00000100000
00001010000
00010001000
00100000100
01000000010
10000000001
Software notes

Ecography 37: 916–920, 2014
doi: 10.1111/ecog.00888

Subject Editor: Thiago Rangel. Accepted 9 May 2014

MDPtoolbox: a multi-platform toolbox to solve stochastic dynamic programming problems

Iadine Chadès, Guillaume Chapron, Marie-Josée Cros, Frédérick Garcia and Régis Sabbadin

I. Chadès (iadine.chades@csiro.au), CSIRO Ecosystem Sciences, GPO Box 2583, Brisbane, QLD 4001, Australia. – G. Chapron, Grimsö Wildlife Research Station, Swedish Univ. of Agricultural Sciences, SE-73091 Riddarhyttan, Sweden. – M.-J. Cros, F. Garcia and R. Sabbadin, INRA, Unité de Mathématiques et Informatique Appliquées, Toulouse, UR875, CS 52627, FR-31326 Castanet Tolosan cedex, France.

Stochastic dynamic programming (SDP) or Markov decision processes (MDP) are increasingly being used in ecology to find the best decisions over time and under uncertainty so that the chance of achieving an objective is maximised. To date, few programs are available to solve SDP/MDP. We present MDPtoolbox, a multi-platform set of functions to solve Markov decision problems (MATLAB, GNU Octave, Scilab and R). MDPtoolbox provides state-of-the-art and ready to use algorithms to solve a wide range of MDPs. MDPtoolbox is easy to use, freely available and has been continuously improved since 2004. We illustrate how to use MDPtoolbox on a dynamic reserve design problem.

Globally, biodiversity is undergoing a rapid decline with only limited resources available to undertake conservation strategies (Wilson et al. 2006). In response, we have seen the development of cost-effective decision making (Possingham 2001). Finding the best sequential decisions under uncertainty is an optimisation problem consisting in maximising the sum of future expected rewards over time. The solution of this problem is a policy function that indicates the action to apply for each possible state of the system, also called state-dependent decisions. This optimisation problem is often referred to by its solution technique as stochastic dynamic programming (SDP) or by the mathematical model as a Markov decision process (MDP). Formally, MDPs are defined as controlled stochastic processes satisfying the Markov property and assigning reward values to state transitions (Puterman 1994, Sigaud and Buffet 2010). MDPs allow a clear and eloquent formulation of optimal resource allocation problems. First developed in the fields of operations research and artificial intelligence (Puterman 1994), MDPs have since been widely applied in conservation science (Chadès et al. 2011, 2012b) and behavioural ecology (McNamara et al. 1987, Venner et al. 2006). We have developed MDPtoolbox, an open-source set of functions that can be used to solve a wide range of Markov decision problems and freely downloaded at <http://www7.inra.fr/mia/T/MDPtoolbox>. To the best of our knowledge MDPtoolbox is the only toolbox that provides a variety of algorithms that suits most optimisation criteria, freely available and multi-platform (Marescot et al. 2013). MDPtoolbox has already been used in applied mathematics, computer science (Zhao et al. 2010, Munir and Gordon-Ross 2012), ecology and agronomy (Chadès et al. 2012b, Pichancourt et al. 2012, Grechi et al. 2014).

Theory of MDP and its implementation in MDPtoolbox

Our toolbox consists of a set of functions related to the resolution of discrete-time MDP (finite horizon, value iteration, policy iteration, linear programming algorithms with some variants) and also proposes some functions related to a Reinforcement Learning method (Q-learning). Simpler functions intended to generate simple MDP e.g. for educational purposes are also available.

Using MDPtoolbox requires describing a MDP by a tuple $<S,A,p,r,(T,r_T)>$ where:

– S is the finite state space that describes the possible configurations of the system;

– A is the finite set of all possible actions or decisions which control the state dynamics;

– p denotes the state transition function and characterises the state dynamics of the system, that is $p(s_{t+1}|s_t,a_t)$ represents the probability of transitioning to state s_{t+1} given the system is in state s_t and action a_t is applied;

– r provides the reward function defined on state transitions: $r(s_t,a_t,\ s_{t+1})$. Desirable transitions usually receive strong rewards; and

– T is the time horizon over which decisions need to be made, and can be either finite or infinite. An intermediate case between finite and infinite horizons exists: indefinite horizon. This is the case where a set of

terminal states (or goal states) exists and the process stops as soon as one of these states is encountered. T only needs to be defined in the finite horizon case. In the case where the horizon T is finite or indefinite, a terminal reward $r_T(s_T)$ is defined, either on the whole set of states or on the set of terminal states only.

