Probabilistic Co-adaptive brain computer interfacing

V. Deepak Raya

Department of Avionics

MDPRL course project presentation , 2020

Outline

Outline

Introduction

- The Probabilistic co-adaptive approach to BCIs, provides a unified frame-work for tackling the problems of uncertainty and co-adaptation.
- Uncertainty: Decoding noisy brain signals
  Co-adaptation: Between brain and interface, co-operatively achieve a common goal
- This frame work is based upon POMDPs and RL
- Experimental paradigm: non-invasive BCI based on SSVEP signals.

Outline

POMDPs

- Partial Observability + MDPs : Not such a trivial addition
- CO-MDPs give values or policy for each state, for this solutions state is to be known completly at all times
- In POMDPs, Partial observability clouds the idea of current state
  - We add a set of observations to the MDP model, which gives hint about the current state.
  - Since the current sate is behind the clouds, the agent maintains a probability distribution over states called as belief
- POMDPs are described by 7-tuple $\{S, A, Z, T, O, R, \gamma\}$

POMDPs

- Z is the set of possible observations,(using sensor measurements), which give indirect access to the hidden states.
- T describes the state transition probabilities, which govern the dynamics of hidden states i.e. $T(s', s, a) = p(s'|s, a)$
- O is the observation model, defined as the set of observation probabilities for given action and resulting state, i.e $O(o, a, s') = p(o|a, s')$.

Beliefs & updating Beliefs

- Accounts for its own degree of uncertainty about the current state
- Posterior probability distribution over the current state, given the history.
- Uses Bayesian filtering for updating the beliefs
-

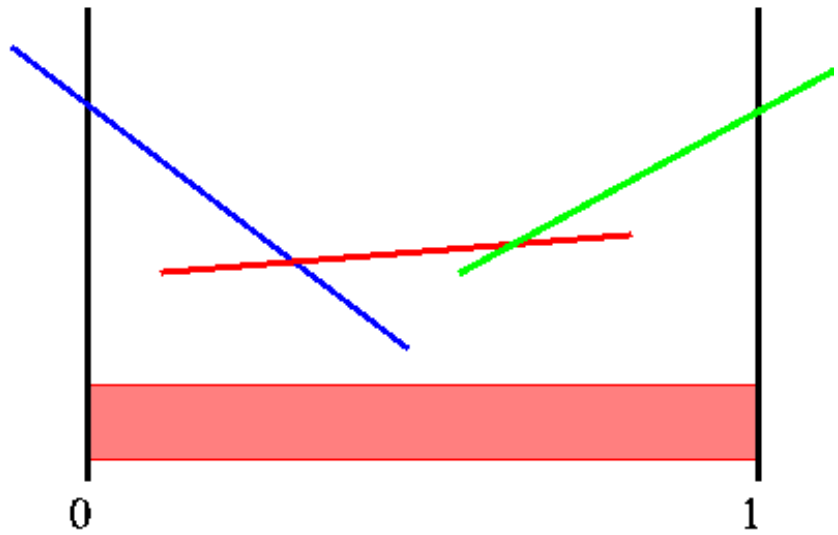
$$
\begin{aligned}
b'(s') &= p(s'|o,a,b) \\
&= \eta\, p(o|s',a)\, p(s'|a,b) \\
&= \eta\, p(o|s',a) \sum_{s \epsilon S} p(s'|s,a)\, p(s|a,b) \\
&= \eta\, p(o|s',a) \sum_{s \epsilon S} p(s'|s,a)\, b(s)
\end{aligned}
$$

General overview of Solving a POMDP

- Our Goal is to derive a policy $\pi : B \longrightarrow A$
  where, B is the belief space
- Which maximizes the total expected reward
- Calculating the optimal policy exactly is intractable, in general
- Algorithms exist for finding approximate solutions in reasonable time

General overview of Solving POMDP

- Keeping things simple, we will consider a 2 state POMDP
- A finite horizon value function of a POMDP, is Piece wise linear and convex, for every horizon length
- Each value hyperplane is represented by a vector of dimensions equal to # of states



General overview of Solving POMDP

- Consider 2 actions, 2 states and 3 observations POMDP.
- since, 2 states and 2 actions, we have four separate possible rewards
- let us take

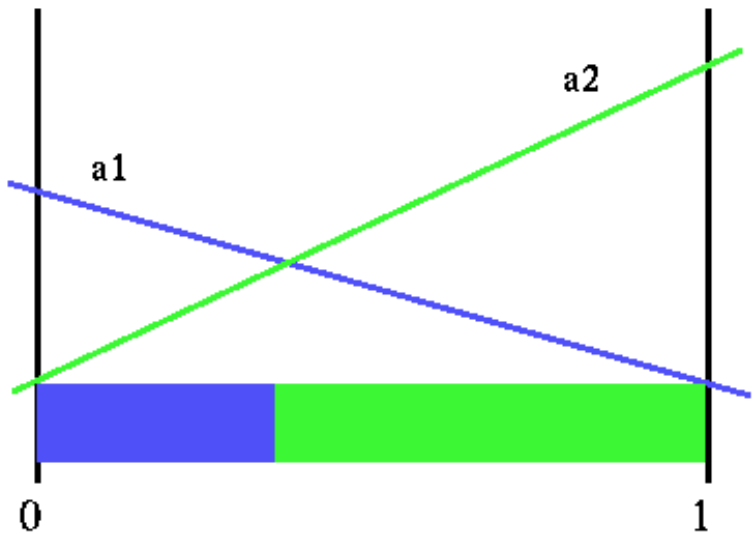
$$a_1, s_1 \longleftrightarrow 1$$

$$a_1, s_2 \longleftrightarrow 0$$

$$a_2, s_1 \longleftrightarrow 0$$

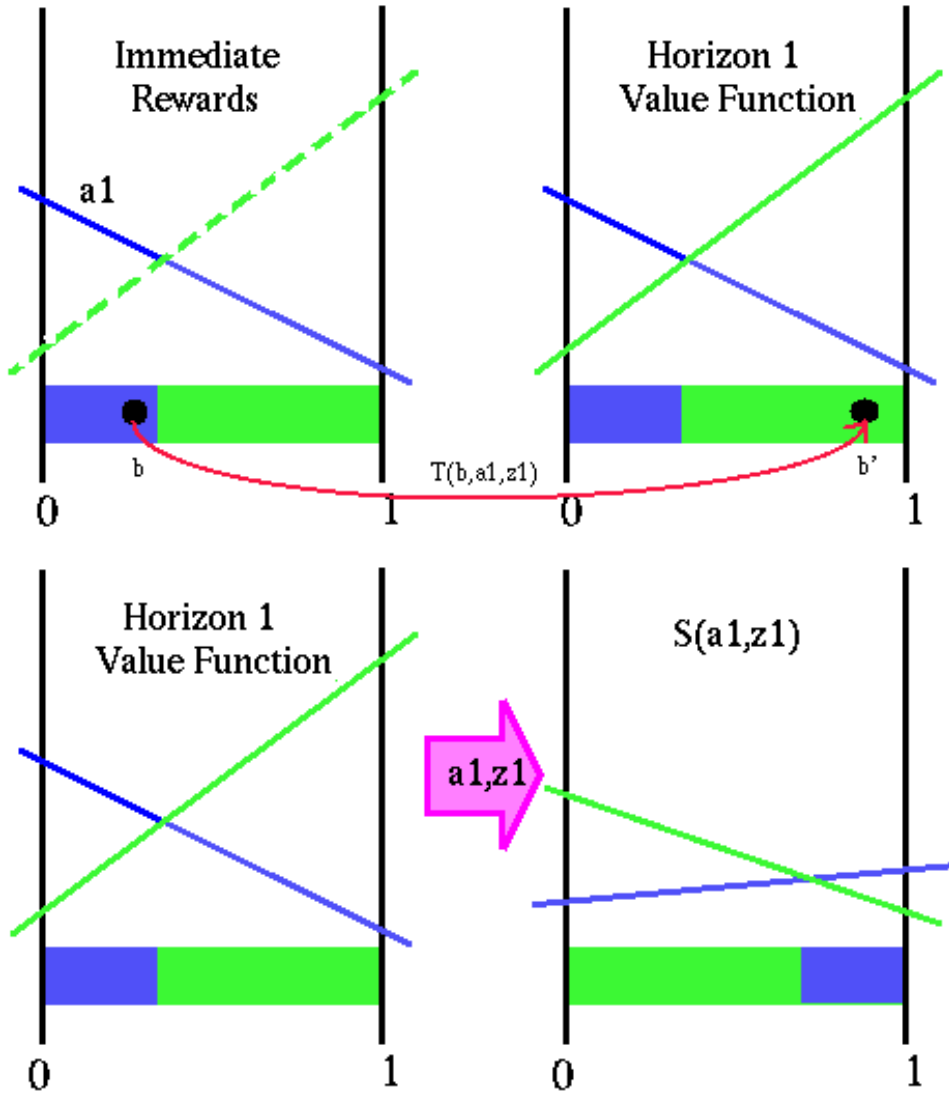
$$a_2, s_2 \longleftrightarrow 1.5$$

- horizon 1 value function are nothing but the immediate rewards, as shown in fig



