

000  
001  
002  
003  
004  
005  
006  
007  
008  
009  
010  
011  
012  
013  
014  
015  
016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030  
031  
032  
033  
034  
035  
036  
037  
038  
039  
040  
041  
042  
043  
044  
045  
046  
047  
048  
049  
050  
051  
052  
053

---

# Symbolic Dynamic Programming for Continuous State and Observation POMDPs

---

Anonymous Author(s)

Affiliation

Address

email

## Abstract

Partially-observable Markov decision processes (POMDPs) provide a powerful model for real-world sequential decision-making problems. In recent years, point-based value iteration methods have proven to be extremely effective techniques for finding (approximately) optimal dynamic programming solutions to POMDPs when an initial set of belief states is known. However, no point-based work has provided exact point-based backups for both continuous state and observation spaces, which we tackle in this paper. Our key insight is that while there may be an infinite number of possible observations, there are only a finite number of observation partitionings that are relevant for optimal decision-making when a finite, fixed set of reachable belief states is known. To this end, we make two important contributions: (1) we show how previous exact symbolic dynamic programming solutions for continuous state MDPs can be generalized to continuous state POMDPs with discrete observations, and (2) we show how this solution can be further extended via recently developed symbolic methods to continuous state and observations to *derive* the minimal relevant observation partitioning for potentially correlated, multivariate observation spaces. We demonstrate our algorithm on a power plant control problem requiring continuous state *and* observations.

## 1 Introduction

Partially-observable Markov decision processes (POMDPs) are a powerful modeling formalism for real-world sequential decision-making problems [3]. In recent years, point-based value iteration methods [6, 11, 12, 8] have proved extremely successful at scaling (approximately) optimal POMDP solutions to large state spaces when a set of initial belief states is known.

More recent work has extended point-based methods to both continuous state and continuous observation spaces, but no work has tackled both jointly without sampling. [7] provides exact point-based backups for continuous state and discrete observation problems (with approximate sample-based extensions to continuous actions and observations), while [2] provides exact point-based backups for discrete state and continuous observation problems (where multivariate observations must be conditionally independent). While restricted to discrete states, [2] provides an important insight that we exploit in this work: *only a finite number of partitionings of the observation space are required to distinguish between the optimal conditional policy over a finite set of belief states.*

Our contributions in this work are two-fold. First, we extend symbolic dynamic programming for continuous state MDPs [10] to the case of continuous state and discrete observation POMDPs. This provides an expressive and concrete instantiation of the framework in [7] (which only abstractly required that integrals were computable) in that it ensures the closed-form of all  $\alpha$ -functions for *all horizons* is a symbolic piecewise case statement, even for POMDPs with *arbitrary* continuous rewards and transitions with discrete noise (i.e., a finite mixture of deterministic transitions). Second,

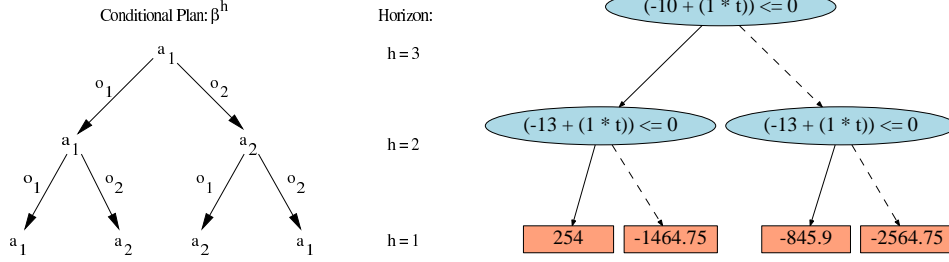


Figure 1: (left) Example conditional plan  $\beta^h$  for discrete observations; (right) an example piecewise  $\alpha$ -vector for continuous state  $t$  represented as a decision diagram: the *true* branch is solid, the *false* branch is dashed.

we extend this symbolic dynamic programming algorithm to the case of continuous observations (while restricting the transition and reward functions to piecewise polynomials) and extend the work of [2] to *derive* relevant observation partitions for potentially correlated multivariate continuous observation spaces by exploiting the piecewise polynomial integration operation of [9] and the multivariate symbolic maximization technique of [10]. We conclude by demonstrating our algorithm on a power plant control problem requiring bivariate continuous state and observations.

