Reinforcement learning project : Linearly Solvable Markov Decision Processes

Aurélien Brouillaud
ENS Paris-Saclay
Cachan, 94230

Armand Lacombe
ENS Paris-Saclay
Cachan, 94230

Abstract

In this report, we investigate the framework of linearly solvable markov decision processes (LSMDP), as this particular class of MDP was introduced by Todorov [1]. In the first part, we produce an overview of this fundamental article. Then, we show some implementation we achieved to test these new methods. Finally, we summarize some other works capitalizing on this framework.

An implementation code can be found at the following url:

`https://github.com/ArmandLacombe/MVA-RL-Project`

1 Linearly solvable MDP definition and main properties

1.1 Standard formalism

We will use a standard Reinforcement Learning formalism as mentioned in the work of Todorov [1].

In the following we will consider a MDP with a finite set of states S on which we define a set of controls U, transitions costs $l(i,u) \geq 0$, and ***P(u)*** a stochastic matrix where we note $p_{ij}(u)$ the probability to go from i to j under control u.

We suppose that there is an empty set of absorbing states that are such that $p_{ij}(u) = \delta_i^j$ and $l(i,u) = 0$.

In such case we know that if an absorbing state can be reached from any other state in a finite number of steps, the optimal value function is the only solution to the Bellman equation :

$$v(i) = \min_{u \in U(i)} \Big\{ l(i,u) + \sum_j p_{ij}(u) v(j) \Big\}$$

We also consider an undiscounted problem, but the results can be extended to a discounted case.

1.2 Definition of linearly solvable MDP

In his work [1] Todorov considers a new class of MDP where the control is no longer a finite list of actions for each state, but a real vector **u** whose length is the number of states itself, that can be chosen differently for each state. In this class, we also suppose given an uncontrolled markov chain of matrix $\overline{P}$ (the transition probabilities are given by $\overline{p_{ij}}$). It represents a passive underlying dynamics. By choosing a control u when in state i, the user can change the transition probabilities from state i that become:

$$p_{ij}(u) = \overline{p_{ij}} exp(u_j).$$

By using this control in state i, the user must pay a cost given by $r(i, \mathbf{u}) = KL(p_i(u)||p_i(0))$, which has the following interpretation: the more the user wants to change the transition probabilities of the passive dynamics, the more it has to pay. Moreover, we consider there is a fixed cost per state given by a vector q.
Then the transition cost can be written $l(i, \mathbf{u}) = q(i) + KL(p_i(u)||p_i(0))$. The crucial point is that it allows the Bellman equation to be now written as:

$$v(i) = \min_{u \in U(i)} \left\{ q(i) + \sum_j \overline{p_{ij}} exp(u_j)(u_j + v(j)) \right\}$$

The Bellman equation becomes then a convex minimization problem under constraints (the fact that $p_{ij}(u)$ for j in all states must form a probability distribution). Hence the best control $u(i)$ to use in state i can be explicitly computed as a function of the objective function: we find that

$$u_j(i) = -v(j) - log(\sum_k \overline{p_{ik}} exp(-v(k)))$$

We thus have the optimally-controlled transition probabilities:

$$p_{ij}(u(i)) = \frac{\overline{p_{ij}} exp(-v(j))}{\sum_k \overline{p_{ik}} exp(-v(k))}$$

Finally, reintroducing those expressions in the Bellman equation, it can also easily be shown, that with G being the diagonal matrix having the elements $exp(-q(i))$ on its diagonal, and $\mathbf{z}$ a vector which has the elements $exp(-v(i))$, the optimal value function v is such that

$$\mathbf{z} = G\overline{P}\mathbf{z}$$

Todorov [1] shows that this equation admits an only solution satisfying the constraints that z must satisfy. Moreover, it is easily computable using an iterative solution.

The optimal value function estimation is thus no longer an exploration task. The problem has been reduced to a simpler eigenvalues problem that can be solved using classical linear algebra algorithms.

1.3 Discrete MDP approximation by linearly solvable MDP

In order to be able to use the remarkable properties of this class of problems, Todorov [1] also suggests a way to transform problems relying on discrete MDP into problems using linearly solvable MDP.

