We thank the reviewers for their highly useful feedback, which we will incorporate in the revised version. To summarize, the main points are:

- Highlight relationship with existing representation formalisms
- Clarify how the ideas can be extended to practical settings

1 Reviewer 1

Put "Related work" in part 2. We agree we can improve the connection with other formalisms, we will emphasize this.

2 Reviewer 2

Isn't the first half of Lemma 1 a well known result? You are right in that the proof makes direct use of existing linear programming results. There is a slight difference, however, since Lemma 1 is specifically about the value polytope: the domain of the functional is nonconvex, and the proof requires first an extension to its convex hull. The statement regarding deterministic policies is also polytope-specific. We will make sure to emphasize what is novel in the statement.

Consider saying why Lemma 1 is useful. The usefulness is the interesting property of the value polytope (which we then make use of). We will emphasize this point also.

Connection to Wasserstein distance and model-based RL. In that space, we know of Farahmand, Barreto, Nikovski (2017) and follow-up work. This is an interesting connection, although to the best of our knowledge there is no model learning equivalent of the value polytope or adversarial value functions. E.g. we would need a small set of models w.r.t. which we want low modelling error. Please let us know in your revised review if there are specific papers that might have a closer connection.

Can you clarify why you used model-based algorithms in experiments? Model-free experiments require dealing with stochasticity in the results, error bars, etc. and would give a murkier picture of the role of the representation. We could learn AVFs using Monte-Carlo samples from the different adversarial policies, and learn these policies through sample-based policy gradient, but we feel our setup gets to the point more clearly. We will highlight this.

The experiments are preliminary / challenges when going to function approximation. To clarify, our experiments use tabular information (e.g., V^π) to produce function approximation, so we read this comment as "how to learn a representation when tabular information is not available".

The path to a "deep" method is clear to us, but not short. There are a few challenges to overcome: 1) how should we represent adversarial policies? 2) what is the effect of non-uniform distributions on the learned representation? this effectively changes the norm in Equation 1; 3) what is the effect of changing the sampling distribution for δ? and 4) we would like to make use of Bellman updates to learn AVFs, but these requires a good off-policy learning method. 1–3 might be answered by considering the auxiliary tasks perspective (Section 3.2). We will add a section discussing these points.

3 Reviewer 3

ϕ becomes part of the computation through its gradient. This is a fair point. At first glance the difference in our perspectives seems to be about how the optimization process discovers the representation, rather than the space of possible representations, but if you are making a more specific distinction please let us know in the revised review. Either way, we will make sure to discuss this point.

Figure 1 (Right) is a cartoon version of a value polytope for a 2-state MDP. The axes are a good suggestion.

Thank you for catching bugs in the math, we will fix.