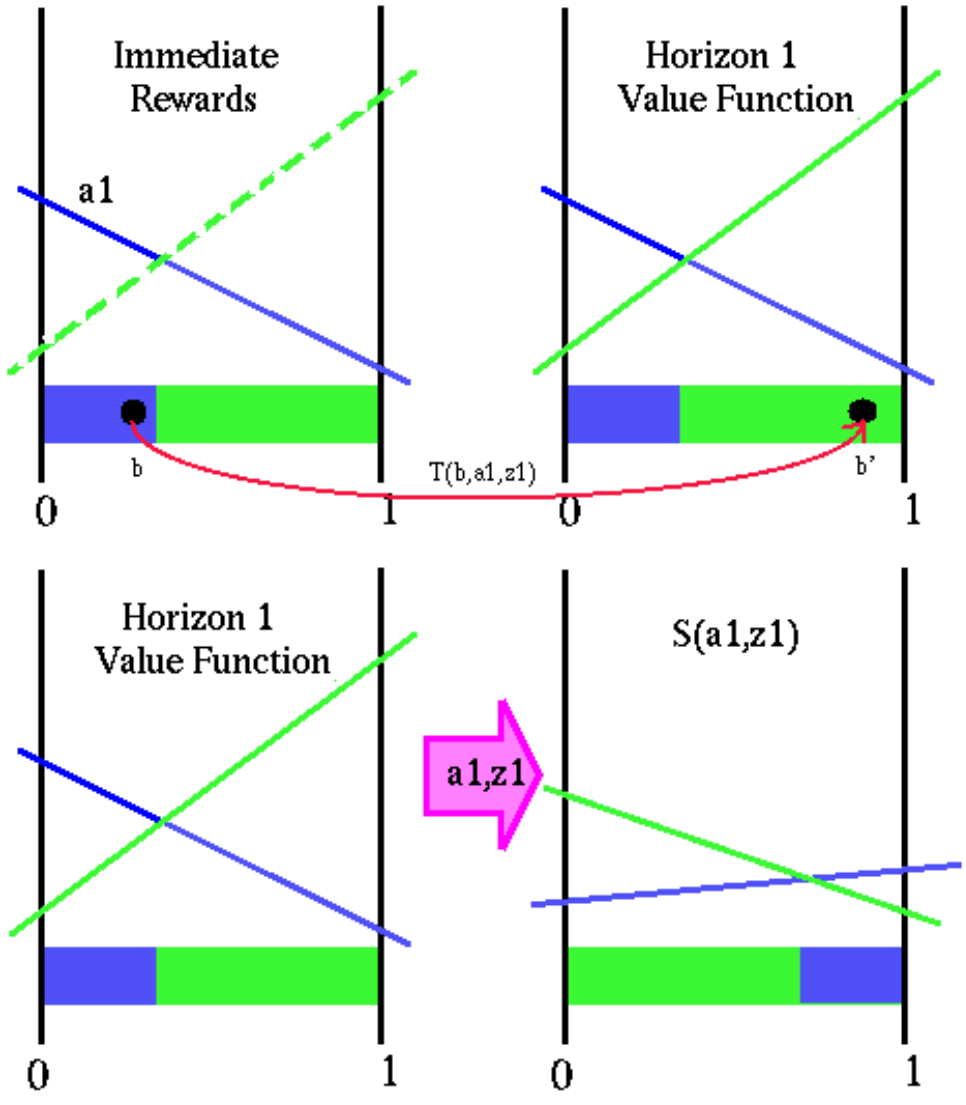
General overview of Solving POMDP

- We will go for construction horizon 2 value function
- 3 step breakdown:
- Compute value of a belief state, given action and observation fixed
- Now, the value of a belief state, for horizon 2 is, the value of immediate action plus the value of the next action



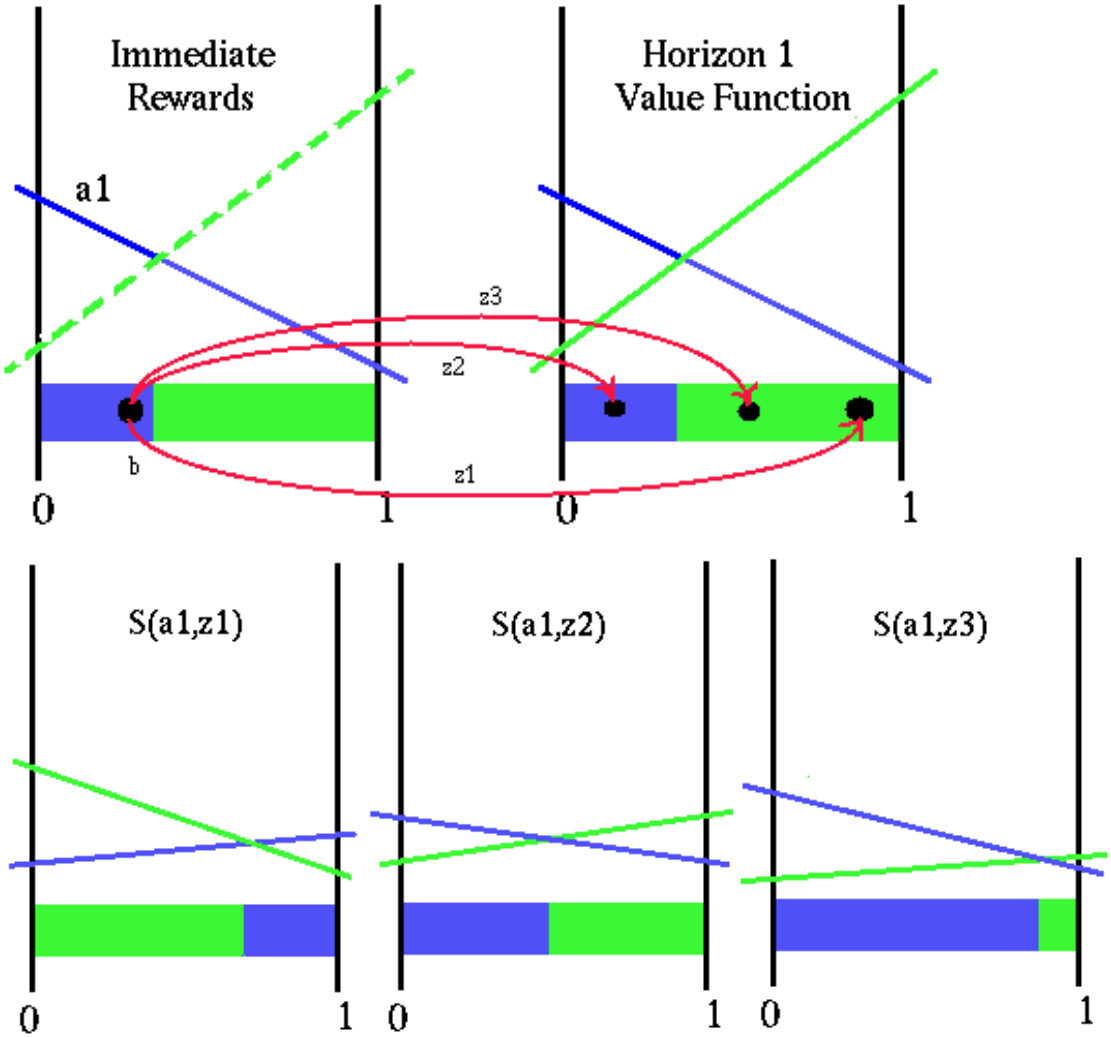
General overview of Solving POMDP

- for value of another belief sate we repeat the same process
- now, we want to find value of all belief points, given fixed action and observation
- Our horizon 1 value function, is a function of transformed belief sate $b^{'}$, and transformed belief state is a function of initial belief sate b, since the action and observation is fixed
- It turns out we can construct a function over the entire belief space from horizon 1