## 2 DC-POMDP Model

We assume familiarity with MDPs and introduce discrete and continuous partially observable MDPs (DC-POMDPs) as an extension to their DC-MDP subset [10]. A DC-POMDP is a tuple  $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \mathcal{R}, \mathcal{Z}, \gamma, h \rangle$ . States  $\mathcal{S}$  are represented by vectors of  $(\vec{d}_s, \vec{x}_s) = (d_{s_1}, \dots, d_{s_n}, x_{s_1}, \dots, x_{s_m})$  where each  $d_{s_i} \in \{0, 1\}$  ( $1 \leq i \leq n$ ) is a discrete boolean and each  $x_{s_j} \in \mathbb{R}$  ( $1 \leq j \leq m$ ) is a continuous variable. We assume a finite, discrete action space  $\mathcal{A} = \{a_1, \dots, a_p\}$ . Observations  $\mathcal{O}$  are given by the vector  $(\vec{d}_o, \vec{x}_o) = (d_{o_1}, \dots, d_{o_p}, x_{o_1}, \dots, x_{o_q})$  where each  $d_{o_i} \in \{0, 1\}$  ( $1 \leq i \leq p$ ) is boolean and each  $x_{o_j} \in \mathbb{R}$  ( $1 \leq j \leq q$ ) is continuous.

Three functions are required for modeling DC-POMDPs: (1)  $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$  a Markovian transition model defined as the probability of the next state given the action and previous state; (2)  $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$  a reward function which returns the immediate reward of taking an action in some state; and (3) an observation function defined as  $\mathcal{Z} : \mathcal{S} \times \mathcal{A} \times \mathcal{O} \rightarrow [0, 1]$  which gives the probability of an observation given the outcome of a state after executing an action. A discount factor  $\gamma$ ,  $0 \leq \gamma \leq 1$  is used to discount rewards  $t$  time steps into the future by  $\gamma^t$ .

We use a dynamic Bayesian network (DBN) to compactly represent the transition model over the factored state variables and we likewise use a Bayesian network to represent the observation model:

$$\mathcal{T} : P(\vec{d}'_s, \vec{x}'_s | \vec{d}_s, \vec{x}_s, a) = \prod_{i=1}^n P(d'_i | \vec{d}_s, \vec{x}_s, a) \prod_{j=1}^m P(x'_j | \vec{d}_s, \vec{x}_s, a).$$

$$\mathcal{Z} : P(\vec{d}_o, \vec{x}_o | \vec{d}_s, \vec{x}_s, a) = \prod_{i=1}^n P(d_{o_i} | d_{s_i}, a) \prod_{j=1}^m P(x_{o_j} | x_{s_j}, a).$$

For simplicity of algorithmic exposition, we do not allow synchronic arcs in the transition representation, but note that to incorporate such arcs only requires restrictions on the variable elimination ordering during dynamic programming. In these transition and observation Bayes nets, probabilities over *discrete* variables are represented using conditional probability functions (CPF) in the form of  $P(d'_i | \vec{d}, \vec{x}, a)$ . Probabilities for *continuous* variables  $P(x'_j | \vec{d}, \vec{x}, a)$  are represented by *piecewise functions* (PWFs) that are first-order Markov and deterministic, i.e. the probabilities are encoded using the Dirac  $\delta[\cdot]$  function. The reward function  $R(\vec{d}, \vec{x}, a)$  is set to be any general piecewise function. We next provide an intuitive example to make this DC-POMDP framework more concrete.

**Example (Power Plant) [1]** The steam generation system of a power plant aims to provide hot steam to the steam turbine which in turn produces electricity. Feed-water is exposed to heat in the tubes leading to the water tank, where water evaporation is performed under specific pressure and

temperature. A reward is obtained when electricity is generated through steam production and the pressure and temperature are within safe ranges. Mixing water and steam makes the respective pressure and temperature observations  $p_o \in \mathbb{R}$  and  $t_o \in \mathbb{R}$  on the underlying state  $p_s \in \mathbb{R}$  and  $t_s \in \mathbb{R}$  highly uncertain. Action choices are either to open or close a pressure valve  $a \in \{\text{open}, \text{close}\}$ .

We initially present two simple examples labeled **1D-Power Plant** using only the temperature state  $t_s$ . First, the transition and reward functions common to both problems are defined as below where simply, the temperate increments or decrements according to the action:

$$P(t'_s | \vec{t}_s, a) = \delta \begin{cases} t'_s - \begin{cases} (a = \text{open}) & : t_s - 5 \\ (a \neq \text{open}) & : t_s + 7 \end{cases} \end{cases} \quad R(t_s, a) = \begin{cases} (a = \text{open}) & : -1 \\ (a \neq \text{open}) \wedge (t_s > T) & : -1000 \\ (a \neq \text{open}) \wedge \neg(t_s > T) & : 100 \end{cases} \quad (1)$$

Next we introduce the **Discrete Obs. 1D-Power Plant** variant using a discrete binary observation space with possible observations  $o \in \text{mathcal{O}} = \{\text{high}, \text{low}\}$ :

$$P(o = \text{high} | t'_s, \text{open}) = \begin{cases} t_s \leq 15 & : 0.9 \\ \neg(t_s \leq 15) & : 0.1 \end{cases} \quad P(o = \text{low} | t'_s, \text{open}) = \begin{cases} t_s \leq 15 & : 0.1 \\ \neg(t_s \leq 15) & : 0.9 \end{cases} \quad (2)$$

