al. (2018c) also constructs a memorization test for mazes and grid-worlds, and showed that more parameters increased the memorization ability of the policy. While it is intuitive that more parameters would incur more memorization, we show in Figure 8 that this is perhaps not a complete picture when implicit regularization is involved.

Using the underlying MDP as a Swimmer-v2 environment, we see that NatureCNN, IMPALA, IMPALA-LARGE have reduced memorization performances. IMPALA-LARGE, which has more depth parameters and more residual layers (and thus technically has more capacity), memorizes *less* than IMPALA due its inherent inductive bias. While memorization performance is dampened in 8, we perform another deconvolution memorization test using an LQR as the underlying MDP in Appendix A.1.1 that shows that there can exist specific hard limits to memorization, which also follows the same ranking above.

3.4 Overparametrization in CoinRun

We further test our overparametrization hypothesis from Sections 3.1, 3.2 to the CoinRun benchmark, using unlimited levels for training. For MLP networks, we downsized CoinRun from native 64×64 to 32×32, and flattened the $32 \times 32 \times 3$ image for input to an MLP. Two significant differences from the synthetic cases are that 1. Inherent dynamics are changing per level in CoinRun, and 2. The relevant and irrelevant CoinRun features change locations across the 1-D input vector. Regardless, in Figure 9, we show that overparametrization can still improve generalization in this more realistic

RL benchmark, much akin to (Neyshabur et al., 2018b) which showed that overparametrization for MLP's improved generalization on $32 \times 32 \times 3$ CIFAR-10.

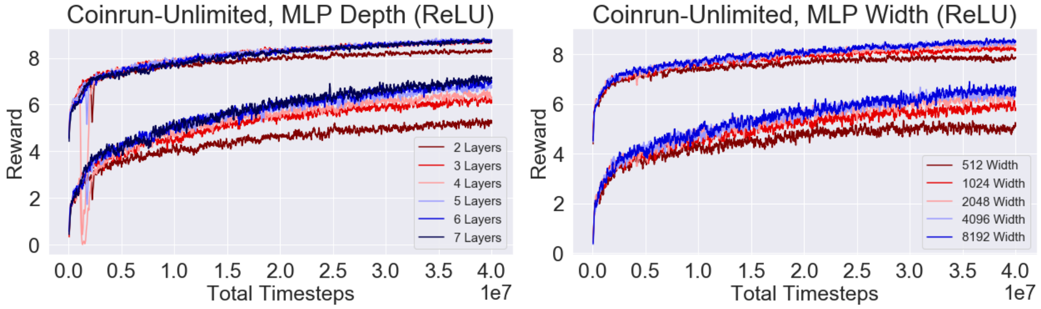


Figure 9: Overparametrization improves generalization for CoinRun.

While we also extend the case of large-parameter convolutional networks using ImageNet networks in Appendix A.2.1, an important question is how to predict the generalization gap only from the training phase. A particular set of metrics, popular in the SL community are *margin distributions* (Jiang et al., 2018; Bartlett et al., 2017), as they deal with the case for softmax outputs which do not explicitly penalize the weight norm of a network, by normalizing the "confidence" margin of the logit outputs. While using margins on state-action pairs (from an on-policy replay buffer) is not technically rigorous, one may be curious to see if they have predictive power, especially as MLPs are relatively simple to norm-bound. We plotted these margin distributions in Appendix A.2.2, but found that the weight norm bounds used in SL are simply too dominant for this RL case. This, with the bound results found earlier for the LQR case, suggests that current norm bounds are simply too loose for the RL case even though we have shown overparametrization helps generalization in RL, and hopefully this motivates more of the study of such theory.

4 Conclusion

We have identified and isolated a key component of overfitting in RL as the particular case of "observational overfitting", which is particularly attractive for studying architectural implicit regularizations. We have analyzed this setting extensively, by examining 3 main components:

1. The analytical case of LQR and linear policies under exact gradient descent, which lays the foundation for understanding theoretical properties of networks in RL generalization.
2. The empirical but principled Projected-Gym case for both MLP and convolutional networks which demonstrates the effects of neural network policies under nonlinear environments.
3. The large scale case for CoinRun, which can be interpreted as a case where relevant features are moving across the input, where empirically, MLP overparametrization also improves generalization.

We noted that current network policy bounds using ideas from SL are unable to explain overparametrization effects in RL, which is an important further direction. In some sense, this area of RL generalization is an extension of static SL classification from adding extra RL components. For instance, adding a nontrivial "combination function" between f and g_θ that is dependent on time (to simulate how object pixels move in a real game) is both an RL generalization issue and potentially video classification issue, and extending results to the memory-based RNN case will also be highly beneficial.

Furthermore, it is unclear whether such overparametrization effects would occur in off-policy methods such as Q-learning and also ES-based methods. In terms of architectural design, recent works (Jacot et al., 2018; Garriga-Alonso et al., 2019; Lee et al., 2019) have shed light on the properties of asymptotically overparametrized neural networks in the infinite width and depth cases and their performance in SL. Potentially such architectures (and a corresponding training algorithm) may be used in the RL setting which can possibly provide benefits, one of which is generalization as shown in this paper. We believe that this work provides an important initial step towards solving these future problems.

Acknowledgements

We would like to thank John Schulman for very helpful guidance over the course of this work. We also wish to thank Chiyuan Zhang, Ofir Nachum, Aurick Zhou, Daniel Seita, Alexander Irpan, and the OpenAI team for fruitful comments and discussions during the course of this work.

References

Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. Understanding the impact of entropy on policy optimization. *CoRR*, abs/1811.11214, 2018.

Martín Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *CoRR*, abs/1907.02893, 2019. URL `http://arxiv.org/abs/1907.02893`.

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. *CoRR*, abs/1802.06509, 2018a. URL `http://arxiv.org/abs/1802.06509`.

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 244–253, 2018b.

Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 254–263, 2018c.

Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. *CoRR*, abs/1905.13655, 2019. URL `http://arxiv.org/abs/1905.13655`.

Kavosh Asadi, Dipendra Misra, and Michael L. Littman. Lipschitz continuity in model-based reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 264–273, 2018.

Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 6240–6249. Curran Associates, Inc., 2017.

Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *CoRR*, abs/1207.4708, 2012.

Joseph K. Bradley, Aapo Kyrola, Danny Bickson, and Carlos Guestrin. Parallel coordinate descent for l1-regularized loss minimization. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pp. 321–328, 2011. URL `https://icml.cc/2011/papers/231_icmlpaper.pdf`.

Krzysztof Choromanski, Mark Rowland, Vikas Sindhwani, Richard E. Turner, and Adrian Weller. Structured evolution with compact architectures for scalable policy optimization. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 969–977, 2018. URL `http://proceedings.mlr.press/v80/choromanski18a.html`.

Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. *CoRR*, abs/1812.02341, 2018.

Sarah Dean, Horia Mania, Nikolai Matni, Benjamin Recht, and Stephen Tu. On the sample complexity of the linear quadratic regulator. *Foundations of Computational Mathematics*, Aug 2019.

Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

Lasse Espeholt, Hubert Soyer, Rémi Munos, Karen Simonyan, Volodymyr Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: scalable distributed deep-rl with importance weighted actor-learner architectures. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 1406–1415, 2018.

Maryam Fazel, Rong Ge, Sham Kakade, and Mehran Mesbahi. Global convergence of policy gradient methods for the linear quadratic regulator. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 1466–1475, 2018.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 1126–1135, 2017. URL http://proceedings.mlr.press/v70/finn17a.html.

Adam Gaier and David Ha. Weight agnostic neural networks. *CoRR*, abs/1906.04358, 2019. URL http://arxiv.org/abs/1906.04358.

Shani Gamrian and Yoav Goldberg. Transfer learning for related reinforcement learning tasks via image-to-image translation. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 2063–2072, 2019. URL http://proceedings.mlr.press/v97/gamrian19a.html.

Adrià Garriga-Alonso, Carl Edward Rasmussen, and Laurence Aitchison. Deep convolutional networks as shallow gaussian processes. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019. URL https://openreview.net/forum?id=Bklfsi0cKm.

Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. In *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018.*, pp. 297–299, 2018. URL http://proceedings.mlr.press/v75/golowich18a.html.

Samuel Greydanus, Anurag Koul, Jonathan Dodge, and Alan Fern. Visualizing and understanding atari agents. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 1787–1796, 2018.

Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit regularization in matrix factorization. In *Advances in Neural Information Processing Systems*, pp. 6151–6159, 2017.