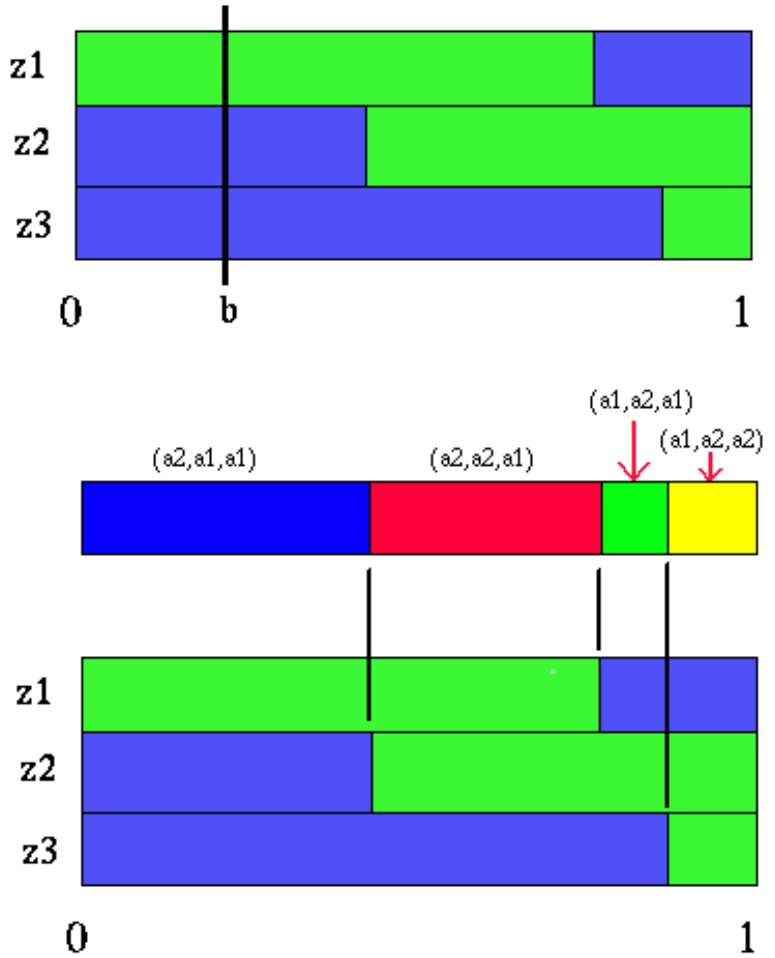
General overview of Solving POMDP

- Computing value of belief state, given only the action.
- Since the observations are probabilistic,we are not guaranteed to see z1
- each observation can lead to a separate resulting belief state
- now the transformed value function, factors in the probabilities of the observations



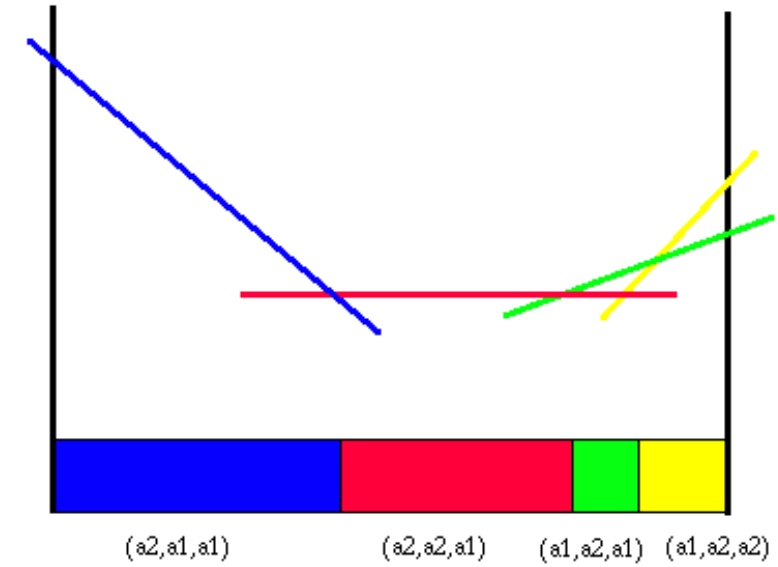
General overview of Solving POMDP

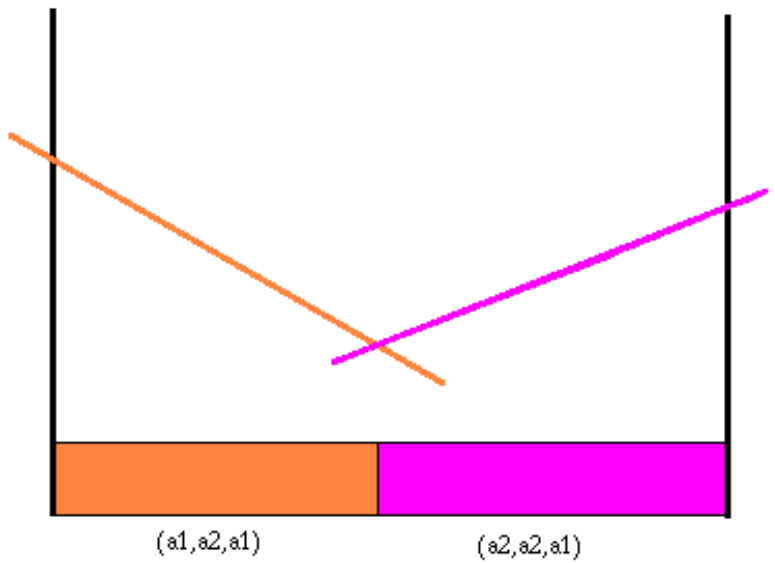
- So, the horizon 2 value of a belief state, given a particular action a1, depends, not only on doing a1, but also on what action we do next.
- Now ,this depends on what observation we get.
- this results in four useful future strategies



General overview of Solving POMDP

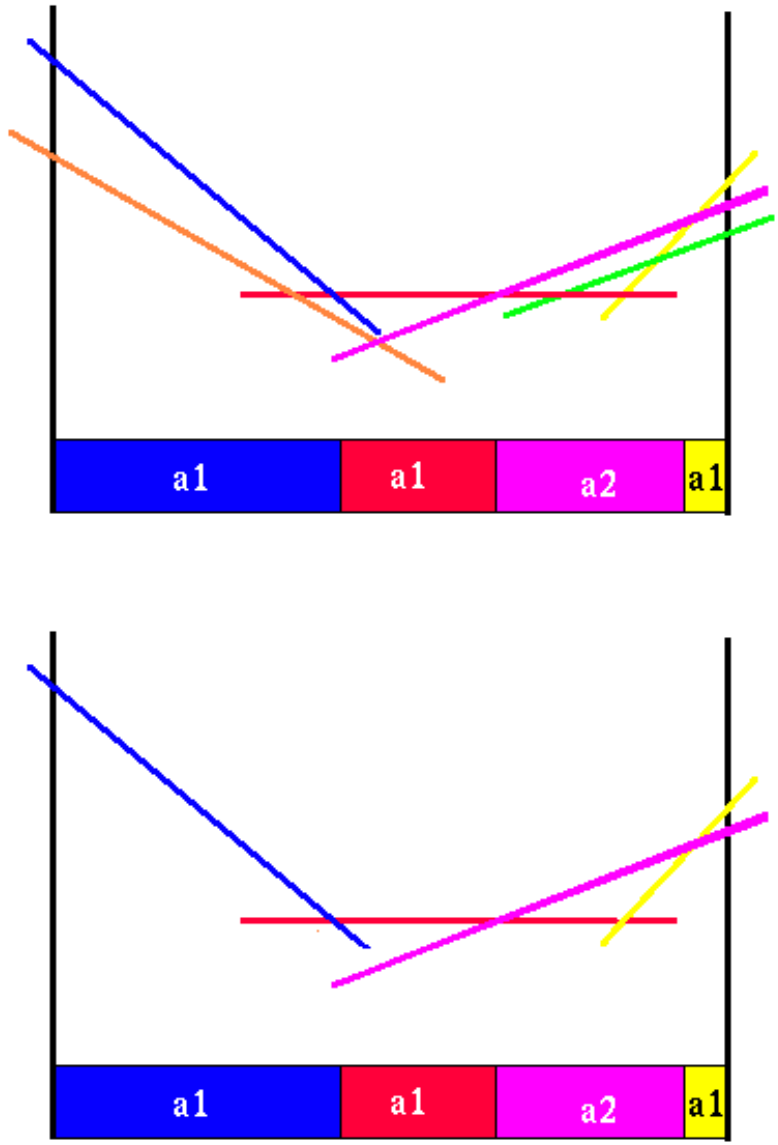
- The updated value functions and partitions, for action $a1$(first figure)
- We have considered only taking action a1, we should also consider for action a2
- The second figure shows the partition and value functions for $a2$
- now, we have to combine both $a1$ and $a2$ value functions, and see which action gives the highest value.