Finally we introduce the **Cont. Obs. 1D-Power Plant** variant with continuous observation  $t_o$ . We define the observation probability over  $t_o$  as uniform on an interval 10 units wide, centered at  $t_s$ .

$$P(t_o | t'_s, \text{open}) = \begin{cases} (t_o > t_s - 5) \wedge (t_o < t_s + 5) & : 0.1 \\ \neg((t_o > t_s - 5) \wedge (t_o < t_s + 5)) & : 0 \end{cases} \quad (3)$$

### 3 DC-POMDP solution

In a DC-POMDP, the agent does not directly observe the states and thus must maintain a belief state  $b(xd_s) = P(xd_s)$ .<sup>1</sup> For a given belief state  $\vec{b} = b(\cdot)$ , a POMDP policy  $\pi$  can be represented by a tree corresponding to a conditional plan  $\beta$ . An  $h$ -step conditional plan  $\beta^h$  can be defined recursively in terms of  $(h - 1)$ -step conditional plans as shown in Fig. 1.

Our goal is to find a policy  $\pi$  that maximizes the value function, defined as the sum of expected discounted rewards over horizon  $h$  starting from initial belief state  $\vec{b}$ :

$$V_\pi^h(\vec{b}) = E_\pi \left[ \sum_{t=0}^h \gamma^t \cdot r_t \mid \vec{b}_0 = \vec{b} \right] \quad (4)$$

where  $r_t$  is the reward obtained at time  $t$  and  $\vec{b}_0$  is the belief state at  $t = 0$ . For finite  $h$  and belief state  $\vec{b}$ , the optimal policy  $\pi$  takes the form of an  $h$ -step conditional plan  $\beta^h$ . For  $h = \infty$ , the optimal discounted ( $\gamma < 1$ ) value function can be approximated arbitrarily closely by using a sufficiently large, but finite  $h$  [3].

Even when the state is continuous (but the actions and observations are discrete), the optimal POMDP value function for finite horizon  $h$  is a piecewise linear and convex function of the belief state  $\vec{b}$  [7] hence  $V^h$  is given by a maximum over a finite set of “ $\alpha$ -vectors”  $\alpha_i^h$ :

$$V^h(\vec{b}) = \max_{\alpha_i^h \in \Gamma^h} \alpha_i^h \cdot \vec{b} \quad (5)$$

Later on when we tackle continuous state *and* observations, we note that we will dynamically derive an optimal, finite partitioning of the observation space for a given belief state and hence reduce the continuous observation problem back to a discrete observation problem at every horizon.

The  $\Gamma^h$  in this optimal  $h$ -stage-to-go value function can be computed via Monahan’s dynamic programming approach to *value iteration* (VI) [4]. Initializing  $\alpha_1^0 = \vec{0}$  and  $\Gamma^0 = \{\alpha_1^0\}$ ,  $\Gamma^h$  is obtained from  $\Gamma^{h-1}$  using the backup operation defined in (2):<sup>2</sup>

$$g_{a,o,j}^h = \int_{x_{s'}} \sum_{d_{s'}} p(o | x_{d_s}', a) p(x_{d_s}' | x_{d_s}, a) \alpha_j^h(x_{d_s}') \quad (6)$$

$$\Gamma_a^h = r_a + \gamma \boxplus_{o \in \mathcal{O}} \left\{ g_{a,o,j}^h(\cdot) \right\}_j ; \forall \alpha_j^{h-1} \in \Gamma^{h-1} \quad (7)$$

<sup>1</sup>For compactness we define  $xd_s := (x_s, d_s)$  and  $xd_o := (x_o, d_o)$ .

<sup>2</sup>The  $\boxplus$  of sets is defined as  $\boxplus_{j \in \{1, \dots, n\}} S_j = S_1 \boxplus \dots \boxplus S_n$  where the pairwise cross-sum  $P \boxplus Q = \{\vec{p} + \vec{q} \mid \vec{p} \in P, \vec{q} \in Q\}$ .

