Journal of Machine Learning Research 17 (2016) 1-51 Submitted 8/15; Revised 11/16; Published 12/16

Approximate Newton Methods for Policy Search in Markov Decision Processes

Thomas Furmston T.Furmston@cs.ucl.ac.uk
Department of Computer Science
University College London
London, WC1E 6BT

Guy Lever G.Lever@cs.ucl.ac.uk
Department of Computer Science
University College London
London, WC1E 6BT

David Barber D.Barber@cs.ucl.ac.uk
Department of Computer Science
University College London
London, WC1E 6BT

Editor: Joelle Pineau

Abstract

Approximate Newton methods are standard optimization tools which aim to maintain the benefits of Newton's method, such as a fast rate of convergence, while alleviating its drawbacks, such as computationally expensive calculation or estimation of the inverse Hessian. In this work we investigate approximate Newton methods for policy optimization in Markov decision processes (MDPs). We first analyse the structure of the Hessian of the total expected reward, which is a standard objective function for MDPs. We show that, like the gradient, the Hessian exhibits useful structure in the context of MDPs and we use this analysis to motivate two Gauss-Newton methods for MDPs. Like the Gauss-Newton method for non-linear least squares, these methods drop certain terms in the Hessian. The approximate Hessians possess desirable properties, such as negative definiteness, and we demonstrate several important performance guarantees including guaranteed ascent directions, invariance to affine transformation of the parameter space and convergence guarantees. We finally provide a unifying perspective of key policy search algorithms, demonstrating that our second Gauss-Newton algorithm is closely related to both the EM-algorithm and natural gradient ascent applied to MDPs, but performs significantly better in practice on a range of challenging domains.

Keywords: Markov decision processes, reinforcement learning, Newton method, function approximation

1. Introduction

Markov decision processes (MDPs) are the standard model for optimal control in a fully observable environment (Bertsekas, 2010). Strong empirical results have been obtained in numerous challenging real-world optimal control problems using the MDP framework. This includes problems of non-linear control (Stengel, 1993; Li and Todorov, 2004; Todorov and

Tassa, 2009; Deisenroth and Rasmussen, 2011; Rawlik et al., 2012; Spall and Cristion, 1998; Levine and Koltun, 2013b; Schulman et al., 2015; Heess et al., 2015; Lillicrap et al., 2016), robotic applications (Kober and Peters, 2011; Kohl and Stone, 2004; Vlassis et al., 2009), biological movement systems (Li, 2006), traffic management (Richter et al., 2007; Srinivasan et al., 2006), helicopter flight control (Abbeel et al., 2007), elevator scheduling (Crites and Barto, 1995) and numerous games, including chess (Veness et al., 2009), go (Silver et al., 2016; Gelly and Silver, 2008), backgammon (Tesauro, 1994) and Atari 2600 video games (Mnih et al., 2015; Schulman et al., 2015).

It is well known that the global optimum of a MDP with finite state and action sets can be obtained through methods based on dynamic programming, such as value iteration (Bellman, 1957) and policy iteration (Howard, 1960). However, these techniques are known to suffer from the curse of dimensionality, which makes them infeasible for many real-world problems of interest. As a result, most research in the reinforcement learning and control theory literature has focused on obtaining approximate or locally optimal solutions. There exists a broad spectrum of such techniques, including approximate dynamic programming methods (Bertsekas, 2010; Mnih et al., 2015), tree search methods (Russell and Norvig, 2009; Kocsis and Szepesvári, 2006; Browne et al., 2012; Silver et al., 2016), local trajectory-optimization techniques, such as differential dynamic programming (Jacobson and Mayne, 1970) and iLQG (Li and Todorov, 2006), and policy search methods (Williams, 1992; Baxter and Bartlett, 2001; Sutton et al., 2000; Marbach and Tsitsiklis, 2001; Kakade, 2002; Kober and Peters, 2011; Levine and Koltun, 2013b; Silver et al., 2014; Schulman et al., 2015; Heess et al., 2015; Lillicrap et al., 2016).

The focus of this paper is on policy search methods, which are a family of algorithms that have proven extremely popular in recent years, and which have numerous desirable properties that make them attractive in practice. Policy search algorithms are typically specialized applications of techniques from numerical optimization (Nocedal and Wright, 2006; Dempster et al., 1977). As such, the controller is given a differentiable parameterisation and an objective function is defined in terms of this parameterisation. Local information about the objective function, such as the gradient, is then used to update the parameters of the controller in an incremental manner until the algorithm converges to a local optimum of the objective function. There are several benefits to such an approach: the smooth updates of the control parameters endow these algorithms with very general convergence guarantees; as performance is improved at each iteration (or at least on average in stochastic policy search methods) these algorithms have good anytime performance properties; it is not necessary to approximate the value function, which is typically a difficult function to approximate—instead it is only necessary to approximate a low-dimensional projection of the value function, an observation which has led to the emergence of so called *actor-critic* methods (Konda and Tsitsiklis, 2003, 1999; Bhatnagar et al., 2008, 2009); policy search methods are easily extendable to models for optimal control in a partially observable environment, such as the Finite State Controllers (Meuleau et al., 1999; Toussaint et al., 2006).

In (stochastic) gradient ascent (Williams, 1992; Baxter and Bartlett, 2001; Sutton et al., 2000) the control parameters are updated by moving in the direction of the gradient of an objective function. While gradient ascent has enjoyed some success, it suffers from serious issues that can hinder its performance: specifically, it is not scale invariant (Nocedal and

Wright, 2006) and the search direction is often poorly-scaled, i.e., the variation of the objective function differs dramatically along the different components of the gradient. Poor scaling of the gradient leads to a poor rate of convergence (Nocedal and Wright, 2006). It also makes the construction of a good step size sequence a difficult problem, which is an important issue in stochastic methods.[1] Poor scaling is a well known problem with gradient ascent and alternative numerical optimization techniques have been considered in the policy search literature. Two approaches that have proven to be particularly popular are expectation maximization (Dempster et al., 1977) and natural gradient ascent (Amari, 1997, 1998; Amari et al., 1992), which have both been successfully applied to various challenging MDPs (see the works of Dayan and Hinton (1997); Kober and Peters (2009); Toussaint et al. (2011); Levine and Koltun (2013a) and Kakade (2002); Bagnell and Schneider (2003) respectively).

An avenue of research that has received less attention is the application of Newton's method to Markov decision processes. Although such an extension of the *GPOMDP* algorithm is provided in the work of Baxter and Bartlett (2001), they give no empirical results in either that article or the accompanying paper of empirical comparisons (Baxter et al., 2001). There has since been only a limited amount of research into using the second order information contained in the Hessian during the parameter update. To the best of our knowledge only two attempts have been made: in the work of Schraudolph et al. (2006) an on-line estimate of a Hessian-vector product is used to adapt the step size sequence in an on-line manner; in the work of Ngo et al. (2011), Bayesian policy gradient methods (Ghavamzadeh and Engel, 2007) are extended to Newton's method. There are several reasons for this lack of interest. Firstly, in many problems the construction and inversion of the Hessian is too computationally expensive to be feasible. Additionally, the objective function of a MDP is typically not concave, and so the Hessian is not guaranteed to be negative-definite. As a result, the search direction of Newton's method may not be an ascent direction, and hence a parameter update could actually lower the objective. Additionally, the variance of sample-based estimators of the Hessian will be larger than that of estimators of the gradient. This is an important point because the variance of gradient estimates can be a problematic issue and various methods, such as baselines (Weaver and Tao, 2001; Greensmith et al., 2004), exist to reduce the variance.

Many of these problems are not particular to Markov decision processes, but are general longstanding issues that plague Newton's method. Various methods have been developed in the optimization literature to alleviate these issues, while also maintaining desirable properties of Newton's method. For instance, quasi-Newton methods were designed to efficiently mimic Newton's method using only evaluations of the gradient obtained during previous iterations of the algorithm. These methods have low computational costs, a super-linear rate of convergence and have proven to be extremely effective in practice. See the work of Nocedal and Wright (2006) for an introduction to quasi-Newton methods. Alternatively, the well-known Gauss-Newton method is a popular approach that aims to efficiently mimic Newton's method. The Gauss-Newton method is particular to non-linear least squares objective functions, for which the Hessian has a particular structure. Due to this structure there exist certain terms in the Hessian that can be used as a useful proxy for the Hessian

1. This is because line search techniques lose much of their desirability in stochastic numerical optimization algorithms, due to variance in the evaluations.

itself, with the resulting algorithm having various desirable properties. For instance, the preconditioning matrix used in the Gauss-Newton method is guaranteed to be positive-semidefinite, so that the non-linear least squares objective is guaranteed not to increase for a sufficiently small step size.

While a straightforward application of quasi-Newton methods will not typically be possible for MDPs,[2] in this paper we consider whether an analogue to the Gauss-Newton method exists, so that the benefits of such methods can be applied to MDPs. The specific contributions are as follows:

- In Section 3, we present an analysis of the Hessian of the total expected reward, which is a standard objective function for MDPs. Our starting point is a derivation of the Hessian for the total expected reward (Theorem 3) and we analyse the behavior of individual terms of the Hessian to provide insight into constructing efficient approximate Newton methods for policy optimization. In particular, we provide conditions under which certain terms become negligible near local optima.

- Motivated by this analysis, in Section 4 we provide two Gauss-Newton type methods for policy optimization in MDPs. These methods retain certain terms of our Hessian decomposition in the preconditioner in a gradient-based policy search algorithm. The first method discards terms which are difficult to approximate and which, for appropriately selected classes of controller, will become negligible near local optima. The second method further discards an additional term which is not guaranteed to be negative semi-definite. We provide an analysis of our Gauss-Newton methods and give several important performance guarantees for the second Gauss-Newton method: We demonstrate that the preconditioning matrix is negative-semidefinite when the controller is log-concave in the control parameters (detailing some widely used controllers for which this condition holds) guaranteeing that the search direction is an ascent direction; We show that the method is invariant to affine transformations of the parameter space and thus does not suffer the significant drawback of gradient ascent. We provide a convergence analysis, demonstrating linear convergence to local optima, in terms of the step size of the update.

  Our methods apply to finite and continuous state and action sets. For simplicity of exposition we have presented results for the finite case to avoid measurability considerations. Some of our experiments have continuous state and action sets. Similarly our method is suitable for unknown transition dynamics, but can also be trivially used in a model-based approach with a known or estimated dynamics model.

- In Section 5 we present a unifying perspective for several policy search methods. In particular we relate the search direction of our second Gauss-Newton algorithm to that of expectation maximization (which provides new insights into the latter algorithm when used for policy search), and we also discuss its relationship to the natural gradient algorithm.

2. In quasi-Newton methods, to ensure an increase in the objective function, it is necessary to satisfy the secant condition (Nocedal and Wright, 2006). This condition is satisfied when the objective is concave/convex or the strong Wolfe conditions are met during a line search. For this reason, stochastic applications of quasi-Newton methods has been restricted to convex/concave objective functions (Schraudolph et al., 2007).

- In Section 6 we present experiments demonstrating state-of-the-art performance on challenging domains including Tetris and robotic arm applications.

2. Preliminaries and Background

In Section 2.1 we introduce Markov decision processes, along with some standard terminology relating to these models that will be required throughout the paper. In Section 2.2 we introduce policy search methods and provide an overview of the literature.

2.1 Markov Decision Processes

In a Markov decision process an *agent*, or *controller*, sequentially interacts with an environment by selecting actions (based on the the current state of the environment) after which the system transitions to a new state (often in a stochastic manner) and the agent receives a scalar reward (dependent upon the selected action and the state of the environment). The optimality of an agent's behavior is measured in terms of the total (discounted) reward the agent can expect to receive, so that optimal control is obtained when this quantity is maximized.

Formally a MDP is described by the tuple $(\mathcal{S}, \mathcal{A}, D, P, R)$, in which $\mathcal{S}$ and $\mathcal{A}$ are sets, known respectively as the state and action space, D is the initial state distribution, which is a distribution over the state space, P encodes the transition dynamics using a set of conditional distributions over the state space, $\{P(\cdot|s,a)\}_{(s,a)\in\mathcal{S}\times\mathcal{A}}$, and $R : \mathcal{S} \times \mathcal{A} \to [0, R_{\max}]$ is the (possibly stochastic[3]) reward function, which is assumed to be bounded and non-negative.[4] We use the notation s_t and a_t to denote the random variable of the state and action of the t^{th} time step, $t \in \mathbb{N}$, respectively. The state at the initial time step is determined by the initial state distribution, $s_1 \sim D(\cdot)$. At any given time step, $t \in \mathbb{N}$, and given the state of the environment, the agent selects an action, $a_t \sim \pi(\cdot|s_t)$, according to the *policy* π. The next state is determined according to the transition dynamics, $s_{t+1} \sim P(\cdot|a_t, s_t)$. At each time step the agent receives a scalar reward, which is determined by the reward function.

In this paper we consider the total expected reward, which is a standard objective function in the reinforcement learning literature (Bertsekas, 2010). We shall consider the infinite horizon discounted reward framework. In this framework there is a discount factor, $\gamma \in [0, 1)$, and the objective function takes the form,

$$U(\pi) := \sum_{t=1}^{\infty} \mathbb{E}_{s_t, a_t \sim p_t} \Big[\gamma^{t-1} R(s_t, a_t); \pi \Big]. \tag{1}$$

3. For notational convenience we shall consider a deterministic reward function in this paper. The extension to the case of a stochastic reward function can be done by considering the expectation over the reward function where appropriate.
4. Given either an episodic finite horizon or a discounted infinite horizon MDP in which the reward function is bounded but not necessarily non-negative, one can construct an auxiliary MDP that has a bounded non-negative reward function and has the same optimal policies as the original MDP. This can be achieved, for example, by setting the reward function of the auxiliary MDP as $R_{\text{auxiliary}}(s,a) = R(s,a) - min_{(s,a)\in\mathcal{S}\times\mathcal{A}} R(s,a)$, which only requires knowledge of the lower bound of the reward function in the original MDP. The same is not true, however, of absorbing state MDPs with a discount factor of 1.

We use the semi-colon to identify parameters of the distribution, rather than conditioning variables. The distribution of s_t and a_t, which we denote by p_t, is given by the marginal at time t of the joint distribution over $(s_{1:t}, a_{1:t})$, with $s_{1:t} = (s_1, s_2, ..., s_t)$, $a_{1:t} = (a_1, a_2, ..., a_t)$, which is given by,

$$p(s_{1:t}, a_{1:t}; \pi) := \pi(a_t|s_t)\Big\{\prod_{\tau=1}^{t-1} P(s_{\tau+1}|s_\tau, a_\tau) \times \pi(a_\tau|s_\tau)\Big\}D(s_1). \tag{2}$$

We use the notation $\xi_t := (s_1, a_1, s_2, a_2, ..., s_t, a_t)$ to denote trajectories through the state-action space of length, $t \in \mathbb{N}$. We use ξ to denote trajectories that are of infinite length, and use Ξ to denote the space of all such trajectories. Given a trajectory, $\xi \in \Xi$, we use the notation $\bar{R}(\xi)$ to denote the total discounted reward of the trajectory, so that $\bar{R}(\xi) = \sum_{t=1}^{\infty} \gamma^{t-1} R(s_t, a_t)$. Similarly, we use the notation $p(\xi; \pi)$ to denote the probability of generating the trajectory ξ under the policy π.

We now introduce several functions that are of central importance. The state value function w.r.t. policy π is defined as the total expected future reward given the current state,

$$V_\pi(s) := \sum_{t=1}^{\infty} \mathbb{E}_{s_t, a_t \sim p_t}\Big[\gamma^{t-1} R(s_t, a_t)\big|s_1 = s; \pi\Big].$$

It can be seen that, $U(\pi) = \mathbb{E}_{s\sim D}[V_\pi(s)]$. The state value function can also be written as the solution of the following fixed-point equation, $V_\pi(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\Big[R(s,a) + \gamma \mathbb{E}_{s'\sim P(\cdot|s,a)}\big[V_\pi(s')\big]\Big]$, which is known as the Bellman equation (Bertsekas, 2010). The state-action value function w.r.t. policy π is given by,

$$Q_\pi(s, a) := R(s, a) + \gamma \mathbb{E}_{s'\sim P(\cdot|s,a)}\Big[V_\pi(s')\Big], \tag{3}$$

and gives the value of performing an action, in a given state, and then following the policy. Note that, $V_\pi(s) = \sum_{a\in\mathcal{A}} \pi(a|s) Q_\pi(s, a)$. Finally, the advantage function, $A_\pi(s, a) := Q_\pi(s, a) - V_\pi(s)$, gives the relative advantage of an action in relation to the other actions available in that state. It can be seen that, $\sum_{a\in\mathcal{A}} \pi(a|s) A_\pi(s, a) = 0$, for each $s \in \mathcal{S}$.

2.2 Policy Search Methods

In policy search methods the policy is given some differentiable parametric form, denoted $\pi(a|s; \boldsymbol{w})$, with policy parameters, $\boldsymbol{w} \in \mathcal{W} \subset \mathbb{R}^n$, $n \in \mathbb{N}$. (We also use the notation, $\pi_{\boldsymbol{w}}(a|s) \equiv \pi(a|s; \boldsymbol{w})$, where appropriate.) Local information, such as the gradient of the objective function, is then used to update the policy in an incremental manner until the algorithm converges to a local optimum of the objective function. We overload notation and write the objective function directly in terms of the parameter vector, i.e.,

$$U(\boldsymbol{w}) := U(\pi_{\boldsymbol{w}}), \qquad \forall \boldsymbol{w} \in \mathcal{W}, \tag{4}$$

while the trajectory distribution is written in the form $p(a_{1:t}, s_{1:t}; \boldsymbol{w}) = p(a_{1:t}, s_{1:t}; \pi_{\boldsymbol{w}})$. Similarly, $V(s; \boldsymbol{w})$, $Q(s, a; \boldsymbol{w})$ and $A(s, a; \boldsymbol{w})$ denote respectively the state value function,

state-action value function and the advantage function in terms of the parameter vector $\boldsymbol{w}$. We introduce the function,

$$p_\gamma(s, a; \boldsymbol{w}) := \sum_{t=1}^{\infty} \gamma^{t-1} p_t(s_t = s, a_t = a; \boldsymbol{w}). \tag{5}$$

Note that $\sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} p_\gamma(s, a; \boldsymbol{w}) = (1-\gamma)^{-1}$. Additionally, the objective function can be written in the form $U(\boldsymbol{w}) = \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} p_\gamma(s, a; \boldsymbol{w}) R(s, a)$.

We shall consider two forms of policy search algorithm in this paper: gradient-based optimization methods and methods based on iteratively optimizing a lower-bound on the objective function. In gradient-based methods the update of the policy parameters takes the form,

$$\boldsymbol{w}^{\text{new}} = \boldsymbol{w} + \alpha \mathcal{M}(\boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w}), \tag{6}$$

where $\alpha \in \mathbb{R}^+$ is a step size parameter and $\mathcal{M}(\boldsymbol{w})$ is some preconditioning matrix that possibly depends on $\boldsymbol{w} \in \mathcal{W}$. If U is smooth, $\mathcal{M}(\boldsymbol{w})$ is positive-definite and α is sufficiently small then such an update will increase the total expected reward. If the preconditioning matrix is always positive-definite, the step size sequence is appropriately selected and U is Lipschitz, which is the case when $\|\frac{\partial}{\partial \boldsymbol{w}} \log \pi(a|s; \boldsymbol{w})\|_2$ is uniformly bounded by $M \in \mathbb{R}$ for all $\boldsymbol{w} \in \mathcal{W}$ and $(a, s) \in \mathcal{A} \times \mathcal{S}$, then iteratively updating the policy parameters according to (6) will result in the policy parameters converging to a local optimum of U. This generic gradient-based policy search algorithm is given in Algorithm 1. Gradient-based methods vary in the form of the preconditioning matrix used in the parameter update. The choice of the preconditioning matrix determines various aspects of the resulting algorithm, such as the computational complexity, the rate at which the algorithm converges to a local optimum and invariance properties of the parameter update. Typically the gradient, $\frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w})$, and the preconditioner, $\mathcal{M}(\boldsymbol{w})$, will not be known exactly and must be approximated by collecting data from the system. In the context of reinforcement learning, the expectation maximization (EM) algorithm searches for the optimal policy by iteratively optimizing a lower bound on the objective function. While EM does not have an update of the form given in (6) we shall see in Section 5.2 that the algorithm is closely related to such an update. We now review specific policy search methods.

2.2.1 Gradient Ascent

Gradient ascent corresponds to the choice $\mathcal{M}(\boldsymbol{w}) = I_n$, where I_n denotes the $n \times n$ identity matrix, so that the parameter update takes the form:

Policy search update using gradient ascent

$$\boldsymbol{w}^{\text{new}} = \boldsymbol{w} + \alpha \frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w}). \tag{7}$$

The gradient, $\frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w})$, can be written in a relatively simple form using the following theorem:

Algorithm 1: Generic gradient-based policy search algorithm

Input: Initial vector of policy parameters, $\boldsymbol{w}_0 \in \mathcal{W}$, and a step size sequence, $(\alpha_k)_{k=0}^{\infty}$, with $\alpha_k \in \mathbb{R}^+$ for $k \in \mathbb{N}$.

Set iteration counter, $k \leftarrow 0$.

repeat

Either calculate or estimate the gradient of the objective, $\frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w})|_{w=w_k}$, and the preconditioner, $\mathcal{M}(\boldsymbol{w}_k)$, at the current point in the parameter space.

Update policy parameters, $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + \alpha_k \mathcal{M}(\boldsymbol{w}_k) \frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w})|_{w=w_k}$.