Christina Heinze-Deml and Nicolai Meinshausen. Conditional variance penalties and domain shift robustness. *CoRR*, abs/1710.11469, 2019. URL https://arxiv.org/abs/1710.11469.

Christina Heinze-Deml, Jonas Peters, and Nicolai Meinshausen. Invariant causal prediction for nonlinear models. *CoRR*, abs/1706.08576, 2019. URL https://arxiv.org/abs/1706.08576.

Maximilian Igl, Kamil Ciosek, Yingzhen Li, Sebastian Tschiatschek, Cheng Zhang, Sam Devlin, and Katja Hofmann. Generalization in reinforcement learning with selective noise injection and information bottleneck. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2019,Long Beach, CA, USA*, 2019.

Arthur Jacot, Clément Hongler, and Franck Gabriel. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pp. 8580–8589, 2018.

Nan Jiang, Alex Kulesza, Satinder Singh, and Richard Lewis. The dependence of effective planning horizon on model accuracy. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '15, pp. 1181–1189, Richland, SC, 2015. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 978-1-4503-3413-6.

Yiding Jiang, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Predicting the generalization gap in deep networks with margin distributions. *CoRR*, abs/1810.00113, 2018. URL http://arxiv.org/abs/1810.00113.

Niels Justesen, Ruben Rodriguez Torrado, Philip Bontrager, Ahmed Khalifa, Julian Togelius, and Sebastian Risi. Procedural level generation improves generality of deep reinforcement learning. *CoRR*, abs/1806.10729, 2018. URL http://arxiv.org/abs/1806.10729.

Sham M. Kakade, Karthik Sridharan, and Ambuj Tewari. On the complexity of linear prediction: Risk bounds, margin bounds, and regularization. In *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*, pp. 793–800, 2008.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *CoRR*, abs/1902.06720, 2019. URL http://arxiv.org/abs/1902.06720.

Tengyuan Liang, Tomaso A. Poggio, Alexander Rakhlin, and James Stokes. Fisher-rao metric, geometry, and complexity of neural networks. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, pp. 888–896, 2019. URL http://proceedings.mlr.press/v89/liang19a.html.

Marlos C. Machado, Marc G. Bellemare, Erik Talvitie, Joel Veness, Matthew J. Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *J. Artif. Intell. Res.*, 61:523–562, 2018. doi: 10.1613/jair.5699.

Horia Mania, Aurelia Guy, and Benjamin Recht. Simple random search provides a competitive approach to reinforcement learning. *CoRR*, abs/1803.07055, 2018.

David A McAllester. Pac-bayesian model averaging. In *COLT*, volume 99, pp. 164–170. Citeseer, 1999.

David A. McAllester. Simplified pac-bayesian margin bounds. In *Computational Learning Theory and Kernel Machines, 16th Annual Conference on Computational Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003, Proceedings*, pp. 203–215, 2003. doi: 10.1007/978-3-540-45167-9_16. URL https://doi.org/10.1007/978-3-540-45167-9_16.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

Vaishnavh Nagarajan and J Zico Kolter. Generalization in deep networks: The role of distance from initialization. *arXiv preprint arXiv:1901.01672*, 2019.

Behnam Neyshabur. Implicit regularization in deep learning. *CoRR*, abs/1709.01953, 2017. URL http://arxiv.org/abs/1709.01953.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. *ICLR*, 2015a.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. Norm-based capacity control in neural networks. In *Conference on Learning Theory*, pp. 1376–1401, 2015b.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 5949–5958, 2017.

Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018a. URL https://openreview.net/forum?id=Skz_WfbCZ.

Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. Towards understanding the role of over-parametrization in generalization of neural networks. *CoRR*, abs/1805.12076, 2018b. URL http://arxiv.org/abs/1805.12076.

Alex Nichol, Vicki Pfau, Christopher Hesse, Oleg Klimov, and John Schulman. Gotta learn fast: A new benchmark for generalization in RL. *CoRR*, abs/1804.03720, 2018.

OpenAI. Openai five. *CoRR*, 2018. URL https://openai.com/blog/openai-five/.

Charles Packer, Katelyn Gao, Jernej Kos, Philipp Krähenbühl, Vladlen Koltun, and Dawn Song. Assessing generalization in deep reinforcement learning. *CoRR*, abs/1810.12282, 2018. URL http://arxiv.org/abs/1810.12282.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pp. 1–8, 2018. doi: 10.1109/ICRA.2018.8460528. URL https://doi.org/10.1109/ICRA.2018.8460528.

Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 2817–2826, 2017.

Aravind Rajeswaran, Kendall Lowrey, Emanuel Todorov, and Sham M. Kakade. Towards generalization and simplicity in continuous control. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 6553–6564, 2017.

Adam Santoro, Ryan Faulkner, David Raposo, Jack W. Rae, Mike Chrzanowski, Theophane Weber, Daan Wierstra, Oriol Vinyals, Razvan Pascanu, and Timothy P. Lillicrap. Relational recurrent neural networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pp. 7310–7321, 2018.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

Samuel L. Smith, Pieter-Jan Kindermans, Chris Ying, and Quoc V. Le. Don't decay the learning rate, increase the batch size. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018. URL https://openreview.net/forum?id=B1Yy1BxCZ.

Felipe Petroski Such, Vashisht Madhavan, Rosanne Liu, Rui Wang, Pablo Samuel Castro, Yulun Li, Ludwig Schubert, Marc G. Bellemare, Jeff Clune, and Joel Lehman. An atari model zoo for analyzing, visualizing, and comparing deep reinforcement learning agents. *CoRR*, abs/1812.07069, 2018.

Stephen Tu and Benjamin Recht. The gap between model-based and model-free methods on the linear quadratic regulator: An asymptotic viewpoint. In *Conference on Learning Theory, COLT 2019, 25-28 June 2019, Phoenix, AZ, USA*, pp. 3036–3083, 2019. URL http://proceedings.mlr.press/v99/tu19a.html.

Vladimir N Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Theory of probability and its applications*, pp. 11–30. Springer, 1971.

Roman Vershynin. Lectures in geometric functional analysis. *CoRR*, 2009.

Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, John Quan, Stephen Gaffney, Stig Petersen, Karen Simonyan, Tom Schaul, Hado van Hasselt, David Silver, Timothy P. Lillicrap, Kevin Calderone, Paul Keet, Anthony Brunasso, David Lawrence, Anders Ekermo, Jacob Repp, and Rodney Tsing. Starcraft II: A new challenge for reinforcement learning. *CoRR*, abs/1708.04782, 2017. URL `http://arxiv.org/abs/1708.04782`.

Huan Wang, Stephan Zheng, Caiming Xiong, and Richard Socher. On the generalization gap in reparameterizable reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 6648–6658, 2019. URL `http://proceedings.mlr.press/v97/wang19o.html`.

Wenhao Yu, Jie Tan, C. Karen Liu, and Greg Turk. Preparing for the unknown: Learning a universal policy with online system identification. In *Robotics: Science and Systems XIII, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, July 12-16, 2017*, 2017. doi: 10.15607/RSS.2017.XIII.048. URL `http://www.roboticsproceedings.org/rss13/p48.html`.

