UCI University of California, Irvine

CS 295: Optimal Control and Reinforcement Learning

Winter 2020

Lecture 12: Advanced Partial Observability Methods

Roy Fox
Department of Computer Science
Bren School of Information and Computer Sciences
University of California, Irvine

Today's lecture

- Belief-state value function
- Point-Based Value Iteration (PBVI)
- Predictive State Representations (PSRs)
- Learning PSRs

Belief-state MDP

- Since the (hidden) state separates the past and the future

$$p(f_t|h_t, a_{\geqslant t}) = \sum_{s_t} p(s_t|h_t)p(f_t|s_t, a_{\geqslant t}) = \sum_{s_t} b_t(s_t)p(f_t|s_t, a_{\geqslant t})$$

  - its posterior distribution, a.k.a the Bayesian belief, is also a separator = state

- No advantage by the agent policy having further dependence on the past

Belief-state value function

$$V_\pi(b_t) = \mathbb{E}[R_{\geqslant t}|b_t]$$
$$= \sum_{s_t,a_t,s_{t+1},o_{t+1}} b_t(s_t)\pi(a_t|b_t)p(s_{t+1}|s_t,a_t)p(o_{t+1}|s_{t+1})(r(s_t,a_t) + \gamma V_\pi(b_{t+1}))$$

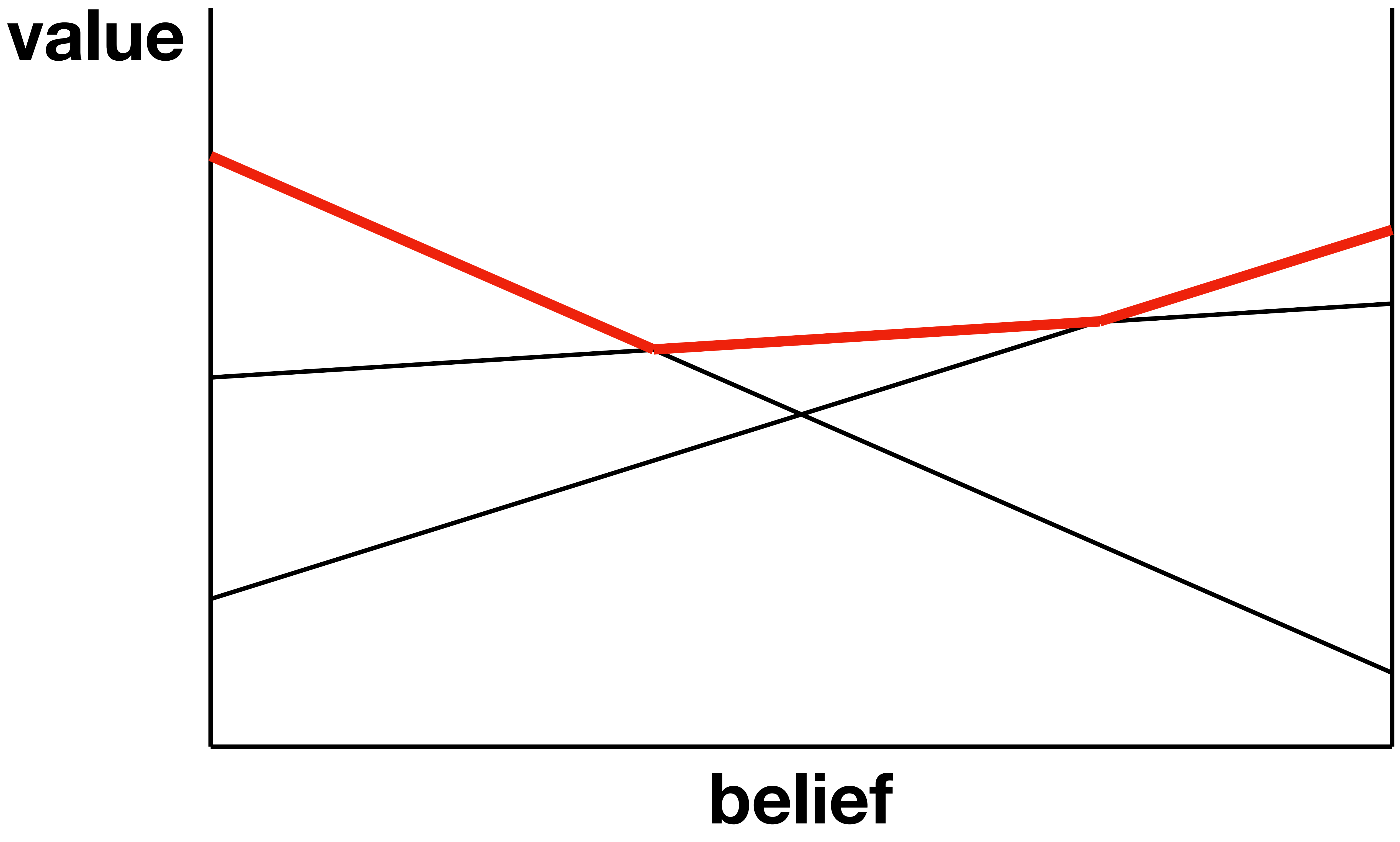
- With $b_{t+1}(\bar{s}_{t+1}) = p(\bar{s}_{t+1}|b_t, a_t, o_{t+1})$
- Note that $V_\pi(b_t)$ is **linear** in b_t
- Therefore, optimal value satisfies