The key idea is to construct an embedding of the original problem in a way such that the controls of the original MDP are associated with a certain set of control vectors in the continuous MDP with same costs and transition probabilities. If the original problem has the transition probabilities and costs p and l, the conditions are then

$$\overline{p}_{ij} exp(u_j^a) = p_{ij}(a) \, \forall i \in S, \forall j \in N(i), \forall a \in U(i)$$
$$q(i) + r(i, u^a) = l(i, a) \, \forall i \in S, \forall a \in U(i)$$

Let B(i) be the matrix of all transitions probabilities from state i, i-e $b_{aj}(i) = p_{ij}(a) \, \forall a \in U(i)$. We need B to be full row-rank and that its columns contain either no zeroes or only zeroes. This condition isn't as restrictive as it might seem. It is possible to perturb slightly B by adding epsilon values where zeroes keep us from verifying this condition. However, choosing a too little value for epsilon might be problematic considering numerical stability since further steps will include exponential and logarithmic functions.

Let $\mathbf{y}$(i) be the vector with elements $l(i, a) - h(i, a)$ (h is the entropy function) and $\mathbf{x}(i)$ having elements $log(\overline{p}_{ij})$, we can obtain the following equation after some computations:

$$q\overline{1} - B\overline{x} = \overline{y}$$

Once again the problem has been reduced to linear algebra, that can be quickly solved. We have to solve it in x and q, B and y depending only of the original problem. We can use for instance the Moore-Penrose pseudoinverse. The condition on B used here to get the embedding (full row-rank) implies that there are not more possible actions from i than possible next states from i for all states i. This condition does not look so strong, hence it makes a remarkable embedding of many MDP problems into LSMDP problems.

We will illustrate this construction in the following part.

1.4 Z-learning

Similarly to Q-learning, a stochastic approximation for the optimal value function can be used when the model of a continuous MDP in not available. Using $\mathbf{z} = G\overline{P}\mathbf{z}$, Todorov [1] suggests the following iteration step :

$$\hat{z}(i_k) \leftarrow (1 - \alpha_k) + \alpha_k exp(-q_k)\hat{z}(j_k)$$

and $\hat{z}$ will converge to z, giving then access to v.

Comparison between Z-learning and Q-learning in [1] demonstrate the benefits of the former approach. However, one must keep in mind that in order to have meaningful results, one should compare the two approaches on a problem designed in a way such that both of them admit the exact same optimal value function. In more general cases, the LSMDP framework appears as a slight relaxation of the initial problem and is useful to get an approximation.

2 An implementation of discrete MDP approximation

We decided to reproduce the paper implementation and then to do some test on the obtained model. We modified the environment of the first homework so that we could run a pathfinding problem on it. Every state is associated to a cost, which is defined as the cost of getting there from any other state to this one. We also add walls, like states with infinite cost, and two absorbent states.

2.1 Weights display

For instance, one can visualize the costs of one random instance of this environment on the following graph (fig. 1), -1 state values corresponding to walls in this case :

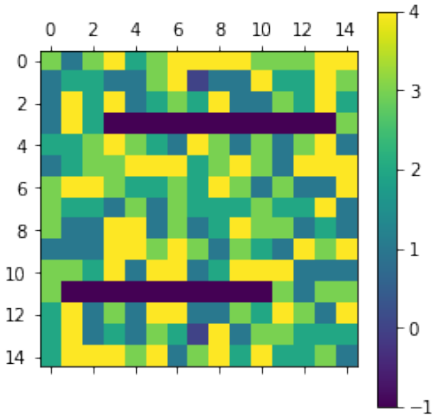
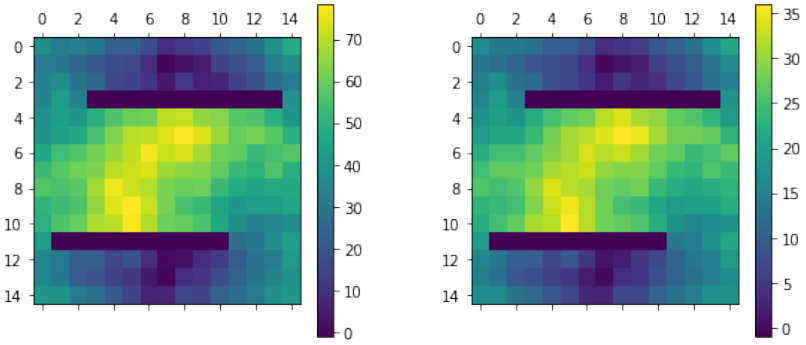


Figure 1: State costs

We can now display the state values computed using a continuous embedding of the original discrete problem (fig 2. a), and compare it to what we would have obtained using an exact algorithm like Dijkstra (fig 2. b).



(a) continuous embedding (b) Dijkstra

Figure 2: 2 methods

Admitting a multiplicative constant due to the continuous problem larger control space, the results are very similar. We may also compute the Pearson correlation coefficient : $R = 0.9978$ here.

2.2 Method robustness

The method implementation raises both sampling difficulties and numerical difficulties which we will investigate.