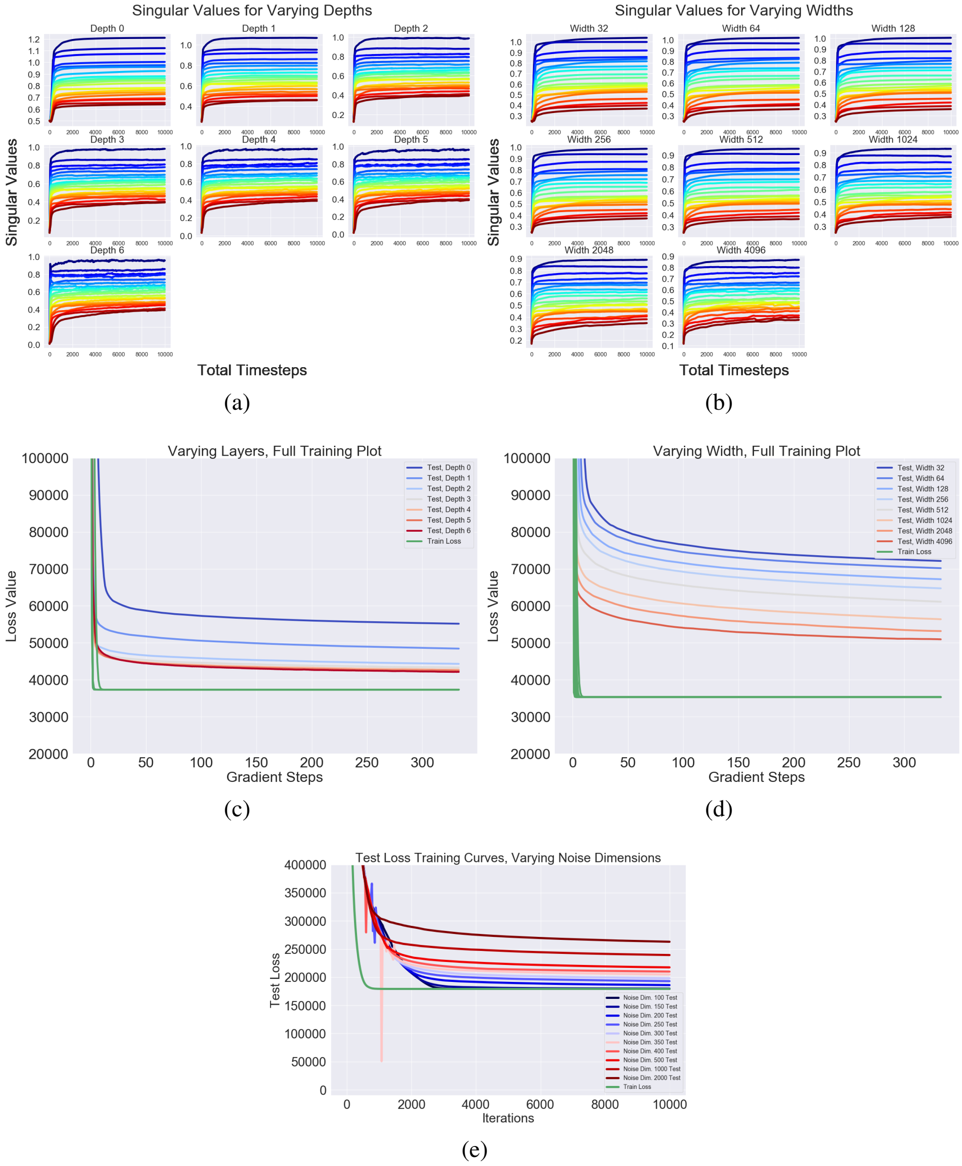
Amy Zhang, Nicolas Ballas, and Joelle Pineau. A dissection of overfitting and generalization in continuous reinforcement learning. *CoRR*, abs/1806.07937, 2018a.

Amy Zhang, Yuxin Wu, and Joelle Pineau. Natural environment benchmarks for reinforcement learning. *CoRR*, abs/1811.06032, 2018b.

Chiyuan Zhang, Oriol Vinyals, Rémi Munos, and Samy Bengio. A study on overfitting in deep reinforcement learning. *CoRR*, abs/1804.06893, 2018c. URL `http://arxiv.org/abs/1804.06893`.

A.1 Full Plots for LQR and fg-Gym

A.1.1 LQR



(a)

(b)

(c)

(d)

(e)

Figure A1: (a,b): Singular Values for varying depths and widths. (c,d): Train and Test Loss for varying widths and depths. (e): Train and Test Loss for varying Noise Dimensions.

We further verify that *explicit regularization* (norm based penalties) also reduces generalization gaps. However, explicit regularization may be explained due to the bias of the synthetic tasks, since the first layer's matrix may be regularized to only ”view” the output of f, especially as regularizing the first layer's weights substantially improves generalization.

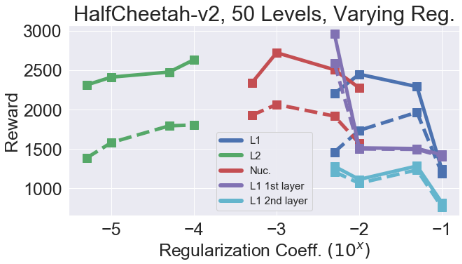


Figure A2: Explicit Regularization on layer norms.

We provide another deconvolution memorization test, using an LQR as the underlying MDP. While fg-Gym-Deconv shows that memorization performance is dampened, this test shows that there can exist specific hard limits to memorization. Specifically, NatureCNN can memorize 30 levels, but not 50; IMPALA can memorize 2 levels but not 5; IMPALA-LARGE cannot memorize 2 levels at all.

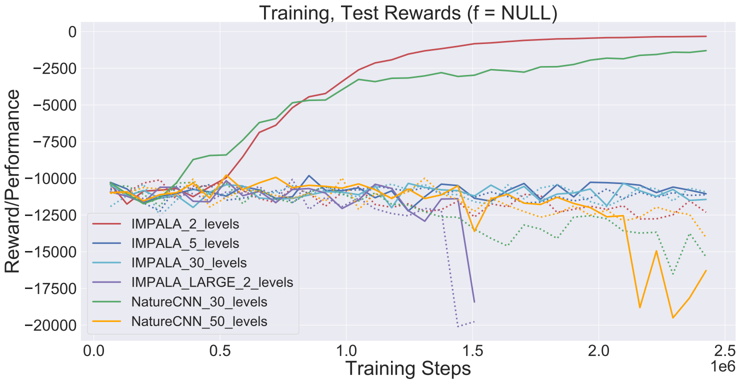


Figure A3: Deconvolution memorization test using LQR as underlying MDP.

A.2 EXTENDED LARGE RL RESULTS

A.2.1 LARGE IMAGENET MODELS FOR COINRUN

For reference, we also extend the case of large-parameter convolutional networks using ImageNet networks. We experimentally verify in Table 1 that large ImageNet models perform very differently in RL than SL. We note that default network with the highest test reward was IMPALA-LARGE-BN (IMPALA-LARGE, with Batchnorm) at ≈ 5.5 test score.

In order to verify that this is inherently a *feature learning* problem rather than a *combinatorial problem* involving objects, such as in (Santoro et al., 2018), we show that state-of-the-art attention mechanisms for RL such as Relational Memory Core (RMC) using pure attention on raw 32×32 pixels does not perform well here, showing that a large portion of generalization and transfer must be based on correct convolutional setups.

Architecture	Coinrun-100 (Train, Test)
AlexNet-v2	(10.0, 3.0)
CifarNet	(10.0, 3.0)
IMPALA-LARGE-BN	(10.0, 5.5)
Inception-ResNet-v2	(10.0, 6.5)
Inception-v4	(10.0, 6.0)
MobileNet-v1	(10.0, 5.5)
MobileNet-v2	(10.0, 5.5)
NASNet-CIFAR	(10.0, 4.0)
NASNet-Mobile	(10.0, 4.5)
ResNet-v2-50	(10.0, 5.5)
ResNet-v2-101	(10.0, 5.0)
ResNet-v2-152	(10.0, 5.5)
RMC32x32	(9.0, 2.5)
ShakeShake	(10.0, 6.0)
VGG-A	(9.0, 3.0)
VGG-16	(9.0, 3.0)

Table 1: Raw Network Performance (rounded to nearest 0.5) on CoinRun, 100 levels. Images scaled to default image sizes (32×32 or 224×224) depending on network input requirement. See Appendix A.2.1 for training curves.

We provide the training/testing curves for the ImageNet/large convolutional models used. Note the following:

1. RMC32x32 projects the native image from CoinRun from 64×64 to 32×32, and uses all pixels as components for attention, after adding the coordinate embedding found in (Santoro et al., 2018). Optimal parameters were (mem_slots = 4, head_size = 32, num_heads = 4, num_blocks = 2, gate_style = 'memory').
2. Auxiliary Loss in ShakeShake was not used during training, only the pure network.
3. VGG-A is a similar but slightly smaller version of VGG-16.