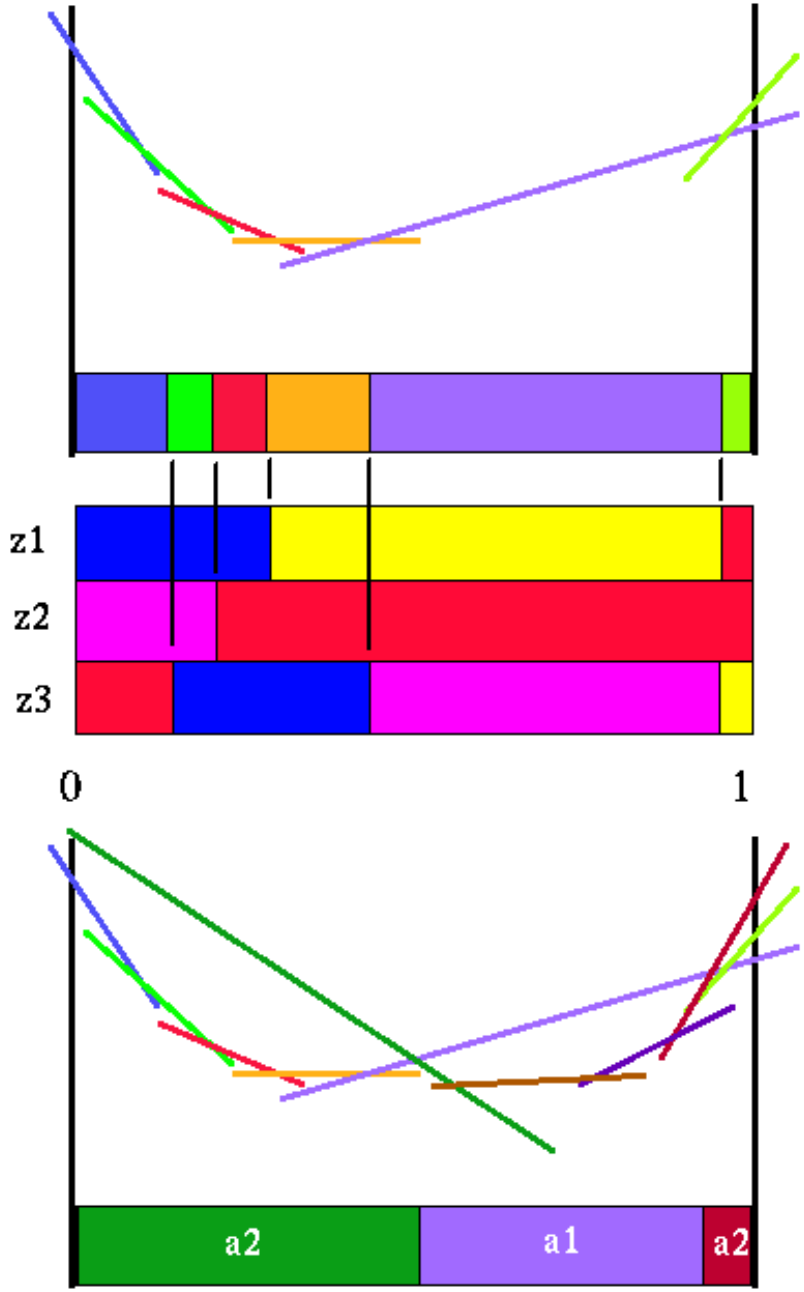


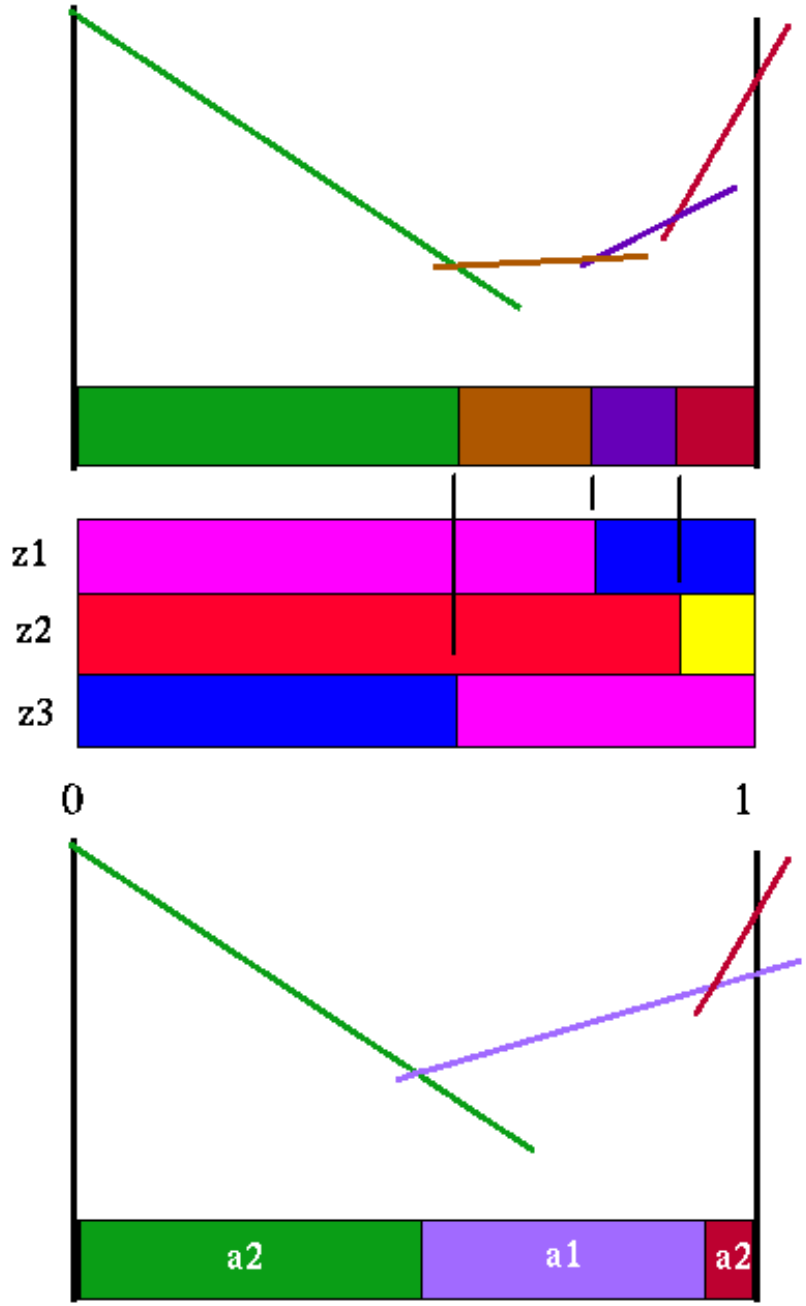
General overview of Solving POMDP

- first figure shows the combined value functions form $a1$ and $a2$
- On selecting the value functions, which maximizes the value at each belief point we get the value function for horizon 2
- Shown in figure 2
-



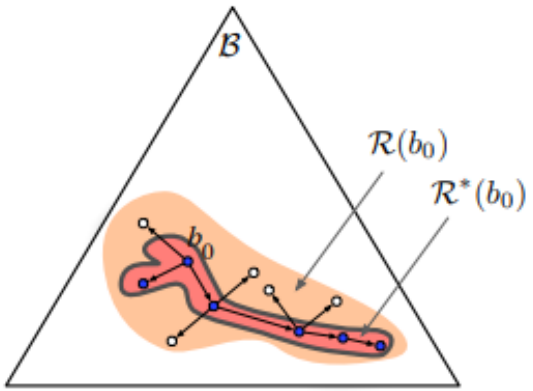
For horizon 3, we do the same

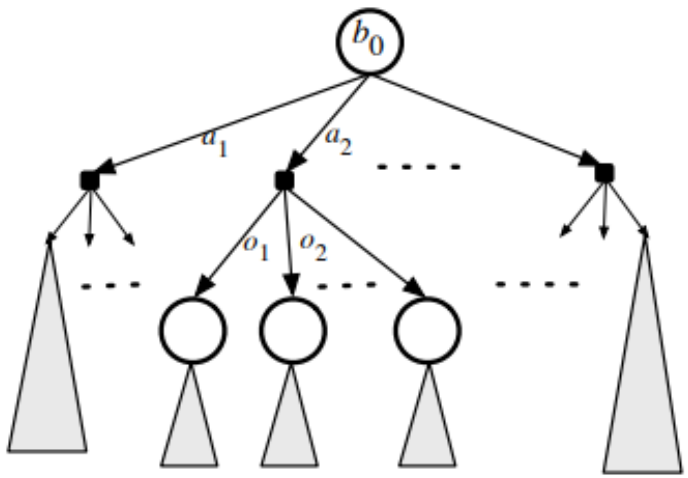




SARSOP algorithm

- Point based POMDP algorithm
- We sample points from belief space $\mathcal{B}$
- and maintain the a set Γ of α vectors, A piece wise linear approximation $\underline{V}$ to V^*
- Each α vector, represents potentially a optimal hyperplane, at the sampled belief point
- To improve $\underline{V}$ back up α vectors at the sampled belief point