The probability distribution over the next state s_{t+1} defined by p follows the Markov property which gives its name to Markov decision processes. The probability of reaching state s_{t+1} consecutively to action a_t only depends on action a_t and the previous state of the system s_t. Function mdp_example_rand() in our toolbox allows the user to become familiar with MDP and mdp_check() performs verifications on the validity of a MDP description.

Markov decision processes are used to model the state dynamics of a stochastic system when this system can be controlled by a decision maker. In our toolbox, we call a strategy or policy the function π: S $\rightarrow$ A, which associates an action (or decision) to each state. A policy assigns a deterministic action to a state configuration of the system and can be seen as a set of rules a decision-maker would follow to choose the action to perform in each state. A policy can be explicitly dependent on a time step t (non stationary policy) or independent of time (stationary policy).

Our toolbox solves a Markov decision process by finding the optimal policy given an optimisation criterion. This criterion characterises the policies which will provide the highest sequence of cumulated rewards, and represents the objective decision makers try to achieve. In our toolbox, we differentiate between four optimisation criteria most commonly used (Table 1, Puterman 1994).

The finite criterion is the expected sum of rewards over a finite time horizon (finite number of time steps). The underlying assumption of this criterion is that the decision-maker has T steps to manage the system. The finite criterion can also be discounted using a discount factor γ ($0 \leq \gamma \leq 1$). The discount factor γ can have an economic interpretation linked to the inflation rate.

The γ-discounted infinite criterion is the expected discounted sum of rewards over an infinite time horizon. The discount factor γ is a real number: $0 \leq \gamma < 1$. The discount factor also guarantees the convergence of the infinite sum of rewards towards a finite value and offers useful mathematical properties.

The total reward criterion is the expected sum of rewards over an infinite time horizon. The total reward criterion assumes that we know the process will terminate in a finite number of steps but an upper bound on this number is not available. This criterion is often used for optimal stopping problems.

The average criterion is the expected average of rewards over an infinite time horizon. This criterion is preferred when decisions have to be made frequently with a discount factor close to one or when it is preferable to average the value of the received rewards instead of considering their discounted sum.

The optimisation criteria define value functions V^{π}: S$\rightarrow$R that evaluate the performance of a policy given an optimisation criterion in any initial state, s, of the system. This value function gives, for each state of the system, the expected performance of the policy π, applied in state s.

The goal of an MDP solver is to find an optimal policy π^* such that its value function $V^{\pi*}$ (s) is maximised: $\forall s \in S,\ V^{\pi}(s) \leq V^{\pi*}(s)$. Our toolbox provides solution methods that correspond to the different optimisation criteria (Table1).

Illustrative example: dynamic reserve design

The creation of conservation reserves is a primary way of reducing biodiversity loss. The establishment of reserves is a gradual, additive process, comprising a sequence of land acquisitions through time because funding is insufficient to acquire all sites at once (Costello and Polasky 2004). Conservation organisations must contend with the risk that

Table 1. Summary of the different optimisation criteria that define optimisation objectives, a value function to maximise and a set of solution methods (algorithms) with T the time horizon, r_t the instant reward, r_T the terminal reward, s_T the terminal state, s_0 the initial state, and γ the discount factor. To each optimisation criterion correspond one or several solution methods and MDPtoolbox functions. Interested readers can refer to Puterman (1994), Sutton and Barto (1998), and Marescot et al. (2013) for detailed descriptions of the algorithms.

Optimisation criteria	Value function V^{π} (s)	Algorithms	MDPtoolbox functions
Finite	$E^{\pi}\left[\sum_{t=0}^{T-1}\left[\gamma^t r_t + \gamma^T r_T(s_T)\middle\vert s_0\right]\right]$	Backwards induction	mdp_finite_horizon
γ-discounted infinite	$E^{\pi}\left[\sum_{t=0}^{\infty}\left[\gamma^t r_t \middle\vert s_0\right]\right]$	Linear programming	mdp_LP
		Value iteration	mdp_value_iteration
		Gauss–Seidel's value iteration	mdp_value_iterationGS
		Policy iteration	mdp_policy_iteration
		Modified policy iteration	mdp_policy_iteration_modified
		Q-learning	mdp_Q_learning
Total	$E^{\pi}\left[\sum_{t=0}^{\infty}\left[r_t \middle\vert s_0\right]\right]$	Value iteration	mdp_value_iteration
		Policy iteration	mdp_policy_iteration
Average infinite	$\lim_{n\rightarrow\infty} E^{\pi}\left[\frac{1}{n}\sum_{t=0}^{n-1}\left[r_t \middle\vert s_0\right]\right]$	Relative value iteration	mdp_relative_value_iteration
		Modified policy iteration	mdp_policy_iteration_modified

sites will be developed before being reserved. Markov decision processes have been proposed to solve reserve selection problems of this type (Costello and Polasky 2004, Sabbadin et al. 2007). In these models, available sites are irreversibly developed each year with a given probability, but only a limited number of sites can be reserved each year. The problem is to design a dynamic reservation policy that results in the maximum expected number of species conserved over time. We apply MDPtoolbox to the model developed by Sabbadin et al. (2007).