$$V^*(b_t) = \max_{\pi\in\Pi} V_\pi(b_t) = \max_{\nu\in\mathcal{V}} b_t\nu$$

- Where for each $\pi(a|b)$ we have $V_\pi(b_t) = \sum_{s_t} b_t(s_t)\nu(s_t)$

Belief-state value function

- Piecewise-linear function:



- Can be represented by set of supporting vectors $\mathcal{V}$

First-action partitioning

$$V^*(b_t) = \max_\pi \sum_{s_t,a_t,s_{t+1},o_{t+1}} b_t(s_t)\pi(a_t|b_t)p(s_{t+1}|s_t,a_t)p(o_{t+1}|s_{t+1})(r(s_t,a_t) + \gamma V^*(b_{t+1}))$$

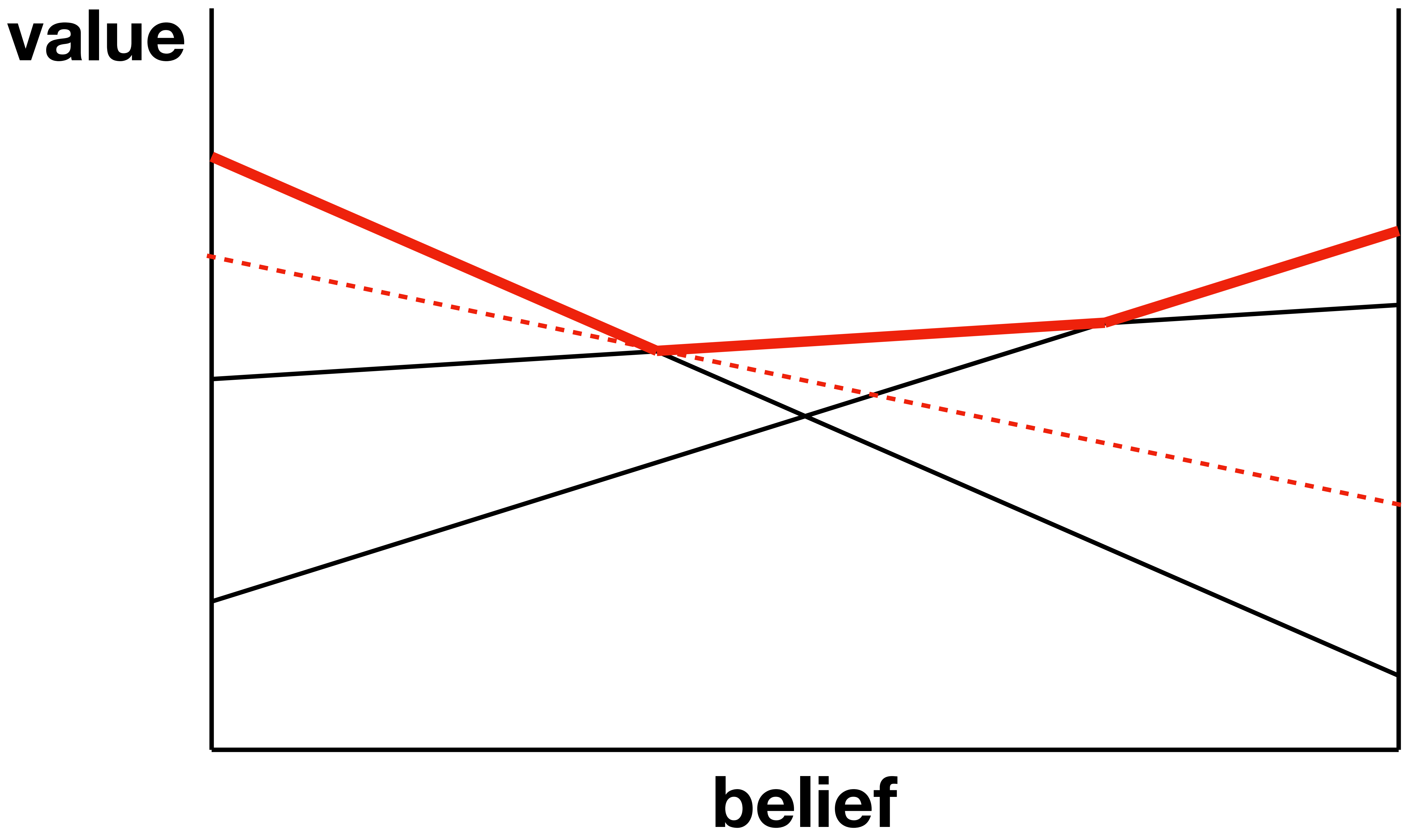
$$= \max_{a_t} \sum_{s_t} b_t(s_t) \left(r(s_t,a_t) + \gamma \sum_{s_{t+1},o_{t+1}} p(s_{t+1}|s_t,a_t)p(o_{t+1}|s_{t+1})V^*(b_{t+1}) \right)$$

