# POMDP and RL for the CMAPPS model

Yannik Pitcan
Panos Lambrianides
Ram Akella
Anil Aswani
Phil Kaminsky

March 15, 2019

## 1 Our Model

Note that these are specific to our NASA CMAPPS dataset.

In our model, we have a vector of sensor measurements as our observations and for our state space, we have a numeric value denoting the health of our system as a number from 0 to 1.

However, we cannot necessarily assume our observed sensor values have a linear relationship with the health of our system. This is where we have some added complexity compared to the "toy problem" mentioned in the next section.

Key idea here because we don't see 'states' (health of the system) – we have to use some form of estimation to get a guess of our system's health. One way of doing this is via a dynamic linear model.

We can note that the end of a cycle corresponds to state 0 and the beginning corresponds to state 1 and use a DLM to infer the hidden states in between, since we have sensor observations throughout.

Another way to do state estimation here is by fitting the health of our system to a Weibull distribution. The question I have with this is "how do we then get knowledge of transition probabilities between states?"

We have a list of RUL values that serves as a 'validation set'. What I'm not sure about: how to combine RUL data with training vs test data. I could just run my DLM on this combination of training and test data and then have state inferences for every timestep.

Define a set of belief states $B$ in our model by $(z, 1-z)$, where $z \in [0,1]$ and $z$ is the probability (or ”belief”) that our system is functioning well and $1-z$ for a failing system.

Although we don't see our true ”state”, we can model our ”belief state” as a non-linear function of our sensor measurements. If $y$ is our vector of sensor measurements, let $f(y) = z$.

## 2 Simpler Model Example

This is to give intuition about what we're doing.

Let this be a $2 \times 2$ matrix where state 1 is a poorly performing machine and 2 is a new machine.

Transition probabilities are

$$P(1) = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$$

and

$$P(2) = \begin{bmatrix} 1 & 0 \\ \theta & 1-\theta \end{bmatrix}$$

where $\theta$ is the probability of deterioration.

$$B = \begin{bmatrix} p & 1-p \\ 1-q & q \end{bmatrix}$$

is the observation probability matrix where $p$ is the probability that a machine in a good state produces quality output and $q$ is the probability a poor machine produces low quality output.

In state $x$, operating cost is $c(x, u = 2)$. Replacing the machine at $x$ costs $R$, i.e. $c(x, u = 1) = R$.

Want to minimize

$$\mathrm{E}_{\mu} \left\{ \sum c(x_k, u_k) | \pi_0 \right\}$$

for a horizon $N$.

# 3 POMDPs

## 3.1 Definitions and Notations

A discrete-time POMDP models the relationship between an agent and its environment. Formally, a POMDP is a 7-tuple $(S, A, T, R, \Omega, O, \gamma)$, where

- $S$ is a set of states,
- $A$ is a set of actions,
- $T$ is a set of conditional transition probabilities between states,
- $R : S \times A \to \mathbb{R}$ is the reward function.
- $\Omega$ is a set of observations,
- $O$ is a set of conditional observation probabilities, and
- $\gamma \in [0, 1]$ is the discount factor.

At each time period, the environment is in some state $s \in S$. The agent takes an action $a \in A$, which causes the environment to transition to state $s'$ with probability $T(s' \mid s, a)$. At the same time, the agent receives an observation $o \in \Omega$ which depends on the new state of the environment, $s'$, and on the just taken action, $a$, with probability $O(o \mid s', a)$. Finally, the agent receives a reward $r$ equal to $R(s, a)$. Then the process repeats. The goal is for the agent to choose actions at each time step that maximize its expected future discounted reward: $E\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$, where $r_t$ is the reward earned at time $t$. The discount factor $\gamma$ determines how much immediate rewards are favored over more distant rewards. When $\gamma = 0$ the agent only cares about which action will yield the largest expected immediate reward; when $\gamma = 1$ the agent cares about maximizing the expected sum of future rewards.

Because the agent does not directly observe the environment's state, the agent must make decisions under uncertainty of the true environment state. However, by interacting with the environment and receiving observations, the agent may update its belief in the true state by updating the probability distribution of the current state. A consequence of this property is that the optimal behavior may often include (information gathering) actions that are taken purely because they improve the agent's estimate of the current state, thereby allowing it to make better decisions in the future.