Update iteration counter, $k \leftarrow k + 1$.

until Convergence of the policy parameters;

return $\boldsymbol{w}_k$

Theorem 1 (Policy Gradient Theorem (Sutton et al., 2000))**.** *Suppose we are given a Markov decision process with objective (1) and Markovian trajectory distribution (2). For any given parameter vector, $\boldsymbol{w} \in \mathcal{W}$, the gradient of (4) takes the form,*

$$\frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w}) = \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} p_\gamma(s, a; \boldsymbol{w}) Q(s, a; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a|s; \boldsymbol{w}). \tag{8}$$

Proof. This is a well-known result that can be found in Sutton et al. (2000). A derivation of (8) is provided in Section A.1 in the Appendix. □

It is not possible to calculate the gradient exactly for many real-world MDPs of interest. For instance, in discrete domains the size of the state-action space may be too large for enumeration over these sets to be feasible. Alternatively, in continuous domains the presence of non-linearities in the transition dynamics makes the calculation of the occupancy marginals an intractable problem. Additionally, it can be the case that P and R are unknown in practice. In such cases it can be preferable to directly estimate the gradient using samples obtained from the environment, rather than building a model of the MDP. Various techniques have been proposed in the literature to estimate the gradient, including the method of finite-differences (Kiefer and Wolfowitz, 1952; Kohl and Stone, 2004; Tedrake and Zhang, 2005), simultaneous perturbation methods (Spall, 1992; Spall and Cristion, 1998; Srinivasan et al., 2006) and likelihood-ratio methods (Glynn, 1986, 1990; Williams, 1992; Baxter and Bartlett, 2001; Konda and Tsitsiklis, 2003, 1999; Sutton et al., 2000; Bhatnagar et al., 2009; Kober and Peters, 2011). Likelihood-ratio methods, which originated in the statistics literature and were later applied to MDPs, are now the prominent method for estimating the gradient. There are numerous such methods in the literature, including Monte-Carlo methods (Williams, 1992; Baxter and Bartlett, 2001) and actor-critic methods (Konda and Tsitsiklis, 2003, 1999; Sutton et al., 2000; Bhatnagar et al., 2009; Kober and Peters, 2011).

Gradient ascent is known to perform poorly on objective functions that are poorly-scaled, that is, if changes to some parameters produce much larger variations to the function than changes in other parameters. In this case gradient ascent zig-zags along the ridges of the

objective in the parameter space (see e.g., the work of Nocedal and Wright, 2006). It can be difficult to gauge an appropriate scale for the steps sizes in poorly-scaled problems and the robustness of optimization algorithms to poor scaling is of significant practical importance.

2.2.2 Natural Gradient Ascent

Natural gradient ascent techniques originated in the neural network and blind source separation literature (Amari, 1997, 1998; Amari et al., 1996, 1992), and were introduced into the policy search literature in the work of Kakade (2002). To address the issue of poor scaling, natural gradient methods take the perspective that the parameter space should be viewed with a manifold structure in which the distance between two points on the manifold captures the discrepancy between the distribution over trajectories parameterized by the two corresponding parameter vectors. In natural gradient ascent $\mathcal{M}(\boldsymbol{w}) = G^{-1}(\boldsymbol{w})$ in (6), with $G(\boldsymbol{w})$ a suitable metric tensor for the manifold, so that the parameter update takes the form:

Policy search update using natural gradient ascent

$$\boldsymbol{w}^{\text{new}} = \boldsymbol{w} + \alpha G^{-1}(\boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w}). \tag{9}$$

In the case of Markov decision processes a standard choice for $G(\boldsymbol{w})$ is the Fisher information matrix of the policy distribution, averaged over the state distribution, which takes the form,

$$G(\boldsymbol{w}) = \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} p_\gamma(s, a; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a|s; \boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} \log \pi(a|s; \boldsymbol{w}). \tag{10}$$

It was shown in the work of Bagnell and Schneider (2003) that (10) corresponds to the Fisher information matrix of the distribution over trajectories in the state-action space, given π. The use of the Fisher information matrix can be viewed as imposing a local norm on the parameter space which is derived from a second order approximation to the KL-divergence between induced trajectory distributions (Coolen et al., 2005). For brevity we refer to this choice of $G(\boldsymbol{w})$ as *the* natural gradient algorithm. When the policy distribution satisfies the Fisher regularity conditions (see Lehmann and Casella, 1998, Lemma 5.3) there is an alternate, equivalent, form of the Fisher information matrix given by,

$$G(\boldsymbol{w}) = -\sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} p_\gamma(s, a; \boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a|s; \boldsymbol{w}). \tag{11}$$

Equation (11) can be derived by differentiating the identity, $\sum_{a \in \mathcal{A}} \pi(a|s; \boldsymbol{w}) \frac{\partial}{\partial w_i} \log \pi(a|s; \boldsymbol{w}) = \sum_{a \in \mathcal{A}} \frac{\partial}{\partial w_i} \pi(a|s; \boldsymbol{w}) = 0$, with respect to w_j, and taking expectation over $s \in \mathcal{S}$. Other metric tensors have also been considered in the policy search literature, such as in the work of Morimura et al. (2008).

There are several desirable properties of the natural gradient approach: the Fisher information matrix is always positive-semidefinite, regardless of the policy parameterisation;

the search direction is invariant to the parameterisation of the policy, (Bagnell and Schneider, 2003; Peters and Schaal, 2008). Additionally, the natural gradient update direction can be obtained by regressing the state-action value function, or the advantage function, using a compatible function approximator (Sutton et al., 2000), and minimizing a square loss weighted by the (discounted) trajectory distribution (Kakade, 2002). Furthermore, natural gradient ascent has been shown to perform well in some difficult MDP environments, including Tetris (Kakade, 2002) and several challenging robotics problems (Peters and Schaal, 2008). However, theoretically, the rate of convergence of natural gradient ascent is the same as gradient ascent, i.e., linear, although, it has been noted to be substantially faster in practice.

2.2.3 Expectation Maximization

An alternative optimization procedure that has been the focus of much research in the planning and reinforcement learning communities is the EM-algorithm (Dayan and Hinton, 1997; Toussaint et al., 2006, 2011; Kober and Peters, 2009, 2011; Hoffman et al., 2009; Furmston and Barber, 2009, 2010; Levine and Koltun, 2013a). The EM-algorithm is a powerful optimization technique popular in the statistics and machine learning community (see e.g., the works of Dempster et al., 1977; Little and Rubin, 2002; Neal and Hinton, 1999) that has been successfully applied to a large number of problems. See the work of Barber (2012) for a general overview of some of the applications of the algorithm in the machine learning literature. Among the strengths of the algorithm are its guarantee of increasing the objective function at each iteration, its often simple update equations, and its generalization to highly intractable models through variational Bayes approximations (Saul et al., 1996).

Given the advantages of the EM-algorithm it is natural to extend the algorithm to the MDP framework. Several derivations of the EM-algorithm for MDPs exist (Kober and Peters, 2011; Toussaint et al., 2011). For reference, we state the lower-bound upon which the algorithm is based in the following theorem. The proof is based on an application of Jensen's inequality and can be found in the work of Kober and Peters (2011).

Theorem 2. *Suppose we are given a Markov decision process with objective (1) and Markovian trajectory distribution (2). Given any distribution, q, over the space of trajectories, Ξ, then the following bound holds,*

$$\log U(\boldsymbol{w}) \geq H(q(\xi)) + \mathbb{E}_{\xi \sim q(\cdot)}\Big[\log\big(p(\xi;\boldsymbol{w})\bar{R}(\xi)\big)\Big], \qquad \forall \boldsymbol{w} \in \mathcal{W}. \tag{12}$$

The distribution, q, in Theorem 2 is often referred to as the variational distribution. An EM-algorithm is obtained through coordinate-wise optimization of (12) with respect to the variational distribution (the E-step) and the policy parameters (the M-step). In the E-step the lower-bound is optimized when $q(\xi) \propto p(\xi;\boldsymbol{w}')\bar{R}(\xi)$, in which $\boldsymbol{w}'$ are the current policy parameters. In the M-step the lower-bound is optimized with respect to $\boldsymbol{w}$, which, given $q(\xi) \propto p(\xi;\boldsymbol{w}')\bar{R}(\xi)$ and the Markovian structure of $\log p(\xi;\boldsymbol{w})$, is equivalent to optimizing the function,

$$\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}') = \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} p_\gamma(s,a;\boldsymbol{w}')Q(s,a;\boldsymbol{w}')\log\pi(a|s;\boldsymbol{w}), \tag{13}$$

with respect to the first parameter, $\boldsymbol{w}$. The E-step and M-step are iterated in this manner until the policy parameters converge to a local optimum of the objective function.

Alternative bound optimisation techniques exist in the policy search literature. For instance, the trust region policy optimisation algorithm (Schulman et al., 2015) is based on a generalisation of a lower-bound of the total expected reward given in the work of Kakade and Langford (2002). While the lower-bound is intractable in large-scale MDPs, the introduction of several heuristics enables the authors to obtain strong empirical results in both non-linear control problems and Atari games.

3. The Hessian of the Total Expected Discounted Return

In this section we provide an analysis of the Hessian of the total expected reward of a MDP. This analysis will then be used in Section 4 to propose Gauss-Newton type methods for MDPs. In Section 3.1 we provide a novel representation of the Hessian of the total expected reward, in Section 3.2 we detail the definiteness properties of certain terms in the Hessian and in Section 3.3 we analyse the behaviour of individual terms of the Hessian in the vicinity of a local optimum.

3.1 The Policy Hessian Theorem

There is a standard expansion of the Hessian of the total expected reward in the policy search literature (Baxter and Bartlett, 2001; Kakade, 2001, 2002) that, as with the gradient, takes a relatively simple form. This is summarized in the following result.

Theorem 3 (Policy Hessian Theorem)**.** *Suppose we are given a Markov decision process with objective (1) and Markovian trajectory distribution (2). For any given parameter vector, $\boldsymbol{w} \in \mathcal{W}$, the Hessian of (4) takes the form,*

$$\mathcal{H}(\boldsymbol{w}) = \mathcal{H}_1(\boldsymbol{w}) + \mathcal{H}_2(\boldsymbol{w}) + \mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w}), \tag{14}$$

in which the matrices $\mathcal{H}_1(\boldsymbol{w})$, $\mathcal{H}_2(\boldsymbol{w})$ and $\mathcal{H}_{12}(\boldsymbol{w})$ can be written in the form,

$$\mathcal{H}_1(\boldsymbol{w}) := \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} p_\gamma(s,a;\boldsymbol{w}) Q(s,a;\boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a|s;\boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} \log \pi(a|s;\boldsymbol{w}), \tag{15}$$

$$\mathcal{H}_2(\boldsymbol{w}) := \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} p_\gamma(s,a;\boldsymbol{w}) Q(s,a;\boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a|s;\boldsymbol{w}), \tag{16}$$

$$\mathcal{H}_{12}(\boldsymbol{w}) := \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} p_\gamma(s,a;\boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a|s;\boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} Q(s,a;\boldsymbol{w}). \tag{17}$$

Proof. A derivation for a sample-based estimator of the Hessian can be found in Baxter and Bartlett (2001). For ease of reference a derivation of (14) is provided in Section A.1 in the Appendix. □

We remark that $\mathcal{H}_1(\boldsymbol{w})$ and $\mathcal{H}_2(\boldsymbol{w})$ are relatively simple to estimate, in the same manner as estimating the policy gradient. The term $\mathcal{H}_{12}(\boldsymbol{w})$ is more difficult to estimate since it

contains terms involving the gradient, $\frac{\partial^\top}{\partial \boldsymbol{w}} Q(s,a;\boldsymbol{w})$, resulting in a double sum over state-actions.

Below we will present a novel form for the Hessian of the total expected reward, with attention given to the term $\mathcal{H}_1(\boldsymbol{w}) + \mathcal{H}_2(\boldsymbol{w})$ in (14), which will require the following notion of a parameterisation with constant curvature.

Definition 1. *A policy parameterisation is said to have constant curvature with respect to the action space, if for each $(s,a) \in \mathcal{S} \times \mathcal{A}$ the Hessian of the log-policy, $\frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a|s;\boldsymbol{w})$, does not depend upon the action, i.e.,*

$$\frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a|s;\boldsymbol{w}) = \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a'|s;\boldsymbol{w}), \qquad \forall a, a' \in \mathcal{A}.$$

A common class of policy which satisfies the property of Definition 1 is, $\pi(a|s;\boldsymbol{w}) \propto \exp(\boldsymbol{w}^\top \boldsymbol{\phi}(a,s))$, in which $\boldsymbol{\phi}(a,s)$ is a vector of features that depends on the state-action pair, $(a,s) \in \mathcal{A} \times \mathcal{S}$. Under this parameterisation,

$$\frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a|s;\boldsymbol{w}) = -\mathrm{Cov}_{a' \sim \pi(\cdot|s;\boldsymbol{w})}\big(\boldsymbol{\phi}(a',s), \boldsymbol{\phi}(a',s)\big),$$

which does not depend on, $a \in \mathcal{A}$. In the case when the action space is continuous, then the policy parameterisation $\pi(a|s;\boldsymbol{w};\Sigma) \propto \exp\big(-\frac{1}{2}(a - \boldsymbol{w}^\top \boldsymbol{\phi}(s))^\top \Sigma^{-1}(a - \boldsymbol{w}^\top \boldsymbol{\phi}(s))\big)$, in which $\boldsymbol{\phi} : \mathcal{S} \to \mathbb{R}^n$ is a given feature map, satisfies the properties of Definition 1 with respect to the mean parameters, $\boldsymbol{w} \in \mathcal{W}$.

We now present a novel decomposition of the Hessian for Markov decision processes.

Theorem 4. *Suppose we are given a Markov decision process with objective (1) and Markovian trajectory distribution (2). For any given parameter vector, $\boldsymbol{w} \in \mathcal{W}$, the Hessian of (4) takes the form,*

$$\mathcal{H}(\boldsymbol{w}) = \mathcal{A}_1(\boldsymbol{w}) + \mathcal{A}_2(\boldsymbol{w}) + \mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w}), \tag{18}$$

with,

$$\mathcal{A}_1(\boldsymbol{w}) := \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} p_\gamma(s,a;\boldsymbol{w}) A(s,a;\boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a|s;\boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} \log \pi(a|s;\boldsymbol{w}),$$

$$\mathcal{A}_2(\boldsymbol{w}) := \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} p_\gamma(s,a;\boldsymbol{w}) A(s,a;\boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a|s;\boldsymbol{w}).$$

When the policy parameterization has constant curvature with respect to the action space, then the Hessian takes the form,

$$\mathcal{H}(\boldsymbol{w}) = \mathcal{A}_1(\boldsymbol{w}) + \mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w}). \tag{19}$$

Proof. See Section A.2 in the Appendix. □

We now present an analysis of the terms of the policy Hessian, simplifying the expansion and demonstrating conditions under which certain terms disappear. The analysis will be used to motivate our Gauss-Newton methods in Section 4.

3.2 Analysis of the Policy Hessian

An interesting comparison can be made between the expansions (14) and (18, 19) in terms of the definiteness properties of the component matrices. As the state-action value function is non-negative over the entire state-action space, it can be seen that $\mathcal{H}_1(\boldsymbol{w})$ is positive-semidefinite for all $\boldsymbol{w} \in \mathcal{W}$. Similarly, it can be shown that under certain common policy parameterisations $\mathcal{H}_2(\boldsymbol{w})$ is negative-semidefinite over the entire parameter space. This is summarized in the following theorem.

Theorem 5. *The matrix $\mathcal{H}_2(\boldsymbol{w})$ is negative-semidefinite for all $\boldsymbol{w} \in \mathcal{W}$ if: 1) the policy is* log*-concave with respect to the policy parameters; or 2) the policy parameterisation has constant curvature with respect to the action space.*

Proof. See Section A.3 in the Appendix. □

It can be seen, therefore, that when the policy parameterisation satisfies the properties of Theorem 5 the expansion (14) gives $\mathcal{H}(\boldsymbol{w})$ in terms of a positive-semidefinite term, $\mathcal{H}_1(\boldsymbol{w})$, a negative-semidefinite term, $\mathcal{H}_2(\boldsymbol{w})$, and a remainder term, $\mathcal{H}_{12}(\boldsymbol{w})+\mathcal{H}_{12}^{\top}(\boldsymbol{w})$. In Section 3.3 we shall show that this remainder term becomes negligible around a local optimum when given a sufficiently rich policy parameterisation, in a sense that we introduce in Definition 2. In contrast to the state-action value function, the advantage function takes both positive and negative values over the state-action space. As a result, the matrices $\mathcal{A}_1(\boldsymbol{w})$ and $\mathcal{A}_2(\boldsymbol{w})$ in (18, 19) can be indefinite over parts of the parameter space.

3.3 Analysis in Vicinity of a Local Optimum

In this section we consider the term $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^{\top}(\boldsymbol{w})$, which is both difficult to estimate and not guaranteed to be negative definite. In particular, we shall consider the conditions under which this term becomes either negligible or vanishes completely at a local optimum. We start by noting that,

$$\begin{aligned}\mathcal{H}_{12}(\boldsymbol{w}) &= \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} p_\gamma(s,a;\boldsymbol{w})\frac{\partial}{\partial \boldsymbol{w}}\log\pi(a|s;\boldsymbol{w})\frac{\partial^{\top}}{\partial \boldsymbol{w}}\left(R(s,a)+\gamma\sum_{s'}p(s'|a,s)V(s';\boldsymbol{w})\right),\\ &= \gamma\sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} p_\gamma(s,a;\boldsymbol{w})\frac{\partial}{\partial \boldsymbol{w}}\log\pi(a|s;\boldsymbol{w})\sum_{s'}p(s'|a,s)\frac{\partial^{\top}}{\partial \boldsymbol{w}}V(s';\boldsymbol{w}). \qquad (20)\end{aligned}$$

Our approach is to obtain a bound of, $\frac{\partial}{\partial w_i}V(s;\boldsymbol{w})\big|_{\boldsymbol{w}=\boldsymbol{w}^*}$, for all $s' \in \mathcal{S}$ and $i \in \{1,...,n\}$, when, $\boldsymbol{w}^* \in \mathcal{W}$, is a local optimum of (4). Given such a bound it is then possible to obtain a bound on the term, $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^{\top}(\boldsymbol{w})$, at a local optimum. In the limit the bound gives conditions under which, $\frac{\partial}{\partial \boldsymbol{w}}V(s';\boldsymbol{w}) = \mathbf{0}$, for all $s' \in \mathcal{S}$, at a local optimum, thus giving sufficient conditions under which the term, $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^{\top}(\boldsymbol{w})$, vanishes at a local optium. We start by introducing the notion of a ϵ-*value-consistent* policy class, with $\epsilon \in \mathbb{R}$, $\epsilon \geq 0$.

Definition 2. *Given $\epsilon \in \mathbb{R}$, with $\epsilon \geq 0$, then a policy parameterisation is said to be ϵ-value-consistent w.r.t. a Markov decision process if whenever $\frac{\partial}{\partial w_i}V(\hat{s};\boldsymbol{w}) \neq 0$ for some $\hat{s} \in \mathcal{S}$, $\boldsymbol{w} \in \mathcal{W}$ and $i \in \{1,...,n\}$, then $\forall s \in \mathcal{S}$ it holds that either,*

$$\operatorname{sign}\left(\frac{\partial}{\partial w_i}V(s;\boldsymbol{w})\right) = \operatorname{sign}\left(\frac{\partial}{\partial w_i}V(\hat{s};\boldsymbol{w})\right), \qquad (21)$$

or

$$\left| \frac{\partial}{\partial w_i} V(s; \boldsymbol{w}) \right| \leq \epsilon. \tag{22}$$

Furthermore, for any state, $s \in \mathcal{S}$, for which, $\frac{\partial}{\partial w_i} V(s; \boldsymbol{w}) = 0$, it also holds that $\frac{\partial}{\partial w_i} \pi(a|s; \boldsymbol{w}) = 0$, $\forall a \in \mathcal{A}$. A policy parameterization is said to be value-consistent if it is 0-value-consistent.

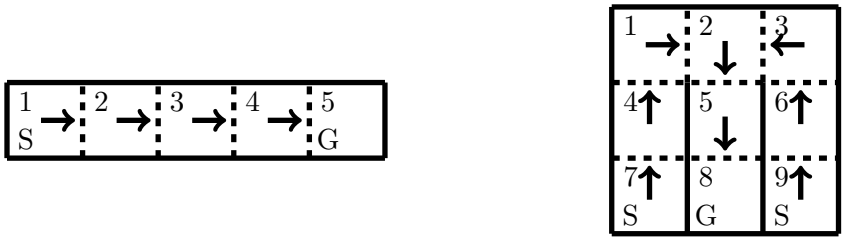
This property of a policy class captures the notion that when maximally improving the value of one state, then the amount that the value of another state can decrease is bounded. When a policy class is value-consistent, then changing a parameter to maximally improve the value in one state, does not worsen the value in another state. i.e., when a policy class is value-consistent, there is no trade-off between improving the value in different states.