We assume that there exist J sites indexed by $j = 1, 2, \ldots, J$ and I species indexed by $i = 1, 2, \ldots, I$, a $J \times I$ matrix M is given, where an element M_{ji} equals 1 if site j is suitable for species i, and 0 if not. At a given time period t, any site j can be in one of the three following states: developed, reserved or available. A species i exists in site j if and only if site j is not developed.

States: the state s_t of sites can be unambiguously described by a vector of size J, $s_t(j) \in \{Av,R,D\}$ (respectively for 'available', 'reserved' and 'developed').

Actions: at any time period, it is possible to select one available site $a_t \in A = \{1,\ldots,J\}$ for reservation, thus changing its state from available ($s_t(a_t) = Av$) to reserved ($s_{t+1}(a_t) = R$).

Transitions: at any time period, any available site j not selected for reservation can become developed at the end of the period with a known probability p_j. The development probability of a site j which is currently undeveloped is: $p_j(s_{t+1}(j) = D|s_t,a_t)$. The transition probabilities between states are defined as

$$p(s_{t+1}|s_t, a_t) = \Pi_j\, p_j(s_{t+1}(j)|s_t,a_t)$$

Rewards: we define a reward function $r(s,a)$ as the number of additional species that are protected when site a is reserved in state s. At time t, if $a_t = j$, $r(s_t, a_t)$ is the number of species for which site j is suitable and for which no suitable site has been reserved so far.

The objective of a reserve selection problem is to minimise species losses, or equivalently to maximise the number of species present in reserved sites, either at the end of a fixed horizon or when the process has reached an absorbing state, for example a state where no additional species can be protected.

The MDP model $<$S, A, p, r$>$ of the reserve selection problem can be solved for undiscounted finite horizon, discounted finite or infinite horizon, using MDPtoolbox (Table 2, Fig. 1). We first generate the species richness matrix (M), which provides for 7 sites the presence/absence information of 20 species we seek to protect (lines 2a, 2b, Table 2). We build the transition probability (P) and reward matrices (R) according to the species richness matrix M and the probability of a site becoming developed ($p_j = 0.2$, lines 5a, 5–7b, Table 2). Note that users will have to define these functions when solving an MDP for a different application. We check the validity of the MDP (lines 8a, 10b). We solve the optimisation problem assuming a discounted infinite time horizon with a discount factor $\gamma = 0.96$ and a stopping criterion of $\varepsilon = 0.001$ required for the value iteration algorithm (lines 11a, 13b, Table 2). The function output provides the optimal strategy (policy), the number of iterations (iter) and the CPU time required (cpu_time). From this point we have solved the reserve design optimisation problem and we can explore the solution by simulation (lines 14a, 17b, Table 2). Figure 1 illustrates a management simulation for our reserve design problem when the initial state is 'all states available'. This single simulation scenario only illustrates the use of the MDPtoolbox. A thorough analysis of the reserve design problem would, of course, require running and analysing several different scenarios to provide guidance to conservation managers. Readers interested in the application can refer to (Costello and Polasky 2004) and (Sabbadin et al. 2007) for more in depth analysis of the dynamic reserve design problem.

Table 2. Implementation of the reserve design problem in (A) MATLAB, GNU Octave, Scilab and (B) R (attached as supplementary material). The function associated to the reserve design problem is not part of the MDPtoolbox. However, it can be downloaded (in Matlab, Octave, Scilab and R format) from a SourceForge deposit, directly accessible from the MDPtoolbox website (<http://www7.inra.fr/mia/T/MDPtoolbox/Install.html>).