---

**Algorithm 1:**  $\text{PBVI}(\text{DC-POMDP}, H, \text{ContObs}, B = \{b_i\}) \longrightarrow (V^h, \pi^{*,h})$

---

```

1 begin
2    $V^0 := 0, h := 0, \Gamma_{PBVI}^0 = \{\alpha_1^0\}$ 
3   while  $h < H$  do
4      $h := h + 1, \Gamma^h := \emptyset, \Gamma_{PBVI}^h := \emptyset$ 
5     foreach  $b_i \in B$  do
6       foreach  $a \in A$  do
7          $\Gamma_a^h := \emptyset$ 
8         if  $\text{ContObs} = \text{true}$  then
9           // Continuous observation so derive observation set  $O_i^h$  for belief  $b_i$ 
10           $(O_i^h, P(O_i^h | x d_s)) := \text{GenRelObs}(\Gamma_{PBVI}^{h-1}, a, b_i)$ 
11          foreach  $o \in O_i^h$  do
12            foreach  $\alpha_j^{h-1} \in \Gamma_{PBVI}^{h-1}$  do
13               $\alpha_j^{h'}, := \text{Prime}(\alpha_j^h) // \forall d_i \rightarrow d'_i \text{ and } \forall x_i \rightarrow x'_i$ 
14               $g_{a,o,j}^h := \text{see Eq (6)}$ 
15             $\Gamma_a^h := \text{see Eq (7)}$ 
16           $\Gamma^h := \Gamma^h \cup \Gamma_a^h$ 
17           $\Gamma^h := \text{Prune}(\Gamma^h)$ 
18        foreach  $b_i \in B$  do
19           $\alpha_{b_i}^h := \arg \max_{\alpha_j \in \Gamma^h} \alpha_j \cdot b_i$ 
20           $\Gamma_{PBVI}^h := \Gamma_{PBVI}^h \cup \alpha_{b_i}^h$ 
21
22        if  $\Gamma_{PBVI}^h = \Gamma_{PBVI}^{h-1}$  then
23          break // Terminate if early convergence
24
25    return  $\Gamma_{PBVI}$ 
26
27 end

```

---

$$\Gamma^h = \bigcup_a \Gamma_a^h \quad (8)$$

where  $r_a = r(x_s, d_s, a)$ .

**Point-based value iteration (PBVI)** computes the optimal value function only for a set of belief states  $\{b_i\}$  where  $b_i := P(x d_s)$ . The idea is straightforward and the main modification needed to Monahan's algorithm in Algorithm 1 is the loop from lines 19–21 where only  $\alpha$ -vectors optimal at some belief state are retained for subsequent iterations. We note that if all beliefs reachable in  $h$  steps from some initial set can be finitely enumerated and PBVI is executed for this set of beliefs, then optimality is guaranteed for the PBVI policy over these initial beliefs up to horizon  $h$ .

We pause for a moment to make a few notes about the PBVI algorithm in the context of DC-POMDPs. If we have a fixed discrete observation set, i.e.,  $\forall h \mathcal{O}^h = \mathcal{O}$ , then the algorithm proceeds as usual, but if we have continuous observation variables then we will need to derive a relevant set of observations on line 10, which we describe in Section 4.3. We need to guarantee that the integral on line 14 can be done in closed-form and we guarantee this for piecewise functions in Section 4.2. Because in general we will deal with  $\alpha$ -functions represented as piecewise functions, we only assume that strict dominance testing is implemented for `Prune` in line 18.

## 4 Symbolic Dynamic Programming

In this section we take a symbolic dynamic programming (SDP) approach to performing PBVI in Algorithm 1. This requires us to define all of the operations used in PBVI on a closed-form representation. For SDP, this closed-form will be a piecewise case statement and it will stored

compactly as a decision diagram [10]. We first review this form and then proceed to define SDP operations specific to the case of DC-POMDPs.

#### 4.1 Case Representation and Extended ADDs

The **Power Plant** example represented all functions using the case structure which can be generally defined as

$$f = \begin{cases} \phi_1 : & f_1 \\ \vdots & \vdots \\ \phi_k : & f_k \end{cases}$$

and this is the form we use to represent all functions in a DC-POMDP. Here  $\phi_i$  are disjoint logical formulae defined over the state  $(\vec{d}, \vec{x})$  with logical  $(\wedge, \vee, \neg)$  combinations of boolean variables and inequalities  $(\geq, >, \leq, <)$  over continuous variables. The  $f_i$  are general functions, we will restrict them later for the case of continuous observation POMDPs.

For *unary operations* such as scalar multiplication  $c \cdot f$  (for some constant  $c \in \mathbb{R}$ ) or negation  $-f$  on case statements is simply to apply the operation on each case partition  $f_i$  ( $1 \leq i \leq k$ ). A *binary operation* on two case statements, takes the cross-product of the logical partitions of each case statement and performs the corresponding operation on the resulting paired partitions. The cross-sum  $\oplus$  of two cases is defined as the following:

$$\begin{cases} \phi_1 : & f_1 \\ \phi_2 : & f_2 \end{cases} \oplus \begin{cases} \psi_1 : & g_1 \\ \psi_2 : & g_2 \end{cases} = \begin{cases} \phi_1 \wedge \psi_1 : & f_1 + g_1 \\ \phi_1 \wedge \psi_2 : & f_1 + g_2 \\ \phi_2 \wedge \psi_1 : & f_2 + g_1 \\ \phi_2 \wedge \psi_2 : & f_2 + g_2 \end{cases}$$