SARSOP algorithm

- The point based algorithms, iterate over three main functions:
  1. SAMPLE
  2. BACKUP
  3. PRUNE
- Until further update produces, very little change in the value approximations
- This is the basic structure of many point based POMDP algorithms
- Only differ in the details of the three main functions

Algorithm 1 SARSOP.

1: Initialize the set Γ of α-vectors, representing the lower bound $\underline{V}$ on the optimal value function V^*. Initialize the upper bound $\overline{V}$ on V^*.
2: Insert the initial belief point b_0 as the root of the tree $T_{\mathcal{R}}$.
3: **repeat**
4: SAMPLE($T_{\mathcal{R}}$, Γ).
5: Choose a subset of nodes from $T_{\mathcal{R}}$. For each chosen node b, BACKUP($T_{\mathcal{R}}, \Gamma, b$).
6: PRUNE($T_{\mathcal{R}}$, Γ).
7: **until** termination conditions are satisfied.
8: **return** Γ.

SARSOP algorithm

- First, we have to build a reachable belief space($\mathcal{R}$), a subset of belief space $\mathcal{B}$
- sampled belief points form a tree $\mathcal{X}$
- we select an initial belief point b_0; generally a high entropy point
- To sample a new point $b^{'}$:
  - pick b from $\mathcal{X}$, select $a \in A$ and $z \in Z$
  - by Bayesian filtering compute the posterior belief $b^{'}$
  - insert $b^{'}$ as child of b
- Every point sampled this way is reachable from b_0

SARSOP algorithm

- After sampling, we perform backup at selected nodes in $\mathcal{X}$
- Invocation of SAMPLE and BACKUP functions, generates new sample points and α vectors.
- however, not all of them are useful for constructing optimal policy, $\therefore$ prune away some belief points and α vectors, to increase the computational efficiency
- Avoid sampling far away from optimal reachable belief space.(belief point pruning)
- Cost of single backup operation is directly propotional to no of α vectors in Γ. (α vector pruning)

SARSOP:

SAMPLE(X, Γ)

1: Pick $b \in X$, $a \in A$, and $z \in Z$.
2: $b' \leftarrow \tau(b, a, o)$.
3: Insert b' into X as a child of b.

BACKUP(X, Γ, b)

1: For all $a \in A$, $z \in Z$, $\alpha_{a,z} \leftarrow \text{argmax}_{\alpha \in \Gamma}(\alpha \cdot \tau(b, a, z))$.
2: For all $a \in A$, $s \in S$, $\alpha_a(s) \leftarrow R(s, a) + \gamma \sum_{z,s'} T(s, a, s') O(s', a, z) \alpha_{a,z}(s')$.
3: $\alpha' \leftarrow \text{argmax}_{a \in A}(\alpha_a \cdot b)$
4: Insert α' into Γ.

Belief point pruning

- To use the sampled points in $\mathcal{X}$ as an approximation of $\mathcal{R}^*$ we must prune those points far away from $\mathcal{R}^*$
- for this we maintain not only lower bound $\underline{V}$, but also and upper bound $\overline{V}$ on V^*
- Consider a node b in $\mathcal{X}$,

$$\underline{Q}(b,a) = \sum_s R(s,a)\, b(s) + \gamma \sum_z p(z|b,a)\, \underline{V}(T(b,a,z))$$

  is the lower bound on the value of taking action a at b belief point.
- similarly the upper bound, $\overline{Q}(b,a)$ with $\overline{V}$

Belief point pruning

- If for some action $a^{'}$ and a,

$$\overline{Q}(b,a) < \underline{Q}(b,a^{'})$$

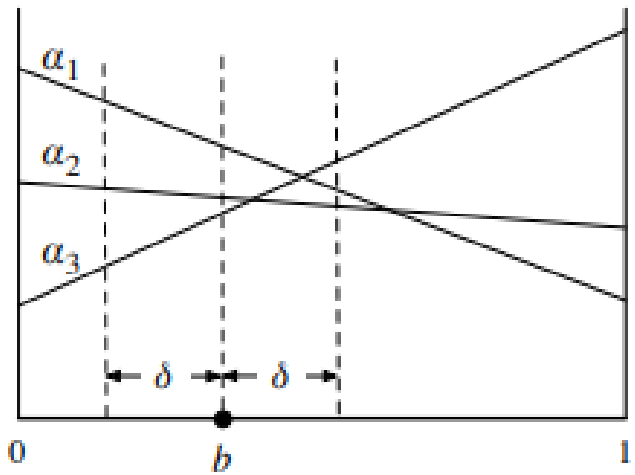
  at b.
- The optimal policy will never take action a, under b, and traverse the subtree underneath it.
- thus, prune the subtree, along with the associated sample points

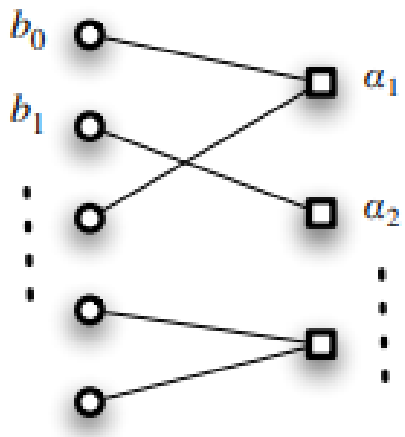
α vectors pruning

- Let P denote the set of sampled belief points in $\mathcal{X}$, SARSOP prunes away an α vector if it is dominated by others over $\mathcal{R}^*$, which is approximated by the current P.
- if $\alpha_1 . b \leq \alpha_2 . b$, α_2 dominates α_1
- but this is not robust,Since SARSOP computes an approximately optimal policy over P only, the computed policy may choose an action that causes it to slightly veer off $\mathcal{R}^*$
- and get into a region in which the value function approximation is poor
- we impose more stringent dominance criteria.

α vectors pruning

- The δ-dominance criteria
- i.e dominance over a δ neighbourhood:
- α_1 dominates α_2 at a belief point b if $\alpha_1 \,.\, b^{'} \geq \alpha_2 \,.\, b^{'}$ at every point $b^{'}$, whose distance to b is less than δ, for some fixed constant δ.

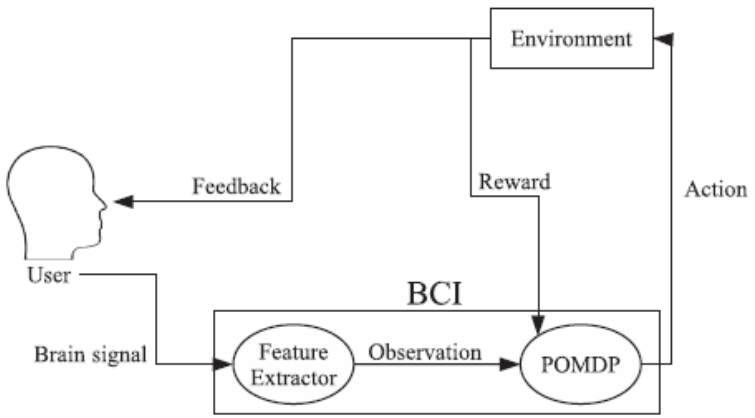




Outline

BCI system Overview

- POMDP - deals with uncertainty in users brain state and to decide the amount of information to make confident control decision.
- RL - To co-adapt with the user.
- User decides
  - The Goal (e.g. Direction to steer a wheel chair)
  - The control mapping (e.g. Which SSVEP channel (frequency of EEG activity) maps to which action)