(A) MATLAB/GNU Octave Code/Scilab code

```
1a   % Generate the species richness matrix
2a   M = round(rand(7,20));
3a
4a   % Generate the transition and reward matrix
5a   [P, R] = mdp_example_reserve(M, 0.2);
6a
7a   % Checks the validity of the MDP
8a   mdp_check(P, R)
9a
10a  % Solve the reserve design problem
11a  [policy, iter, cpu_time] = mdp_value_iteration
       (P, R, 0.96, 0.001);
12a
13a  % Explore solution with initial state all sites
       available
14a  explore_solution_reserve([0 0 0 0 0 0 0],
       policy,M,P,R)
```

(B) R code

```
1b   # Generate the species richness matrix
2b   M <- round(matrix(nrow = 7, ncol = 20,
       data = runif(7*20,0,1)))
3b
4b   # Generate the transition and reward matrix
5b   PR <- mdp_example_reserve(M, 0.2)
6b   P <- PR$P
7b   R <- PR$R
8b
9b   # Checks the validity of the MDP
10b  mdp_check(P, R)
11b
12b  # Solve the reserve design problem
13b  results <- mdp_value_iteration(P, R,
       0.96, 0.001);
14b  policy <- results$policy
15b
16b  # Explore solution with initial state all sites
       available
17b  explore_solution_reserve(numeric(7), policy,
       M, P, R)
```

Conclusion

Since 2004, we have continually maintained and improved MDPtoolbox. Now in ver. 4.0, it is available on multiple

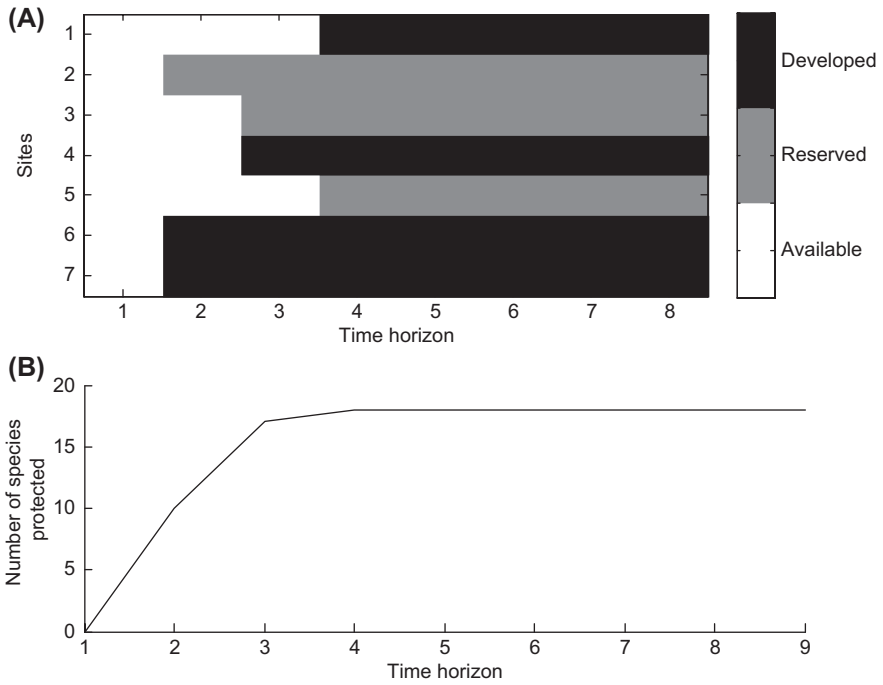

Figure 1. Simulation of the optimal strategy for the reserve design problem. The top graph (A) represents the status of 7 sites over an 8 time-step horizon. At t = 1, all sites are available and the optimal action is to reserve site 2, while sites 6 and 7 become developed. The bottom graph represents the performance achieved as the cumulative number of species protected. After the first action, 10 species are protected. At time horizon four, 18 species are protected out of 20. By the end of the time horizon it was not possible to protect additional species, therefore 2 species remained unprotected due to the development of sites.

platforms and provides an entry point to solving stochastic dynamic programming problems across multiple areas of science. Because of the inherent computational complexity, solving large state or action space MDP can be prohibitive even for modern computers. We encourage interested readers to refer to advanced SDP methods to avoid the so-called 'curse of dimensionality' as adding a new state variable to a model may results in an exponential increase in the state space (Nicol and Chadès 2011, Sabbadin et al. 2012). While SDP algorithms provide one optimal strategy, several strategies might be optimal. Alternative strategies should be evaluated and compared to optimal strategies. We recommend comparing the performance of different strategies so that gains or losses of performance are acknowledged and understood by decision makers. Our toolbox offers functions to perform such analyses (MDP_eval_policy functions).

MDPs have underpinned the development of recommendations regarding key conservation issues, such as how best to manage both invasive and threatened species (Chadès et al. 2011, Sabbadin et al. 2012). In the field of behavioural ecology, MDPtoolbox could help testing whether or not species have evolved to forage and reproduce optimally to maximize their reproductive success or overall fitness over time (McNamara et al. 1987, Mangel and Clark 1988). MDPs are also central to solving adaptive management problems (Hauser and Possingham 2008, Williams 2009, Chadès et al. 2012a). A future development of our toolbox will incorporate adaptive management solution methods (Walters and Hilborn 1978, Nicol et al. 2013).