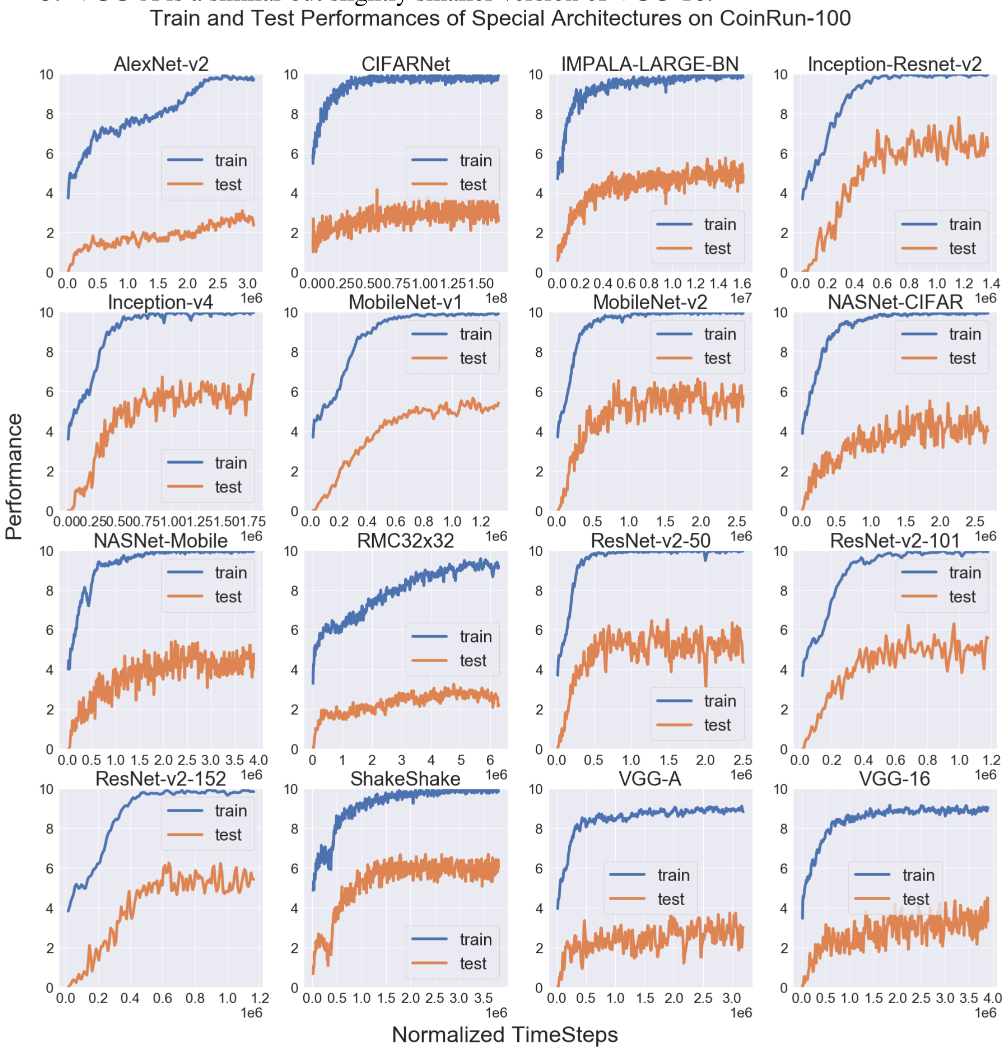


Figure A4: Large Architecture Training/Testing Curves (Smoothed).

A.2.2 Do State-Action Margin Distributions Predict Generalization in RL?

A key question is how to predict the generalization gap only from the training phase. A particular set of metrics, popular in the SL community are *margin distributions* (Jiang et al., 2018; Bartlett et al., 2017), as they deal with the case for softmax categorical outputs which do not explicitly penalize the weight norm of a network, by normalizing the ”confidence” margin of the logit outputs. While using margins on state-action pairs (from an on-policy replay buffer) is not technically rigorous, one may be curious to see if they have predictive power, especially as MLP’s are relatively simple to norm-bound, and as seen from the LQR experiments, the norm of the policy may be correlated with the generalization performance.

For a policy, the the margin distribution will be defined as $(x, y) \rightarrow \frac{F_\pi(x)_y - \max_{i \neq y} F_\pi(x)_i}{\mathcal{R}_\pi \|S\|_2 / n}$, where $F_\pi(x)_y$ is the logit value (before applying softmax) of output y given input x, and S is the matrix of states in the replay buffer, and $\mathcal{R}_\pi$ is a norm-based Lipschitz measure on the policy network logits. In general, $\mathcal{R}_\pi$ is a bound on the Lipschitz constant of the network but can also be simply expressions which allow the margin distribution to have high correlation with the generalization gap. Thus, we use measures inspired by recent literature in SL in which we designate Spectral-L1, Distance, and Spectral-Frobenius measures for $\mathcal{R}_\pi$, and we replace the classical supervised learning pair $(x, y) = (s, a)$ with the state-action pairs found on-policy. [2]

The expressions for $\mathcal{R}_\pi$ (after removing irrelevant constants) are as follows, with their analogous papers:

1. Spectral-L1 measure: $\left(\prod_{i=1}^{d} \|W_i\|\right) \left(\sum_{i=1}^{d} \frac{\|W_i\|_1^{2/3}}{\|W_i\|^{2/3}}\right)^{3/2}$ (Bartlett et al., 2017)
2. Distance measure: $\sqrt{\sum_{i=1}^{d} \|W_i - W_i^0\|_F^2}$ (Nagarajan & Kolter, 2019)
3. Spectral-Fro measure: $\sqrt{\ln(d) \prod_{i=1}^{d} \|W_i\|^2 \sum_{j=1}^{d} \frac{\|W_j - W_j^0\|_F^2}{\|W_j\|^2}}$ (Neyshabur et al., 2018a)

We verify in Figure A5, that indeed, simply measuring the raw norms of the policy network is a poor way to predict generalization, as it generally increases even as training begins to plateau. This is inherently because the softmax on the logit output does not penalize arbitrarily high logit values, and hence proper normalization is needed.

The margin distribution converges to a fixed distribution even long after training has plateaued. However, unlike SL, the margin distribution is conceptually not fully correlated with RL generalization on the total reward, as a policy overconfident in some state-action pairs does not imply bad testing performance. This correlation is stronger if there are Lipschitz assumptions on state-action transitions, as noted in (Wang et al., 2019). For empirical datasets such as CoinRun, a metric-distance between transitioned states is ill-defined however. Nevertheless, the distribution over the on-policy replay buffer at each policy gradient iteration is a rough measure of overall confidence.

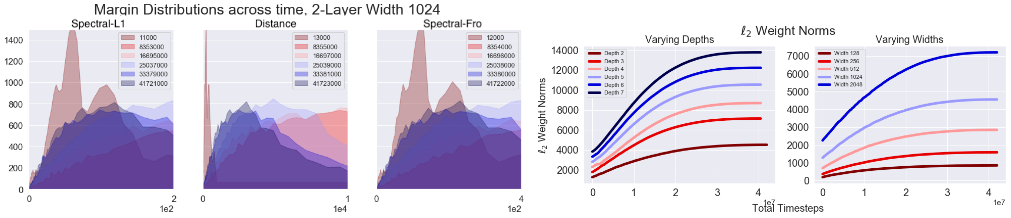


Figure A5: Margin Distributions across training.

We note that there are two forms of modifications, *network dependent* (explicit modifications to the policy - norm regularization, dropout, etc.) and *data dependent* (modifications only to the data in the replay buffer - action stochasticity, data augmentation, etc.). Ultimately however, we find that current

[2] We removed the training sample constant m from all original measures as this is ill-defined for the RL case, when one can generate infinitely many (s, a) pairs. Furthermore, we used the original $\|W_i\|_1$ in the numerator found in the first version of (Bartlett et al., 2017) rather than the current $\|W_i\|_{1,2}$.

norm measures R_π become too dominant in the fraction, leading to the monotonic decreases in the means of the distributions as we increase parametrization.