Example. To illustrate the concept of a value-consistent policy parameterisation we now consider two simple maze navigation MDPs, one with a value-consistent policy parameterisation, and one with a policy parameterisation that is not value-consistent. The two MDPs are displayed in Figure 1. Walls of the maze are solid lines, while the dotted lines indicate state boundaries and are passable. The agent starts, with equal probability, in one of the states marked with an 'S'. The agent receives a positive reward for reaching the goal state, which is marked with a 'G', and is then reset to one of the start states. All other state-action pairs return a reward of zero. The discount factor in both MDPs is 0.95. There are four possible actions (up, down, left, right) in each state, and the optimal policy is to move, with probability one, in the direction indicated by the arrow. We define the mapping, $o : \mathcal{S} \to \{0, 1\}^4$, which indicates the presence of a wall on each of the four state boundaries. We use the notation, $\mathcal{O} := \{o(s) | s \in \mathcal{S}\}$. We consider the policy parameterisation, $\pi(a|s; \boldsymbol{w}) \propto \exp(\boldsymbol{w}^\top \boldsymbol{\phi}(a, s))$, in which, $\boldsymbol{\phi} : \mathcal{A} \times \mathcal{S} \to \mathbb{R}^{3|\mathcal{O}|}$, is a feature map. We consider the feature map,

$$\boldsymbol{\phi}(a, s) = \begin{cases} \mathbf{0} & \textit{if } a = \text{up}, \\ \boldsymbol{e}_{i(a)+3j(s)} & \textit{otherwise} \ , \end{cases}$$

in which, $i : \mathcal{A} \backslash \{up\} \to \{1, 2, 3\}$, is an index function over the actions, $\mathcal{A} \backslash \{up\}$, and $j : \mathcal{S} \to \{0, 1, ..., |\mathcal{O}| - 1\}$, is an index function over the aliased states, i.e., $j(s) = j(s')$ iff $o(s) = o(s')$. We use the notation, $\boldsymbol{e}_i \in \mathbb{R}^{3|\mathcal{O}|}$, $i \in \{1, ..., 3|\mathcal{O}|\}$, to denote the i^{th} standard basis vector. Perceptual aliasing (Whitehead, 1992) occurs in both MDPs under this policy parameterisation, with states 2, 3 & 4 aliased in the hallway problem, and states 4, 5 & 6 aliased in McCallum's grid. In the hallway problem all of the aliased states have the same optimal action, and the value of these states all increase/decrease in unison. Hence, it can be seen that the policy parameterisation is value-consistent for the hallway problem. In McCallum's grid, however, the optimal action for states 4 & 6 is to move upwards, while in state 5 it is to move downwards. In this example increasing the probability of moving downwards in state 5 will also increase the probability of moving downwards in states 4 & 6. There is a point, therefore, at which increasing the probability of moving downwards in state 5 will decrease the value of states 4 & 6. Thus this policy parameterisation is not value-consistent for McCallum's grid. Numerical inspection of the parameter space indicates that this policy parameterisation is ϵ-value-consistent, with $\epsilon \approx 0.2$.

We now show that tabular policies—i.e., policies such that, for each state $s \in \mathcal{S}$, the conditional distribution $\pi(a|s; \boldsymbol{w}_s)$ is parametrized by a separate parameter vector $\boldsymbol{w}_s \in \mathbb{R}^{n_s}$ for some $n_s \in \mathbb{N}$—are value-consistent, regardless of the given Markov decision process.



(a) Hallway Problem (b) McCallum Grid

Figure 1: (a) The hallway problem. Under the feature map, $\boldsymbol{\phi}$, states 2, 3 and 4 map to the the same feature, and the optimal policy is identical on these states. (b) McCallum's grid. Under the feature map, $\boldsymbol{\phi}$, states 4, 5 and 6 map to the same feature, but now the optimal policy differs among these states.

Theorem 6. *Suppose that a given Markov decision process has a tabular policy parameterisation, then the policy parameterisation is value-consistent.*

Proof. See Section A.4 in the Appendix. □

We now show that given an ϵ-value-consistent policy parameterisation and a local optimum, $\boldsymbol{w}^* \in \mathcal{W}$, the terms, $\frac{\partial}{\partial_{w_i}} V(s;\boldsymbol{w})\big|_{\boldsymbol{w}=\boldsymbol{w}^*}$, are bounded in magnitude by ϵ, for all $s' \in \mathcal{S}$ and $i \in \{1, ..., n\}$.

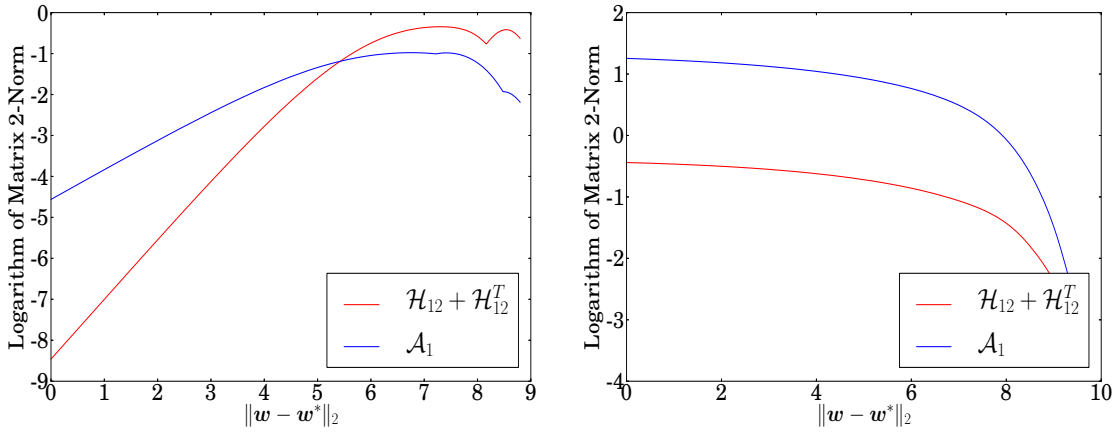
Theorem 7. *Suppose that $\boldsymbol{w}^* \in \mathcal{W}$ is a local optimum of the differentiable objective function, $U(\boldsymbol{w}) = \mathbb{E}_{s\sim p_1(\cdot)}\big[V(s;\boldsymbol{w})\big]$. Suppose that the Markov chain induced by $\boldsymbol{w}^*$ is ergodic. Suppose that the policy parameterisation is ϵ-value-consistent, $\epsilon \in \mathbb{R}$, $\epsilon \geq 0$, w.r.t. the given Markov decision process. Then*

$$\left| \frac{\partial}{\partial_{w_i}} V(s;\boldsymbol{w})\big|_{\boldsymbol{w}=\boldsymbol{w}^*} \right| \leq \epsilon, \qquad \forall s \in \mathcal{S}. \tag{23}$$

Proof. See Appendix A.5 □

If a policy parameterisation is ϵ-value-consistent and the parameterisation satisfies the bound, $|\frac{\partial}{\partial \boldsymbol{w}} \log \pi(a|s;\boldsymbol{w})| \leq M$, $M \in \mathbb{R}$, for all $(s,a) \in \mathcal{S} \times \mathcal{A}$ and $\boldsymbol{w} \in \mathcal{W}$, then it follows from Theorem 7 that the terms of the matrix, $\mathcal{H}_{12}(\boldsymbol{w}^*) + \mathcal{H}_{12}^\top(\boldsymbol{w}^*)$, are bounded by $2\epsilon\gamma M/(1-\gamma)$. A further corollary of Theorem 7 is that when a policy class is value-consistent the term, $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w})$, vanishes near local optima. Furthermore, when we have the additional condition that the gradient of the state value function is continuous in $\boldsymbol{w}$ (at $\boldsymbol{w} = \boldsymbol{w}^*$) then $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w}) \to \mathbf{0}$ as $\boldsymbol{w} \to \boldsymbol{w}^*$. This condition will be satisfied if, for example, the policy is continuously differentiable w.r.t. the policy parameters.

Example (continued)**.** Returning to the MDPs given in Figure 1, we now empirically observe the behaviour of the term $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w})$ as the policy approaches a local optimum of the objective function. Figure 2 gives the magnitude of $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w})$, in terms of the spectral norm, in relation to the distance from the local optimum. In correspondence with



(a) Hallway Problem (b) McCallum's Grid

Figure 2: Graphical illustration of the logarithm of the spectral norm of $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w})$ and $\mathcal{A}_1(\boldsymbol{w})$ in terms of $\|\boldsymbol{w} - \boldsymbol{w}^*\|_2$ for the hallway problem (a) and McCallum's grid (b). For the given policy parameterisation, $\mathcal{H}(\boldsymbol{w}) = \mathcal{A}_1(\boldsymbol{w}) + \mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w})$, so the plot displays the two components of the Hessian as the policy converges to a local optimum. As expected, in the hallway problem, $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w}) \to \mathbf{0}$ as $\boldsymbol{w} \to \boldsymbol{w}^*$. Conversely, in McCallum's grid, $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w}) \not\to \mathbf{0}$ as $\boldsymbol{w} \to \boldsymbol{w}^*$, but the term, $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w})$, is still dominated by, $\mathcal{A}_1(\boldsymbol{w})$, in the vicinty of a local optima. In this example the magnitude of $\mathcal{A}_1(\boldsymbol{w})$ is roughly five times greater than that of $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w})$ when $\|\boldsymbol{w} - \boldsymbol{w}^*\|_2 \approx 0.0045$.

the theory, $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w}) \to \mathbf{0}$ as $\boldsymbol{w} \to \boldsymbol{w}^*$ in the hallway problem, while this is not the case in McCallum's grid. This simple example illustrates the fact that if the feature representation is well chosen, in the sense that it is value-consistent, the term $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w})$ vanishes in the vicinity of a local optimum.

4. Gauss-Newton Methods for Markov Decision Processes

In this section we propose several Gauss-Newton type methods for MDPs, motivated by the analysis of Section 3. The algorithms are outlined in Section 4.1, and key performance analysis is provided in Section 4.2.

4.1 The Gauss-Newton Methods

The first Gauss-Newton method we propose drops the Hessian terms which are difficult to estimate, but are expected to be negligible in the vicinity of local optima. Specifically, it was shown in Section 3.3 that if the policy parameterisation is value-consistent with a given MDP, then $\mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w}) \to \mathbf{0}$ as $\boldsymbol{w}$ converges towards a local optimum of the objective function. In such cases $\mathcal{A}_1(\boldsymbol{w}) + \mathcal{A}_2(\boldsymbol{w})$, as defined in Theorem 4, will be a good approximation to the Hessian in the vicinity of a local optimum. For this reason, the

first Gauss-Newton method that we propose for MDPs is to precondition the gradient with $\mathcal{M}(\boldsymbol{w}) = -(\mathcal{A}_1(\boldsymbol{w}) + \mathcal{A}_2(\boldsymbol{w}))^{-1}$ in (6), so that the update is of the form:

Policy search update using the first Gauss-Newton method

$$\boldsymbol{w}^{\text{new}} = \boldsymbol{w} - \alpha(\mathcal{A}_1(\boldsymbol{w}) + \mathcal{A}_2(\boldsymbol{w}))^{-1} \frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w}). \tag{24}$$

When the policy parameterisation has constant curvature with respect to the action space $\mathcal{A}_2(\boldsymbol{w}) = 0$ it is sufficient to calculate just $\mathcal{A}_1^{-1}(\boldsymbol{w})$.

The second Gauss-Newton method we propose removes further terms from the Hessian which are not guaranteed to be negative-semidefinite. As was seen in Section 3.1, when the policy parameterisation satisfies the properties of Theorem 5 then $\mathcal{H}_2(\boldsymbol{w})$ is negative-semidefinite over the entire parameter space.[5] Recall that in (6) it is necessary that $\mathcal{M}(\boldsymbol{w})$ is positive-definite (in Newton's method this corresponds to requiring the Hessian to be negative-definite) to ensure an increase of the objective function. That $\mathcal{H}_2(\boldsymbol{w})$ is negative-semidefinite over the entire parameter space is therefore a highly desirable property of a preconditioning matrix, and for this reason the second Gauss-Newton method that we propose for MDPs is to precondition the gradient with $\mathcal{M}(\boldsymbol{w}) = -\mathcal{H}_2^{-1}(\boldsymbol{w})$ in (6), so that the update is of the form:

Policy search update using the second Gauss-Newton method

$$\boldsymbol{w}^{\text{new}} = \boldsymbol{w} - \alpha \mathcal{H}_2^{-1}(\boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w}). \tag{25}$$

We shall see that the second Gauss-Newton method has important performance guarantees including: a guaranteed ascent direction; linear convergence to a local optimum under a step size which does not depend upon unknown quantities; invariance to affine transformations of the parameter space; and efficient estimation procedures for the preconditioning matrix. We will also show in Section 5 that the second Gauss-Newton method is closely related to both the EM and natural gradient algorithms.

We shall also consider a diagonal form of the approximation for both forms of Gauss-Newton methods. Denoting the diagonal matrix formed from the diagonal elements of $\mathcal{A}_1(\boldsymbol{w}) + \mathcal{A}_2(\boldsymbol{w})$ and $\mathcal{H}_2(\boldsymbol{w})$ by $\mathcal{D}_{\mathcal{A}_1+\mathcal{A}_2}(\boldsymbol{w})$ and $\mathcal{D}_{\mathcal{H}_2}(\boldsymbol{w})$, respectively, then we shall consider the methods that use $\mathcal{M}(\boldsymbol{w}) = -\mathcal{D}_{\mathcal{A}_1+\mathcal{A}_2}^{-1}(\boldsymbol{w})$ and $\mathcal{M}(\boldsymbol{w}) = -\mathcal{D}_{\mathcal{H}_2}^{-1}(\boldsymbol{w})$ in (6). We call these methods the diagonal first and second Gauss-Newton methods, respectively. This

5. That the preconditioning matrix is negative-semidefinite and not negative-definite is a standard problem with many optimisation techniques that require the inversion of a preconditioning matrix. This includes natural gradient ascent when the Fisher information matrix is used to precondition the gradient (Thomas, 2014). Various approaches can be taken with regard to this problem: Add a ridge term to the preconditioning matrix; Minimize $||\mathcal{H}_2(\boldsymbol{w})\boldsymbol{p} + \frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w})||^2$ with respect to $\boldsymbol{p}$ by gradient descent, and use $\boldsymbol{p}$ as the search direction; Precondition the gradient with the pseudoinverse $-\mathcal{H}_2^{+}(\boldsymbol{w})$.

diagonalization amounts to performing the approximate Newton methods on each parameter independently, but simultaneously.

4.1.1 Estimation of the Preconditioners and the Gauss-Newton Update Direction

It is possible to extend typical techniques used to estimate the policy gradient to estimate the preconditioner for the Gauss-Newton method, by including either the Hessian of the log-policy, the outer product of the derivative of the log-policy, or the respective diagonal terms. As an example, in Section B.1 of the Appendix we detail the extension of the recurrent state formulation of gradient evaluation in the average reward framework (Williams, 1992) to the second Gauss-Newton method. We use this extension in the Tetris experiment that we consider in Section 6. Given n_s sampled state-action pairs, the complexity of this extension scales as $\mathcal{O}(n_s n^2)$ for the second Gauss-Newton method, while it scales as $\mathcal{O}(n_s n)$ for the diagonal version of the algorithm. We provide more details of situations in which the inversion of the preconditioning matrices can be performed more efficiently in Section B.2 of the Appendix.

4.2 Performance Guarantees and Analysis

4.2.1 Ascent Directions

In general the objective (4) is not concave, which means that the Hessian will not be negative-definite over the entire parameter space. In such cases Newton's method can actually lower the objective and this is an undesirable aspect of Newton's method. We now consider ascent directions for the Gauss-Newton methods, and in particular demonstrate that the proposed second Gauss-Newton method guarantees an ascent direction in typical settings.

Ascent directions for the first Gauss-Newton method: As mentioned previously, the matrix $\mathcal{A}_1(\boldsymbol{w}) + \mathcal{A}_2(\boldsymbol{w})$ will typically be indefinite, and so a straightforward application of the first Gauss-Newton method will not necessarily result in an increase in the objective function. There are, however, standard correction techniques that one could consider to ensure that an increase in the objective function is obtained, such as adding a ridge term to the preconditioning matrix. A survey of such correction techniques can be found in Boyd and Vandenberghe (2004).

Ascent directions for the second Gauss-Newton method: It was seen in Theorem 5 that $\mathcal{H}_2(\boldsymbol{w})$ will be negative-semidefinite over the entire parameter space if either the policy is log-concave with respect to the policy parameters, or the policy has constant curvature with respect to the action space. It follows that in such cases an increase of the objective function will be obtained when using the second Gauss-Newton method with a sufficiently small step-size. Additionally, the diagonal terms of a negative-semidefinite matrix are non-positive, so that $\mathcal{D}_{\mathcal{H}_2}(\boldsymbol{w})$ is negative-semidefinite whenever $\mathcal{H}_2(\boldsymbol{w})$ is negative-semidefinite, and thus similar performance guarantees exist for the diagonal version of the second Gauss-Newton algorithm.

To motivate this result we now briefly consider some widely used policies that are either log-concave or blockwise log-concave. Firstly, consider the linear softmax policy parameterisation, $\pi(a|s;\boldsymbol{w}) \propto \exp \boldsymbol{w}^T \boldsymbol{\phi}(a,s)$, in which $\boldsymbol{\phi}(a,s) \in \mathbb{R}^n$ is a feature vector. This

policy is widely used in discrete systems and is log-concave in $\boldsymbol{w}$, which can be seen from the fact that $\log \pi(a|s;\boldsymbol{w})$ is the sum of a linear term and a negative *log-sum-exp* term, both of which are concave (Boyd and Vandenberghe, 2004). In systems with a continuous state-action space a common choice of controller is $\pi(a|s;K,\Sigma) = \mathcal{N}(a|K\boldsymbol{\phi}(s),\Sigma)$, in which $\boldsymbol{\phi}(s) \in \mathbb{R}^n$ is a feature vector. This controller is not jointly log-concave in K and Σ, but it is blockwise log-concave in K and Σ^{-1}. In terms of K the log-policy is quadratic and the coefficient matrix of the quadratic term is negative-semidefinite. In terms of Σ^{-1} the log-policy consists of a linear term and a log-determinant term, both of which are concave.

4.2.2 Affine Invariance

An undesirable aspect of gradient ascent is that its performance is dependent on the choice of basis used to represent the parameter space. An important and desirable property of Newton's method is that it is invariant to non-singular affine transformations of the parameter space (Boyd and Vandenberghe, 2004). The proposed approximate Newton methods have various invariance properties, and these properties are summarized in the following theorem.

Theorem 8. *The first and second Gauss-Newton methods are invariant to (non-singular) affine transformations of the parameter space. The diagonal versions of these algorithms are invariant to (non-singular) rescalings of the parameter space.*

Proof. See Section A.6 in the Appendix. □

4.2.3 Convergence Analysis

We now provide a local convergence analysis of the Gauss-Newton framework. We shall focus on the full Gauss-Newton methods, with the analysis of the diagonal Gauss-Newton method following similarly. Additionally, we shall focus on the case in which a constant step size is considered throughout, which is denoted by $\alpha \in \mathbb{R}^+$. We say that an algorithm converges linearly to a limit L at a rate $r \in (0,1)$ if $\lim_{k\to\infty} \frac{|U(\boldsymbol{w}_{k+1})-L|}{|U(\boldsymbol{w}_k)-L|} = r$. If $r = 0$ then the algorithm converges super-linearly. We denote the parameter update function of the first and second Gauss-Newton methods by G_1 and G_2, respectively, so that $G_1(\boldsymbol{w}) = \boldsymbol{w} - \alpha(\mathcal{A}_1(\boldsymbol{w}) + \mathcal{A}_2(\boldsymbol{w}))^{-1}\frac{\partial}{\partial \boldsymbol{w}}U(\boldsymbol{w})$ and $G_2(\boldsymbol{w}) = \boldsymbol{w} - \alpha\mathcal{H}_2^{-1}(\boldsymbol{w})\frac{\partial}{\partial \boldsymbol{w}}U(\boldsymbol{w})$. Given a matrix, $A \in L(\mathbb{R}^n)$ we denote the spectral radius of A by $\rho(A) = \max_i |\lambda_i|$, where $\{\lambda_i\}_{i=1}^n$ are the eigenvalues of A. Throughout this section we shall use $\nabla G(\boldsymbol{w}^*)$ to denote $\frac{\partial}{\partial \boldsymbol{w}}G(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*}$.

Theorem 9 (Convergence analysis for the first Gauss-Newton method)**.** *Suppose that $\boldsymbol{w}^* \in \mathcal{W}$ is such that $\frac{\partial}{\partial \boldsymbol{w}}U(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} = \mathbf{0}$ and $\mathcal{A}_1(\boldsymbol{w}^*) + \mathcal{A}_2(\boldsymbol{w}^*)$ is invertible, then G_1 is Fréchet differentiable at $\boldsymbol{w}^*$ and $\nabla G_1(\boldsymbol{w}^*)$ takes the form,*

$$\nabla G_1(\boldsymbol{w}^*) = I - \alpha(\mathcal{A}_1(\boldsymbol{w}^*) + \mathcal{A}_2(\boldsymbol{w}^*))^{-1}\mathcal{H}(\boldsymbol{w}^*). \tag{26}$$

If $\mathcal{H}(\boldsymbol{w}^*)$ and $\mathcal{A}_1(\boldsymbol{w}^*) + \mathcal{A}_2(\boldsymbol{w}^*)$ are negative-definite, and the step size is in the range,

$$\alpha \in \left(0, 2/\rho\left((\mathcal{A}_1(\boldsymbol{w}^*) + \mathcal{A}_2(\boldsymbol{w}^*))^{-1}\mathcal{H}(\boldsymbol{w}^*)\right)\right) \tag{27}$$

then $\boldsymbol{w}^*$ is a point of attraction of the first Gauss-Newton method, the convergence is at least linear and the rate is given by $\rho(\nabla G_1(\boldsymbol{w}^*)) < 1$. When the policy parameterisation is

value-consistent with respect to the given Markov decision process, then (26) simplifies to,

$$\nabla G_1(\boldsymbol{w}^*) = (1-\alpha)I, \tag{28}$$

and whenever $\alpha \in (0,2)$ then $\boldsymbol{w}^*$ is a point of attraction of the first Gauss-Newton method, and the convergence to $\boldsymbol{w}^*$ is linear if $\alpha \neq 1$ with a rate given by $\rho(\nabla G_1(\boldsymbol{w}^*)) < 1$, and convergence is super-linear when $\alpha = 1$.