Likewise  $\ominus$  and  $\otimes$  are defined by subtracting or multiplying partition values. Inconsistent partitions can be discarded when they are irrelevant to the function value. A *symbolic case maximization* is defined as below:

$$\max \left( \begin{cases} \phi_1 : & f_1 \\ \phi_2 : & f_2 \end{cases}, \begin{cases} \psi_1 : & g_1 \\ \psi_2 : & g_2 \end{cases} \right) = \begin{cases} \phi_1 \wedge \psi_1 \wedge f_1 > g_1 : & f_1 \\ \phi_1 \wedge \psi_1 \wedge f_1 \leq g_1 : & g_1 \\ \phi_1 \wedge \psi_2 \wedge f_1 > g_2 : & f_1 \\ \phi_1 \wedge \psi_2 \wedge f_1 \leq g_2 : & g_2 \\ \vdots & \vdots \end{cases}$$

The following SDP operations on case statements require more detail than can be provided here, hence we refer the reader to the relevant literature:

- Restriction  $f|_{\phi}$ : Takes a function  $f$  to restrict only in cases that satisfy some formula  $\phi$  as defined in [10].
- Substitution  $f\sigma$ : Takes a set  $\sigma$  of variables and their substitutions, where the LHS of the substitution operator  $/$  represents the substitution variable and the RHS of the  $/$  represents the expression that should be substituted in its place as in [10].
- Integration  $\int_{x_1} f dx_1$ : Eliminates continuous variables by taking the integral of case statements  $f$  restricted to linear constraints and polynomial values [9].

The data structure of the *extended algebraic decision diagram* (XADD) [10] is used to support case statements and the required operations. Figure 1 (right) is an example of an XADD representation.

#### 4.2 PBVI for DC State and Discrete Observations

For discrete and continuous states and discrete observations, we use the symbolic version of Monahan's algorithm as the DC-POMDP solution represented in PBVI. The same approach is required for continuous observations only based on point-based dynamic programming backups.

Given an initial set of beliefs, we can compute the relevant observation partitions explained in the next section and then continue in a similar fashion of having discrete observations. Note that if the input to this algorithm is continuous observations  $ContObs = true$  then line 8 finds the relevant observation partitions, else discrete observations are assumed and their probabilities are given in the problem specifications such as section 2. Also a set of beliefs are given as the input as often

---

**Algorithm 2:**  $\text{GenRelObs}(\Gamma^h, a, b_i) \longrightarrow P(z)$ 


---

```

1 begin
2    $z := 0$ ; for all  $\alpha_j$  in  $\Gamma^{h-1}$  do
3     // Point-based backup operation
4      $\alpha_j(xd_s, xd_o) := \int_{s'} P(xd_{o_i}|xd'_{s_i}, a) \cdot P(xd'_{s_i}|xd_{s_i}, a) \cdot \alpha_j$ 
5   for all  $j$  in  $\alpha_j(xd_s, xd_o)$  do
6     // Generate value of  $\alpha$ -vector for belief  $b_i$  as a function of observations
7      $\delta_j(xd_o) := \int_{xd_{s_i}} b_i \cdot \alpha_j(xd_s, xd_o) \}$ 
8   for all  $j$  in  $\delta_j(xd_o)$  do
9     // Finding partition points on the observation space
10     $z := \max(\delta_j(xd_o), z)$ 
11  //  $\phi_{z_k}$  logical term of each partition of  $z$ 
12  // Generate probabilities and partitions of observations that distinguish optimal
     $\alpha$ -vector (conditional plan)
13   $P(z_k) := \int_{xd_o} \int_{xd_s} P(xd_o|xd'_s, a) * P(xd'_s|xd_s, a) * b_i * \mathbb{I}[\phi_{z_k}] d_{xd_o} d_{xd_s}$ 
14  return  $P(z)$ 
15 end

```

---

several beliefs are required to cover the belief space and thus the algorithm is executed for each one of  $b_i \in B$ . For each iteration, lines 11–12 product the following equation:

$$g_{a,o,j}^h = \int_{x_{s'}} \oplus_{d_{s'}} p(xd_o|xd'_s, a) p(xd'_s|xd_s, a) \alpha_j^h(xd'_s)$$

where the first probability is from equation (2) and the next probability is from equation (1) of the example. The  $\alpha$ -vectors are case functions primed using substitution in `Prime`. `Backup` performs regression on the  $\alpha$ -vectors where *Boolean regression* is performing a symbolic restriction and *Continuous regression* is a symbolic substitution, both SDP operations. Lines 14–16 cover closed form operations on the  $\Gamma$ -set such as cross-sum and union. Finally lines 18–20 find the maximum  $\alpha$ -vector within the  $\Gamma^h$  set and add this to the final  $\Gamma_{P_{BVI}}^h$  if it is not already in the set. In the next section we discuss the main part of our algorithm which is generating relevant observation partitions.