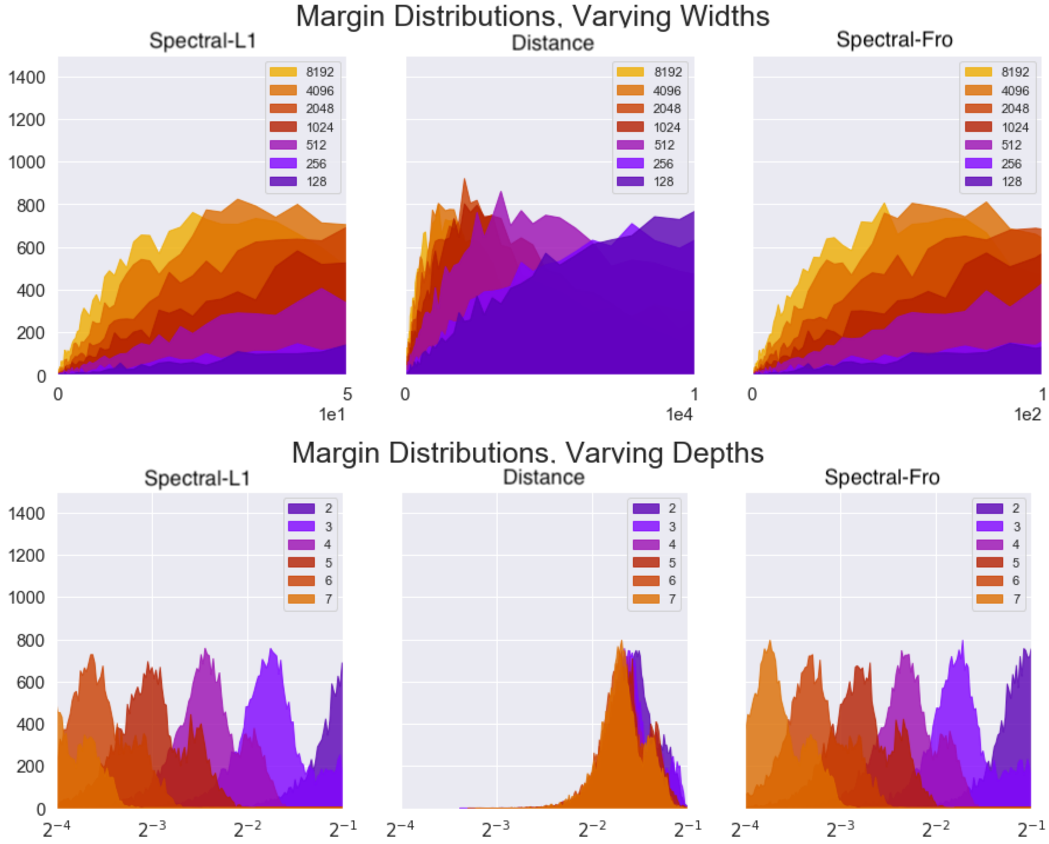


Figure A6: Margin Distributions at the end of training.

A.2.3 Gym-Retro (Sonic)

In the Gym-Retro benchmark using Sonic (Nichol et al., 2018), the agent is given 47 training levels with rewards corresponding to increases in horizontal location. The policy is trained until 5k reward. At test time, 11 unseen levels are partitioned into starting positions, and the rewards are measured and averaged.

We briefly mention that the agent strongly overfits to the scoreboard (i.e. an artifact correlated with progress in the level), which may be interpreted as part of the output of $g_\theta(\cdot)$. In fact, the agent is still able to train to 5k reward from purely observing the timer as the observation. By blacking out this scoreboard with a black rectangle, we see an increase in test performance.

Settings	IMPALA	NatureCNN
Blackout	1250 ± 40	1141 ± 40
NoBlackout	1130 ± 40	1052 ± 40

Table 2: IMPALA vs NatureCNN test rewards, with and without Blackout.

A.3 Hyperparameters and Exact Setups

A.3.1 Exact infinite LQR

For infinite horizon case, see (Fazel et al., 2018) for the the full solution and notations. Using the same notation (A, B, Q, R), denote $C(K) = \sum_{x_0 \sim \mathcal{D}} x_0^T P_K x_0$ as the cost and $u_t = -K x_t$ as the policy, where P_K satisfies the infinite case for the Lyapunov equation:

$$P_K = Q + K^T RK + (A - BK)^T P_K (A - BK) \tag{3}$$

We may calculate the precise LQR cost by vectorizing (i.e. flattening) both sides' matrices and using the Kroncker product $\otimes$, which leads to a linear regression problem on P_K, which has a precise solution, implementable in TensorFlow:

$$\text{vec}(P_K) = \text{vec}(Q) + \text{vec}(K^T RK) + \left[(A - BK)^T \otimes (A - BK)^T)\right] \text{vec}(P_K) \tag{4}$$

$$\left[I_{n^2} - (A - BK)^T \otimes (A - BK)^T\right] \text{vec}(P_K) = \text{vec}(Q) + \text{vec}(K^T RK) \tag{5}$$

Parameter	Generation
A	Orthogonal initialization, scaled 0.99
B	I_n
Q	I_n
R	I_n
n	10
$K_i \quad \forall i$	Orthogonal Initialization, scaled 0.5

Table 3: Hyperparameters for LQR

A.3.2 Projection Method

The basis for producing f, g_θ outputs is due to using batch matrix multiplication operations, or "BMV", where the same network architecture uses different network weights for each batch dimension, and thus each entry in a batchsize of B will be processed by the same architecture, but with different network weights. This is to simulate the effect of g_{θ_i}. The numeric ID i of the environment is used as an index to collect a specific set of network weights θ_i from a global memory of network weights (e.g. using `tensorflow.gather`). We did not use nonlinear activations for the BMV architectures, as they did not change the outcome of the results.

Architecture	Setup
BMV-Deconv	(filtersize = 2, stride = 1, outchannel = 8, padding = "VALID") (filtersize = 4, stride = 2, outchannel = 4, padding = "VALID") (filtersize = 8, stride = 2, outchannel = 4, padding = "VALID") (filtersize = 8, stride = 3, outchannel = 3, padding = "VALID")
BMV-Dense	f : Dense 30, g : Dense 100

A.3.3 ImageNet Models

For the networks used in the supervised learning tasks, we direct the reader to the following repository: `https://github.com/tensorflow/models/blob/master/research/slim/nets/nets_factory.py`. We also used the RMC: `deepmind/sonnet/blob/master/sonnet/python/modules/relational_memory.py`

A.3.4 PPO Parameters

For the projected gym tasks, we used for PPO2 Hyperparameters:

PPO2 Hyperparameters	Values
nsteps	2048
nenvs	16
nminibatches	64
λ	0.95
γ	0.99
noptepochs	10
entropy	0.0
learning rate	$3 \cdot 10^{-4}$
vf coeffiicent	0.5
max-grad-norm	0.5
total time steps	Varying

See (Cobbe et al., 2018) for the default parameters used for CoinRun. We only varied nminibatches in order to fit memory onto GPU. We also did not use RNN additions, in order to measure performance only from the feedforward network - the framestacking/temporal aspect is replaced by the option to present the agent velocity in the image.

A.4 Theoretical (LQR)

In this section, we use notation consistent with (Fazel et al., 2018) for our base proofs. However, in order to avoid confusion with a high dimensional policy K we described in 3.1, we denote our low dimensional base policy as P and state as s_t rather than x_t.

A.4.1 Notation and Setting

Let $\|\cdot\|$ be the spectral norm of a matrix (i.e. largest singular value). Suppose $C(P)$ was the infinite horizon cost for an (A, B, Q, R)-LQR where action $a_t = -P \cdot s_t$, s_t is the state at time t, state transition is $s_{t+1} = A \cdot s_t + B \cdot a_t$, and timestep cost is $s_t^T Q s_t + a_t^T R a_t$.

$C(P)$ for an infinite horizon LQR, while known to be non-convex, still possess the property that when $\nabla C(P^*) = 0$, P^* is a global minimizer, or the problem statement is rank deficient. To ensure that our cost $C(P)$ always remains finite, we restrict our analysis when $P \in \mathcal{P}$, where $\mathcal{P} = \{P : \|P\| \leq \alpha$ and $\|A - BP\| \leq 1\}$ for some constant α, by choosing A, B and the initialization of P appropriately, using the hyperparameters found in A.3.1. We further define the observation modified cost as $C(K; W_\theta) = C\left(K \begin{bmatrix} W_c \\ W_\theta \end{bmatrix}^\intercal\right)$.

A.4.2 Smoothness Bounds

As described in Lemma 16 of (Fazel et al., 2018), we define

$$T_P(X) = \sum_{t=0}^{\infty} (A - BP)^t X [(A - BP)^T]^t \tag{6}$$

and $\|T_P\| = \sup_X \frac{T_P(X)}{\|X\|}$ over all non-zero symmetric matrices X.