Proof. See Section A.7 in the Appendix. □

Theorem 10 (Convergence analysis for the second Gauss-Newton method)**.** *Suppose that $\boldsymbol{w}^* \in \mathcal{W}$ is such that $\frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} = \mathbf{0}$ and $\mathcal{H}_2(\boldsymbol{w}^*)$ is invertible, then G_2 is Fréchet differentiable at $\boldsymbol{w}^*$ and $\nabla G_2(\boldsymbol{w}^*)$ takes the form,*

$$\nabla G_2(\boldsymbol{w}^*) = I - \alpha \mathcal{H}_2^{-1}(\boldsymbol{w}^*)\mathcal{H}(\boldsymbol{w}^*). \tag{29}$$

If $\mathcal{H}(\boldsymbol{w}^*)$ is negative-definite and the step size is in the range,

$$\alpha \in (0, 2/\rho(\mathcal{H}_2^{-1}(\boldsymbol{w}^*)\mathcal{H}(\boldsymbol{w}^*))) \tag{30}$$

then $\boldsymbol{w}^*$ is a point of attraction of the second Gauss-Newton method, convergence to $\boldsymbol{w}^*$ is at least linear and the rate is given by $\rho(\nabla G_2(\boldsymbol{w}^*)) < 1$. Furthermore, $\alpha \in (0,2)$ implies condition (30). When the policy parameterisation is value-consistent with respect to the given Markov decision process, then (29) simplifies to,

$$\nabla G_2(\boldsymbol{w}^*) = I - \alpha \mathcal{H}_2^{-1}(\boldsymbol{w}^*)\mathcal{A}_1(\boldsymbol{w}^*). \tag{31}$$

Proof. See Section A.7 in the Appendix. □

The conditions of Theorem 10 look analogous to those of Theorem 9, but they differ in important ways: in Theorem 10 it is not necessary to assume that the preconditioning matrix is negative-definite and the sets in (27) will not be known in practice, whereas the condition $\alpha \in (0,2)$ in Theorem 10 is more practical, i.e., for the second Gauss-Newton method convergence is guaranteed for a constant step size which is easily selected and does not depend upon unknown quantities.

It will be seen in Section 5.2 that the second Gauss-Newton method has a close relationship to the EM-algorithm. For this reason we postpone additional discussion about the rate of convergence of the second Gauss-Newton method until then.

5. Relation to Existing Policy Search Methods

In this section we consider the relationship between the second Gauss-Newton method and existing policy search methods; In Section 5.1 we examine its relation to natural gradient ascent and in Section 5.2 to the EM-algorithm.

5.1 Natural Gradient Ascent and the Second Gauss-Newton Method

Comparing the form of the Fisher information matrix given in (11) with $\mathcal{H}_2$ (16) it can be seen that there is a close relationship between natural gradient ascent and the second Gauss-Newton method: in $\mathcal{H}_2$ there is an additional weighting of the integrand from the state-action value function. Hence, $\mathcal{H}_2$ incorporates information about the reward structure of the objective function that is not present in the Fisher information matrix.

We now consider how this additional weighting affects the search direction for natural gradient ascent and the Gauss-Newton approach. Given a norm on the parameter space, $||\cdot||$, the steepest ascent direction at $\boldsymbol{w} \in \mathcal{W}$ with respect to that norm is given by,

$$\hat{\boldsymbol{p}} = \text{argsup}_{\{\boldsymbol{p}:||\boldsymbol{p}||=1\}} \lim_{\alpha \to 0} \frac{U(\boldsymbol{w} + \alpha \boldsymbol{p}) - U(\boldsymbol{w})}{\alpha}.$$

Natural gradient ascent is obtained by considering the (local) norm $||\cdot||_{G(\boldsymbol{w})}$ given by $||\boldsymbol{w} - \boldsymbol{w'}||^2_{G(\boldsymbol{w})} := (\boldsymbol{w} - \boldsymbol{w'})^\top G(\boldsymbol{w})(\boldsymbol{w} - \boldsymbol{w'})$, with $G(\boldsymbol{w})$ as in (10). The natural gradient method allows less movement in the directions that have high norm which, as can be seen from the form of (10), are those directions that induce large changes to the policy over the parts of the state-action space that are likely to be visited under the current policy parameters. More movement is allowed in directions that either induce a small change in the policy, or induce large changes to the policy, but only in parts of the state-action space that are unlikely to be visited under the current policy parameters. In a similar manner the second Gauss-Newton method can be obtained by considering the (local) norm $||\cdot||_{\mathcal{H}_2(\boldsymbol{w})}$, given by $||\boldsymbol{w} - \boldsymbol{w'}||^2_{\mathcal{H}_2(\boldsymbol{w})} := -(\boldsymbol{w} - \boldsymbol{w'})^\top \mathcal{H}_2(\boldsymbol{w})(\boldsymbol{w} - \boldsymbol{w'})$ so that each term in (11) is additionally weighted by the state-action value function, $Q(s, a; \boldsymbol{w})$. Thus, the directions which have high norm are those in which the policy is rapidly changing in state-action pairs that are not only likely to be visited under the current policy, but also have high value. Thus the second Gauss-Newton method updates the parameters more conservatively if the behaviour in high value states is affected. Conversely, directions which induce a change only in state-action pairs of low value have low norm, and larger increments can be made in those directions.

5.2 Expectation Maximization and the Second Gauss-Newton Method

It has previously been noted (Kober and Peters, 2011) that the parameter update of gradient ascent and the EM-algorithm can be related through the function $\mathcal{Q}$ defined in (13). In particular, the gradient (8) evaluated at $\boldsymbol{w}_k$ can be written in terms of $\mathcal{Q}$ as follows,

$$\frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w})|_{w=w_k} = \frac{\partial}{\partial \boldsymbol{w}} \mathcal{Q}(\boldsymbol{w}, \boldsymbol{w}_k)|_{w=w_k},$$

while the parameter update of the EM-algorithm is given by $\boldsymbol{w}_{k+1} = \text{argmax}_{\boldsymbol{w} \in \mathcal{W}} \ \mathcal{Q}(\boldsymbol{w}, \boldsymbol{w}_k)$. In other words, gradient ascent moves in the direction that most rapidly increases $\mathcal{Q}$ with respect to the first variable, while the EM-algorithm maximizes $\mathcal{Q}$ with respect to the first variable. While this relationship is true, it is also quite a negative result. It states that in situations in which it is not possible to explicitly maximize $\mathcal{Q}$ with respect to its first variable, then the alternative, in terms of the EM-algorithm, is a generalized EM-algorithm, which is equivalent to gradient ascent. Given that the EM-algorithm is typically used to

overcome the negative aspects of gradient ascent, this is an undesirable alternative. It is possible to find the optimum of (13) numerically, but this is also undesirable as it results in a double-loop algorithm that could be computationally expensive. Finally, this result provides no insight into the behaviour of the EM-algorithm, in terms of the direction of its parameter update, when the maximization over $\boldsymbol{w}$ in (13) can be performed explicitly.

We now demonstrate that the step-direction of the EM-algorithm has an underlying relationship with the second of our proposed Gauss-Newton methods. In particular, we show that under suitable regularity conditions the direction of the EM-update, $\boldsymbol{w}_{k+1} - \boldsymbol{w}_k$, is the same, up to first order, as the direction of the second Gauss-Newton method.

Theorem 11. *Suppose we are given a Markov decision process with objective (1) and Markovian trajectory distribution (2). Consider the parameter update (M-step) of expectation maximization at the k^{th} iteration of the algorithm, i.e., $\boldsymbol{w}_{k+1} = \operatorname{argmax}_{\boldsymbol{w}\in\mathcal{W}} \mathcal{Q}(\boldsymbol{w}, \boldsymbol{w}_k)$. Provided that $\mathcal{Q}(\boldsymbol{w}, \boldsymbol{w}_k)$ is twice continuously differentiable in the first parameter we have that,*

$$\boldsymbol{w}_{k+1} - \boldsymbol{w}_k = -\mathcal{H}_2^{-1}(\boldsymbol{w}_k)\frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w})|_{w=w_k} + \mathcal{O}(\|\boldsymbol{w}_{k+1} - \boldsymbol{w}_k\|^2). \tag{32}$$

Additionally, in the case where the log*-policy is quadratic the relation to the approximate Newton method is exact, i.e., the second term on the* r.h.s. *of (32) is zero.*

Proof. See Section A.8 in the Appendix. □

Given a sequence of parameter vectors, $(\boldsymbol{w}_k)_{k=1}^{\infty}$, generated through an application of the EM-algorithm, then $\lim_{k\to\infty} \|\boldsymbol{w}_{k+1} - \boldsymbol{w}_k\| = 0$. This means that the rate of convergence of the EM-algorithm will be the same as that of the second Gauss-Newton method when considering a constant step size of one. We formalize this intuition and provide the convergence properties of the EM-algorithm when applied to Markov decision processes in the following theorem. This is, to our knowledge, the first formal derivation of the convergence properties for this application of the EM-algorithm.

Theorem 12. *Suppose that the sequence, $(\boldsymbol{w}_k)_{k\in\mathbb{N}}$, is generated by an application of the EM-algorithm, where the sequence converges to $\boldsymbol{w}^*$. Denote the update operation of the EM-algorithm by G_{EM}, so that $\boldsymbol{w}_{k+1} = G_{\text{EM}}(\boldsymbol{w}_k)$. Using $\nabla G_{\text{EM}}(\boldsymbol{w}^*)$ to denote $\frac{\partial}{\partial \boldsymbol{w}} G_{\text{EM}}(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*}$, then,*

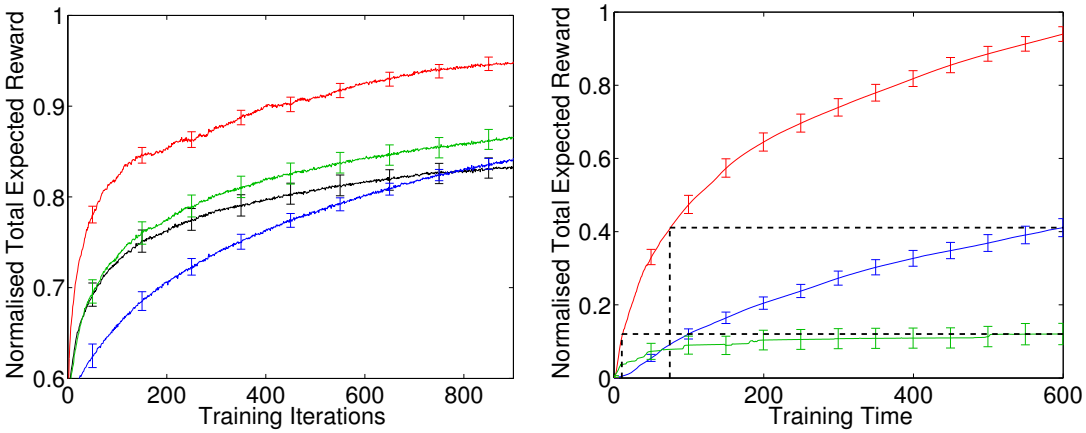
$$\nabla G_{\text{EM}}(\boldsymbol{w}^*) = I - \mathcal{H}_2^{-1}(\boldsymbol{w}^*)\mathcal{H}(\boldsymbol{w}^*).$$

When the policy parameterisation is value-consistent with respect to the given Markov decision process this simplifies to $\nabla G_{\text{EM}}(\boldsymbol{w}^*) = I - \mathcal{H}_2^{-1}(\boldsymbol{w}^*)\mathcal{A}_1(\boldsymbol{w}^*)$. When the Hessian, $\mathcal{H}(\boldsymbol{w}^*)$, is negative-definite then $\rho(\nabla G_{\text{EM}}(\boldsymbol{w}^*)) < 1$ and $\boldsymbol{w}^*$ is a local point of attraction for the EM-algorithm.

Proof. See Section A.9 in the Appendix. □

6. Experiments

In this section we provide an empirical evaluation of the Gauss-Newton methods on a variety of domains. We summarize the experimental results here. For reproducibility, more details can be found in Appendix C.



(a) Non-Linear Navigation Task : Results (b) 3-Link Manipulator Experiment : Results

Figure 3: (a) Results from the non-linear navigation task, with the results for gradient ascent (black), expectation maximization (blue), natural gradient ascent (green) and the second Gauss-Newton method (red). (b) Normalized total expected reward plotted against training time (in seconds) for the 3-link rigid manipulator. The plot shows the results expectation maximization (blue), the second Gauss-Newton method (red) and natural gradient ascent (green).

6.1 Non-Linear Navigation Experiment

The first domain that we consider is the synthetic two-dimensional non-linear MDP considered in the work of Vlassis et al. (2009). In this experiment we consider gradient ascent, natural gradient ascent, expectation maximisation and the second Gauss-Newton method. Details of the domain and the experiment settings are given in Section C.1. The experiment was repeated 100 times and the results of the experiment are given in Figure 3a, which gives the mean and standard error of the results. The step size sequences of gradient ascent, natural gradient ascent and the Gauss-Newton method were all tuned for performance and the results shown were obtained from the best step size sequence for each algorithm.

6.2 N-link Rigid Manipulator Experiment

The N-link rigid robot arm manipulator is a standard continuous model, consisting of an end effector connected to an N-linked rigid body (Khalil, 2001). A typical continuous control problem for such systems is to apply appropriate torque forces to the joints of the manipulator so as to move the end effector into a desired position. More details on the settings of the domain used in this experiment can be found in Section C.2. We consider a policy of the form,

$$\pi(a|s;\boldsymbol{w}) = \mathcal{N}(a|Ks + \boldsymbol{m}, \sigma^2 I), \tag{33}$$

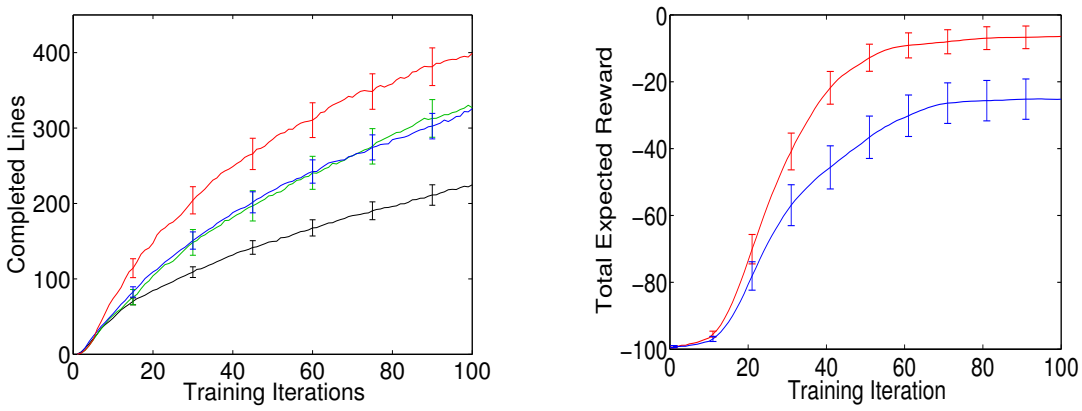
with $\boldsymbol{w} = (K, \boldsymbol{m}, \sigma)$ and $s \in \mathbb{R}^{n_s}$, $a \in \mathbb{R}^{n_a}$, for some $n_s, n_a \in \mathbb{N}$. We consider a 3-link rigid manipulator, which results in a parameter space with 22 dimensions.

In this experiment we compare gradient ascent, natural gradient ascent, expectation maximization and the second Gauss-Newton method. The step size sequences of gradient ascent, natural gradient ascent and the Gauss-Newton method were all tuned for performance. Details of the experiment settings and the procedure used to tune the step size sequences are described in Section C.2. We repeated the experiment 100 times, each time with a different random initialisation of the system. The final results, obtained using the best step size sequence for each algorithm, are given in Figure 3b. We omit the result of gradient ascent as we were unable to obtain any meaningful results for this domain with this algorithm. In this experiment the maximal value of the objective function varied dramatically depending on the random initialization of the system. To account for this variation the results from each run of the experiment are normalized by the maximal value achieved between the algorithms in that run. This means that the results displayed are the percentages of reward received in comparison to the best results among the algorithms considered in the experiment. The second Gauss-Newton method significantly outperforms all of the comparison algorithms. In the experiment the Gauss-Newton method only took around 50 seconds to obtain the same performance as 300 seconds of training with expectation maximization. Furthermore expectation maximization was only able to obtain 40% of the performance of the Gauss-Newton method, while natural gradient ascent was only able to obtain around 15% of the performance. The step direction of expectation maximization is very similar to the search direction of the second Gauss-Newton method in this problem. In fact, given that the log-policy is quadratic in the mean parameters, they are the same for the mean parameters. The difference in performance between the Gauss-Newton method and expectation maximization is largely explained by the tuning of the step size in the Gauss-Newton method, compared to the constant step size of 1.0 in expectation maximization.

6.3 Tetris Experiment

In this experiment we consider the Tetris domain, which is a popular computer game designed by Alexey Pajitnov in 1985. Firstly, we compare the performance of the full and diagonal second Gauss-Newton methods to other policy search methods. We model the policy using a linear softmax paramaterisation. We used the same set of features as used in the works of Bertsekas and Ioffe (1996) & Kakade (2002). Under this paramaterisation it is not possible to obtain the explicit maximum over $\boldsymbol{w}$ in (13), so a straightforward application of the EM-algorithm is not possible in this problem. We therefore compare the diagonal and full versions of the second Gauss-Newton method with steepest and natural gradient ascent. Due to computational costs we consider a 10×10 board in this experiment, which results in a state space with roughly 7×2^{100} states (Bertsekas and Ioffe, 1996). We ran 100 repetitions of the experiment, each consisting of 100 training iterations, and the mean and standard error of the results are given in Figure 4a. It can be seen that the full Gauss-Newton method outperforms all of the other methods, while the performance of the diagonal Gauss-Newton method is comparable to natural gradient ascent.

We also ran several training runs of the full approximate Newton method on the full-sized 20×10 board and were able to obtain a score in the region of $14,000$ completed lines, which was obtained after roughly 40 training iterations. An approximate dynamic programming based method has previously been applied to the Tetris domain in the work of Bertsekas and



(a) Tetris Results

(b) Robot Arm Results

Figure 4: (a) Results from the Tetris experiment, with results for gradient ascent (black), natural gradient ascent (green), the diagonal Gauss-Newton method (blue) and the Gauss-Newton method (red). (b) Results from the robot arm experiment, with results for the second Gauss-Newton method (red) and the EM-algorithm (blue).

Ioffe (1996). The same set of features were used and a score of roughly $4,500$ completed lines was obtained after around 6 training iterations, after which the solution then deteriorated. More recently a modified policy iteration approach (Gabillon et al., 2013) was able to obtain significantly better performance in the game of Tetris, completing approximately 51 million lines in a 20×10 board. However, these results were obtained through an entirely different set of features, and analysis of the results in the work of (Gabillon et al., 2013) indicates that this difference in features makes a substantial difference in performance. On a 10×10 board using the same features as in the work of Bertsekas and Ioffe (1996) the approach was able to complete approximately 500 lines on average.

6.4 Robot Arm Experiment

In the final experiment we consider a robotic arm application. We use the Simulation Lab (Schaal, 2006) environment, which provides a physically realistic engine of a Barrett WAM$^{\text{TM}}$ robot arm. We consider the ball-in-a-cup domain (Kober and Peters, 2009), which is a challenging motor skill problem that is based on the traditional children's game. In this domain a small cup is attached to the end effector of the robot arm. A ball is attached to the cup through a piece of string. At the beginning of the task the robot arm is stationary and the ball is hanging below the cup in a stationary position. The aim of the task is for the robot arm to learn an appropriate set of joint movements to first swing the ball above the cup and then to catch the ball in the cup when the ball is in its downward trajectory. More details of the domain and the experiment settings are provided in Section C.4.

In this experiment we compare gradient ascent, natural gradient ascent, expectation maximization, the first Gauss-Newton method and the second Gauss-Newton method. We

repeated the experiment 50 times and the results are given in Figure 4b. We were unable to successfully learn to catch the ball in the cup using either gradient ascent, natural gradient ascent or the first Gauss-Newton method. For this reason the results for these algorithms are omitted. It can be seen that the second Gauss-Newton method significantly outperforms the EM-algorithm in this domain. Out of the 50 runs of the experiment, the second Gauss-Newton method was successfully able to learn to catch the ball in the cup 45 times. The EM-algorithm successfully learnt the task 36 times. We note that in this experiment the policy took the form, $\pi(a;\boldsymbol{w}) = \mathcal{N}\big(a|\boldsymbol{\mu}, \big(LL^*\big)^{-1}\big)$, with, $\boldsymbol{w} = (\boldsymbol{\mu}, L)$. (More details of the policy parameterisation can be found in Section C.4.) A fixed step size of 1.0 was used in the second Gauss-Newton method, which means that, as the log-policy is quadratic in $\boldsymbol{\mu}$, the update of $\boldsymbol{\mu}$ in the second Gauss-Newton method and the EM-algorithm were the same. The difference in performance can therefore be attributed to the difference in the updates of L between the two algorithms.

7. Conclusions

Approximate Newton methods, such as quasi-Newton methods and the Gauss-Newton method, are standard optimization techniques. These methods aim to maintain the benefits of Newton's method, while alleviating its shortcomings. In this paper we have considered approximate Newton methods in the context of policy optimization in MDPs. The first contribution of this paper was to provide a novel analysis of the Hessian of the total expected reward, which is a standard objective function for policy optimization. This included providing a novel form for the Hessian, as well as detailing the positive/negative semidefiniteness properties of certain terms in the Hessian. Furthermore, we have shown that when the policy parameterisation is sufficiently rich, in the sense that it is ϵ-value-consistent with an appropriately small value of ϵ, then the remaining terms in the Hessian become negligible in the vicinity of a local optimum. Motivated by this analysis we introduced two Gauss-Newton Methods for MDPs. Like the Gauss-Newton method for non-linear least squares, these methods involve approximating the Hessian by ignoring certain terms in the Hessian. The approximate Hessians possess desirable properties, such as negative-semidefiniteness, and we demonstrated several important performance guarantees including guaranteed ascent directions, invariance to affine transformation of the parameter space, and convergence guarantees. We also demonstrated our second Gauss-Newton algorithm is closely related to both the EM-algorithm and natural gradient ascent applied to MDPs, providing novel insights into both of these algorithms. We have compared the proposed Gauss-Newton methods with other techniques in the policy search literature over a range of challenging domains, including Tetris and a robotic arm application. We found that the second Gauss-Newton method performed significantly better than other methods in all of the domains that we considered.