### 4.3 PBVI for DC State and DC Observations

The main operations required for value iteration in DC-POMDP is generating relevant observations to perform the bellman backup operation. The `GenRelObs` algorithm defines the steps required to generate these partitions. We use the second iteration of the **Power Plant** example to demonstrate partitioning of the entire continuous observation space. The main assumption is to consider some belief state and obtaining the relevant partitions according to that belief. We choose these beliefs such that they cover the entire probable state space; e.g.  $b_1 : U[t_s; 2, 6]$  and  $b_2 : U[t_s; 6, 11]$ .

Starting with an empty set of relevant observation partitions  $z$ , the algorithm first eliminates the next state variable  $xd'_s$ . It multiplies the observation model and the transition model in the set of  $\alpha$ -vectors of the previous horizon:  $\alpha_j(xd_s, xd_o) := \int_{s'} P(xd_{o_i}|xd'_{s_i}, a) * P(xd'_{s_i}|xd_{s_i}, a) * \alpha_j$ . This operation of line 3 in `GenRelObs` is similar to that of `Regress` multiplied by the observation regressed model. For continuous variables the continuous regression and for discrete variables boolean restriction is used as defined in the previous section. Assuming that after  $h = 1$  the set of  $\alpha$ -vectors are defined equal to the reward function of equation (1). For  $h = 2$  the resulting  $\alpha$ -vectors after the regression step are functions of the observation and current state:

$$\alpha_1^2(t_s, t_o) = \begin{cases} (t_s < 15) \wedge (t_s - 10 < t_o < t_s) & : 10 \\ (t_s > 15) \wedge (t_s - 10 < t_o < t_s) & : -100 \\ \neg(t_s - 10 < t_o < t_s) & : 0 \end{cases} \quad \alpha_2^2(t_s, t_o) = \begin{cases} (t_s - 10 < t_o < t_s) & : -0.1 \\ \neg(t_s - 10 < t_o < t_s) & : 0 \end{cases}$$

We need the  $\alpha$ -vectors to partition on the observation space, thus next we eliminate the current state variable in line 4–5. During this integration the current state variable is factored out by multiplying the belief in each of the current  $\alpha$ -vectors. The resulting  $\delta$ -functions only depend on the observation:  $\delta_j(xd_o) := \int_{xd_{s_i}} b_i * \alpha_j(xd_s, xd_o)$ . For  $b_1 : U[t_s; 2, 6]$  assume the following simple  $\delta$ -functions:

$$\delta_1(t_o) = \begin{cases} (-\infty < t_o < \infty) & : -0.4 * t_o + 6.8 \end{cases} \quad \delta_2(t_o) = \begin{cases} (-\infty < t_o < \infty) & : 1.3 * t_o + 3.4 \end{cases}$$

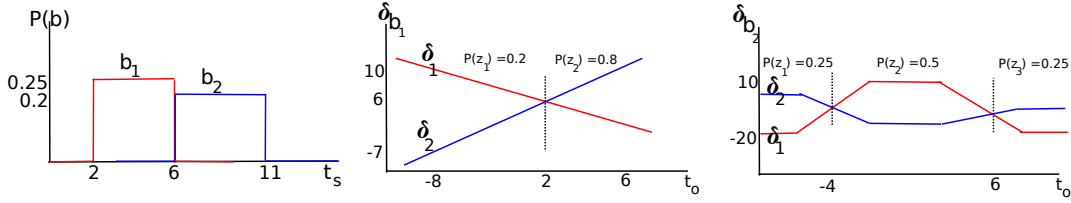


Figure 2: (left) Beliefs  $b_1, b_2$  for 1D **Power Plant** example; (middle) Observation dependent function for  $b_1$  that partition the observation space into 2 regions with different probabilities  $P(z_i)$ ; (right) 3 Relevant observation partitions and their probabilities for  $b_2$ .

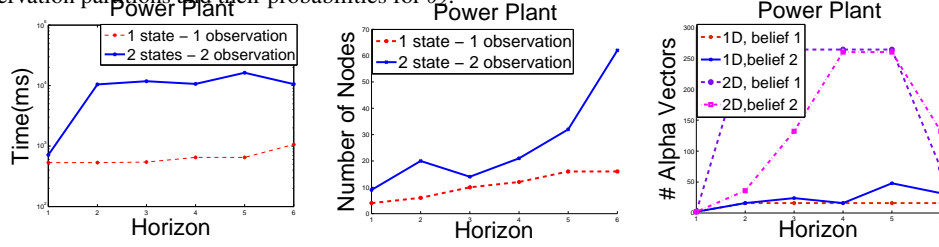


Figure 3: (left) Space vs Horizon; (middle) Time vs Horizon; (right) Number of  $\alpha$ -vectors vs Horizon.