Lemma 27 of (Fazel et al., 2018) provides a bound on the difference $C(P') - C(P)$ for two different policies P, P' when LQR parameters A, B, Q, R are fixed. During the derivation, it states that when $\|P - P'\| \leq \min\left(\frac{\sigma_{min}(Q)\mu}{4C(P)\|B\|(\|A-BP\|+1)}, \|P\|\right)$, then:

$$\begin{aligned} C(P') - C(P) \leq 2\|T_P\|(2\|P\|\|R\|\|P' - P\| + \|R\|\|P' - P\|^2) + \\ 2\|T_P\|^2 2\|B\|(\|A - BP\| + 1)\|P - P'\|\|P\|^2\|R\| \end{aligned} \tag{7}$$

Lemma 17 also states that:

$$\|T_P\| \leq \frac{C(P)}{\mu\sigma_{min}(Q)} \tag{8}$$

where

$$\mu = \sigma_{min}(\mathbb{E}_{x_0 \sim D}[x_0 x_0^T]) \tag{9}$$

Assuming that in our problem setup, x_0, Q, R, A, B were fixed, this means many of the parameters in the bounds are constant, and thus we conclude:

$$C(P') - C(P) \leq \mathcal{O}\left(C(P)^2 \left[\|P\|^2 \|P - P'\| (\|A - BP\| + \|B\| + 1) + \|P\| \|P - P'\|^2\right]\right) \tag{10}$$

Since we assumed $\|A - BP\| \leq 1$ or else $T_P(X)$ is infinite, we thus collect the terms:

$$C(P') - C(P) \leq \mathcal{O}\left(C(P)^2 \left[\|P\|^2 \|P - P'\| + \|P\| \|P - P'\|^2\right]\right) \tag{11}$$

Since α is a bound on $\|P\|$ for $P \in \mathcal{P}$, note that

$$\|P\|^2 \|P - P'\| + \|P\| \|P - P'\|^2 = \|P - P'\| (\|P\|^2 + \|P\| + \|P - P'\|) \tag{12}$$

$$\leq \|P - P'\| (\|P\|^2 + \|P\| (\|P\| + \|P'\|) \leq (3\alpha^2) \|P - P'\| \tag{13}$$

From (11), this leads to the bound:

$$C(P') - C(P) \leq \mathcal{O}\left(C(P)^2 \|P - P'\|\right) \tag{14}$$

Note that this directly implies a similar bound in the high dimensional observation case - in particular, if $P = K \begin{bmatrix} W_c \\ W_\theta \end{bmatrix}^\mathsf{T}$ and $P' = K \begin{bmatrix} W_c \\ W_\theta \end{bmatrix}^\mathsf{T}$ then $\|P - P'\| \leq \|K - K'\| \left\| \begin{bmatrix} W_c \\ W_\theta \end{bmatrix}^\mathsf{T} \right\| = \|K - K'\|$.

A.4.3 Gradient Dynamics in 1-Step LQR

We first start with a convex cost 1-step LQR toy example under this regime, which shows that linear components such as $\beta \begin{bmatrix} 0 \\ W_\theta \end{bmatrix}^\mathsf{T}$ cannot be removed from the policy by gradient descent dynamics to improve generalization. To shorten notation, let $W_c \in \mathbb{R}^{n \times n}$ and $W_\theta \in \mathbb{R}^{p \times n}$, where $n \ll p$. This is equivalent to setting $d_{signal} = d_{state} = n$ and $d_{noise} = p$, and thus the policy $K \in \mathbb{R}^{n \times (n+p)}$.

In the 1-step LQR, we allow $s_0 \sim \mathcal{N}(0, I)$, $a_0 = K \begin{bmatrix} W_c \\ W_\theta \end{bmatrix} s_0$ and $s_1 = s_0 + a_0$ with cost $\frac{1}{2} \|s_1\|^2$, then

$$C(K; W_\theta) = \mathbb{E}_{s_0} \left[\frac{1}{2} \left\| x_0 + K \begin{bmatrix} W_c \\ W_\theta \end{bmatrix} x_0 \right\|^2 \right] = \frac{1}{2} \left\| I + K \begin{bmatrix} W_c \\ W_\theta \end{bmatrix} \right\|_F^2 \tag{15}$$

and

$$\nabla C(K; W_\theta) = \left(I + K \begin{bmatrix} W_c \\ W_\theta \end{bmatrix} \right) \begin{bmatrix} W_c \\ W_\theta \end{bmatrix}^\mathsf{T} . \tag{16}$$

Define the population cost as $C(K) := \mathbb{E}_{W_\theta}[C(K; W_\theta)]$. Let the notation $O(p, n)$ denote the following set of orthogonal matrices:

$$O(p, n) = \{W \in \mathbb{R}^{p \times n} : W^\mathsf{T} W = I\} .$$

We use the shorthand $O(n) = O(n, n)$.

Proposition 1. *Suppose that $W_\theta \sim \mathrm{Unif}(O(p, n))$ and $W_c \sim \mathrm{Unif}(O(n))$. Then*

(i) The minimizer of $C(K)$ is unique and given by $K_\star = \begin{bmatrix} -W_c^\mathsf{T} & 0 \end{bmatrix}$.
(ii) Thus, the minimizer cost is $C(K_\star) = 0$.

Proof. By standard properties of the Haar measure on $O(p, n)$, we have that $\mathbb{E}[W_\theta] = 0$ and $\mathbb{E}[W_\theta W_\theta^\mathsf{T}] = \frac{n}{p} I$. Therefore,

$$C(K) = \frac{n}{2} + \frac{1}{2}\mathbb{E}\left[\left\| K \begin{bmatrix} W_c \\ W_\theta \end{bmatrix} \right\|_F^2\right] + \mathbb{E}\left[\operatorname{tr}\left(K \begin{bmatrix} W_c \\ W_\theta \end{bmatrix}\right)\right]$$
$$= \frac{n}{2} + \frac{1}{2}\operatorname{tr}\left(K^\mathsf{T} K \begin{bmatrix} I & 0 \\ 0 & \frac{n}{p} I \end{bmatrix}\right) + \operatorname{tr}\left(K \begin{bmatrix} W_c \\ 0 \end{bmatrix}\right) .$$

We can now differentiate $C(K)$:

$$\nabla C(K) = K \begin{bmatrix} I & 0 \\ 0 & \frac{n}{p} I \end{bmatrix} + \begin{bmatrix} W_c^\mathsf{T} & 0 \end{bmatrix} .$$

Both claims now follow. □

A.4.3.1 Finite Sample Generalization Gap

As an instructive example, we consider the case when we only possess one sample W_θ. Note that if $K = K' + \beta \begin{bmatrix} 0 \\ W_\theta \end{bmatrix}^\mathsf{T}$, then $\nabla C(K; W_\theta) = \nabla C(K'; W_\theta) + \beta \begin{bmatrix} 0 \\ W_\theta \end{bmatrix}^\mathsf{T}$. In particular, if we perform gradient descent dynamics $K_{t+1} = K_t - \eta \nabla C(K_t; W_\theta)$, then we have

$$K_t = K_0 (I - \eta M)^t + B (I - \eta M)^{t-1} + ... + B \tag{17}$$

where $M = \begin{bmatrix} W_c \\ W_\theta \end{bmatrix} \begin{bmatrix} W_c \\ W_\theta \end{bmatrix}^\mathsf{T}$ is the Hessian of $C(K; W_\theta)$ and $B = -\eta \begin{bmatrix} W_c \\ W_\theta \end{bmatrix}$. Note that M has rank at most $n \ll n + p$, and thus at a high level, $K_0 (I - \eta M)^t$ does not diminish if some portion of K_0 lies in the orthogonal complement of the range of M. If the initialization is $K_0 \sim \mathcal{N}(0, I)$, then it is highly likely we can find a subspace $Q \subseteq \operatorname{range}(M)^\perp$ for which $K_0 Q \neq 0$.

This is a specific example of the general case where if the Hessian of a function $f(x)$ is degenerate everywhere (e.g. has rank $k < n$), then an x_0 initialized with e.g. an isotropic Gaussian distribution cannot converge under gradient descent to a minimizer that lives in the span of the Hessian, as the non-degenerate components do not change. In particular, Proposition 4.7 in (Vershynin, 2009) points to the exact magnitude of the non-degenerate component in the relevant subspace Q: $\mathbb{E}_{x \sim \mathcal{N}(0, I)}\left[\|\operatorname{Proj}_Q(x)\|^2\right] = \frac{n-k}{n}$.

The generalization gap may decrease if the number of level samples is high enough. This can be seen by the sample Hessian of $\widehat{C}(K) = \frac{1}{m}\sum_{i=1}^m C(K; W_{\theta_i})$ being $\widehat{M} = \frac{1}{m}\sum_{i=1}^m M_i$ where $M_i = \begin{bmatrix} W_c \\ W_{\theta_i} \end{bmatrix} \begin{bmatrix} W_c \\ W_{\theta_i} \end{bmatrix}^\mathsf{T}$ $\forall i$. In particular, as m increases, the rank of $\widehat{M}$ increases, which allows gradient descent to recover the minimizer $K_\star$ better.

We calculate the exact error of gradient descent on m samples of the 1-step LQR problem.