2.2.1 Sampling influence

As the algorithm includes a step where the transitions probabilities of the discrete problem are sampled in order to realize the embedding, one can wonder to what extend this sampling phase accounts for the results.

Thus, we will consider a little set of different grids, and for each of them we will run the algorithm, sampling an increasing number of transitions per state and action in order to determine the transition probabilities : 1, 10, ... , 10000. We then compute the R coefficient of the correlation between the continuous MDP result and that of Dijkstra. The results are shown (fig 3.) :

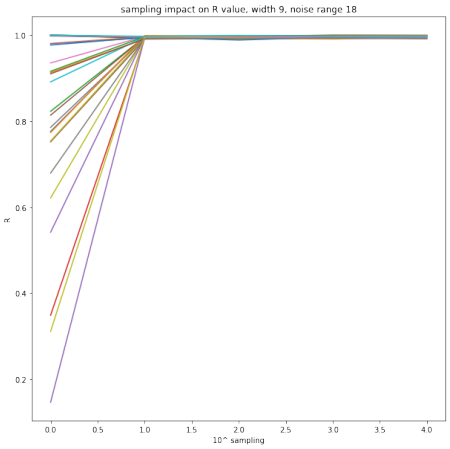


Figure 3: R coefficients for different sampling size and different experiments

It appears that for a number of samples superior to 10 per state, the influence of sampling is insignificant compared to later results. We will then take 1000 samples per state and action in order

to determine the transition probabilities of the discrete model, and assume that this will be of no influence on the results no matter how little or large the grid will be.

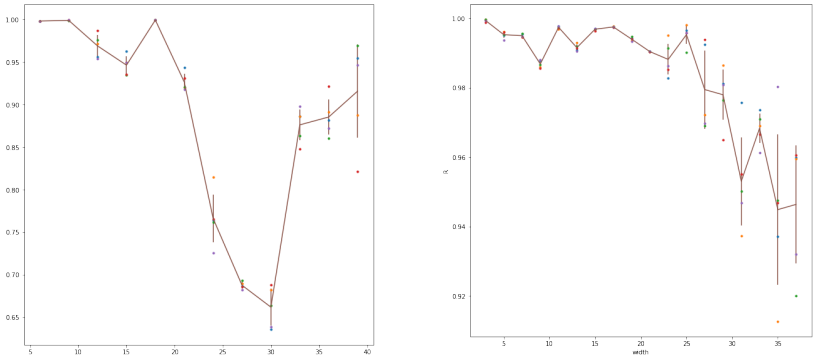
2.2.2 Numerical aspects

The method implementation underlines its numerical difficulties. At some point of the computation, the algorithm has to deal with exponential of very little numbers, and the iterative way to discover the required eigenvector is approximating.

Hence, one might wonder to what extent the obtained results are reliable, and how much their trustworthiness depends on the grid width and on the state costs range. Running several times the same algorithm on the same problem offer no assurance of obtaining the exact same results.

For varying parameters of the grid, we run 5 times the continuous embedding algorithm (fig 4.), and compute the R coefficient between what is obtained and what Dijkstra predicts. We underline that for each set of parameters, the 5 runs are realized on the exact same problem. Without any numerical error, we would obtain the same R value each time (or at least very close values, since the algorithm implies a phase of transition probabilities estimation on the original problem, hence a sampling).

We will limit our study to widths of maximum 39 states (corresponding to 1.5e3 states) because of computation times. On the following graphs, the bars are error bars and their size is proportional to the measures standard deviations.



(a) cost range influence, width 12 (b) width influence, cost range 12

Figure 4: robustness to the two influences

It appears that as the width and cost range increase, the numerical precision degrades. This appears to account for a large part in the global performance decrease. We will study this topic more in detail in the next subsection.

2.3 Grid parameters influence on the algorithm performance on numerical performance

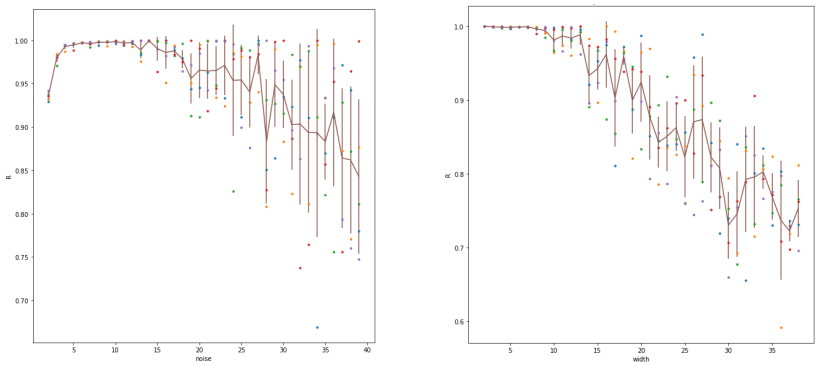
We also investigate on the influence of the grid parameters on the continuous embedding performance. For each set of parameters, we simulate 5 grids and compute the R value of the two algorithms outputs (on the previous study, we ran a few experiments using the same environment).