- The optimal **value** can be found by a deterministic action
  - But the optimal **policy** can be stochastic, a mixture of optimal actions
- Optimal supporting set can be partitioned by first action

$$\mathcal{V} = \bigcup_a \mathcal{V}_a$$

So do we need stochastic policies?

- For some beliefs, the optimal policy may be stochastic



- The value function is still supported by deterministic policies ("backward")
    - But their "forward" may lead to worse belief-states

Value Iteration in belief-state MDP

- Recalling that

$$b_{t+1}(s_{t+1}|b_t, a_t, o_{t+1}) = \frac{\sum_{s_t} b_t(s_t)p(s_{t+1}|s_t, a_t)p(o_{t+1}|s_{t+1})}{\sum_{s_t,\bar{o}_{t+1}} b_t(s_t)p(s_{t+1}|s_t, a_t)p(\bar{o}_{t+1}|s_{t+1})}$$

we have

$$V^*(b_t, a_t) = \sum_{s_t} b_t(s_t)\left(r(s_t, a_t) + \gamma \sum_{s_{t+1},o_{t+1}} p(s_{t+1}|s_t, a_t)p(o_{t+1}|s_{t+1})V^*(b_{t+1})\right)$$

$$= \sum_{s_t} b_t(s_t)r(s_t, a_t) + \gamma \sum_{o_{t+1}} \max_{\nu'} \sum_{s_t,s_{t+1}} b_t(s_t)p(s_{t+1}|s_t, a_t)p(o_{t+1}|s_{t+1})\nu'(s_{t+1})$$

- And so

$$\mathcal{V}_{a,o'} = \left\{ \nu(s) = \sum_{s'} p(s'|s, a)p(o'|s')\nu'(s') \,\middle|\, \nu' \in \mathcal{V} \right\}$$

$$\mathcal{V}_a = r(\cdot, a) + \gamma \bigoplus_{o'} \mathcal{V}_{a,o'} \qquad \mathcal{V} = \bigcup_a \mathcal{V}_a$$

Representing belief value by its support

- Another curse of history: the support of $\mathcal{V}$ has at worst $|\mathcal{A}|^{|\mathcal{O}|^{T-t}}$ vectors
  - For infinite horizon, value function may be **uncomputable!**
- Do we need all of them?
  - Some may be optimal only in unreachable beliefs
  - Some may be optimal for beliefs not reached by an optimal policy
  - Some may be optimal for beliefs with low probability of being reached
  - Some may only be slightly better than others on likely beliefs

Point-Based Value Iteration (PBVI)