To simplify notation, we rewrite W_{θ_i} as W_i, and interchangeably use the abbreviations for the finite $C_m(K) = C_m(\cdot; \{W_i\}) = \frac{1}{m}\sum_{i=1}^m C(K; W_i)$ when necessary. We consider gradient descent $K_{t+1} = K_t - \eta \nabla C_m(K_t)$ starting from a random K_0 with each entry iid $\mathcal{N}(0, \psi^2)$ and η sufficiently small so that gradient descent converges. Let K_∞ denote the limit point of gradient descent. We prove the following generalization gap between K_∞ and K_{opt}.

Theorem 1. *Suppose that n divides p. Fix an $m \in \{1, ..., p/n\}$. Suppose that the samples $\{W_1, ..., W_m\}$ are generated by first sampling a $W \sim \operatorname{Unif}(O(p))$, and then partitioning the first $n \cdot m$ columns of W into $W_1, ..., W_m$. Suppose gradient descent is run with a sufficiently small step size η so that it converges to a unique limit point K_∞. The limit point K_∞ has the following generalization error:*

$$\mathbb{E}[C(K_\infty)] = \underbrace{\frac{n}{2} + \frac{m^2 n}{2(m+1)^2} + \frac{n^2 m}{2p(m+1)^2} - \frac{m}{m+1} n}_{=:E_1} + \underbrace{\frac{\psi^2 n^2}{2}\left(\frac{m+2}{m+1} - \frac{m^2}{m+1}\frac{n}{p}\right)}_{=:E_2} .$$

Here, the expectation is over the randomness of the samples $\{W_i\}_{i=1}^{m}$ and the initalization K_0. The contribution from E_1 is due to the generalization error of the minimum-norm *stationary point of $C_m(\cdot;\{W_i\})$. The contribution from E_2 is due to the full-rank initialization of K_0.*

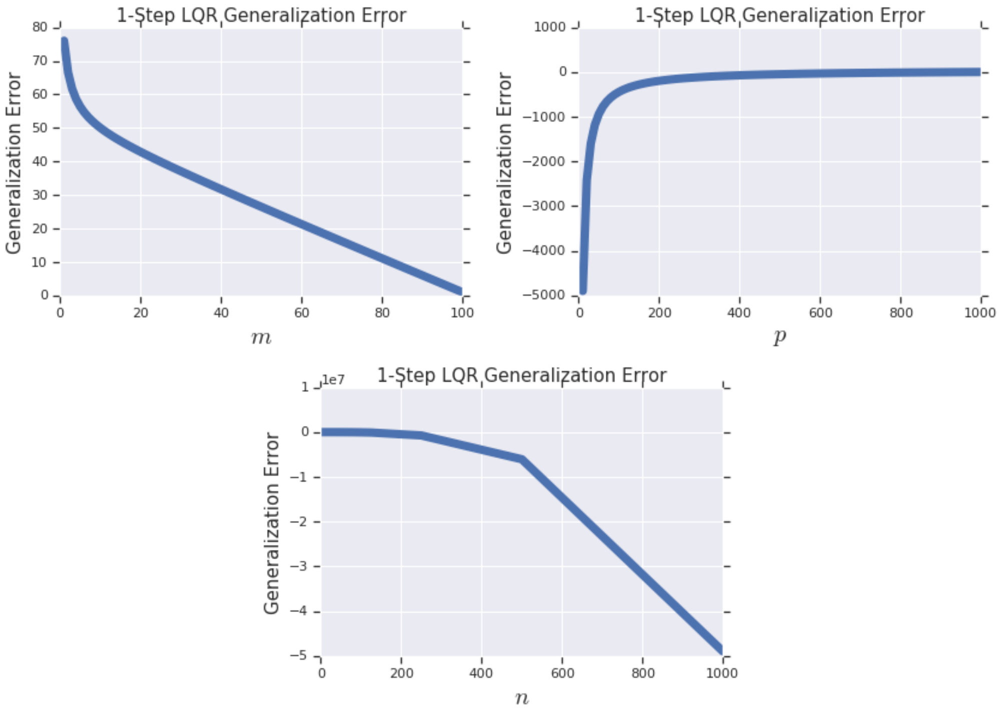


Figure A7: Plot of $\mathbb{E}[C(K_\infty)]$ as a function of m, p, n with elsewhere fixed default values $n = 10$, $p = 1000$, $m = 10$, and $\psi = 1$.

We remark that our proof specifically relies on the rank of the Hessian as m increases, rather than a more common concentration inequality used in empirical risk minimization arguments, which leads to a $\frac{1}{\sqrt{m}}$ scaling. Furthermore, the above expression for $\mathbb{E}[C(K_\infty)]$ does not scale increasingly with $poly(p)$ for the convex 1-Step LQR case, while empirically, the non-convex infinite LQR case does indeed increase from increasing the noise dimension p (as shown in Section 3.1). Interestingly, this suggests that there is an extra contribution from the non-convexity of the cost, where the observation-modified gradient dynamics tends to reach worse optima.

A.4.3.2 Proof of Theorem 1

Fix integers n, p with $p \geq n$ and suppose that n divides p. Draw a random $W \in O(p)$ uniformly from the Haar measure on $O(p)$ and divide W column-wise into $W_1, W_2, ..., W_{p/n}$ (that is $W_i \in \mathbb{R}^{p \times n}$, $W_i^\mathsf{T} W_i = I_n$ and $W_i^\mathsf{T} W_j = 0$ when $i \neq j$). Also draw a $W_c \in O(n)$ uniformly and independent from W. Now we consider the matrix $Z_m \in \mathbb{R}^{m \cdot n \times (n+p)}$ for $m = 1, ..., p/n$ defined as:

$$Z_m := \begin{bmatrix} W_c^\mathsf{T} & W_1^\mathsf{T} \\ \vdots & \vdots \\ W_c^\mathsf{T} & W_m^\mathsf{T} \end{bmatrix} .$$

Proposition 2. *We have that $Z_m^\dagger$ is given as:*

$$Z_m^\dagger = \begin{bmatrix} \frac{1}{m+1} W_c & \cdots & \frac{1}{m+1} W_c \\ \frac{m}{m+1} W_1 - \frac{1}{m+1} \sum_{i \neq 1} W_i & \cdots & \frac{m}{m+1} W_m - \frac{1}{m+1} \sum_{i \neq m} W_i \end{bmatrix} .$$

Proof. Using the fact that $W_c^\mathsf{T} W_c = I$, $W_i^\mathsf{T} W_i = I_n$, and $W_i^\mathsf{T} W_j = 0$ for $i \neq j$, we can compute:

$$Z_m Z_m^\mathsf{T} = I_{m \cdot n} + \begin{bmatrix} I_n \\ \vdots \\ I_n \end{bmatrix} \begin{bmatrix} I_n \\ \vdots \\ I_n \end{bmatrix}^\mathsf{T} .$$

By the matrix inversion formula we can compute the inverse $(Z_m Z_m^\mathsf{T})^{-1}$ as the $m \times m$ block matrix where the diagonal blocks are $\frac{m}{m+1} I$ and the off-diagonal blocks are $-\frac{1}{m+1} I$. We can now write $Z_m^\dagger$ using the formula:

$$Z_m^\dagger = Z_m^\mathsf{T} (Z_m Z_m^\mathsf{T})^{-1} .$$

The result is seen to be:

$$\begin{bmatrix} \frac{1}{m+1} W_c & \cdots & \frac{1}{i+1} W_c \\ \frac{m}{m+1} W_1 - \frac{1}{m+1} \sum_{i \neq 1} W_i & \cdots & \frac{m}{m+1} W_m - \frac{1}{m+1} \sum_{i \neq m} W_i \end{bmatrix} .$$

□

Proposition 3. *We have that:*

$$P_{Z_m^\mathsf{T}} = \begin{bmatrix} \frac{m}{m+1} W_c W_c^\mathsf{T} & \frac{1}{m+1} \sum_{i=1}^m W_c W_i^\mathsf{T} \\ \frac{1}{m+1} \sum_{i=1}^m W_i W_c^\mathsf{T} & \frac{m}{m+1} \sum_{i=1}^m W_i W_i^\mathsf{T} - \frac{1}{m+1} \sum_{i \neq j} W_i W_j^\mathsf{T} \end{bmatrix} .$$

Proof. We use the formula $P_{Z_m^\mathsf{T}} = Z_m^\mathsf{T} (Z_m Z_m^\mathsf{T})^{-1} Z_m$, combined with the calculations in the previous proposition. □