Properties to be satisfied by a BCI control problem

- State, action and observation spaces must be discrete
- Quantifiable connection between hidden state and observations
- Problem must allow control in discrete time steps
- User and BCI must have joint awareness of the feedback from the environment/system

SSVEP based BCI

- This BCI is EEG based BCI
- more specifically SSVEPs Steady state visually evoked potentials are used
- SSVEP: Oscillatory potentials observed in the occipital areas of brain, when the subject is focusing on a flickering stimuli of certain frequency
- Response frequency is equal to stimuli frequency or harmonics of stimuli frequency.

hardware description

- Mounted LEDs around the lower perimeter for the LCD screen
- Five frequencies 12, 15, 17, 20 and 22 Hz
- EEG recording system (256 Hz sampling frequnecy)

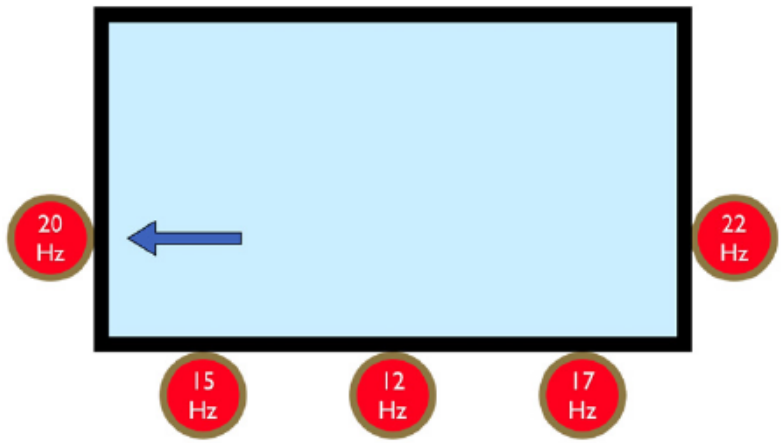


Figure: Stimulus device

Features and observation model

- **EEG** $\xrightarrow[1.0\,s\ window]{FFT}$ [12hz: ,15hz: ,17hz: ,20hz: ,22hz:]
- Normalization with subject's baseline EEG activity
- The extracted EEG features are used to build the observation model of the POMDP
- Since the extracted features lie in continuous space, we need to discretize it (since the observation space should be discrete for this framework)

Discretization of observation space

- Partition the feature space and label each partition
- Easy way: uniform discretization, i.e divide each dimension of the feature space uniformly (Grid like partition)
- Drawback: fixed grid size - too big resolution is lost, too small results in large number of cells, scales poorly
- we need an informed method of non-uniform discretization
- use A priori information about observation function

Discretization of observation space

- Use A priori information about the observation function
  1. Gaussian distribution approximation to observation function (One gaussian for each state)
  2. Use this gaussian model to make non uniform partitioning of feature space
  3. Numerically integrate each of the gaussians within each partition to derive the discretized observation function $p(O|S)$
- Estimate a Gaussian distribution for the features, given each brain state
- i.e. we fit a gaussian in our feature space, for each state based on users training data

Discretization of observation space

- Estimate a Gaussian distribution for the features, given each brain state, which gives, continuous approximation to $p(O|S)$
- All points in the space, that yield equivalent "evidence" forms a partition
- Since, no observations once we reach final state, and other transitions leading to non-terminal state are determinsitic, the belief update equation reduces to:

$$b_{t+1}(s_{t+1}) = \eta\, p(o|s_{t+1} b_t(s_{t+1})$$

- $$evidence \;=\; \frac{b_{t+1}(s_{t+1})}{b_t(s_{t+1})} = \eta\, p(o|s_{t+1})$$

- $$evidence = \frac{b_{t+1}(s_{t+1})}{b_t(s_{t+1})} = \eta\, p(o|s_{t+1})$$
- Suppose, we have two states and our prior belief is:

$$b_t = \begin{pmatrix} p(s_1) = 0.3 \\ p(s_2) = 0.7 \end{pmatrix}$$

suppose that two possible observations give

$$\begin{pmatrix} p(O|S = s_1) = 0.0025 \\ p(O|S = s_2) = 0.0075 \end{pmatrix}$$

and

$$\begin{pmatrix} p(O|S = s_1) = 0.005 \\ p(O|S = s_2) = 0.0015 \end{pmatrix}$$

this results in a posterior belief

$$b_{t+1} = \begin{pmatrix} p(s_1) = 0.125 \\ p(s_2) = 0.875 \end{pmatrix}$$

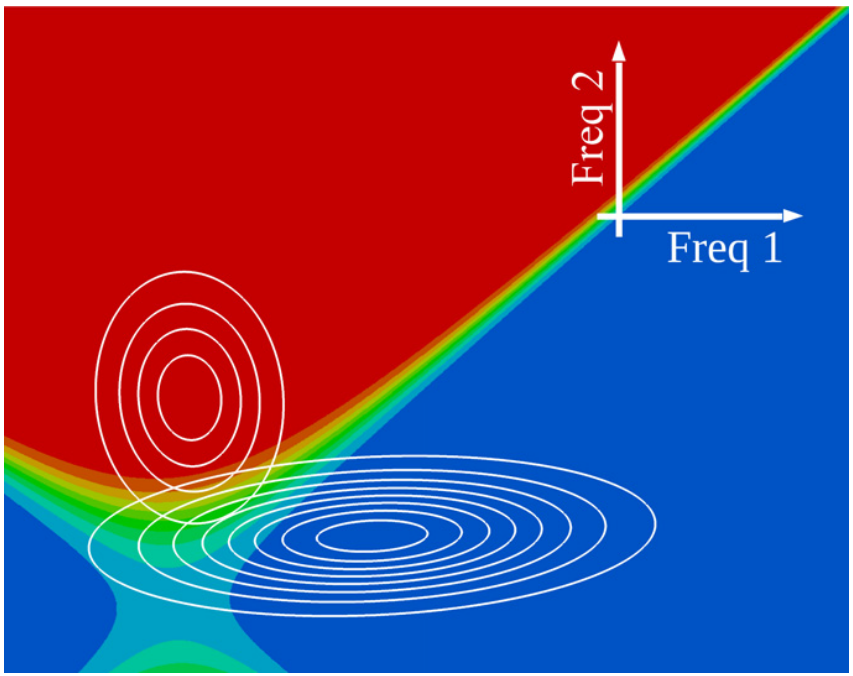
- Two observations yield equivalent evidence.

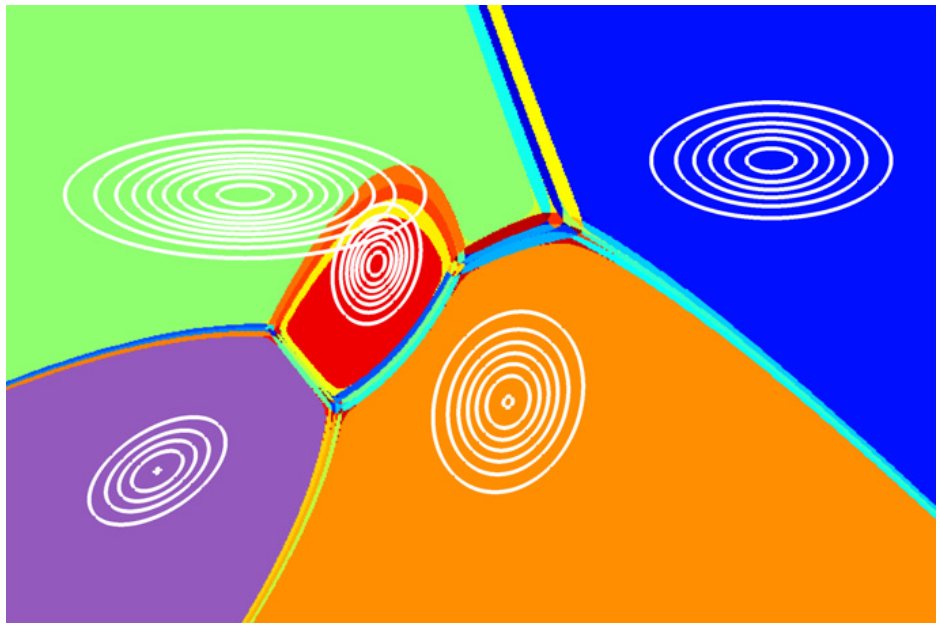
Discretization of observation space

- Calculate partition based on evidence
- Create a high reoslution grid in our feature space within some finite bounds
- finite bounds - rectangle sides a large mahalalobis distance from the means of gaussian for all classes.
- cal vectors $p(o|S)$, at each grid point o, which we normalize to sum 1.
- call this vector an evidence vector
- each element of the evidence vector is given an integer label in $[1, n]$
- $label_i = round(\,(n-1) * (evidence_i) + 1\,)$
  The output of this is a vector of labels, for each point in the grid.
- Finally, we assign a discrete observation label, to every unique value these vectors take

Discretization of observation space

- This method generates more discretized areas as the dimensionality of the observation model increase
- K-means among centroids of discretized areas to prevent the combinatorial explosion





POMDP model of the BCI system

- The POMDP has one state for each available SSVEP stimulus, and one terminal state for each possible control action.