The observation dependent  $\delta$ -functions divide the observation space into regions which can yield the optimal policy according to the belief state  $b_1$ . According to continuous observation space defined in [2], for a 1-dimensional observation space with two or more functions of the observation, we need to find the optimal boundaries or partitions of the space. In their work, numerical solutions are proposed to find the roots of any two observation dependent function. Here we use the symbolic power of the max-operator to find all the maximum boundary regions of two or more  $\delta$ -functions at the same time. For the two  $\delta$ -functions above the following partitions of the observation space is derived after taking their maximum in line 6–7:

$$\max \left( \delta_1(t_o), \delta_2(t_o) \right) = \begin{cases} z_1 : (t_o < 2 : -0.4 * t_o + 6.8) \\ z_2 : (t_o > 2 : 1.3 * t_o + 3.4) \end{cases}$$

The two partitions define observation regions which we can now use in a similar fashion to a discrete observation set if only the probability of each of the two distinct observation partitions is found. If we integrate out the state and observation from the observation model and the belief state regardless of a particular  $\alpha$ -vector, it integrates to one:  $\int_{x_{d_o}} \int_{x_{d_s}} P(x_{d_o}|x_{d_s}, a) * P(x_{d_s}|x_{d_s}, a) * P(s) = 1$ . Thus we only need to multiply the indicators of each case partition as as in line 9 in this formula to obtain the probability of each partition such that it sums to one:  $P(z_k) := \int_{x_{d_o}} \int_{x_{d_s}} P(x_{d_o}|x_{d_s}, a) * P(x_{d_s}|x_{d_s}, a) * b_i * \mathbb{I}[\phi_{z_k}] dx_{d_o} dx_{d_s}$ . For our example the probabilities are defined as  $P(z_1) = 0.2$  and  $P(z_2) = 0.8$ .

Hence we can now use the algorithms and methods of a discrete observation setting using the probabilities of the partitioned observation space in Monahan’s algorithm. Next we present some results for 2-Dimensional continuous observation spaces.

## 5 Empirical Results

The 2-dimensional model description is provided below:

**2D-Power Plant:** We consider the more complex model of the power plant similar to [1] where the pressure inside the water tank must be controlled to avoid mixing water into the steam or explosion of the tank. We model the pressure variable  $p$  as a partially observable variable from the observation readings of the pressure  $po$ . The two actions of increase and decrease are defined based on the change in both the temperature and the pressure. For the increase action we define:

$$P(p'_s | \vec{p}_s, inc) = \delta \left[ p'_s - \begin{cases} (p + 10 > 20) & : 20 \\ \neg(p + 10 > 20) & : p_s + 10 \end{cases} \right] \quad P(t'_s | \vec{t}_s, inc) = \delta [t'_s - (t_s + 10)]$$

There is a high reward for staying within the safe temperature and pressure range since it produces power, else depending on how safe it is to have values higher or lower than the safe range, penalty is defined.

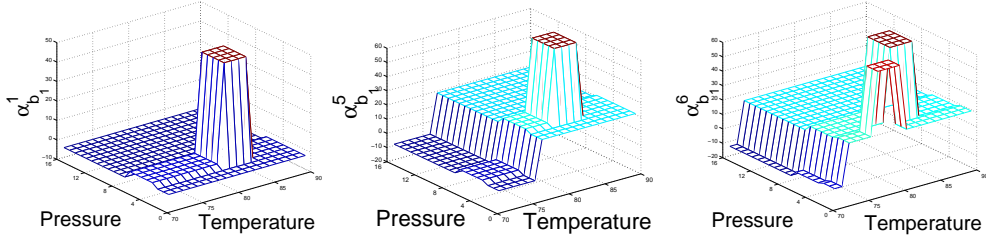


Figure 4: (left) Maximum  $\alpha$ -vector for  $b_1$  in first iteration; (middle)  $\alpha_{max}^5(b_1)$ ; (right)  $\alpha_{max}^6(b_1)$ .

$$R(t_s, p_s, inc) = \begin{cases} (5 \leq p_s \leq 15) \wedge (95 \leq t_s \leq 105) & : 50 \\ (5 \leq p_s \leq 15) \wedge (t_s \leq 95) & : -1 \\ (p_s \geq 15) & : -5 \\ else & : -3 \end{cases}$$

As for the decrease action, the transition functions reduce the temperature by 5 units and the pressure by 10 units as long as the pressure stays above zero. For the reward function, we assume that there is always a small penalty for decreasing the values because power can not be generated. For the observation model we consider two continuous uniform distributions such as the following:

$$P(t_o|t'_s) = \begin{cases} (t_s + 80 < t_o < t_s + 105) & : 0.4 \\ \neg(t_s + 80 < t_o < t_s + 105) & : 0 \end{cases} \quad P(p_o|p'_s) = \begin{cases} (p_s < p_o < p_s + 10) & : 0.1 \\ \neg(p_s < p_o < p_s + 10) & : 0 \end{cases}$$