We have provided a convergence analysis of the two proposed Gauss-Newton methods for the setting in which the gradient and the preconditioning matrices can be calculated exactly. An interesting piece of future work is to extend this analysis to the stochastic setting, in which these quantities are estimated from samples of the MDP, either through a Monte-Carlo approach or in a stochastic approximation framework.

Acknowledgments

We would like to thank Peter Dayan, David Silver, Nicolas Heess for helpful discussions on this work and Gerhard Neumann and Christian Daniel for their assistance in the robot arm experiment. We also thank the anonymous reviewers for their suggested improvements. This work was supported by the European Community Seventh Framework Programme (FP7/2007-2013) under grant agreement 270327 (CompLACS), and by the EPSRC under grant agreement EP/M006093/1 (C-PLACID).

Appendix A. Proofs

A.1 Proofs of Theorems 1 and 3

We begin with an auxiliary Lemma.

Lemma 1. *Suppose we are given a Markov decision process with objective (1) and Markovian trajectory distribution (2). For any given parameter vector, $\boldsymbol{w} \in \mathcal{W}$, the following identities hold,*

$$\frac{\partial}{\partial \boldsymbol{w}} V(s;\boldsymbol{w}) = \sum_{t=1}^{\infty} \sum_{s_t \in \mathcal{S}} \sum_{a_t \in \mathcal{A}} \gamma^{t-1} p(s_t, a_t | s_1 = s; \boldsymbol{w}) Q(s_t, a_t; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a_t | s_t; \boldsymbol{w}) \tag{34}$$

$$\frac{\partial}{\partial \boldsymbol{w}} Q(s,a;\boldsymbol{w}) = \sum_{t=2}^{\infty} \sum_{s_t \in \mathcal{S}} \sum_{a_t \in \mathcal{A}} \gamma^{t-1} p(s_t, a_t | s_1 = s, a_1 = a; \boldsymbol{w}) Q(s_t, a_t; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a_t | s_t; \boldsymbol{w}). \tag{35}$$

Proof. We start by writing the state value function in the form

$$V(s;\boldsymbol{w}) = \sum_{t=1}^{\infty} \sum_{s_{1:t}} \sum_{a_{1:t}} \gamma^{t-1} p(s_{1:t}, a_{1:t} | s_1 = s; \boldsymbol{w}) R(s_t, a_t), \tag{36}$$

so that,

$$\frac{\partial}{\partial \boldsymbol{w}} V(s;\boldsymbol{w}) = \sum_{t=1}^{\infty} \sum_{s_{1:t}} \sum_{a_{1:t}} \gamma^{t-1} p(s_{1:t}, a_{1:t} | s_1 = s; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log p(s_{1:t}, a_{1:t} | s_1 = s; \boldsymbol{w}) R(s_t, a_t).$$

Using the fact that

$$\frac{\partial}{\partial \boldsymbol{w}} \log p(s_{1:t}, a_{1:t} | s_1 = s; \boldsymbol{w}) = \sum_{\tau=1}^{t} \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a_\tau | s_\tau; \boldsymbol{w}), \tag{37}$$

we have that,

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{w}} V(s;\boldsymbol{w}) &= \sum_{t=1}^{\infty} \sum_{s_t, a_t} \sum_{\tau=1}^{t} \sum_{s_\tau, a_\tau} \gamma^{t-1} p(s_\tau, a_\tau, s_t, a_t | s_1 = s; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a_\tau | s_\tau; \boldsymbol{w}) R(s_t, a_t) \\
&= \sum_{\tau=1}^{\infty} \sum_{s_\tau, a_\tau} \gamma^{\tau-1} p(s_\tau, a_\tau | s_1 = s; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a_\tau | s_\tau; \boldsymbol{w}) \sum_{t=\tau}^{\infty} \sum_{s_t, a_t} \gamma^{t-\tau} p(s_t, a_t | s_\tau, a_\tau; \boldsymbol{w}) R(s_t, a_t) \\
&= \sum_{\tau=1}^{\infty} \sum_{s_\tau, a_\tau} \gamma^{\tau-1} p(s_\tau, a_\tau | s_1 = s; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a_\tau | s_\tau; \boldsymbol{w}) Q(s_\tau, a_\tau; \boldsymbol{w}).
\end{aligned} \tag{38}$$

where in the second line we swapped the order of summation and the third line follows from the definition (3). Identity (35) now follows by applying (3):

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{w}} Q(s,a;\boldsymbol{w}) &= \gamma \sum_{s'} P(s'|s,a) \frac{\partial}{\partial \boldsymbol{w}} V(s';\boldsymbol{w}) \\ &= \gamma \sum_{s'} P(s'|s,a) \sum_{t=2}^{\infty} \sum_{s_t \in \mathcal{S}} \sum_{a_t \in \mathcal{A}} \gamma^{t-2} p(s_t, a_t | s_2 = s'; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a_t|s_t;\boldsymbol{w}) Q(s_t,a_t;\boldsymbol{w}) \\ &= \sum_{t=2}^{\infty} \sum_{s_t \in \mathcal{S}} \sum_{a_t \in \mathcal{A}} \gamma^{t-1} p(s_t, a_t | s_1 = s, a_1 = a; \boldsymbol{w}) Q(s_t,a_t;\boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a_t|s_t;\boldsymbol{w}).\end{aligned}$$

□

Theorem 1. *Proof.* Theorem 1 follows immediately from Lemma 1 by taking the expectation over s_1 w.r.t. the start state distribution p_1 and using the definition (5) of the discounted trajectory distribution. □

Theorem 3. *Proof.* Starting from $U(\boldsymbol{w}) = \sum_{t=1}^{\infty} \sum_{s_{1:t}} \sum_{a_{1:t}} \gamma^{t-1} p(s_{1:t}, a_{1:t}; \boldsymbol{w}) R(s_t, a_t)$, the Hessian of (4) takes the form

$$\begin{aligned}\frac{\partial^2}{\partial \boldsymbol{w}^2} U(\boldsymbol{w}) &= \sum_{t=1}^{\infty} \sum_{s_{1:t}} \sum_{a_{1:t}} \gamma^{t-1} p(s_{1:t}, a_{1:t}; \boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log p(s_{1:t}, a_{1:t}; \boldsymbol{w}) R(s_t, a_t) \\ &+ \sum_{t=1}^{\infty} \sum_{s_{1:t}} \sum_{a_{1:t}} \gamma^{t-1} p(s_{1:t}, a_{1:t}; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log p(s_{1:t}, a_{1:t}; \boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} \log p(s_{1:t}, a_{1:t}; \boldsymbol{w}) R(s_t, a_t). \quad (39)\end{aligned}$$

Using the fact that $\frac{\partial^2}{\partial \boldsymbol{w}^2} \log p(s_{1:t}, a_{1:t} | s_1 = s; \boldsymbol{w}) = \sum_{\tau=1}^{t} \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a_\tau | s_\tau; \boldsymbol{w})$ we will show that the first term in (39) is equal to $\mathcal{H}_2(\boldsymbol{w})$ as defined in (16):

$$\begin{aligned}&\sum_{t=1}^{\infty} \sum_{s_{1:t}} \sum_{a_{1:t}} \gamma^{t-1} p(s_{1:t}, a_{1:t}; \boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log p(s_{1:t}, a_{1:t}; \boldsymbol{w}) R(s_t, a_t) \\ &= \sum_{t=1}^{\infty} \sum_{s_{1:t}} \sum_{a_{1:t}} \gamma^{t-1} p(s_{1:t}, a_{1:t}; \boldsymbol{w}) \sum_{\tau=1}^{t} \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a_\tau | s_\tau; \boldsymbol{w}) R(s_t, a_t) \\ &= \sum_{\tau=1}^{\infty} \gamma^{\tau-1} \sum_{s_\tau, a_\tau} p(s_\tau, a_\tau; \boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a_\tau | s_\tau; \boldsymbol{w}) \sum_{t=\tau}^{\infty} \gamma^{t-\tau} \sum_{s_t, a_t} p(s_t, a_t | s_\tau, a_\tau; \boldsymbol{w}) R(s_t, a_t) \\ &= \sum_{\tau=1}^{\infty} \gamma^{\tau-1} \sum_{s_\tau, a_\tau} p(s_\tau, a_\tau; \boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a_\tau | s_\tau; \boldsymbol{w}) Q(s_\tau, a_\tau; \boldsymbol{w}). \\ &= \mathcal{H}_2(\boldsymbol{w})\end{aligned}$$

where in the third line we swapped the order of summation.

Using (37) we can write the second term in (39) as,

$$\begin{aligned}
&\sum_{t=1}^{\infty}\sum_{s_{1:t}}\sum_{a_{1:t}}\gamma^{t-1}p(s_{1:t},a_{1:t};\boldsymbol{w})\frac{\partial}{\partial\boldsymbol{w}}\log p(s_{1:t},a_{1:t};\boldsymbol{w})\frac{\partial^\top}{\partial\boldsymbol{w}}\log p(s_{1:t},a_{1:t};\boldsymbol{w})R(s_t,a_t)\\
&=\sum_{t=1}^{\infty}\sum_{\tau=1}^{t}\sum_{s_{1:t}}\sum_{a_{1:t}}\gamma^{t-1}p(s_{1:t},a_{1:t};\boldsymbol{w})\frac{\partial}{\partial\boldsymbol{w}}\log\pi(a_\tau|s_\tau;\boldsymbol{w})\frac{\partial^\top}{\partial\boldsymbol{w}}\log\pi(a_\tau|s_\tau;\boldsymbol{w})R(s_t,a_t)\\
&\quad+\sum_{t=1}^{\infty}\sum_{\substack{\tau_1,\tau_2=1\\ \tau_1\neq\tau_2}}^{t}\sum_{s_{1:t}}\sum_{a_{1:t}}\gamma^{t-1}p(s_{1:t},a_{1:t};\boldsymbol{w})\frac{\partial}{\partial\boldsymbol{w}}\log\pi(a_{\tau_1}|s_{\tau_1};\boldsymbol{w})\frac{\partial^\top}{\partial\boldsymbol{w}}\log\pi(a_{\tau_2}|s_{\tau_2};\boldsymbol{w})R(s_t,a_t).
\end{aligned}\tag{40}$$

By swapping the order of summation and following analogous calculations to those above, it can be shown that the first term in (40) is equal to $\mathcal{H}_1(\boldsymbol{w})$ as defined in (15). It remains to show that the second term in (40) is given by $\mathcal{H}_{12}(\boldsymbol{w})+\mathcal{H}_{12}^\top(\boldsymbol{w})$, with $\mathcal{H}_{12}(\boldsymbol{w})$ as given in (17). Splitting the second term in (40) into two terms,

$$\begin{aligned}
&\sum_{t=1}^{\infty}\sum_{\substack{\tau_1,\tau_2=1\\ \tau_1\neq\tau_2}}^{t}\sum_{s_{1:t}}\sum_{a_{1:t}}\gamma^{t-1}p(s_{1:t},a_{1:t};\boldsymbol{w})\frac{\partial}{\partial\boldsymbol{w}}\log\pi(a_{\tau_1}|s_{\tau_1};\boldsymbol{w})\frac{\partial^\top}{\partial\boldsymbol{w}}\log\pi(a_{\tau_2}|s_{\tau_2};\boldsymbol{w})R(s_t,a_t)\\
&=\sum_{t=1}^{\infty}\sum_{\tau_2=1}^{t}\sum_{\tau_1=1}^{\tau_2-1}\sum_{s_{1:t}}\sum_{a_{1:t}}\gamma^{t-1}p(s_{1:t},a_{1:t};\boldsymbol{w})\frac{\partial}{\partial\boldsymbol{w}}\log\pi(a_{\tau_1}|s_{\tau_1};\boldsymbol{w})\frac{\partial^\top}{\partial\boldsymbol{w}}\log\pi(a_{\tau_2}|s_{\tau_2};\boldsymbol{w})R(s_t,a_t)\\
&+\sum_{t=1}^{\infty}\sum_{\tau_1=1}^{t}\sum_{\tau_2=1}^{\tau_1-1}\sum_{s_{1:t}}\sum_{a_{1:t}}\gamma^{t-1}p(s_{1:t},a_{1:t};\boldsymbol{w})\frac{\partial}{\partial\boldsymbol{w}}\log\pi(a_{\tau_1}|s_{\tau_1};\boldsymbol{w})\frac{\partial^\top}{\partial\boldsymbol{w}}\log\pi(a_{\tau_2}|s_{\tau_2};\boldsymbol{w})R(s_t,a_t),
\end{aligned}\tag{41}$$

we will show that the first term is equal to $\mathcal{H}_{12}(\boldsymbol{w})$. Given this, it immediately follows that the second term is equal to $\mathcal{H}_{12}^\top(\boldsymbol{w})$. Using the Markov property of the transition dynamics and the policy it follows that the first term in (41) is given by,

$$\begin{aligned}
&\sum_{t=1}^{\infty}\sum_{\tau_2=1}^{t}\sum_{\tau_1=1}^{\tau_2-1}\sum_{s_{\tau_1},a_{\tau_1}}\gamma^{\tau_1-1}p(s_{\tau_1},a_{\tau_1};\boldsymbol{w})\frac{\partial}{\partial\boldsymbol{w}}\log\pi(a_{\tau_1}|s_{\tau_1};\boldsymbol{w})\\
&\times\sum_{s_{\tau_2},a_{\tau_2}}\gamma^{\tau_2-\tau_1}p(s_{\tau_2},a_{\tau_2}|s_{\tau_1},a_{\tau_1};\boldsymbol{w})\frac{\partial^\top}{\partial\boldsymbol{w}}\log\pi(a_{\tau_2}|s_{\tau_2};\boldsymbol{w})\sum_{s_t,a_t}\gamma^{t-\tau_2}p(s_t,a_t|s_{\tau_2},a_{\tau_2};\boldsymbol{w})R(s_t,a_t).
\end{aligned}$$

Rearranging the summation over t, τ_1 and τ_2 this can be rewritten in the form,

$$\begin{aligned}
&\sum_{\tau_1=1}^{\infty} \sum_{s_{\tau_1}, a_{\tau_1}} \gamma^{\tau_1 - 1} p(s_{\tau_1}, a_{\tau_1}; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a_{\tau_1} | s_{\tau_1}; \boldsymbol{w}) \\
&\quad \times \Bigg\{ \sum_{\tau_2=\tau_1+1}^{\infty} \sum_{s_{\tau_2}, a_{\tau_2}} \gamma^{\tau_2 - \tau_1} p(s_{\tau_2}, a_{\tau_2} | s_{\tau_1}, a_{\tau_1}; \boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} \log \pi(a_{\tau_2} | s_{\tau_2}; \boldsymbol{w}) \\
&\qquad \sum_{t=\tau_2}^{\infty} \sum_{s_t, a_t} \gamma^{t - \tau_2} p(s_t, a_t | s_{\tau_2}, a_{\tau_2}; \boldsymbol{w}) R(s_t, a_t) \Bigg\} \\
&= \sum_{\tau_1=1}^{\infty} \sum_{s_{\tau_1}, a_{\tau_1}} \gamma^{\tau_1 - 1} p(s_{\tau_1}, a_{\tau_1}; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a_{\tau_1} | s_{\tau_1}; \boldsymbol{w}) \\
&\quad \times \sum_{\tau_2=\tau_1+1}^{\infty} \sum_{s_{\tau_2}, a_{\tau_2}} \gamma^{\tau_2 - \tau_1} p(s_{\tau_2}, a_{\tau_2} | s_{\tau_1}, a_{\tau_1}; \boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} \log \pi(a_{\tau_2} | s_{\tau_2}; \boldsymbol{w}) Q(s_{\tau_2}, a_{\tau_2}; \boldsymbol{w}) \\
&= \sum_{\tau_1=1}^{\infty} \sum_{s_{\tau_1}, a_{\tau_1}} \gamma^{\tau_1 - 1} p(s_{\tau_1}, a_{\tau_1}; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a_{\tau_1} | s_{\tau_1}; \boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} Q(s_{\tau_1}, a_{\tau_1}; \boldsymbol{w}) \\
&= \mathcal{H}_{12}(\boldsymbol{w})
\end{aligned}$$

Where the penultimate line follows from (35). This completes the proof. □

A.2 Proof of Theorem 4

Recalling that the state-action value function takes the form, $Q(s, a; \boldsymbol{w}) = V(s; \boldsymbol{w}) + A(s, a; \boldsymbol{w})$, the matrices $\mathcal{H}_1(\boldsymbol{w})$ and $\mathcal{H}_2(\boldsymbol{w})$ can be written in the following forms,

$$\mathcal{H}_1(\boldsymbol{w}) = \mathcal{A}_1(\boldsymbol{w}) + \mathcal{V}_1(\boldsymbol{w}), \qquad \mathcal{H}_2(\boldsymbol{w}) = \mathcal{A}_2(\boldsymbol{w}) + \mathcal{V}_2(\boldsymbol{w}), \tag{42}$$

where,

$$\begin{aligned}
\mathcal{A}_1(\boldsymbol{w}) &= \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} p_\gamma(s, a; \boldsymbol{w}) A(s, a; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a | s; \boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} \log \pi(a | s; \boldsymbol{w}) \\
\mathcal{A}_2(\boldsymbol{w}) &= \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} p_\gamma(s, a; \boldsymbol{w}) A(s, a; \boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a | s; \boldsymbol{w}) \\
\mathcal{V}_1(\boldsymbol{w}) &= \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} p_\gamma(s, a; \boldsymbol{w}) V(s, a; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a | s; \boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} \log \pi(a | s; \boldsymbol{w}) \\
\mathcal{V}_2(\boldsymbol{w}) &= \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} p_\gamma(s, a; \boldsymbol{w}) V(s, a; \boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a | s; \boldsymbol{w}).
\end{aligned}$$

We begin with the following auxiliary lemmas.

Lemma 2. *Suppose we are given a Markov decision process with objective (1) and Markovian trajectory distribution (2). Provided that the policy satisfies the Fisher regularity conditions, then for any given parameter vector, $\boldsymbol{w} \in \mathcal{W}$, the matrices $\mathcal{V}_1(\boldsymbol{w})$ and $\mathcal{V}_2(\boldsymbol{w})$ satisfy the following relation $\mathcal{V}_1(\boldsymbol{w}) = -\mathcal{V}_2(\boldsymbol{w})$.*

Proof. As the policy satisfies the Fisher regularity conditions, then for any state, $s \in \mathcal{S}$, the following relation holds

$$\sum_{a\in\mathcal{A}} \pi(a|s;\boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a|s;\boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} \log \pi(a|s;\boldsymbol{w}) = -\sum_{a\in\mathcal{A}} \pi(a|s;\boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a|s;\boldsymbol{w}).$$

This means that $\mathcal{V}_1(\boldsymbol{w})$ can be written in the form

$$\begin{aligned}\mathcal{V}_1(\boldsymbol{w}) &= \sum_{s\in\mathcal{S}} p_\gamma(s;\boldsymbol{w}) V(s;\boldsymbol{w}) \sum_{a\in\mathcal{A}} \pi(a|s;\boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a|s;\boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} \log \pi(a|s;\boldsymbol{w}),\\ &= -\sum_{s\in\mathcal{S}} p_\gamma(s;\boldsymbol{w}) V(s;\boldsymbol{w}) \sum_{a\in\mathcal{A}} \pi(a|s;\boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a|s;\boldsymbol{w}) = -\mathcal{V}_2(\boldsymbol{w}),\end{aligned}$$

which completes the proof. □

Lemma 3. *Suppose we are given a Markov decision process with objective (1) and Markovian trajectory distribution (2). If the policy parameterisation has constant curvature with respect to the action space, then $\mathcal{A}_2(\boldsymbol{w}) = \mathbf{0}$.*

Proof. When a policy parameterisation has constant curvature with respect to the action space, then we use, $\mathcal{H}_\pi(s,\boldsymbol{w})$, to denote $\frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a|s;\boldsymbol{w})$, for each $a \in \mathcal{A}$. Recalling Definition 2, the matrix $\mathcal{A}_2(\boldsymbol{w})$ takes the form,

$$\begin{aligned}\mathcal{A}_2(\boldsymbol{w}) &= \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} p_\gamma(s,a;\boldsymbol{w}) A(s,a;\boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a|s;\boldsymbol{w}),\\ &= \sum_{s\in\mathcal{S}} p_\gamma(s;\boldsymbol{w}) \mathcal{H}_\pi(s,\boldsymbol{w}) \sum_{a\in\mathcal{A}} \pi(a|s;\boldsymbol{w}) A(s,a;\boldsymbol{w}).\end{aligned}$$

The relation $\mathcal{A}_2(\boldsymbol{w}) = \mathbf{0}$ follows because $\sum_{a\in\mathcal{A}} \pi(a|s;\boldsymbol{w}) A(s,a;\boldsymbol{w}) = 0$, for all $s \in \mathcal{S}$. □

Lemmas 2 & 3, along with the relation (42), directly imply the result of Theorem 4.

A.3 Proof of Theorem 5 and Definiteness Results

Theorem 5. *Proof.* The first result follows from the fact that when the policy is log-concave with respect to the policy parameters, then $\mathcal{H}_2(\boldsymbol{w})$ is a non-negative mixture of negative-definite matrices, which again is negative-definite (Boyd and Vandenberghe, 2004).