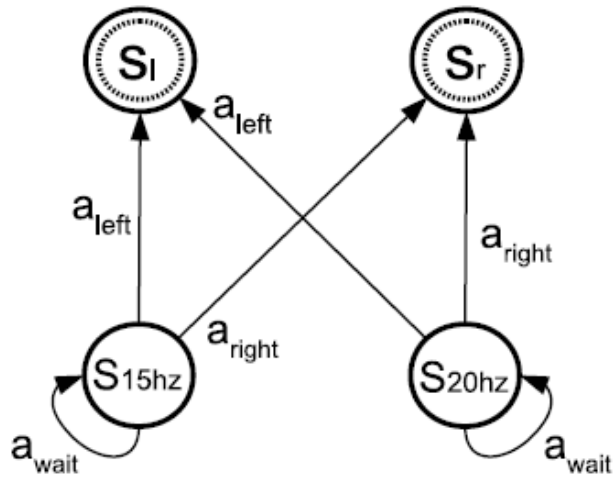


Figure: State transition diagram

- For the this simplest experiment,the state space is $S = \{s_{15Hz}, s_{20Hz}, s_{left}, s_{right}\}$
- action space is $A = \{a_{wait}, a_{left}, a_{right}\}$

POMDP model of the BCI system

- The observation space is obtained by discretizing the EEG feature space. $O = \{o_1, o_2, ..., o_n\}$
- The transition model, in present implementation is deterministic
- The observation model depends highly on specific user, as does the discretization of users feature space.
- The BCI automatically adapts the reward function during co-adaptive experiments, as it accumulates experience through reinforcement learning.
- In few other experiments, the reward is fixed, for evaluation.

	a_{wait}	a_{left}	a_{right}
$S_{15\,\text{Hz}}$	r_{wait}	r_{failure}	r_{success}
$S_{20\,\text{Hz}}$	r_{wait}	r_{success}	r_{failure}
S_{left}	0	0	0
S_{right}	0	0	0

Figure: Reward matrix

Reinforcement based learning and co-adaptation

- Basic POMDP model $\xrightarrow{+\,feedback\ mechanism}$ Adaptive control model.
- To infer users intended control mapping (which stimulus to which control o/p)
- feedback tells us whether, co-operation between the user and BCI was successful at accomplishing the task.
- The BCI will identify the users control mapping over time.
- The BCI receives adverse feedback from environment, If the user and BCI do not agree on a control mapping.

Assumptions to implement the system

1. All available brain states map to some control output.
2. All possible control output map to at least one brain state.
3. The user/system designer can express the relative cost of failure and benefit of success.($r_{failure}$, $r_{success}$)

- Using these 3 assumptions, the authors demonstrate three co-adaptive systems
  - Two frequencies and two outputs case
  - five frequencies and two outputs case
  - three frequencies and three outputs case

Two frequencies and two outputs

- Given the three assumptions , this case exploration is not necessary
- There are only two admissible control mappings, suppose that BCI identifies current state as s_{15Hz}, choose action a_{left}, receives feedback $r_{success}$, since the control was successful in this case, we can conclude that exploiting the control mapping from s_{20Hz} to a_{right} is also a success
- we also know the cost of failure $r_{failure}$, $\therefore$ regardless of the decision we make and feedback we get, we will effectively receive the same information and will be able to update all four elements of reward.
- Thus, there is no extra information that we get by exploration
- Hence, exploitation is strictly better choice.

Two frequencies and two outputs

- The reinforcement algorithm for reward update is

$$\hat{R}(s,a) \leftarrow \alpha\,\beta(r_{success} - \hat{R}(s,a)) + \hat{R}(s,a)$$

- (s,a), is a member of successful control mapping
  α, is learning rate
  $\beta = max(b_t)$, is confidence in current state

Five frequencies and two outputs

- In this case it is not possible to update every element of the reward function in each trial.
- however,we don't make explicit use of exploration, user is implicitly engaged in a form of exploration.
- if user chooses 15Hz channel with an intent of a_{left}
  (15Hz, a_{left}) $\leftarrow$ positive reinforcement
  (15Hz, a_{right}) $\leftarrow$ negative reinforcement
- User explores various input channels, each will independently learn a control mapping.

Three frequencies and three outputs

- Since, there are more than two control outputs, we need an explicit exploration.
- For exploration we use "softmax action selection" strategy.

$$for\ (a_i \in A) : A_w(a_i) = \sum_{s_i \in S} b(s) R(s, a_i)$$

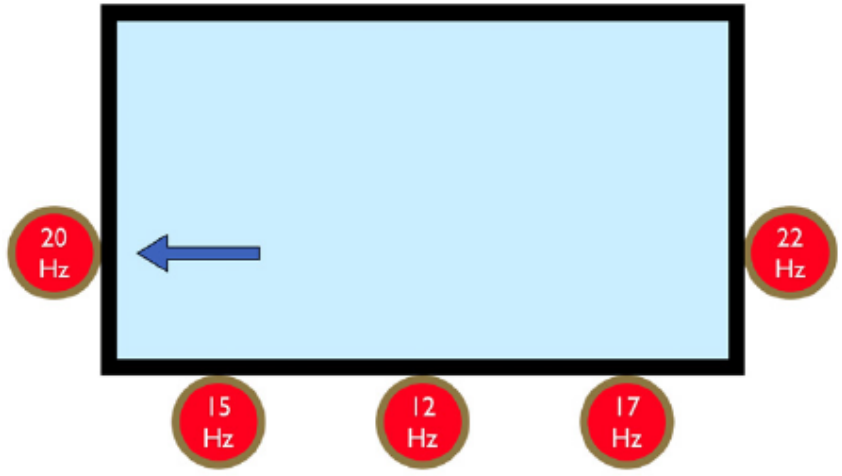
This is used to derive distribution over action

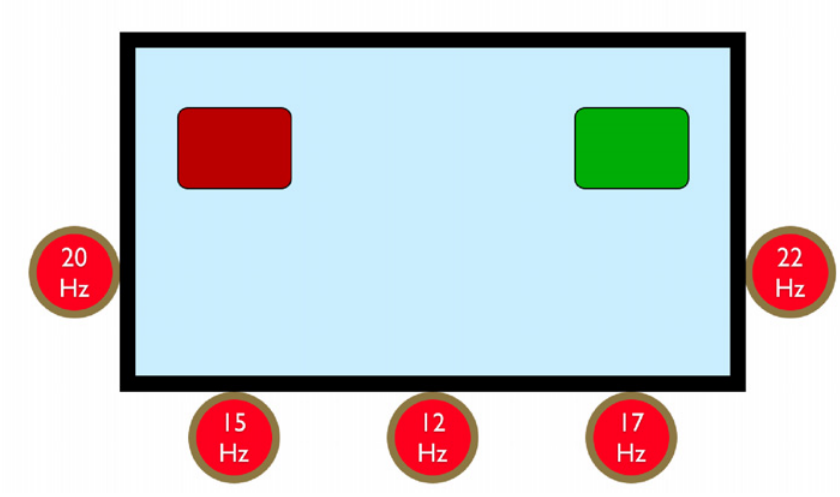
$$P(a_i) = \frac{\exp \frac{A_w(a_i)}{\tau}}{\sum_{a_j \in A} \exp \frac{A_w(a_j)}{\tau}}$$