- Only try to optimize the value for a finite set of belief points $\mathcal{B}$
- That means having a small subset $\mathcal{V}^{\mathcal{B}}$ of all support vectors
- As before we have

$$\mathcal{V}^{\mathcal{B}}_{a,o'} = \left\{ \nu(s) = \sum_{s'} p(s'|s,a)p(o'|s')\nu'(s') \middle| \nu' \in \mathcal{V}^{\mathcal{B}} \right\}$$

- But now we optimize the policy suffix for a specific belief point

$$\nu^b_a = r(\cdot, a) + \gamma \sum_{o'} \operatorname*{argmax}_{\nu' \in \mathcal{V}^{\mathcal{B}}_{a,o'}} b \cdot \nu'$$

- Then optimize the first action, and repeat for all belief points

$$\mathcal{V}^{\mathcal{B}} = \left\{ \operatorname*{argmax}_{\{\nu^b_a\}_a} b \cdot \nu^b_a \middle| b \in \mathcal{B} \right\}$$

PBVI belief set expansion

- With fixed $\mathcal{B}$, repeat the approximate VI backward until near-convergence
- Then expand the belief set to improve belief-space coverage
  - For each $b \in \mathcal{B}$ and a, sample the following observation o', compute $b'(\cdot|b, a, \cdot)$
  - For each $b \in \mathcal{B}$, add belief farthest from $\mathcal{B}$ in L_1
- To use: $\pi(b) = \operatorname*{argmax}_a b \cdot \nu_a^b$
- Proposition: let $\epsilon = \max_{b \text{ reachable}} \min_{b' \in \mathcal{B}} \|b' - b\|_1$ be the density of $\mathcal{B}$, then

$$\|V^* - V^{\mathcal{B}}\|_\infty \leqslant \frac{1}{(1-\gamma)^2} R_{\max} \epsilon$$

Learning with partial observation

- Learning with partial observation is particularly challenging
  - If we never see states, how do we know
    - how to represent them?
    - how many there are?
  - New challenge of exploration
  - New challenge of model-selection
    - how to choose robust representations among equivalent ones?
    - how to discover the causal structure?

Learning: exponentially harder than planning

- In MDPs, we had polynomial model-based learning (E^3, R-max)
- In POMDPs, learning can be exponentially harder than planning
- Password game: guess n bits, unobservable, reward on success
  - Planning: with the dynamics known, password is known
  - Learning: have to brute-force, exponentially many guesses
- What if we can pay to observe state?
  - Can be set up such that optimal policy cannot pay → only used in training
  - Polynomial sample complexity in some classes

Predictive State Representations (PSR)

- Model environment using just observable elements
- Test: future action–observation sequence $a_t, o_{t+1}, \dots, a_{t+k-1}, o_{t+k}$
- History: past action–observation sequence $a_{t-\ell}, o_{t-\ell+1}, \dots, a_{t-1}, o_t$
- Predictive state: $m(h) = \{p(\tau_o | h, \tau_a) | \tau \in \mathcal{T}\}$, for a set of *core tests* $\mathcal{T}$
- m is a sufficient statistic (i.e. state)
  - if and only if the probability of *all* tests can be computed from it

Linear PSR

- Suppose that for every test τ there exists a vector u_τ with

$$\forall h: \;\; p(\tau_o|h,\tau_a) = m(h)\cdot u_\tau$$

- Let $U_{a,o'} = \{u_{a,o',\tau}|\tau\in\mathcal{T}\}$
- Then $u_{a_t,o_{t+1},\ldots,a_{t+k-1},o_{t+k}} = U_{a_t,o_{t+1}}\cdots U_{a_{t+k-1},o_{t+k}} u_\epsilon$
- We can update the state using

$$m(h,a,o')_\tau = \frac{p(o',\tau_o|h,a,\tau_a)}{p(o'|h,a)} = \frac{m(h)\cdot u_{a,o',\tau}}{m(h)\cdot u_{a,o'}} = \frac{m(h)(U_{a,o'})_\tau}{m(h)U_{a,o'}u_\epsilon}$$

- Core test set $\mathcal{T}$ is **minimal** if the tests are linearly independent