The second result follows because when the policy parameterisation has constant curvature with respect to the action space, then by Lemma 3 in Section A.2 $\mathcal{A}_2(\boldsymbol{w}) = \mathbf{0}$, so that $\mathcal{H}_2(\boldsymbol{w}) = \mathcal{A}_2(\boldsymbol{w}) + \mathcal{V}_2(\boldsymbol{w}) = \mathcal{V}_2(\boldsymbol{w}) = -\mathcal{V}_1(\boldsymbol{w})$, with

$$\begin{aligned}\mathcal{V}_1(\boldsymbol{w}) &= \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} p_\gamma(s,a;\boldsymbol{w}) V(s,a;\boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a|s;\boldsymbol{w}) \frac{\partial^\top}{\partial \boldsymbol{w}} \log \pi(a|s;\boldsymbol{w})\\ \mathcal{V}_2(\boldsymbol{w}) &= \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} p_\gamma(s,a;\boldsymbol{w}) V(s,a;\boldsymbol{w}) \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a|s;\boldsymbol{w}).\end{aligned}$$

The result now follows because $-\mathcal{V}_1(\boldsymbol{w})$ is negative-semidefinite for all $\boldsymbol{w} \in \mathcal{W}$. □

Lemma 4. *For any $\boldsymbol{w} \in \mathcal{W}$ the matrix $\mathcal{H}_{11}(\boldsymbol{w}) = \mathcal{H}_1(\boldsymbol{w}) + \mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w})$ is positive-semidefinite.*

Proof. This follows immediately from the form of $\mathcal{H}_1(\boldsymbol{w}) + \mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w})$ given by (40) in Theorem 3, which is positive-semidefinite since the reward function is assumed to be non-negative. □

A.4 Proof of Theorem 6

We first prove an auxiliary lemma about the gradient of the state value function in the case of a tabular policy. As we are considering a tabular policy we have a separate parameter vector $\boldsymbol{w}_s$ for each state $s \in \mathcal{S}$. We denote the parameter vector of the entire policy by $\boldsymbol{w}$, in which this is given by the concatenation of the parameter vectors of the different states. The dimension of $\boldsymbol{w}$ is given by $n = \sum_{s \in \mathcal{S}} n_s$. In order to show that tabular policies are value-consistent we start by relating the gradient of $V(\hat{s}; \boldsymbol{w})$ to the gradient of $V(\bar{s}; \boldsymbol{w})$, where the gradient is taken with respect to the policy parameters of state $\bar{s}$, while the policy parameters of the remaining states are held fixed.

Lemma 5. *Suppose we are given a Markov decision process with a tabular policy such that $V(s; \boldsymbol{w})$ is differentiable for each $s \in \mathcal{S}$. Given $\bar{s}, \hat{s} \in \mathcal{S}$, such that $\bar{s} \neq \hat{s}$, then we have that*

$$\frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} V(\hat{s}; \boldsymbol{w}) = p_{\text{hit}}(\hat{s} \to \bar{s}) \frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} V(\bar{s}; \boldsymbol{w}), \tag{43}$$

where the notation $\frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} V(\hat{s}; \boldsymbol{w})$ is used to denote the gradient of the state value function w.r.t. the policy parameter of state $\bar{s}$, with the policy parameters of all other states considered fixed. The term $p_{\text{hit}}(\hat{s} \to \bar{s})$ in (43) is given by

$$p_{\text{hit}}(\hat{s} \to \bar{s}) = \sum_{t=2}^{\infty} \gamma^{t-1} p(s_t = \bar{s} | s_1 = \hat{s}, s_\tau \neq \bar{s}, \tau = 1, ..., t-1; \boldsymbol{w}).$$

Furthermore, when Markov chain induced by the policy parameters is ergodic then $p_{\text{hit}} > 0$.

Proof. Given the equality $V(s; \boldsymbol{w}) = \sum_{a \in \mathcal{A}} \pi(a|s; \boldsymbol{w}) Q(s, a; \boldsymbol{w})$, we have that

$$\frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} V(\hat{s}; \boldsymbol{w}) = \sum_{a \in \mathcal{A}} \left(\frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} \pi(a|\hat{s}; \boldsymbol{w}) Q(\hat{s}, a; \boldsymbol{w}) + \pi(a|\hat{s}; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} Q(\hat{s}, a; \boldsymbol{w}) \right).$$

As the policy is tabular and $\hat{s} \neq \bar{s}$ we have that $\frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} \pi(a|\hat{s}; \boldsymbol{w}) = \mathbf{0}$, so that this simplifies to

$$\frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} V(\hat{s}; \boldsymbol{w}) = \sum_{a \in \mathcal{A}} \pi(a|\hat{s}; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} Q(\hat{s}, a; \boldsymbol{w}).$$

Using the fact that $Q(s, a; \boldsymbol{w}) = R(s, a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s, a) V(s'; \boldsymbol{w})$, we have

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} V(\hat{s}; \boldsymbol{w}) &= \gamma \sum_{s' \in \mathcal{S}} p(s'|\hat{s}; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} V(s'; \boldsymbol{w}) \\ &= \gamma p(\bar{s}|\hat{s}; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} V(\bar{s}; \boldsymbol{w}) + \gamma \sum_{\substack{s' \in \mathcal{S} \\ s' \neq \bar{s}}} p(s'|\hat{s}; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} V(s'; \boldsymbol{w}). \end{aligned} \tag{44}$$

Applying equation (44) recursively gives

$$\frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} V(\hat{s}; \boldsymbol{w}) = \sum_{t=2}^{\infty} \gamma^{t-1} p(s_t = \bar{s} | s_1 = \hat{s}, s_\tau \neq \bar{s}, \tau = 1, ..., t-1; \boldsymbol{w}) \frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} V(\bar{s}; \boldsymbol{w})$$
$$= p_{\text{hit}}(\hat{s} \to \bar{s}) \ \frac{\partial}{\partial \boldsymbol{w}_{\bar{s}}} V(\bar{s}; \boldsymbol{w}), \tag{45}$$

which completes the proof. The probability, $p(s_t = \bar{s} | s_1 = \hat{s}, s_\tau \neq \bar{s}, \tau = 1, ..., t-1; \boldsymbol{w})$, is equivalent to the probability that the first hitting time (of hitting state $\bar{s}$ when starting in state $\hat{s}$) is equal to t. The strict inequality, $p_{\text{hit}}(\hat{s} \to \bar{s}) > 0$, follows from the ergodicity of the Markov chain induced by $\boldsymbol{w}$. □

We are now ready to prove Theorem 6.

Theorem 6. *Proof.* Suppose that there exists $i \in \{1, ..., n\}$, $\boldsymbol{w} \in \mathcal{W}$ and $\hat{s} \in \mathcal{S}$ such that $\frac{\partial}{\partial w_i} V(\hat{s}; \boldsymbol{w}) \neq 0$, for some $\hat{s} \in \mathcal{S}$. As the policy parameterisation is tabular, then the i^{th} component of $\boldsymbol{w}$ corresponds to a policy parameter for a particular state, $\bar{s} \in \mathcal{S}$. From Lemma 5 it follows that

$$\frac{\partial}{\partial w_i} V(s; \boldsymbol{w}) = p_{\text{hit}}(s \to \bar{s}) \ \frac{\partial}{\partial w_i} V(\bar{s}; \boldsymbol{w}),$$

for all $s \in \mathcal{S}$. It follows that for states, $s \in \mathcal{S}$, for which $p_{\text{hit}}(s \to \bar{s}) > 0$ that we have

$$\text{sign}\left(\frac{\partial}{\partial w_i} V(s; \boldsymbol{w}) \right) = \text{sign}\left(\frac{\partial}{\partial w_i} V(\hat{s}; \boldsymbol{w}) \right),$$

while in states for which $p_{\text{hit}}(s \to \bar{s}) = 0$ we have $\text{sign}\big(\frac{\partial}{\partial w_i} V(s; \boldsymbol{w})\big) = 0$.

It remains to show that for states in which $p_{\text{hit}}(s \to \bar{s}) = 0$ that $\text{sign}\big(\frac{\partial}{\partial w_i} \pi(a|s; \boldsymbol{w})\big) = 0$, $\forall a \in \mathcal{A}$. This property follows immediately from the fact that the policy parameterisation is tabular and $p_{\text{hit}}(\bar{s} \to \bar{s}) \neq 0$. □

A.5 Proof of Theorem 7

Lemma 6. *Given a Markov decision process and a policy parameterisation that is ϵ-value-consistent, if there exists $i \in \{1, ..., n\}$ and $\hat{s} \in \mathcal{S}$ such that,*

$$\left| \frac{\partial}{\partial w_i} V(\hat{s}; \boldsymbol{w}) \big|_{\boldsymbol{w} = \boldsymbol{w}^*} \right| > \epsilon, \tag{46}$$

then for each $s \in \mathcal{S}$,

$$\text{sign}\left(\frac{\partial}{\partial w_i} V(s; \boldsymbol{w}) \right) = \text{sign}\left(\frac{\partial}{\partial w_i} V(\hat{s}; \boldsymbol{w}) \right).$$

Proof. In order to obtain a contradiction suppose that there exists $s \in \mathcal{S}$ such that,

$$\text{sign}\left(\frac{\partial}{\partial w_i} V(s; \boldsymbol{w}) \right) \neq \text{sign}\left(\frac{\partial}{\partial w_i} V(\hat{s}; \boldsymbol{w}) \right). \tag{47}$$

By definition 2 it follows that for all $s' \in \mathcal{S}$,

$$\text{sign}\left(\frac{\partial}{\partial w_i} V(s'; \boldsymbol{w})\right) = \text{sign}\left(\frac{\partial}{\partial w_i} V(s; \boldsymbol{w})\right), \tag{48}$$

or

$$\left|\frac{\partial}{\partial w_i} V(s'; \boldsymbol{w})\right| \leq \epsilon. \tag{49}$$

From (47) it follows that,

$$\left|\frac{\partial}{\partial w_i} V(\hat{s}; \boldsymbol{w})\right| \leq \epsilon. \tag{50}$$

This is a contradiction of (46), which completes the proof. □

Theorem 7. *Proof.* In order to obtain a contradiction suppose that there exists $i \in \{1, ..., n\}$ and $\hat{s} \in \mathcal{S}$ such that,

$$\left|\frac{\partial}{\partial w_i} V(\hat{s}; \boldsymbol{w})\Big|_{\boldsymbol{w}=\boldsymbol{w}^*}\right| > \epsilon. \tag{51}$$

We suppose that $\frac{\partial}{\partial w_i} V(\hat{s}; \boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} > \epsilon$ (an identical argument can be used for the case $\frac{\partial}{\partial w_i} V(\hat{s}; \boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} < -\epsilon$). As the policy parameterisation is ϵ-value-consistent it follows from lemma 6 that, for each $s \in \mathcal{S}$,

$$\frac{\partial}{\partial w_i} V(s; \boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} \geq 0. \tag{52}$$

In order to obtain a contradiction we will show that there is no $s \in \mathcal{S}$ for which (52) holds with equality. Given this property a contradiction is obtained because it follows that

$$\frac{\partial}{\partial w_i} U(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} = \mathbb{E}_{p_1(\boldsymbol{s})}\left[\frac{\partial}{\partial w_i} V(\boldsymbol{s}; \boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*}\right] > 0,$$

contradicting the fact that $\boldsymbol{w}^*$ is a local optimum of the objective function. Introducing the notation

$$\mathcal{S}_= = \left\{s \in \mathcal{S} \,\middle|\, \frac{\partial}{\partial w_i} V(s; \boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} = 0\right\},$$
$$\mathcal{S}_> = \left\{s \in \mathcal{S} \,\middle|\, \frac{\partial}{\partial w_i} V(s; \boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} > 0\right\},$$

we wish to show that $\mathcal{S}_= = \emptyset$. In particular, for a contradiction, suppose that $\mathcal{S}_= \neq \emptyset$. This means, given the ergodicity of the Markov chain induced by $\boldsymbol{w}^*$ and the fact that $\mathcal{S}_> \neq \emptyset$, that there exists $s \in \mathcal{S}_=$ and $s' \in \mathcal{S}_>$ such that $p(s'|s; \boldsymbol{w}^*) = \sum_{a \in \mathcal{A}} p(s'|s, a)\pi(a|s; \boldsymbol{w}^*) > 0$. We now consider the form of $\frac{\partial}{\partial \boldsymbol{w}} V(s; \boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*}$. In particular, we have

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{w}} V(s; \boldsymbol{w}) = &\sum_{a \in \mathcal{A}} \frac{\partial}{\partial \boldsymbol{w}} \pi(a|s; \boldsymbol{w})\left(R(a, s) + \gamma \sum_{s_{\text{next}} \in \mathcal{S}} p(s_{\text{next}}|s, a) V(s_{\text{next}}; \boldsymbol{w})\right) \\ &+ \gamma \sum_{a \in \mathcal{A}} \pi(a|s; \boldsymbol{w}) \sum_{s_{\text{next}} \in \mathcal{S}} p(s_{\text{next}}|s, a) \frac{\partial}{\partial \boldsymbol{w}} V(s_{\text{next}}; \boldsymbol{w}).\end{aligned}$$

As $s \in \mathcal{S}_=$, we have by value consistency that $\frac{\partial}{\partial w_i}\pi(a|s;\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} = 0$. This means that

$$\frac{\partial}{\partial w_i}V(s;\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} = \gamma \sum_{a\in\mathcal{A}} \pi(a|s;\boldsymbol{w}) \sum_{s_{\text{next}}\in\mathcal{S}} p(s_{\text{next}}|s,a)\frac{\partial}{\partial w_i}V(s_{\text{next}};\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} > 0.$$

The inequality follows from the fact that $p(s'|s;\boldsymbol{w}^*) > 0$, for some $s' \in \mathcal{S}_>$. This is a contradiction of the fact that $s_= \in \mathcal{S}_=$, so it follows that $\mathcal{S}_= = \emptyset$ and for all $s \in \mathcal{S}$ we have $\frac{\partial}{\partial w_i}V(s;\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} > 0$, which completes the proof. □

A.6 Proof of Theorem 8

Theorem 8. *Proof.* A optimisation method is said to affine invariant if, given any objective function (for which the optimisation technique is applicable), $f : \mathcal{W} \to \mathbb{R}$, and non-singular affine mapping, $T \in \mathbb{R}^{n\times n}$, the update of the objective $\tilde{f}(\boldsymbol{w}) = f(T\boldsymbol{w})$ is related to the update of the original objective through the same affine mapping, i.e., $\boldsymbol{v} + \Delta\boldsymbol{v}_{\text{step}} = T\big(\boldsymbol{w} + \Delta\boldsymbol{w}_{\text{step}}\big)$, in which $\boldsymbol{v} = T\boldsymbol{w}$ and $\Delta\boldsymbol{v}_{\text{step}}$ and $\Delta\boldsymbol{w}_{\text{step}}$ denote the respective steps in the parameter space.

We shall consider the second Gauss-Newton method, with the result for the diagonal approximate Newton method following similarly. Given a non-singular affine transformation, $T \in \mathbb{R}^{n\times n}$, define the objective, $\widehat{U}(\boldsymbol{w}) = U(T\boldsymbol{w}) = U(\boldsymbol{v})$, with $\boldsymbol{v} = T\boldsymbol{w}$, and denote the approximate Hessian of $\widehat{U}(\boldsymbol{w})$ by $\widehat{\mathcal{H}}_2(\boldsymbol{w})$. Given $\boldsymbol{w} \in \mathcal{W}$, then it is sufficient to show that,

$$T\boldsymbol{w}_{\text{new}} = T\left(\boldsymbol{w} - \alpha\widehat{\mathcal{H}}_2^{-1}(\boldsymbol{w})\frac{\partial}{\partial\boldsymbol{w}}\widehat{U}(\boldsymbol{w})\right) = \boldsymbol{v} - \alpha\mathcal{H}_2^{-1}(\boldsymbol{v})\frac{\partial}{\partial\boldsymbol{v}}U(\boldsymbol{v}) = \boldsymbol{v}_{\text{new}}, \qquad \forall\alpha \in \mathbb{R}^+.$$

Following calculations analogous to those in Section A.1 it can be shown that,

$$\frac{\partial}{\partial\boldsymbol{w}}\widehat{U}(\boldsymbol{w}) = \sum_{s,a} p_\gamma(s,a;T\boldsymbol{w})Q(s,a;T\boldsymbol{w})\frac{\partial}{\partial\boldsymbol{w}}\log p(\boldsymbol{a}|\boldsymbol{s};T\boldsymbol{w}),$$

$$\widehat{\mathcal{H}}_2(\boldsymbol{w}) = \sum_{s,a} p_\gamma(s,a;T\boldsymbol{w})Q(s,a;T\boldsymbol{w})\frac{\partial^2}{\partial\boldsymbol{w}^2}\log p(\boldsymbol{a}|\boldsymbol{s};T\boldsymbol{w}).$$

Using the relations

$$\frac{\partial}{\partial\boldsymbol{w}}\log\pi(a|s;T\boldsymbol{w}) = T^\top\frac{\partial}{\partial\boldsymbol{v}}\log\pi(a|s;\boldsymbol{v}),$$
$$\frac{\partial^2}{\partial\boldsymbol{w}^2}\log\pi(a|s;T\boldsymbol{w}) = T^\top\frac{\partial^2}{\partial\boldsymbol{v}^2}\log\pi(a|s;\boldsymbol{v})T,$$

it follows that

$$\frac{\partial}{\partial\boldsymbol{w}}\widehat{U}(\boldsymbol{w}) = T^\top\frac{\partial}{\partial\boldsymbol{v}}U(\boldsymbol{v}),$$
$$\widehat{\mathcal{H}}_2(\boldsymbol{w}) = T^\top\mathcal{H}_2(\boldsymbol{v})T.$$

From this we have, for any $\alpha \in \mathbb{R}^+$, that

$$T\boldsymbol{w}_{\text{new}} = T\left(\boldsymbol{w} - \alpha\widehat{\mathcal{H}}_2^{-1}(\boldsymbol{w})\frac{\partial}{\partial\boldsymbol{w}}\widehat{U}(\boldsymbol{w})\right) = \boldsymbol{v} - \alpha\mathcal{H}_2^{-1}(\boldsymbol{v})\frac{\partial}{\partial\boldsymbol{v}}U(\boldsymbol{v}) = \boldsymbol{v}_{\text{new}}, \qquad \forall\alpha \in \mathbb{R}^+.$$

which completes the proof. □

A.7 Proofs of Theorems 9 and 10

We begin by stating a well-known tool for analysis of convergence of iterative optimization methods. Given an iterative optimization method, defined through a mapping $G : \mathcal{W} \to \mathbb{R}^n$, where $\mathcal{W} \subseteq \mathbb{R}^n$, the local convergence at a point $\boldsymbol{w}^* \in \mathcal{W}$ is determined by the spectral radius of the Jacobian of G at $\boldsymbol{w}^*$, $\nabla G(\boldsymbol{w}^*)$. This is formalized through the well-known Ostrowski's Theorem, a formal proof of which can be found in the work of Ortega and Rheinboldt (1970).

Lemma 7 (Ostrowski's Theorem)**.** *Suppose that we have a mapping $G : \mathcal{W} \to \mathbb{R}^n$, where $\mathcal{W} \subset \mathbb{R}^n$, such that $\boldsymbol{w}^* \in \mathrm{int}(\mathcal{W})$ is a fixed-point of G and, furthermore, G is Fréchet differentiable at $\boldsymbol{w}^*$. If the spectral radius of $\nabla G(\boldsymbol{w}^*)$ satisfies $\rho(\nabla G(\boldsymbol{w}^*)) < 1$, then $\boldsymbol{w}^*$ is a point of attraction of G. Furthermore, if $\rho(\nabla G(\boldsymbol{w}^*)) > 0$, then the convergence towards $\boldsymbol{w}^*$ is linear and the rate is given by $\rho(\nabla G(\boldsymbol{w}^*))$.*

We now prove Theorems 9 and 10.

Theorem 9 (Convergence analysis for the first Gauss-Newton method)**.** *Proof.* A formal proof that G_1 is Fréchet differentiable can be found in Section 10.2.1 of Ortega and Rheinboldt (1970). We now demonstrate the form of $\nabla G_1(\boldsymbol{w}^*)$. For simplicity we shall assume that $(\mathcal{A}_1(\boldsymbol{w}^*) + \mathcal{A}_2(\boldsymbol{w}^*))^{-1}$ is differentiable. This is not a necessary condition, and a proof that does not make this assumption can be found in Section 10.2.1 of Ortega and Rheinboldt (1970). We have that,

$$G_1(\boldsymbol{w}) = \boldsymbol{w} - \alpha(\mathcal{A}_1(\boldsymbol{w}) + \mathcal{A}_2(\boldsymbol{w}))^{-1}\frac{\partial^\top}{\partial \boldsymbol{w}}U(\boldsymbol{w}),$$

so that $\nabla G_1(\boldsymbol{w})$ is given by

$$\nabla G_1(\boldsymbol{w}) = I - \alpha\frac{\partial}{\partial \boldsymbol{w}}(\mathcal{A}_1(\boldsymbol{w}) + \mathcal{A}_2(\boldsymbol{w}))^{-1}\frac{\partial^\top}{\partial \boldsymbol{w}}U(\boldsymbol{w}) - \alpha(\mathcal{A}_1(\boldsymbol{w}) + \mathcal{A}_2(\boldsymbol{w}))^{-1}\frac{\partial^2}{\partial \boldsymbol{w}^2}U(\boldsymbol{w}).$$

The fact that $\frac{\partial}{\partial \boldsymbol{w}}U(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^*} = \boldsymbol{0}$ means that

$$\nabla G_1(\boldsymbol{w}^*) = I - \alpha(\mathcal{A}_1(\boldsymbol{w}^*) + \mathcal{A}_2(\boldsymbol{w}^*))^{-1}\mathcal{H}(\boldsymbol{w}^*).$$

As $\mathcal{H}(\boldsymbol{w}^*)$ and $\mathcal{A}_1(\boldsymbol{w}^*) + \mathcal{A}_2(\boldsymbol{w}^*)$ are negative-definite, it follows that the eigenvalues of $(\mathcal{A}_1(\boldsymbol{w}^*) + \mathcal{A}_2(\boldsymbol{w}^*))^{-1}\mathcal{H}(\boldsymbol{w}^*)$ are positive. Hence,

$$\rho(\nabla G_1(\boldsymbol{w}^*)) = \max\left\{|1 - \alpha\lambda_{\min}|, |1 - \alpha\lambda_{\max}|\right\}, \tag{53}$$

with $\lambda_{\min}$ and $\lambda_{\max}$ respectively denoting the minimal and maximal eigenvalues of $(\mathcal{A}_1(\boldsymbol{w}^*)+\mathcal{A}_2(\boldsymbol{w}^*))^{-1}\mathcal{H}(\boldsymbol{w}^*)$. Hence, $\rho(\nabla G_1(\boldsymbol{w}^*)) < 1$ provided that $\alpha \in (0, 2\lambda_{\max}^{-1})$, or, written in terms of the spectral radius, $\alpha \in (0, 2/\rho((\mathcal{A}_1(\boldsymbol{w}^*) + \mathcal{A}_2(\boldsymbol{w}^*))^{-1}\mathcal{H}(\boldsymbol{w}^*)))$.