Outline

Experimental setup

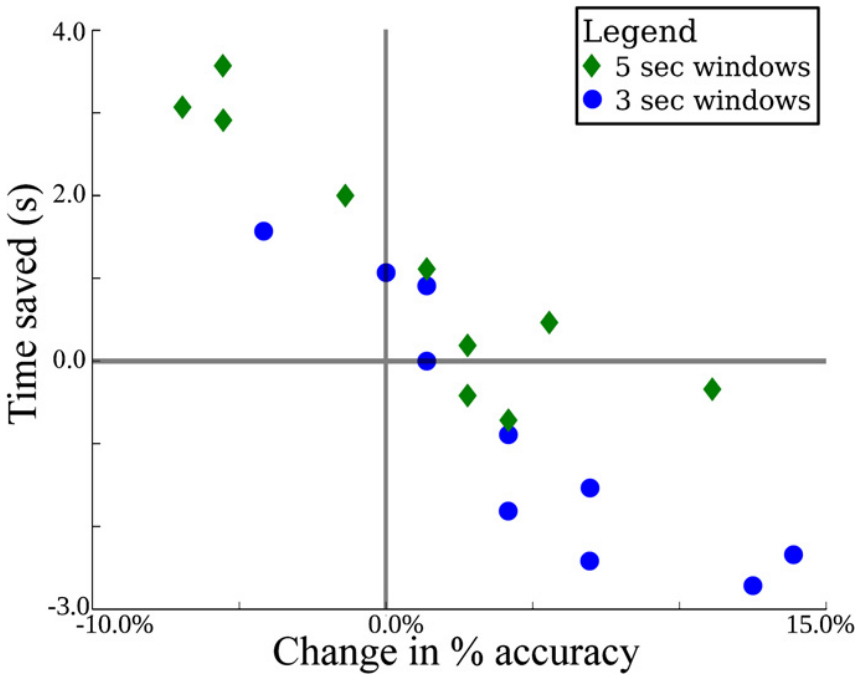
- Session 1: open loop experiment, to collect training data
  six trails per state, arrow on LCD as a cue to focus on one SSVEP stimulus
  This training data was used to train the POMDP observation model.
- Session 2: Baseline reading of EEG for each user
- Next 6 Sessions: six trials per state/stimulus
  used as test data set for offline experiments
- Subsequent session: closed loop co-adaptive experiments
  three sets of co-adaptive experiments.

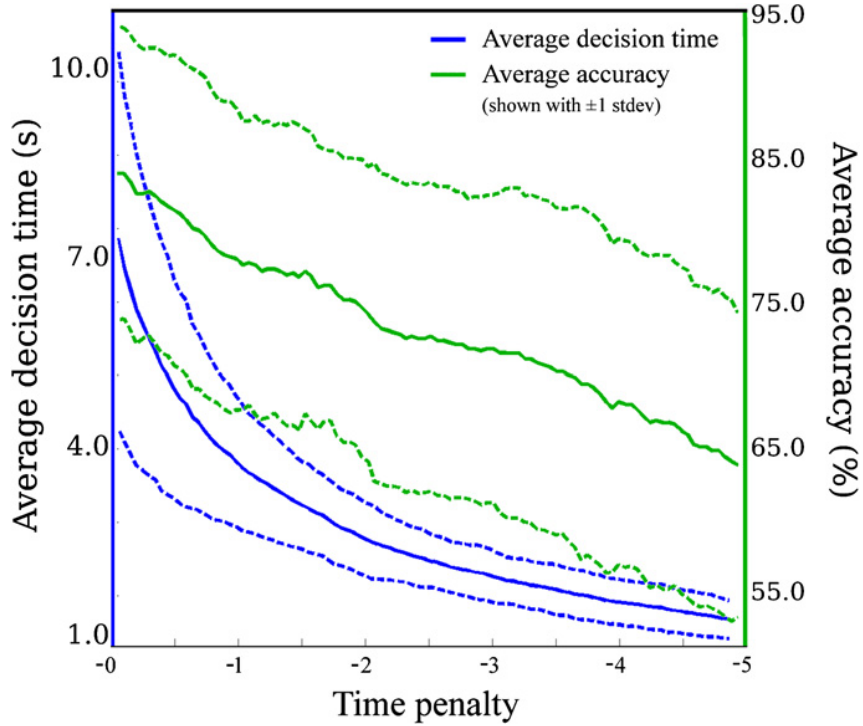

20
Hz
22
Hz
15
Hz
12
Hz
17
Hz


20
Hz
22
Hz
15
Hz
12
Hz
17
Hz

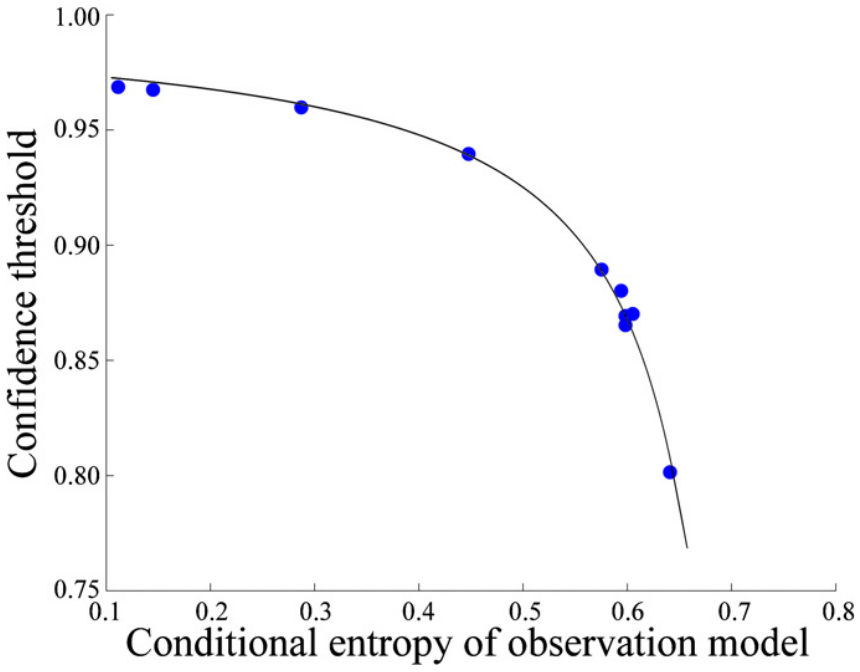
Outline

Speed accuracy trade-off



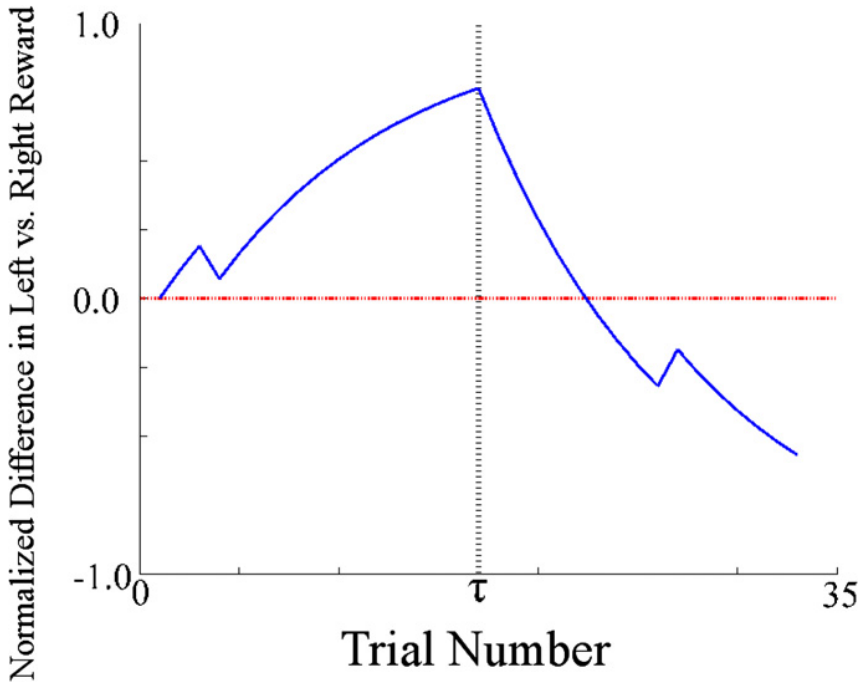


Adaptation to user's SNR



- optimal POMDP policy not only depends on observation model for specific user
- If the observations are noisy, collecting additional information is less useful
- BCI will lower the confidence threshold, avoid some of the penalty for waiting.
- $H(S|O)$ is used to quantify this effect.

Co-adaptation in the POMDP BCI



References


Matthew J Bryan, Stefan A Martin,Willy Cheung and Rajesh P N Rao.
Probabilistic co-adaptive brain computer interfacing
Journal of Neural engineering 2013.


David Hsu, Wee Sun Lee, and Nan Rong.
Accelerating Point-Based POMDP Algorithms through Successive Approximations of the Optimal Reachable Space
Technical report, NUS, School of computing, Singapore 2006.


POMDPs for Dummies article, by computer science department of Brown university.
http://cs.brown.edu/research/ai/pomdp/tutorial/index.html