POMDPs are PSRs

- Every test is a linear function of the belief

$$p(o_{t+1},\dots,o_{t+k}|h_t,a_t,\dots,a_{t+k-1}) = \sum_{s_t,\dots,s_{t+k}} b_t(s_t|h_t) \prod_{t'=t}^{t+k-1} p(s_{t'+1}|s_{t'},a_{t'})p(o_{t'+1}|s_{t'+1})$$

  having

$$w_\tau(s_t) = \sum_{s_{t+1},\dots,s_{t+k}} \prod_{t'=t}^{t+k-1} p(s_{t'+1}|s_{t'},a_{t'})p(o_{t'+1}|s_{t'+1})$$

- If we find a set of $|\mathcal{S}|$ linearly independent tests consisting the columns of W

  then $$m(h) = b(h)W \qquad u_\tau = W^{-1}w_\tau$$

- Model-based **discovery** of core tests using depth-first search

Two PSRs problems

- **Discovery:** find an (approximately) spanning set of core tests
  - Easy to do given the POMDP
  - In general, this is the hard part
- **Learning:** given the core tests, find $m(o_0)$, $U_{a,o'}$, and u_ϵ
  - Can be estimated purely from observable interaction data

What can the agent experience

- Fix some partition of histories $\mathcal{H}$, large set of tests $\hat{\mathcal{T}}$ with $\hat{U} = \{u_\tau | \tau \in \hat{\mathcal{T}}\}$
- Empirical probability of a test in initial history:

$$P_{o_0,\tau} = p(\tau_o | o_0, \tau_a) = (m(o_0)\hat{U})_\tau$$

- Empirical joint probability of history and test:

$$P_{i,\tau} = p_\pi(h \in \mathcal{H}_i, \tau_o | \tau_a) = p_\pi(\mathcal{H}_i)\,\mathbb{E}[m(h) | h \in \mathcal{H}_i] u_\tau = (DS\hat{U})_{i,\tau}$$

  - with $D = \operatorname{diag}(p_\pi(\mathcal{H}_i))_i$ and $S_{i,\tau} = \mathbb{E}[m(h)_\tau | h \in \mathcal{H}_i]$ in core tests
- Empirical one-step joint probability:

$$P_{i,a,o',\tau} = p_\pi(h \in \mathcal{H}_i, o', \tau_o | a, \tau_a) = (DSU\hat{U})_{i,\tau}$$

Transformed PSRs (TPSRs)

- Everything we observe is in the space of the large set of tests $\hat{U}$
- We should make $\hat{\mathcal{T}}$ (and the history partition) diverse enough to span U
- If we knew the core tests, multiplying by $\hat{U}^\dagger$ would recover them
- Otherwise, we can only recover the PSR up to invertible transform W

Recovering the TPSR

- Recall:

$$P_{o_0,\mathcal{T}} = m(o_0)\hat{U}$$

$$P_{\mathcal{H},\mathcal{T}} = DS\hat{U}$$

$$P_{\mathcal{H},a,o',\mathcal{T}} = DSU\hat{U}$$

- With $\hat{W} = \hat{U}^\dagger W$ we can recover

$$\tilde{m}(o_0) = m(o_0)W = P_{o_0,\mathcal{T}}\hat{W}$$

$$\tilde{U} = W^{-1}UW = (P_{\mathcal{H},\mathcal{T}}\hat{W})^\dagger P_{\mathcal{H},a,o',\mathcal{T}}\hat{W}$$

- To recover the ϵ-test "marginalizer", estimate $P_{\mathcal{H}} = DSu_\epsilon$

  and compute $$\tilde{u}_\epsilon = W^{-1}u_\epsilon = (P_{\mathcal{H},\mathcal{T}}\hat{W})^\dagger P_{\mathcal{H}}$$

How to find good transformed test basis

- Compute the singular value decomposition (SVD) of

$$P_{\mathcal{H},\mathcal{T}} = DS\hat{U} = V_1 \Sigma V_2^\intercal$$

- and take $\hat{W} = \hat{U}^\dagger W$ to include the right singular vectors in V_2
    - Most interesting and stable tests correspond to the largest singular values in Σ

Recap

- Belief-state value function is piecewise linear
  - Can be represented by supporting vectors
  - But there are exponentially many
  - We can approximate by using a subset of the supporting vectors
    - PBVI: choose vectors by (recursive) optimality for beliefs we care about
- We can learn partially observable models from just observable interaction
  - PSR: how is the observable future distributed given the observable past
  - Can discover (transformed) tests and learn state updates
  - Use this in a model-based algorithm