## 3.2 Belief Updates

After having taken the action $a$ and observing $o$, an agent needs to update its belief in the state the environment may (or not) be in. Since the state is Markovian, maintaining a belief over the states solely requires knowledge of the previous belief state, the action taken, and the current observation. The operation is denoted $b' = \tau(b, a, o)$. Below we describe how this belief update is computed.

After reaching $s'$, the agent observes $o \in \Omega$ with probability $O(o \mid s', a)$. Let $b$ be a probability distribution over the state space $S$. $b(s)$ denotes the probability that the environment is in state $s$. Given $b(s)$, then after taking action $a$ and observing $o$,

$$b'(s') = \eta O(o \mid s', a) \sum_{s \in S} T(s' \mid s, a) b(s)$$

where $\eta = 1/ \Pr(o \mid b, a)$ is a normalizing constant with

$$\Pr(o \mid b, a) = \sum_{s' \in S} O(o \mid s', a) \sum_{s \in S} T(s' \mid s, a) b(s)$$

.

## 3.3 Belief MDP

A Markovian belief state allows a POMDP to be formulated as a Markov decision process where every belief is a state. The resulting ”belief MDP” will thus be defined on a continuous state space (even if the "originating" POMDP has a finite number of states: there are infinite belief states (in $B$) because there are an infinite number of mixtures of the originating states (of $S$)), since there are infinite beliefs for any given POMDP.

Formally, the belief MDP is defined as a tuple $(B, A, \tau, r, \gamma)$ where

- $B$ is the set of belief states over the POMDP states,
- $A$ is the same finite set of action as for the original POMDP,
- $\tau$ is the belief state transition function,
- $r : B \times A \to \mathbb{R}$ is the reward function on belief states,
- $\gamma$ is the discount factor equal to the $\gamma$ in the original POMDP.

Of these, $\tau$ and $r$ need to be derived from the original POMDP. $\tau$ is

$$\tau(b, a, b') = \sum_{o \in \Omega} \Pr(b'|b, a, o) \Pr(o|a, b),$$

where $\Pr(o|a, b)$ is the value derived in the previous section and

$$Pr(b'|b, a, o) = \begin{cases} 1 & \text{if the belief update with arguments } b, a, o \text{ returns } b' \\ 0 & \text{otherwise} \end{cases}.$$

The belief MDP reward function ($r$) is the expected reward from the POMDP reward function over the belief state distribution:

$$r(b, a) = \sum_{s \in S} b(s) R(s, a)$$

.

The belief MDP is not partially observable anymore, since at any given time the agent knows its belief, and by extension the state of the belief MDP.

### 3.4 Policy and Value Function

Unlike the "originating" POMDP (where each action is available from only one state), in the corresponding Belief MDP all belief states allow all actions, since you (almost) always have ”some” probability of believing you are in any (originating) state. As such, $\pi$ specifies an action $a = \pi(b)$ for any belief $b$.

Here it is assumed the objective is to maximize the expected total discounted reward over an infinite horizon. When $R$ defines a cost, the objective becomes the minimization of the expected cost.

The expected reward for policy $\pi$ starting from belief $b_0$ is defined as

$$V^\pi(b_0) = \sum_{t=0}^{\infty} \gamma^t r(b_t, a_t) = \sum_{t=0}^{\infty} \gamma^t E\Big[R(s_t, a_t) \mid b_0, \pi\Big]$$

where $\gamma < 1$ is the discount factor. The optimal policy $\pi^*$ is obtained by optimizing the long-term reward.

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\ V^\pi(b_0)$$

where $b_0$ is the initial belief.

The optimal policy, denoted by $\pi^*$, yields the highest expected reward value for each belief state, compactly represented by the optimal value function $V^*$. This value function is solution to the Bellman optimality equation

$$V^*(b) = \max_{a \in A}\Big[r(b, a) + \gamma \sum_{o \in \Omega} O(o \mid b, a) V^*(\tau(b, a, o))\Big]$$

For finite-horizon POMDPs, the optimal value function is piecewise-linear and convex. It can be represented as a finite set of vectors. In the infinite-horizon formulation, a finite vector set can approximate $V^*$ arbitrarily closely, whose shape remains convex. Value iteration applies dynamic programming update to gradually improve on the value until convergence to an $\epsilon$-optimal value function, and preserves its piecewise linearity and convexity. By improving the value, the policy is implicitly improved. Another dynamic programming technique called policy iteration explicitly represents and improves the policy instead.