When the policy parameterisation is value-consistent with respect to the given MDP, then from Theorem 7 $\mathcal{H}_{12}(\boldsymbol{w}^*) + \mathcal{H}_{12}^\top(\boldsymbol{w}^*) = \boldsymbol{0}$, so that $\mathcal{H}(\boldsymbol{w}^*) = \mathcal{A}_1(\boldsymbol{w}^*) + \mathcal{A}_2(\boldsymbol{w}^*)$. It then follows that $\nabla G_1(\boldsymbol{w}^*) = (1-\alpha)I$. Convergence for this case follows in the same manner. □

Theorem 10 (Convergence analysis for the second Gauss-Newton method)**.** *Proof.* The formulas (29) and (31) follow as in the proof of Theorem 9. Using the same approach as in Theorem 9, it can be shown that $\rho(\nabla G_2(\boldsymbol{w}^*)) < 1$ provided that, $\alpha \in (0, 2/\rho(\mathcal{H}_2(\boldsymbol{w}^*)^{-1}\mathcal{H}(\boldsymbol{w}^*)))$.

As $\mathcal{H}(\boldsymbol{w}^*)$ and $\mathcal{H}_2(\boldsymbol{w}^*)$ are negative-definite the eigenvalues of $\mathcal{H}_2(\boldsymbol{w}^*)^{-1}\mathcal{H}(\boldsymbol{w}^*)$ are positive. Furthermore, as $\mathcal{H}(\boldsymbol{w}^*) = \mathcal{H}_{11}(\boldsymbol{w}^*) + \mathcal{H}_2(\boldsymbol{w}^*)$, and, by Lemma 4, $\mathcal{H}_{11}(\boldsymbol{w}^*)$ is positive-semidefinite, it follows that the eigenvalues of $\mathcal{H}_2(\boldsymbol{w}^*)^{-1}\mathcal{H}(\boldsymbol{w}^*)$ all lie in the range $(0, 1]$. This means that $\alpha \in (0, 2)$ is sufficient to ensure that $\rho(\nabla G_2(\boldsymbol{w}^*)) < 1$.

□

A.8 Proof of Theorem 11

Theorem 11. *Proof.* We use the notation $\nabla^{10}\mathcal{Q}(\boldsymbol{w}_j, \boldsymbol{w}_k)$ to denote the derivative with respect to the first variable of Q, evaluated at $(\boldsymbol{w}_j, \boldsymbol{w}_k)$, and similarly $\nabla^{20}\mathcal{Q}(\boldsymbol{w}_j, \boldsymbol{w}_k)$ for the second derivative and $\nabla^{01}\mathcal{Q}(\boldsymbol{w}_j, \boldsymbol{w}_k)$ for the derivative with respect to the second variable etc. The idea of the proof is simple and consists of performing a Taylor expansion of $\nabla^{10}\mathcal{Q}(\boldsymbol{w}, \boldsymbol{w}_k)$. As $\mathcal{Q}$ is assumed to be twice continuously differentiable in the first component this Taylor expansion is possible and gives

$$\nabla^{10}\mathcal{Q}(\boldsymbol{w}_{k+1}, \boldsymbol{w}_k) = \nabla^{10}\mathcal{Q}(\boldsymbol{w}_k, \boldsymbol{w}_k) + \nabla^{20}\mathcal{Q}(\boldsymbol{w}_k, \boldsymbol{w}_k)(\boldsymbol{w}_{k+1} - \boldsymbol{w}_k) + \mathcal{O}(\|\boldsymbol{w}_{k+1} - \boldsymbol{w}_k\|^2). \quad (54)$$

As $\boldsymbol{w}_{k+1} = \operatorname{argmax}_{\boldsymbol{w}\in\mathcal{W}} \mathcal{Q}(\boldsymbol{w}, \boldsymbol{w}_k)$ it follows that $\nabla^{10}\mathcal{Q}(\boldsymbol{w}_{k+1}, \boldsymbol{w}_k) = 0$. This means that, upon ignoring higher order terms in $\boldsymbol{w}_{k+1} - \boldsymbol{w}_k$, the Taylor expansion (54) can be rewritten into the form

$$\boldsymbol{w}_{k+1} - \boldsymbol{w}_k = -\nabla^{20}\mathcal{Q}(\boldsymbol{w}_k, \boldsymbol{w}_k)^{-1}\nabla^{10}\mathcal{Q}(\boldsymbol{w}_k, \boldsymbol{w}_k). \quad (55)$$

The proof is completed by observing that

$$\nabla^{10}\mathcal{Q}(\boldsymbol{w}_k, \boldsymbol{w}_k) = \frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w})|_{w=w_k}, \qquad \nabla^{20}\mathcal{Q}(\boldsymbol{w}_k, \boldsymbol{w}_k) = \mathcal{H}_2(\boldsymbol{w}_k).$$

The second statement follows because in the case where the log-policy is quadratic the higher order terms in the Taylor expansion vanish. □

A.9 Proof of Theorem 12

Theorem 12. *Proof.* In the EM-algorithm the update of the policy parameters takes the form

$$G_{\mathrm{EM}}(\boldsymbol{w}_k) = \operatorname{argmax}_{\boldsymbol{w}\in\mathcal{W}} \mathcal{Q}(\boldsymbol{w}, \boldsymbol{w}_k),$$

where the function $\mathcal{Q}(\boldsymbol{w}, \boldsymbol{w}')$ is given by

$$\mathcal{Q}(\boldsymbol{w}, \boldsymbol{w}') = \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} p_\gamma(s, a; \boldsymbol{w}')Q(s, a; \boldsymbol{w}')\Big[\log \pi(a|s; \boldsymbol{w})\Big].$$

Note that $\mathcal{Q}$ is a two parameter function, where the first parameter occurs inside the bracket, while the second parameter occurs outside the bracket. Also note that $\mathcal{Q}(\boldsymbol{w}, \boldsymbol{w}')$ satisfies

the following identities

$$\nabla^{10}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}') = \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} p_\gamma(s,a;\boldsymbol{w}')Q(s,a;\boldsymbol{w}')\left[\frac{\partial}{\partial \boldsymbol{w}}\log\pi(a|s;\boldsymbol{w})\right],$$

$$\nabla^{20}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}') = \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} p_\gamma(s,a;\boldsymbol{w}')Q(s,a;\boldsymbol{w}')\left[\frac{\partial^2}{\partial \boldsymbol{w}^2}\log\pi(a|s;\boldsymbol{w})\right],$$

$$\nabla^{11}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}') = \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} \frac{\partial}{\partial \boldsymbol{w}}\left(p_\gamma(s,a;\boldsymbol{w}')Q(s,a;\boldsymbol{w}')\right)\frac{\partial^\top}{\partial \boldsymbol{w}}\log\pi(a|s\boldsymbol{s};\boldsymbol{w}).$$

Here we have used the notation ∇^{ij} to denote the i^{th} derivative with respect to the first parameter and the j^{th} derivative with respect to the second parameter. Note that when we set $\boldsymbol{w}=\boldsymbol{w}'$ in the first two of these terms we have $\nabla^{10}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}) = \frac{\partial}{\partial \boldsymbol{w}}U(\boldsymbol{w})$, $\nabla^{20}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}) = \mathcal{H}_2(\boldsymbol{w})$. A key identity that we need for the proof is that $\nabla^{11}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}) = \mathcal{H}_1(\boldsymbol{w}) + \mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w})$. This follows from the observation that $\frac{\partial}{\partial \boldsymbol{w}}U(\boldsymbol{w}) = \nabla^{10}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w})$, so that

$$\frac{\partial^2}{\partial \boldsymbol{w}^2}U(\boldsymbol{w}) = \frac{\partial}{\partial \boldsymbol{w}}\Big(\nabla^{10}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w})\Big) = \nabla^{20}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}) + \nabla^{11}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}),$$

so that

$$\begin{aligned}\mathcal{H}_1(\boldsymbol{w}) + \mathcal{H}_{12}(\boldsymbol{w}) + \mathcal{H}_{12}^\top(\boldsymbol{w}) = \mathcal{H}(\boldsymbol{w}) - \mathcal{H}_2(\boldsymbol{w}) &= \nabla^{20}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}) + \nabla^{11}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}) - \nabla^{20}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}),\\ &= \nabla^{11}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}),\end{aligned}$$

as claimed.

Now, to calculate the matrix $\nabla G_{\text{EM}}(\boldsymbol{w}^*)$ we perform a Taylor series expansion of $\nabla^{10}\mathcal{Q}(\boldsymbol{w},\boldsymbol{w}')$ in both parameters around the point $(\boldsymbol{w}^*,\boldsymbol{w}^*)$, and evaluated at $(\boldsymbol{w}_{k+1},\boldsymbol{w}_k)$, which gives

$$\begin{aligned}\nabla^{10}\mathcal{Q}(\boldsymbol{w}_{k+1},\boldsymbol{w}_k) = \nabla^{10}\mathcal{Q}(\boldsymbol{w}^*,\boldsymbol{w}^*) + \nabla^{20}\mathcal{Q}(\boldsymbol{w}^*,\boldsymbol{w}^*)\big(\boldsymbol{w}_{k+1}-\boldsymbol{w}^*\big)\\ + \nabla^{11}\mathcal{Q}(\boldsymbol{w}^*,\boldsymbol{w}^*)\big(\boldsymbol{w}_k-\boldsymbol{w}^*\big) + \ldots\end{aligned}$$

As $\boldsymbol{w}^*$ is a local optimum of $U(\boldsymbol{w})$ we have that $\nabla^{10}\mathcal{Q}(\boldsymbol{w}^*,\boldsymbol{w}^*) = \mathbf{0}$. Furthermore, as the sequence $\{\boldsymbol{w}_k\}_{k\in\mathbb{N}}$ was generated by the EM-algorithm, we have, for each $k\in\mathbb{N}$, that $\boldsymbol{w}_{k+1} = \text{argmax}_{\boldsymbol{w}\in\mathcal{W}}\ \mathcal{Q}(\boldsymbol{w},\boldsymbol{w}_k)$, which implies that $\nabla^{10}\mathcal{Q}(\boldsymbol{w}_{k+1},\boldsymbol{w}_k) = \mathbf{0}$. Finally, as $\nabla^{20}\mathcal{Q}(\boldsymbol{w}^*,\boldsymbol{w}^*) = \mathcal{H}_2(\boldsymbol{w}^*)$ and $\nabla^{11}\mathcal{Q}(\boldsymbol{w}^*,\boldsymbol{w}^*) = \mathcal{H}_1(\boldsymbol{w}^*)$ we have

$$\mathbf{0} = \mathcal{H}_2(\boldsymbol{w}^*)(\boldsymbol{w}_{k+1}-\boldsymbol{w}^*) + \big(\mathcal{H}_1(\boldsymbol{w}^*) + \mathcal{H}_{12}(\boldsymbol{w}^*) + \mathcal{H}_{12}^\top(\boldsymbol{w}^*)\big)(\boldsymbol{w}_k-\boldsymbol{w}^*) + \ldots$$

Using the fact that $\boldsymbol{w}_{k+1} = G_{\text{EM}}(\boldsymbol{w}_k)$ and $\boldsymbol{w}^* = G_{\text{EM}}(\boldsymbol{w}^*)$, taking the limit $k\to\infty$ gives

$$\mathbf{0} = \mathcal{H}_2(\boldsymbol{w}^*)\nabla G_{\text{EM}}(\boldsymbol{w}^*) + \mathcal{H}_1(\boldsymbol{w}^*) + \mathcal{H}_{12}(\boldsymbol{w}^*) + \mathcal{H}_{12}^\top(\boldsymbol{w}^*),$$

so that

$$\nabla G_{\text{EM}}(\boldsymbol{w}^*) = -\mathcal{H}_2^{-1}(\boldsymbol{w}^*)\big(\mathcal{H}_1(\boldsymbol{w}^*) + \mathcal{H}_{12}(\boldsymbol{w}^*) + \mathcal{H}_{12}^\top(\boldsymbol{w}^*)\big) = I - \mathcal{H}_2^{-1}(\boldsymbol{w}^*)\mathcal{H}(\boldsymbol{w}^*).$$

In the case where the policy parameterisation value-consistent with respect to the given MDP then we have $\mathcal{H}_{12}(\boldsymbol{w}^*) + \mathcal{H}_{12}(\boldsymbol{w}^*)^\top = \mathbf{0}$, so that $\nabla G_{\text{EM}}(\boldsymbol{w}^*) = I - \mathcal{H}_2^{-1}(\boldsymbol{w}^*)\mathcal{A}_1(\boldsymbol{w}^*)$. The rest of the proof follows from the result in Theorem 10 when considering $\alpha = 1$. □

Appendix B. Further Details for Estimation of Preconditioners and the Gauss-Newton Update Direction

B.1 Recurrent State Search Direction Evaluation for Second Gauss-Newton Method

In the work of Williams (1992) a sampling algorithm was provided for estimating the gradient of an infinite horizon MDP with average rewards. This algorithm makes use of a recurrent state, which we denote by $\boldsymbol{s}^*$. In Algorithm 2 we detail a straightforward extension of this algorithm to the estimation the approximate Hessian, $\mathcal{H}_2(\boldsymbol{w})$, in this MDP framework. The analogous algorithm for the estimation of the diagonal matrix, $\mathcal{D}_2(\boldsymbol{w})$, follows similarly. In Algorithm 2 we make use of an *eligibility trace* for both the gradient and the approximate Hessian, which we denote by $\boldsymbol{\Phi}^1$ and $\boldsymbol{\Phi}^2$ respectively. The estimates (up to a positive scalar) of the gradient and the approximate Hessian are denoted by $\boldsymbol{\Delta}^1$ and $\boldsymbol{\Delta}^2$ respectively.

B.2 Inversion of Preconditioning Matrices

A computational bottleneck of Newton's method is the inversion of the Hessian matrix, which scales with $\mathcal{O}(n^3)$. In a standard application of Newton's method this inversion is performed during each iteration, and in large parameter systems this becomes prohibitively costly. We now consider the inversion of the preconditioning matrix in proposed Gauss-Newton methods.

Firstly, in the diagonal forms of the Gauss-Newton methods the preconditioning matrix is diagonal, so that the inversion of this matrix is trivial and scales linearly in the number of parameters. In general the preconditioning matrix of the full Gauss-Newton methods will have no form of sparsity, and so no computational savings will be possible when inverting the preconditioning matrix. There is, however, a source of sparsity that allows for the efficient inversion of $\mathcal{H}_2$ in certain cases of interest. In particular, any product structure (with respect to the control parameters) in the model of the agent's behaviour will lead to sparsity in $\mathcal{H}_2$. For example, in partially observable Markov decision processes in which the behaviour of the agent is modeled through a finite state controller (Meuleau et al., 1999) there are three functions that are to be optimized, the initial belief distribution, the belief transition dynamics and the policy. In this case the dynamics of the system are given by,

$$p(s', o', b', a'|s, o, b, a; \boldsymbol{v}, \boldsymbol{w}) = p(s'|s, a)p(o'|s')p(b'|b, o'; \boldsymbol{v})\pi(a'|b', o'; \boldsymbol{w}),$$

in which $o \in \mathcal{O}$ is an observation from a finite observation space, $\mathcal{O}$, and $b \in \mathcal{B}$ is the belief state from a finite belief space, $\mathcal{B}$. The initial belief is given by the initial belief distribution, $p(b|o; \boldsymbol{u})$. The parameters to be optimized in this system are $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$. It can be seen that in this system $\mathcal{H}_2(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ is block-diagonal (across the parameters $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$) and the matrix inversion can be performed more efficiently by inverting each of the block matrices individually. By contrast, the Hessian $\mathcal{H}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ does not exhibit any such sparsity properties.

Algorithm 2: Recurrent state sampling algorithm to estimate the search direction of the second Gauss-Newton method. The algorithm is applicable to Markov decision processes with an infinite planning horizon and average rewards.

Input: Policy parameter, $\boldsymbol{w} \in \mathcal{W}$,
Number of restarts, $N \in \mathbb{N}$.

Sample a state from the initial state distribution:

$$s_1 \sim p_1(\cdot).$$

for $i = 1, ..., N$ **do**

Given the current state, sample an action from the policy:

$$a_t \sim \pi(\cdot|s_t; \boldsymbol{w}).$$

if $s_t \neq s^*$, **then**

update the eligibility traces:

$$\boldsymbol{\Phi}^1 \leftarrow \boldsymbol{\Phi}^1 + \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a_t|s_t; \boldsymbol{w}) \qquad \boldsymbol{\Phi}^2 \leftarrow \boldsymbol{\Phi}^2 + \frac{\partial^2}{\partial \boldsymbol{w}^2} \log \pi(a_t|s_t; \boldsymbol{w})$$

else

reset the eligibility traces:

$$\boldsymbol{\Phi}^1 = \mathbf{0}, \qquad \boldsymbol{\Phi}^2 = \mathbf{0}.$$

end

Update the estimates of the $\frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w})$ and $\mathcal{H}_2(\boldsymbol{w})$:

$$\boldsymbol{\Delta}^1 \leftarrow \boldsymbol{\Delta}^1 + R(a_t, s_t)\boldsymbol{\Phi}^1, \qquad \boldsymbol{\Delta}^2 \leftarrow \boldsymbol{\Delta}^2 + R(a_t, s_t)\boldsymbol{\Phi}^2.$$

Sample state from the transition dynamics:

$$s_{t+1} \sim p(\cdot|a_t, s_t).$$

Update time-step, $t \leftarrow t + 1$.

end

return $\boldsymbol{\Delta}^1$ and $\boldsymbol{\Delta}^2$, which, up to a positive multiplicative constant, are estimates of $\frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w})$ and $\mathcal{H}_2(\boldsymbol{w})$.

Appendix C. Experiments

C.1 Non-Linear Navigation Experiment

The state-space of the problem is two-dimensional, $\boldsymbol{s} = (s^1, s^2)$, in which s^1 is the agent's position and s^2 is the agent's velocity. The control is one-dimensional and the dynamics of

the system is given as follows,

$$s^1_{t+1} = s^1_t + \frac{1}{1+e^{-u_t}} - 0.5 + \kappa,$$
$$s^2_{t+1} = s^2_t - 0.1 s^1_{t+1} + \kappa,$$

with κ a zero-mean Gaussian random variable with standard deviation $\sigma_\kappa = 0.02$. The agent starts in the state $\boldsymbol{s} = (0, 1)$, with the addition of Gaussian noise with standard deviation 0.001, and the objective is for the agent to reach the target state, $\boldsymbol{s}_{\text{target}} = (0, 0)$. We use the same policy as in Vlassis et al. (2009), which is given by $a_t = (\boldsymbol{w} + \boldsymbol{\epsilon}_t)^\top \boldsymbol{s}_t$, with control parameters, $\boldsymbol{w}$, and $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\boldsymbol{\epsilon}_t; \mathbf{0}, \sigma^2_\epsilon I)$. The objective function is non-trivial for $\boldsymbol{w} \in [0, 60] \times [-8, 0]$. In the experiment the initial control parameters were sampled from the region $\boldsymbol{w}_0 \in [0, 60] \times [-8, 0]$. In all algorithms 50 trajectories were sampled during each training iteration and used to estimate the search direction. We consider a finite planning horizon, $H = 80$.

C.2 N-link Rigid Manipulator Experiment

The state of the system is given by $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}} \in \mathbb{R}^N$, where $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ denote the angles, velocities and accelerations of the joints respectively, while the control variables are the torques applied to the joints $\boldsymbol{\tau} \in \mathbb{R}^N$. The nonlinear state equations of the system are given by (Spong et al., 2005),

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\dot{\boldsymbol{q}}, \boldsymbol{q})\dot{\boldsymbol{q}} + g(\boldsymbol{q}) = \boldsymbol{\tau}, \tag{56}$$

where $M(\boldsymbol{q})$ is the inertia matrix, $C(\dot{\boldsymbol{q}}, \boldsymbol{q})$ denotes the Coriolis and centripetal forces and $g(\boldsymbol{q})$ is the gravitational force. While this system is highly nonlinear it is possible to define an appropriate control function $\hat{\boldsymbol{\tau}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ that results in linear dynamics in a different state-action space. This technique is known as feedback linearisation (Khalil, 2001), and in the case of an N-link rigid manipulator recasts the torque action space into the acceleration action space. This means that the state of the system is now given by $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$, while the control is $\boldsymbol{a} = \ddot{\boldsymbol{q}}$. Ordinarily in such problems the reward would be a function of the generalized co-ordinates of the end effector, which results in a non-trivial reward function in terms of $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$. This can be accounted for by modelling the reward function as a mixture of Gaussians (Hoffman et al., 2009), but for simplicity we consider the simpler problem where the reward is a function of $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ directly.

We consider the finite horizon undiscounted problem in this section, so that the gradient of the objective function takes the form

$$\frac{\partial}{\partial \boldsymbol{w}} U(\boldsymbol{w}) = \int\int ds da \frac{\partial}{\partial \boldsymbol{w}} \log \pi(a|s; \boldsymbol{w}) \sum_{t=1}^{H} p_t(s, a; \boldsymbol{w}) Q(s, a, t; \boldsymbol{w}),$$

with the preconditioning matrices of natural gradient ascent and the Gauss-Newton methods taking analogous forms. It can be shown that for the policy parameterisation given in (33) the derivative of $\log \pi(a|s; \boldsymbol{w})$ is quadratic in (s, a), for all $(s, a) \in \mathcal{S} \times \mathcal{A}$. This means that to calculate the search directions of gradient ascent, natural gradient ascent, expectation maximization and the Gauss-Newton methods it is sufficient to calculate the

first two moments of $p_t(s, a; \boldsymbol{w})Q(s, a, t; \boldsymbol{w})$ w.r.t. (s, a), for each $t \in \{1, ..., H\}$. These calculations can be done using the methods presented in the work of Furmston (2012).