The continuous embedding performance (fig 5.) appears to decrease as the grid width and cost range increase. We also confirm a phenomenon the previous study had suggested : the continuous embedding is more precise with a noise range around 5 than 1 or 2.

We thus visualize the performance decrease when the grid width and cost range get higher. More experiments would be needed in order to reduce variance and get a finer view of the relation between parameters and the R value, yet as these computation are computationally expensive.

We may also visualize the join (fig 6) influence of the width and cost range :

Since the levels of equal R are roughly lines, it appears that the join performance decrease seems to be a simple sum of the partial performance decrease due to width and cost range. Further analyses



(a) cost range influence, width 9 (b) width influence, cost range 12

Figure 5: comparison of the two influences on the algorithm performance

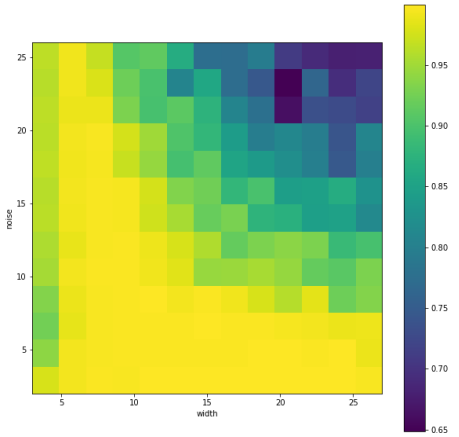


Figure 6: join influence of the width and cost range

could prove or disprove this theory, yet these computation are again very expensive and would require more computing power.

Therefore, as a conclusion of this study, we will conclude that for problems requiring large grids or a large range of states costs, one should be careful and look closely at the possible numerical or approximation errors. A finer handling of very little values might prevent these difficulties from happening.

3 Two applications

Linearly solvable MDP constitue a rich framework that opened the way to a large panel of applications, as in the work of Kappen *et al.* [2-4] or Todorov *et al.* [3-5]. In this part we will briefly evoke its utility in large scale problems, before we analyze more in detail how it can be useful in online learning.

3.1 Large scale problems

In Abbasi-Yadkori *et al.* work [7], the main concern consists in the the learning of good approximations of cost functions in large state spaces MDP in a reasonable time. They show that the search for a good low-dimensional policy is made easier in the context of LSMDP since the key equation of the problem turns into a convex optimization problem. Then, they show how the problem can be

tackled by reducing the number of constraints using sampling and using gradient-descent methods in order to find an acceptable solution.

The major contribution -thought we will not discuss it here- consists in developing an algorithm that finds a reasonable policy in a restricted class using the LSMDP framework, yet whose computation is independent from the state space width (which is relevant since in real world problems, the states space typically grows exponentially).

3.2 Online learning

The framework of LSMDP has been used in many different contexts, with numerous applications. Here we developp on an application to online learning. In their paper *Fast rates for online learning in Linearly Sovable Markov decision processes* [6], Neu *et al.* use the framework of LSMDP to adress a class of online learning problems, and they manage to find a policy that shows great improvement compared to what was previously known. This article is also interesting because it gives another way to see the optimization problem of LSMDP.
The framework of LMDP is used, written in the following way: the learner chooses a transition kernel Q_t at each step t as a control to change an underlying dynamics given by p. The cost of such a policy at time t, when the learner is in state X_t, is then:

$$l(X_t, Q_t) = c(X_t) + D(Q_t(.|X_t)||P(.|X_t))$$

They address the problem of long-term average cost per state:

$$limsup \frac{1}{T} \sum_{t=1}^{T} l(X_t, Q_t)$$

With this form, the problem is identical to that of Todorov, except that it is not the problem with absorbing states. However, they show that the optimal kernel to choose has the same expression as the one given by Todorov:

$$Q(x'|x) = \frac{P(x'|x)z(x')}{\sum_y P(y|x)z(y)}$$

Now, the problem can be transformed in an online learning problem when the fixed cost c becomes a time-dependant cost function which is unknown to the learner.
That is at time t, the learner now incurs a cost $l_t(X_t, Q_t) = c_t(X_t) + D(Q_t(.|X_t)||P(.|X_t))$, and the environment reveals the state-cost function c_t only after the learner has chosen Q_t. It then gives a different problem, which is to find the best series of policies $Q_1, ..., Q_T$ minimizing the regret at time T.