Now we consider the gradient of C_m:

$$\nabla C_m(K) = \frac{1}{m} \sum_{i=1}^m \begin{bmatrix} W_c \\ W_i \end{bmatrix}^\mathsf{T} + K \left(\frac{1}{m} \sum_{i=1}^m \begin{bmatrix} W_c \\ W_i \end{bmatrix} \begin{bmatrix} W_c \\ W_i \end{bmatrix}^\mathsf{T} \right) .$$

Setting $\nabla C_m(K) = 0$, we see that minimizers of C_m are solutions to:

$$K \left(\sum_{i=1}^m \begin{bmatrix} W_c \\ W_i \end{bmatrix} \begin{bmatrix} W_c \\ W_i \end{bmatrix}^\mathsf{T} \right) = -\sum_{i=1}^m \begin{bmatrix} W_c \\ W_i \end{bmatrix}^\mathsf{T} .$$

Using our notation above, this is the same as:

$$K Z_m^\mathsf{T} Z_m = -\begin{bmatrix} I_n & \cdots & I_n \end{bmatrix} Z_m .$$

The minimum norm solution, denoted $K_{\min}$, to this equation is:

$$K_{\min} = -\begin{bmatrix} I_n & \cdots & I_n \end{bmatrix} (Z_m^\dagger)^\mathsf{T} .$$

Next, we recall that $C(K) = \mathbb{E}[C_m(K)]$ is:

$$C(K) = \frac{n}{2} + \frac{1}{2} \operatorname{tr} \left(K^\mathsf{T} K \begin{bmatrix} I & 0 \\ 0 & \frac{n}{p} I \end{bmatrix} \right) + \operatorname{tr} \left(K \begin{bmatrix} W_c \\ 0 \end{bmatrix} \right) .$$

Our next goal is to compute $\mathbb{E}[C(K_{\min})]$. First, we observe that:

$$\begin{aligned} \operatorname{tr} \left(K_{\min} \begin{bmatrix} W_c \\ 0 \end{bmatrix} \right) &= -\left\langle Z_m^\dagger, \begin{bmatrix} W_c & \cdots & W_c \\ 0 & \cdots & 0 \end{bmatrix} \right\rangle \\ &= -\frac{m}{m+1} \operatorname{tr}(W_c^\mathsf{T} W_c) \\ &= -\frac{m}{m+1} n . \end{aligned}$$

Next, defining $B_i := \frac{m}{m+1}W_i - \frac{1}{m+1}\sum_{j\neq i} W_j$, we compute $K_{\min}^\mathsf{T} K_{\min}$ as:

$$\begin{aligned}
K_{\min}^\mathsf{T} K_{\min} &= \begin{bmatrix} \frac{1}{m+1}W_c & \cdots & \frac{1}{m+1}W_c \\ B_1 & \cdots & B_m \end{bmatrix} \begin{bmatrix} I & \cdots & I \\ \vdots & \ddots & \vdots \\ I & \cdots & I \end{bmatrix} \begin{bmatrix} \frac{1}{m+1}W_c^\mathsf{T} & B_1^\mathsf{T} \\ \vdots & \vdots \\ \frac{1}{m+1}W_c^\mathsf{T} & B_m^\mathsf{T} \end{bmatrix} \\
&= \begin{bmatrix} \frac{m^2}{(m+1)^2}I & * \\ * & (\sum_{i=1}^m B_i)(\sum_{i=1}^m B_i)^\mathsf{T} \end{bmatrix} \\
&= \begin{bmatrix} \frac{m^2}{(m+1)^2}I & * \\ * & \frac{1}{(m+1)^2}(\sum_{i=1}^m W_i)(\sum_{i=1}^m W_i)^\mathsf{T} \end{bmatrix} .
\end{aligned}$$

Therefore:

$$\operatorname{tr}\left(K_{\min}^\mathsf{T} K_{\min} \begin{bmatrix} I & 0 \\ 0 & \frac{n}{p}I \end{bmatrix}\right) = \frac{m^2}{(m+1)^2}n + \frac{n}{p(m+1)^2}(nm) .$$

Putting everything together, we have that:

$$C(K_{\min}) = \frac{n}{2} + \frac{m^2 n}{2(m+1)^2} + \frac{n^2 m}{2p(m+1)^2} - \frac{m}{m+1}n .$$

Notice that this final value is not a function of the actual realization of W.

We now consider the second source of error, which comes from the initialization K_0. Recall that each entry of K_0 is drawn iid from $\mathcal{N}(0, \psi^2)$.

Let us consider the gradient descent dynamics:

$$K_{t+1} = K_t - \eta \nabla C_m(K_t) .$$

Unrolling the dynamics under our notation:

$$K_t = K_0(I - (\eta/m)Z_m^\mathsf{T} Z_m)^t - (\eta/m)\begin{bmatrix} I_n & \ldots & I_n \end{bmatrix} Z_m \sum_{k=0}^{t-1}(I - (\eta/m)Z_m^\mathsf{T} Z_m)^k .$$

It is not hard to see that for η sufficiently small, as $t \to \infty$

$$\lim_{t\to\infty} -(\eta/m)\begin{bmatrix} I_n & \ldots & I_n \end{bmatrix} Z_m \sum_{k=0}^{t-1}(I - (\eta/m)Z_m^\mathsf{T} Z_m)^k = K_{\min} .$$

Hence:

$$K_\infty = K_0(I - (\eta/m)Z_m^\mathsf{T} Z_m)^\infty + K_{\min} .$$

Call the matrix $M_\infty := (I - (\eta/m)Z_m^\mathsf{T} Z_m)^\infty$. We observe that:

$$\begin{aligned}
C(K_\infty) = \frac{n}{2} &+ \frac{1}{2}\operatorname{tr}\left(M_\infty^\mathsf{T} K_0^\mathsf{T} K_0 M_\infty \begin{bmatrix} I & 0 \\ 0 & \frac{n}{p}I \end{bmatrix}\right) \\
&+ \frac{1}{2}\operatorname{tr}\left((M_\infty^\mathsf{T} K_0^\mathsf{T} K_{\min} + K_{\min}^\mathsf{T} K_0 M_\infty) \begin{bmatrix} I & 0 \\ 0 & \frac{n}{p}I \end{bmatrix}\right) \\
&+ \frac{1}{2}\operatorname{tr}\left(K_{\min}^\mathsf{T} K_{\min} \begin{bmatrix} I & 0 \\ 0 & \frac{n}{p}I \end{bmatrix}\right) \\
&+ \operatorname{tr}\left(K_0 M_\infty \begin{bmatrix} W_c \\ 0 \end{bmatrix}\right) \\
&+ \operatorname{tr}\left(K_{\min} M_\infty \begin{bmatrix} W_c \\ 0 \end{bmatrix}\right) .
\end{aligned}$$

Noticing that K_0 is independent of Z_m, taking expectations

$$\begin{aligned}
\mathbb{E}[C(K_\infty)] &= \mathbb{E}[C(K_{\min})] + \frac{1}{2}\mathbb{E}\operatorname{tr}\left(M_\infty^\mathsf{T} K_0^\mathsf{T} K_0 M_\infty \begin{bmatrix} I & 0 \\ 0 & \frac{n}{p}I \end{bmatrix}\right) \\
&= \mathbb{E}[C(K_{\min})] + \frac{\psi^2 n}{2}\mathbb{E}\operatorname{tr}\left(M_\infty^\mathsf{T} M_\infty \begin{bmatrix} I & 0 \\ 0 & \frac{n}{p}I \end{bmatrix}\right) .
\end{aligned}$$

Above, we use the fact that $\mathbb{E}[K_0^\mathsf{T} K_0] = \psi^2 n I_{n+p}$. Now it is not hard to see that for η sufficiently small, we have that $M_\infty = I - P_{Z_m^\mathsf{T}}$. On the other hand, using $\mathbb{E}[W_i W_j^\mathsf{T}] = 0$, we have that

$$\mathbb{E}[P_{Z_m^\mathsf{T}}] = \begin{bmatrix} \frac{m}{m+1} I & 0 \\ 0 & \frac{m^2}{m+1} \frac{n}{p} I \end{bmatrix} .$$

Therefore:

$$\mathbb{E}[M_\infty^\mathsf{T} M_\infty] = \begin{bmatrix} (1 - m/(m+1))I & 0 \\ 0 & (1 - m^2 n/((m+1)p)I \end{bmatrix} .$$

Plugging in to the previous calculations, this yields

$$\mathbb{E}[C(K_\infty)] = \mathbb{E}[C(K_{\min})] + \frac{\psi^2 n^2}{2} \left(\frac{m+2}{m+1} - \frac{m^2}{m+1} \frac{n}{p} \right) .$$