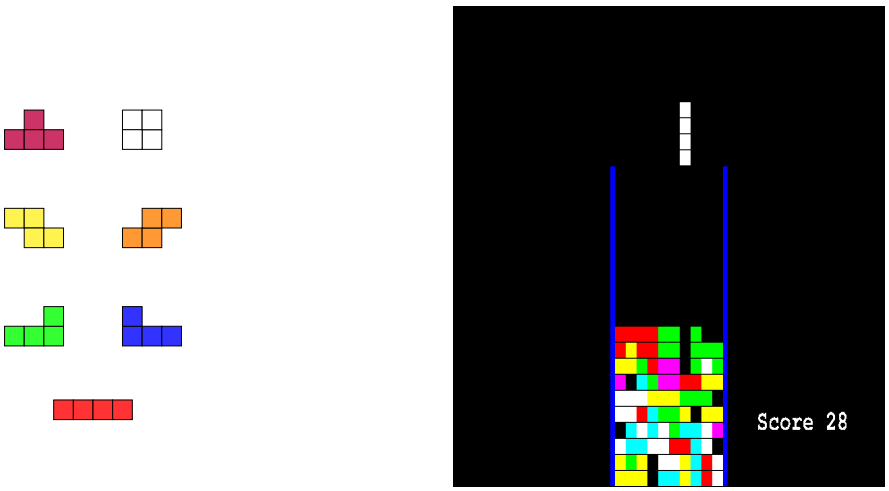
For all algorithms that required the specification of a step size we ran the experiment over a collection of step size sequences and use the optimal step size sequence in the final experiment. In both steepest gradient ascent and natural gradient ascent we considered the following fixed step sizes: 0.001, 0.01, 0.1, 1, 10, 20, 30, 100 and 250. We were unable to obtain any reasonable results with steepest gradient ascent with any of these fixed step sizes, for which reason the results are omitted. In natural gradient ascent we found 30 to be the best step size of those considered. In the Gauss-Newton method we considered the following fixed step sizes: 10, 20, 30, 100 and 250 and found that the fixed step size of 30 gave consistently good results without overstepping in the parameter space. The smaller step sizes obtained better results than expectation maximization, but less than the fixed step size of 30. The larger step sizes often found superior results, but would sometimes overstep in the parameter space. For these reasons we used the fixed step size of 30 in the final experiment.

C.3 Tetris Experiment

In Tetris there exists a board, which is typically a 20×10 grid, which is empty at the beginning of a game. During each stage of the game a four block piece, called a tetrzoid, appears at the top of the board and begins to fall down the board. While the tetrzoid is moving the player is allowed to rotate the tetrzoid and to move it left or right. The tetrzoid stops moving once it reaches either the bottom of the board or a previously positioned tetrzoid. In this manner the board begins to fill up with tetrzoid pieces. There are seven different variations of tetrzoid, as shown in Figure 5a. When a horizontal line of the board is completely filled with (pieces of) tetrzoids the line is removed from the board and the player receives a score of one. The game terminates when the player is not able to fully place a tetrzoid on the board due to insufficient space remaining on the board. An example configuration of the board during a game of Tetris is given in Figure 5b. More details on the game of Tetris can be found in the work of Fahey (2003). As in other applications of Tetris in the reinforcement learning literature (Kakade, 2002; Bertsekas and Ioffe, 1996) we consider a simplified version of the game in which the current tetrzoid remains above the board until the player decides upon a desired rotation and column position for the tetrzoid.

We use the same procedure to evaluate the search direction for all the algorithms in the experiment. Irrespective of the policy, a game of Tetris is guaranteed to terminate after a finite number of turns (Bertsekas and Ioffe, 1996). We therefore model each game as an absorbing state MDP. The reward at each time step is equal to the number of lines deleted. We use a recurrent state approach (Williams, 1992) to estimate the gradient, using the empty board as a recurrent state. (Since a new game starts with an empty board this state is recurrent.[6]) We use analogous versions of this recurrent state approach for natural gradient ascent, the diagonal Gauss-Newton method and the full Gauss-Newton

6. This is actually an approximation because it does not take into account that the state is given by the configuration of the board and the current piece, so this particular 'recurrent state' ignores the current piece. Empirically we found that this approximation gave better results, presumably due to reduced variance in the estimands, and there is no reason to believe that it is unfairly biasing the comparison between the various parametric policy search methods.



(a) Tetris : Tetrzoids

(b) Tetris : Game Board

Figure 5: A graphical illustration of the game of tetris with (a) the collection of possible pieces, or tetrozoids, of which there are seven (b) a possible configuration of the board, which in this example is of height 20 and width 10.

method. As in the work of Kakade (2002), we use the sample trajectories obtained during the gradient evaluation to estimate the Fisher information matrix. During each training iteration an approximation of the search direction is obtained by sampling 1000 games, using the current policy to sample the games. It is computationally very expensive to perform experiments on the Tetris domain. When performing the experiment we found that it would be prohibitively expensive to perform an extensive sweep over different step size sequences for all of the different algorithms. For this reason we decided to implement a simple line search in this domain. Given the current approximate search direction we use the following basic line search method to obtain a step size: For every step size in a given finite set of step sizes sample a set number of games and then return the step size with the maximal score over these games. In practice, in order to reduce the susceptibility to random noise, we used the same simulator seed for each possible step size in the set. In this line search procedure we sampled 1000 games for each of the possible step sizes. We use the same set of step sizes

$$\big\{0.1, 0.5, 1.0, 2.0, 4.0, 8.0, 16.0, 32.0, 64.0, 128.0\big\}.$$

in all of the different training algorithms in the experiment. To reduce the amount of noise in the results we use the same set of simulator seeds in the search direction evaluation for each of the algorithms considered in the experiment. In particular, we generate a $n_{\text{experiments}} \times n_{\text{iterations}}$ matrix of simulator seeds, with $n_{\text{experiments}}$ the number of repetitions of the experiment and $n_{\text{iterations}}$ the number of training iterations in each experiment. We use this one matrix of simulator seeds in all of the different training algorithms, with the element in the j^{th} column and i^{th} row corresponding to the simulator seed of the j^{th} training iteration of the i^{th} experiment. In a similar manner, the set of simulator seeds we use for the line search procedure is the same for all of the different training algorithms. Finally, to

make the line search consistent among all of the different training algorithms we normalize the search direction and use the resulting unit vector in the line search procedure.

C.4 Robot Arm Experiment

The domain in the robot arm experiment is episodic, with each episode 20 seconds in length. The state of the domain is given by the angles and velocities of the seven joints in the robot arm, along with the Cartesian coordinates of the ball. The action is given by the joint accelerations of the robot arm. We denote the position of the cup and the ball by $(x_c, y_c, z_c) \in \mathbb{R}^3$ and $(x_b, y_b, z_b) \in \mathbb{R}^3$ respectively. The reward function is given by,

$$r(x_c, y_c, x_b, y_b, t) = \begin{cases} -20\big((x_c - x_b)^2 + (y_c - y_b)^2\big) & \text{if } t = t_c, \\ 0 & \text{if } t \neq t_c, \end{cases}$$

in which t_c is the moment the ball crosses the z-plane (level with the cup) in a downward direction. If no such t_c exists then the reward of the episode is given by -100.

We use the motor primitive framework (Ijspeert et al., 2002, 2003; Schaal et al., 2007; Kober and Peters, 2011) in this domain, applying a separate motor primitive to each dimension of the action space. Each motor primitive consists of a parametrized curve that models the desired action sequence (for the respective dimension of the action space) through the course of the episode. Given this collection of motor primitives the control engine within the simulator tries to follow the desired action sequence as closely as possible while also satisfying the constraints on the system, such as the physical constraints on the torques that can safely be applied without damaging the robot arm. As in the work of Kober and Peters (2011) we use dynamic motor primitives, using 10 shape parameters for each of the individual motor primitives. The robot arm has 7 joints, so that there are 70 motor primitive parameters in total. We optimize the parameters of the motor primitives by considering the MDP induced by this motor primitive framework. The action space corresponds to the space of possible motor primitives, so that $\mathcal{A} = \mathbb{R}^{70}$. There is no state space in this MDP and the planning horizon is 1, so that this MDP is effectively a bandit problem. The reward of an action is equal to the total reward of the episode induced by the motor primitive. We consider a policy of the form,

$$\pi(a; \boldsymbol{w}) = \mathcal{N}\big(a|\boldsymbol{\mu}, \big(LL^*\big)^{-1}\big),$$

with $\boldsymbol{w} = (\boldsymbol{\mu}, L)$, $\boldsymbol{\mu}$ the mean of the Gaussian and LL^* the Cholesky decomposition of the precision matrix. We consider a diagonal precision matrix, which results in a total of 140 policy parameters.

In this experiment we compare gradient ascent, natural gradient ascent, expectation maximization, the first Gauss-Newton method and the second Gauss-Newton method. As the planning horizon is of length 1 it follows that $\mathcal{H}_{12}(\boldsymbol{w}) = \mathbf{0}$, $\forall \boldsymbol{w} \in \mathcal{W}$, so that the first Gauss-Newton method coincides with Newton's method for this MDP. The policy is blockwise log-concave in $\boldsymbol{\mu}$ and L, but not jointly log-concave in $\boldsymbol{\mu}$ and L. As a result we construct block diagonal forms of the preconditioning matrices for the first and second Gauss-Newton methods, with a separate block for $\boldsymbol{\mu}$ and L. Additionally, since the planning horizon is of length 1 it is possible to calculate the Fisher information exactly in this domain. For gradient ascent and natural gradient ascent we considered several different step size sequences. Each

sequence considered had a constant step size throughout, and the sequences differed in the size of this step size. We considered step sizes of length 1, 0.1, 0.01 and 0.001. For both Gauss-Newton methods we considered a fixed step size of one throughout training (i.e., no tuning of the step size sequence was performed for either the first or the second Gauss-Newton methods). As in the work of Kober and Peters (2009) the initial value of $\boldsymbol{\mu}$ is set so that the trajectory of the robot arm mimics that of a given human demonstration. The diagonal elements of the precision matrix are initialized to 0.01. During each training iteration we sampled 15 actions from the policy and used the episodes generated from these samples to estimate the search direction. To deal with this low number of samples we used the samples from the last 10 training iterations when calculating the search direction, taking the 'effective' sample size up to 150. Finally, we used the reward/fitness shaping approach of Wierstra et al. (2014) in all the algorithms considered, using the same shaping function as in Wierstra et al. (2014). In each run of the experiment we performed 100 updates of the policy parameters.

References

P. Abbeel, A. Coates, M. Quigley, and A. Ng. An application of reinforcement learning to aerobatic helicopter flight. *NIPS*, 19:1–8, 2007.

S. Amari. Neural learning in structured parameter spaces - natural Riemannian gradient. *NIPS*, 9:127–133, 1997.

S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10:251–276, 1998.

S. Amari, K. Kurata, and H. Nagaoka. Information geometry of Boltzmann machines. *IEEE Transactions on Neural Networks*, 3(2):260–271, 1992.

S. Amari, A. Cichocki, and H. Yang. A new learning algorithm for blind signal separation. *NIPS*, 8:757–763, 1996.

J. Bagnell and J. Schneider. Covariant policy search. *IJCAI*, 18:1019–1024, 2003.

D. Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

J. Baxter and P. Bartlett. Infinite horizon policy gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

J. Baxter, P. Bartlett, and L. Weaver. Experiments with infinite horizon policy gradient estimation. *Journal of Artificial Intelligence Research*, 15:351–381, 2001.

R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

D. P. Bertsekas. Approximate policy iteration: a survey and some new methods. Research report, Massachusetts Institute of Technology, 2010.

D. P. Bertsekas and S. Ioffe. Temporal differences-based policy iteration and applications in neuro-dynamic programming. Research Report LIDS-P-2349, Massachusetts Institute of Technology, 1996.

S. Bhatnagar, R. Sutton, M. Ghavamzadeh, and M. Lee. Incremental natural actor-critic algorithms. *NIPS*, 20:105–112, 2008.

S. Bhatnagar, R. Sutton, M. Ghavamzadeh, and L. Mark. Natural actor-critic algorithms. *Automatica*, 45:2471–2482, 2009.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

C. Browne, E. Powley, D. Whitehouse, S. Lucas, P. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4:1–43, 2012.

A. Coolen, R. Kuehn, and P. Sollich. *Theory of Neural Information Processing Systems*. OUP Oxford, 2005.

R. Crites and A. Barto. Improving elevator performance using reinforcement learning. *NIPS*, 8:1017–1023, 1995.

P. Dayan and G. E. Hinton. Using expectation-maximization for reinforcement learning. *Neural Computation*, 9:271–278, 1997.

M. P. Deisenroth and C. E. Rasmussen. PILCO: A model-based and data-efficient approach to policy search. *ICML*, 28, 2011.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

C. Fahey. Tetris AI, computers play Tetris `http://colinfahey.com/tetris/tetris_en.html`, 2003.

T. Furmston. *Applications of Probabilistic Inference to Planning & Reinforcement Learning*. PhD thesis, University College London, 2012.

T. Furmston and D. Barber. Solving deterministic policy (PO)MPDs using expectation-maximisation and antifreeze. *ECML*, 1:50–65, 2009. Workshop on Learning and data Mining for Robotics.

T. Furmston and D. Barber. Variational methods for reinforcement learning. *AISTATS*, 9: 241–248, 2010.

V. Gabillon, M. Ghavamzadeh, and B. Scherrer. Approximate dynamic programming finally performs well in the game of Tetris. *NIPS*, 26, 2013.

S. Gelly and D. Silver. Achieving master level play in 9 x 9 computer Go. *AAAI*, 23: 1537–1540, 2008.

M. Ghavamzadeh and Y. Engel. Bayesian policy gradient algorithms. *NIPS*, 19:457–464, 2007.

P. W. Glynn. Stochastic approximation for Monte-Carlo optimisation. *Proceedings of the 1986 ACM Winter Simulation Conference*, 18:356–365, 1986.

P. W. Glynn. Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM*, 33:97–84, 1990.

E. Greensmith, P. Bartlett, and J. Baxter. Variance reduction techniques for gradient based estimates in reinforcement learning. *Journal of Machine Learning Research*, 5:1471–1530, 2004.

N. Heess, G. Wayne, D. Silver, T. Lillicrap, T. Erez, and Y. Tassa. Learning continuous control policies by stochastic value gradients. *NIPS*, 27:2926–2934, 2015.

M. Hoffman, N. de Freitas, A. Doucet, and J. Peters. An expectation maximization algorithm for continuous Markov decision processes with arbitrary rewards. *AISTATS*, 12(5): 232–239, 2009.

R. A. Howard. *Dynamic Programming and Markov Processes*. M.I.T. Press, 1960.

A. Ijspeert, J. Nakanishi, and S. Schaal. Motor imitation with nonlinear dynamical systems in humanoid robots. *IEEE International Conference on Robotic and Automation*, pages 1398–1403, 2002.

A. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. *NIPS*, 15:1547–1554, 2003.

D. Jacobson and D. Mayne. *Differential Dynamic Programming*. Elsevier, 1970.

S. Kakade. Optimizing average reward using discounted rewards. *COLT*, 14:605–615, 2001.

S. Kakade. A natural policy gradient. *NIPS*, 14, 2002.

S. Kakade and J. Langford. Approximately optimal approximate reinforcement learning. *ICML*, 2:267–274, 2002.

H. Khalil. *Nonlinear Systems*. Prentice Hall, 2001.

J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *Annals of Mathematical Statistics*, 23:462–466, 1952.

J. Kober and J. Peters. Policy search for motor primitives in robotics. *NIPS*, 21:849–856, 2009.

J. Kober and J. Peters. Policy search for motor primitives in robotics. *Machine Learning*, 84(1-2):171–203, 2011.

L. Kocsis and C. Szepesvári. Bandit based Monte-Carlo planning. *ECML*, 17:282–293, 2006.

N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. *IEEE International Conference on Robotics and Automation*, 2004.

V. Konda and J. Tsitsiklis. Actor-critic algorithms. *NIPS*, 11:1008–1014, 1999.

V. R. Konda and J. N. Tsitsiklis. On actor-critic algorithms. *SIAM J. Control Optim.*, 42 (4):1143–1166, 2003.

E. L. Lehmann and G. Casella. *Theory of Point Estimation*. Springer, 1998.

S. Levine and V. Koltun. Variational policy search via trajectory optimization. *NIPS*, 27: 207–215, 2013a.

S. Levine and V. Koltun. Guided policy search. *ICML*, 30, 2013b.

W. Li. *Optimal Control for Biological Movement Systems*. PhD thesis, University of San Diego, 2006.

W. Li and E. Todorov. Iterative linear quadratic regulator design for nonlinear biological movement systems. *International Conference on Informatics in Control, Automation and Robotics*, 1, 2004.

W. Li and E. Todorov. Iterative optimal control and estimation design for nonlinear stochastic systems. *IEEE Conference on Decision and Control*, 45, 2006.

T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *ICLR*, 4, 2016.

R. Little and D. Rubin. *Statistical Analysis with Missing Data*. Wiley-Blackwell, 2002.

P. Marbach and J. Tsitsiklis. Simulation-based optimisation of Markov reward processes. *IEEE Transactions on Automatic Control*, 46(2):191–209, 2001.

N. Meuleau, L. Peshkin, K Kim, and L. Kaelbling. Learning finite-state controllers for partially observable environments. *UAI*, 15:427–436, 1999.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015.

T. Morimura, E. Uchibe, J. Yoshimoto, and K. Doya. A new natural policy gradient by stationary distribution metric. *ECML*, 19:82–97, 2008.

R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse and other variants. *Learning in Graphical Models*, pages 355–368, 1999.

A Ngo, Y. Hwanjo, and C. TaeChoong. Hessian matrix distribution for Bayesian policy gradient reinforcment learning. *Information Sciences*, 181:1671–1685, 2011.

J. Nocedal and S. Wright. *Numerical Optimisation*. Springer, 2006.

J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, first edition, 1970.

J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

K. Rawlik, M. Toussaint, and S. Vijayakumar. On stochastic optimal control and reinforcement learning by approximate inference. *International Conference on Robotics Science and Systems*, 2012.

S. Richter, D. Aberdeen, and J. Yu. Natural actor-critic for road traffic optimisation. *NIPS*, 19:1169–1176, 2007.

S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2009.

L. Saul, T. Jaakkola, and M. Jordan. Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research*, 4:61–76, 1996.

S. Schaal. The SL simulation and real-time control software package. Technical report, University of Southern California, 2006.

S. Schaal, P. Mohajerian, and A. Ijspeert. Dynamics systems vs. optimal control - a unifying view. *Progress in Brain Research*, 165(1):425–445, 2007.

N. Schraudolph, J. Yu, and D. Aberdeen. Fast online policy gradient learning with SMD gain vector adaptation. *NIPS*, 18:1185–1192, 2006.

N. Schraudolph, J. Yu, and S. Gunter. A stochastic quasi-Newton method for online convex optimization. *AISTATS*, 11:433–440, 2007.

J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. *ICML*, 32:1889–1897, 2015.

D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. Deterministic policy gradient algorithms. *ICML*, 31:387–395, 2014.

D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–503, 2016.

J. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37:332–341, 1992.

J. Spall and J. Cristion. Model-free control of nonlinear stochastic systems with discrete-time measurements. *IEEE Transactions on Automatic Control*, 43:1198–1210, 1998.

M. Spong, S. Hutchinson, and M. Vidyasagar. *Robot Modelling and Control*. John Wiley & Sons, 2005.

D. Srinivasan, M. C. Choy, and R. L. Cheu. Neural networks for real-time traffic signal control. *IEEE Transactions on Intelligent Transportation Systems*, 7:261–272, 2006.

R. Stengel. *Optimal Control and Estimation*. Dover, 1993.

R. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. *NIPS*, 13, 2000.

R. Tedrake and T. Zhang. Learning to walk in 20 minutes. *Proceedings of the Fourteenth Yale Workshop on Adaptive and Learning Systems*, 2005.

G. Tesauro. TD-Gammon, a self-teaching backgammon program achieves master-level play. *Neural Computation*, 6:215–219, 1994.

P. S. Thomas. Genga: A generalization of natural gradient ascent with positive and negative convergence results. *ICML*, 20:1575–1583, 2014.

E. Todorov and Y. Tassa. Iterative local dynamic programming. *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, pages 90–95, 2009.

M. Toussaint, S. Harmeling, and A. Storkey. Probabilistic inference for solving (PO)MDPs. Research Report EDI-INF-RR-0934, University of Edinburgh, School of Informatics, 2006.

M. Toussaint, A. Storkey, and S. Harmeling. *Bayesian Time Series Models*, chapter Expectation-maximization methods for solving (PO)MDPs and optimal control problems. Cambridge University Press, 2011.

J. Veness, D. Silver, A. Blair, and W. Uther. Bootstrapping from game tree search. *NIPS*, 19:1937–1945, 2009.

N. Vlassis, M. Toussaint, G. Kontes, and S. Piperidis. Learning model-free robot control by a Monte-Carlo EM algorithm. *Autonomous Robots*, 27(2):123–130, 2009.

L. Weaver and N. Tao. The optimal reward baseline for gradient based reinforcement learning. *UAI*, 17(29), 2001.

S. Whitehead. *Reinforcement Learning for Adaptive Control of Perception and Action*. PhD thesis, University of Rochester, 1992.

D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber. Natural evolution strategies. *Journal of Machine Learning Research*, 15:949–980, 2014.

R. Williams. Simple statistical gradient following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.