The regret is defined by

$$R_T = E(\sum_{t=1}^{T} l_t(X_t, Q_t)) - E(\sum_{t=1}^{T} l_t(X_t, Q_T)$$

where Q_T is the best stationary policy at time T, i-e the stationary policy minimizing $L_T(Q) = E(\sum_{t=1}^{T} l_t(X_t, Q))$.

In a previous paper, with the same framework (LSMDP) and the same two assumptions that the authors make afterwards (that are: p the passive dynamics is irreductible, aperiodic, and has a Markov-Dobrushin ergodicity coefficient strictly less than 1), some authors had found a strategy which allowed a O($T^{\frac{3}{4}+\epsilon}$) regret for any ϵ. In the present paper, the authors find a spectacular improvement with a strategy allowing a regret in O($log(T)^2$), that should be improvable in O($log(T)$) according to them.
To find this result, they go a bit further in the framework of LSMDP, proposing another view about it. Let's come back to the classic case, with fixed costs independent of time. Let μ_Q be the stationary distribution induced by a kernel Q, then we can define a new matrix π_Q as $\pi_Q(x, x') = \mu_Q(x)\pi_Q(x, x')$. As the system is now controlled to evolve under Q, the limit distribution of it is given by μ_Q. The long-term average per cost can then be written:

$$lim \frac{1}{T} \sum_{i=1}^{T} l(X_t, Q) = \sum_x \mu_Q(x)(c(x) + D(Q(.|x)||P(.|x)))$$

which, after some computations, can be written as a convex function in π and c, noted $f(\pi, c)$, also affine in c. Hence, fixed costs per state being given, the optimization problem is reduced to a convex optimization problem in the matrix π under constraints.

Now let's come back to the problem of online learning. We can then define an idealized regret replacing the function of (X_t, Q_t) by a function of the π_t associated with the Q_t:

$$\overline{R}_T = \sum_{t=1}^{T} f(\pi_t, c_t) - \sum_{t=1}^{T} f(\pi, c_t)$$

where π denotes here the optimal policy.

In this paper, the authors show that the "follow-the leader-strategy" (FTL) is giving a real regret in O($log(T)^2$). This strategy is the following: π_t is chosen such that it minimizes $\sum_{s=1}^{t-1} f(\pi, c_s)$. Due to the affinity of f in c, π_t is also the minimizer of

$$f(\pi, \overline{c_t})$$

where $\overline{c_t} = \frac{1}{t-1} \sum_{s=1}^{t-1} c_s$

Hence we can compute π_t as we compute the optimal contol in the article from Todorov [1], using $\overline{c_t}$ as the fixed cost function.
The authors can then finally show that using this series of policies $\pi_1, ..., \pi_T$ ensures a O(O($log(T)^2$)) regret. The complete proof relies on a long series of lemmas. The key idea is to show first than the FTL strategy allows an idealized regret in $O(log(T))$, and then to control the difference between the true regret and the idealized regret.

4 Conclusion

The framework of LSMDP is hence powerful because it allows to solve the normally highly complex Bellman optimality equations directly, using tools from linear algebra. Even if it may look a very special class of MDP at first sight, Todorov's work shows that classic MDP problems can often be approximated with LSMDP, making them useful in practice. The implementation put light on the technical and numerical difficulties of such a problem, and justifies further analysis of the error control. That's why many works keep using this framework, like the one presented in part 3.

[1] Todorov, E. (2006) Linearly-solvable Markov decision problems. *Advances in neural information processing systems*

[2] Kappen, H. (2005) Linear theory for control of nonlinear stochastic systems. *Physical review letters 95.20.*

[3] Todorov, E. (2008) General duality between optimal control and estimations. *Decision and Control*

[4] Kappen, H., Gmez V. & Opper, M. (2012) Optimal control as a graphical model inference problem. *Machine learning 87.2*

[5] Todorov, E. (2009) Compositionality of optimal control laws. *Advances in Neural Information Processing Systems*

[6] Neu G. & Gmez V. (2017) Fast rates for online learning in Linearly Solvable Markov Decision Processes. *Proceedings of the 2017 Conference on Learning Theory*

[7] Abbasi-Yadkori Y., Bartlett P. L., Chen X. & Malek A. (2015) Large-Scale Markov Decision Problems with KL Control Cost *Proceedings of Machine Learning Research* and its Application to Crowdsourcing