We define two rectangular uniform beliefs around the regions of rewarding, so that one needs to increase the values while the other should decrease them:  $b_1 : U[t_s; 90, 100] * U[p_s; 0, 10]$  and  $b_2 : U[t_s; 90, 130] * U[p_s; 10, 30]$ . In Figure 3, a time and space analysis of the two versions of **Power Plant** have been performed for up to 6 horizons. As the algorithm progresses, the time required to compute the probability of the partitions and finding the maximum  $\alpha$ -vector with respect to beliefs increases for both problem sizes and significantly more for the 2D version. Increase in the problem size, increases the partition numbers on the observation space and this produces more  $\alpha$ -vectors which also effects the space required to perform the algorithm. The number of vectors stays the same for most horizons and they drop after convergence in the 2D problem instance. This shows that although the 2D instance takes more time and space than the 1D instance, it still converges within reasonable resources.

In Figure 4 we present plots of the maximum  $\alpha$ -vectors of belief  $b_1$  for different iterations of the 2D problem instance. Starting with the first iteration, the value is highest for the reward range  $(5 < p < 15 \wedge 95 < t < 105)$  and -1 or less for other places. In the fifth iteration, the value function has partitioned into more pieces, showing how higher temperatures can increase the value without considering the effect of the pressure. In the last plot, horizon 6 has better tuned the value function so that higher temperatures and pressures increase the value of the maximum  $\alpha$ -vector and also within the reward range, finer grain partitions have been formed.

## 6 Conclusion

We presented the first exact solution to continuous states and observations in a DC-POMDP framework. Transitions and reward functions are defined as piecewise linear functions of the continuous states. Continuous observations require definition of explicit partitions based on the observation model and the belief state of the agent. The key contribution is to define the partitions on the observation space and derive the related probabilities using symbolic integration and maximization. We extend the application to more than one state-observation setting and present the results.

## References

- [1] Mario Agueda and Pablo Ibarguengoytia. An architecture for planning in uncertain domains. In *Proceedings of the ICTAI 2002 Conference*, Dallas, Texas, 2002.
- [2] Jesse Hoey and Pascal Poupart. Solving pomdps with continuous or large discrete observation spaces. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Edinburgh, Scotland, 2005.



- 432 [3] Leslie P. Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in  
433 partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.
- 434 [4] G. E. Monahan. Survey of partially observable markov decision processes: Theory, models,  
435 and algorithms. *Management Science*, 28(1):1–16, 1982.
- 436 [5] J. Scott Penberthy and Daniel S. Weld. Temporal planning with continuous change. In *National*  
437 *Conference on Artificial Intelligence AAAI*, pages 1010–1015, 1994.
- 438 [6] Joelle Pineau, Geoffrey J. Gordon, and Sebastian Thrun. Anytime point-based approximations  
439 for large pomdps. *J. Artif. Intell. Res. (JAIR)*, 27:335–380, 2006.
- 440 [7] J. M. Porta, N. Vlassis, M.T.J. Spaan, and P. Poupart. Point-based value iteration for continuous  
441 pomdps. *Journal of Machine Learning Research*, 7:195220, 2006.
- 442 [8] Pascal Poupart, Kee-Eung Kim, and Dongho Kim. Closing the gap: Improved bounds on  
443 optimal pomdp solutions. In *In Proceedings of the 21st International Conference on Automated*  
444 *Planning and Scheduling (ICAPS-11)*, 2011.
- 445 [9] Scott Sanner and Ehsan Abbasnejad. Symbolic variable elimination for discrete and continuous  
446 graphical models. In *In Proceedings of the 26th AAAI Conference on Artificial Intelligence*  
447 *(AAAI-12)*, Toronto, Canada, 2012.
- 448 [10] Scott Sanner, Karina Valdivia Delgado, and Leliane Nunes de Barros. Symbolic dynamic  
449 programming for discrete and continuous state mdps. In *Proceedings of the 27th Conference*  
450 *on Uncertainty in AI (UAI-2011)*, Barcelona, 2011.
- 451 [11] Trey Smith and Reid G. Simmons. Point-based POMDP algorithms: Improved analysis and  
452 implementation. In *Proc. Int. Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2005.
- 453 [12] M. Spaan and N. Vlassis. Perseus: Randomized point-based value iteration for pomdps. *Journal*  
454 *of Artificial Intelligence Research (JAIR)*, page 195220, 2005.
- 455
- 456
- 457
- 458
- 459
- 460
- 461
- 462
- 463
- 464
- 465
- 466
- 467
- 468
- 469
- 470
- 471
- 472
- 473
- 474
- 475
- 476
- 477
- 478
- 479
- 480
- 481
- 482
- 483
- 484
- 485