To cite MDPtoolbox or acknowledge its use, cite this Software note as follows, substituting the version of the application that you used for 'version 0':

Chadès, I., Chapron, G., Cros, M.-J., Garcia, F. and Sabbadin, R. 2014. MDPtoolbox: a multi-platform toolbox to solve stochastic dynamic programming problems. – Ecography 37: 916–920 (ver. 0).

Acknowledgements – We are grateful to J. Carwardine, A. I. T. Tulloch, T. G. Martin and S. Nicol for commenting on earlier versions of this manuscript.

References

Chadès, I. et al. 2011. General rules for managing and surveying networks of pests, diseases, and endangered species. – Proc. Natl Acad. Sci. USA 108: 8323–8328.

Chadès, I. et al. 2012a. MOMDPs: a solution for modelling adaptive management problems. – The Twenty-sixth AAAI Conference on Artificial Intelligence (AAAI-12), pp. 267–273.

Chadès, I. et al. 2012b. Setting realistic recovery targets for interacting endangered species. – Conserv. Biol. 26: 1016–1025.

Costello, C. and Polasky, S. 2004. Dynamic reserve site selection. – Resour. Energy Econ. 26: 157–174.

Grechi, I. et al. 2014. A decision framework for management of conflicting production and biodiversity goals for a commercially valuable invasive species. – Agric. Syst. 125: 1–11.

Hauser, C. E. and Possingham, H. P. 2008. Experimental or precautionary? Adaptive management over a range of time horizons. – J. Appl. Ecol. 45: 72–81.

Mangel, M. and Clark, T. W. 1988. Dynamic modeling in behavioural ecology. – Princeton Univ. Press.

Marescot, L. et al. 2013. Complex decisions made simple: a primer on stochastic dynamic programming. – Methods Ecol. Evol. 4: 872–884.
McNamara, J. M. et al. 1987. Optimal daily routines of singing and foraging in a bird singing to attract a mate. – Behav. Ecol. Sociobiol. 20: 399–405.
Munir, A. and Gordon-Ross, A. 2012. An MDP-based dynamic optimization methodology for wireless sensor networks. – IEEE Trans. Parallel Distrib. Syst. 23: 616–625.
Nicol, S. and Chadès, I. 2011. Beyond stochastic dynamic programming: a heuristic sampling method for optimizing conservation decisions in very large state spaces. – Methods Ecol. Evol. 2: 221–228.
Nicol, S. et al. 2013. Adaptive management of migratory birds under sea level rise. – International Joint Conference on Artificial Intelligence (IJCAI2013).
Pichancourt, J. B. et al. 2012. Simple rules to contain an invasive species with a complex life cycle and high dispersal capacity. – J. Appl. Ecol. 49: 52–62.
Possingham, H. P. 2001. The business of biodiversity: applying decision theory principles to nature conservation. – Environ. Econ. Soc. 9: 1–37.
Puterman, M. L. 1994. Markov decision processes: discrete stochastic dynamic programming. – Wiley.
Sabbadin, R. et al. 2007. Dynamic reserve site selection under contagion risk of deforestation. – Ecol. Model. 201: 75–81.
Sabbadin, R. et al. 2012. A framework and a mean-field algorithm for the local control of spatial processes. – Int. J. Approx. Reason 53: 66–86.
Sigaud, O. and Buffet, O. 2010. Markov decision processes in artificial intelligence: MDPs, beyond MDPs and applications. – ISTE/Wiley.
Sutton, R. S. and Barto, A. G. 1998. Reinforcement learning: an introduction. – MIT Press.
Venner, S. et al. 2006. Dynamic optimization over infinite-time horizon: web-building strategy in an orb-weaving spider as a case study. – J. Theor. Biol. 241: 725–733.
Walters, C. J. and Hilborn, R. 1978. Ecological optimization and adaptive management. – Annu. Rev. Ecol. Syst. 9: 157–188.
Williams, B. K. 2009. Markov decision processes in natural resources management: observability and uncertainty. – Ecol. Model. 220: 830–840.
Wilson, K. A. et al. 2006. Prioritizing global conservation efforts. – Nature 440: 337–340.
Zhao, Q. H. et al. 2010. A variable neighborhood search based algorithm for finite-horizon Markov decision processes. – Appl. Math. Comput. 217: 3480–3492.

Supplementary material (Appendix ECOG-00888 at <www.ecography.org/readers/appendix>).