Learning Mixtures of Markov Chains and MDPs

Chinmaya Kausik, Kevin Tan and Ambuj Tewari

University of Michigan

(a) Paper (b) Slides

Motivation

- **Markovian dynamics:** Real-life time-series data can often be modeled reasonably with a Markovian assumption.

Motivation

- **Markovian dynamics:** Real-life time-series data can often be modeled reasonably with a Markovian assumption.
- **Multiple models, unlabelled trajectories:** It's more reasonable to assume that there are multiple underlying models, and model labels are often not recorded.

Motivation

- **Markovian dynamics:** Real-life time-series data can often be modeled reasonably with a Markovian assumption.
- **Multiple models, unlabelled trajectories:** It's more reasonable to assume that there are multiple underlying models, and model labels are often not recorded.
- **Examples:**

Trajectories of what?	*Unrecorded model labels*
Medical data	Pre-existing health condition or socio-economic status
Driving data	Type of environment
Education data	Type of learner

Prior Work

Year	*Authors*	*Mixtures of what?*
2004	Vempala and Wang	Gaussians
2020	Kong et al	Linear models
2022	Chen and Poor	Linear dynamical systems
2023, Us	Kausik, Tan and Tewari	Markov chains and MDPs

First setting handling control input!

Problem Setup

- We have a state and action set $\mathcal{S}, \mathcal{A}$ and K different hidden labels. At the start of each trajectory, we draw:
  - Hidden label $k \sim \text{Categorical}(f_1, ..., f_K)$
  - Starting state according to the distribution p_k on $\mathcal{S}$
  - Generate the rest of the trajectory under the policy $\pi_k(a \mid s)$ interacting with the transition structure $\mathbb{P}^{(k)}(\cdot \mid s, a)$.

Problem Setup

- We have a state and action set $\mathcal{S}, \mathcal{A}$ and K different hidden labels. At the start of each trajectory, we draw:
  - Hidden label $k \sim \text{Categorical}(f_1, ..., f_K)$
  - Starting state according to the distribution p_k on $\mathcal{S}$
  - Generate the rest of the trajectory under the policy $\pi_k(a \mid s)$ interacting with the transition structure $\mathbb{P}^{(k)}(\cdot \mid s, a)$.
- In many applications, there are not too many "kinds of behaviors." That is, $K << S, A$.

Problem Setup

- We have a state and action set $\mathcal{S}, \mathcal{A}$ and K different hidden labels. At the start of each trajectory, we draw:
  - Hidden label $k \sim \text{Categorical}(f_1, ..., f_K)$
  - Starting state according to the distribution p_k on $\mathcal{S}$
  - Generate the rest of the trajectory under the policy $\pi_k(a \mid s)$ interacting with the transition structure $\mathbb{P}^{(k)}(\cdot \mid s, a)$.
- In many applications, there are not too many “kinds of behaviors." That is, $K << S, A$.
- What about Markov chains? Just set $\mathcal{A} = \{*\}$!

Problem Formulation

To illustrate, let $K = 5$. Imagine that $\mathcal{S}$ and $\mathcal{A}$ are huge.

Traj 1 $[k = 3 \implies \mathbb{P}^{(3)}, \pi_3, p_3]$ $s_1, a_1, s_3, a_3, s_5, a_5, s_1, a_1, s_2, a_2, \ldots$

Traj 2 $[k = 1 \implies \mathbb{P}^{(1)}, \pi_1, p_1]$ $s_2, a_2, s_4, a_4, s_2, a_2, s_1, a_1, s_5, a_5, \ldots$

Traj 3 $[k = 5 \implies \mathbb{P}^{(5)}, \pi_5, p_5]$ $s_4, a_4, s_2, a_2, s_5, a_5, s_3, a_3, s_1, a_1, \ldots$

$\ldots$

Problem Formulation

To illustrate, let $K = 5$. Imagine that $\mathcal{S}$ and $\mathcal{A}$ are huge.

Traj 1 $[k = 3 \implies \mathbb{P}^{(3)}, \pi_3, p_3]$ $s_1, a_1, s_3, a_3, s_5, a_5, s_1, a_1, s_2, a_2, \ldots$

Traj 2 $[k = 1 \implies \mathbb{P}^{(1)}, \pi_1, p_1]$ $s_2, a_2, s_4, a_4, s_2, a_2, s_1, a_1, s_5, a_5, \ldots$

Traj 3 $[k = 5 \implies \mathbb{P}^{(5)}, \pi_5, p_5]$ $s_4, a_4, s_2, a_2, s_5, a_5, s_3, a_3, s_1, a_1, \ldots$

$\ldots$

Models and labels are unknown: We do not know the parameters $\mathbb{P}^{(k)}, f_k, p_k, \pi_k(\cdot \mid s)$ of any model, or the model label k_n for any trajectory n.

Problem Formulation

To illustrate, let $K = 5$. Imagine that $\mathcal{S}$ and $\mathcal{A}$ are huge.

Traj 1 $[k = 3 \implies \mathbb{P}^{(3)}, \pi_3, p_3]$ $s_1, a_1, s_3, a_3, s_5, a_5, s_1, a_1, s_2, a_2, \ldots$

Traj 2 $[k = 1 \implies \mathbb{P}^{(1)}, \pi_1, p_1]$ $s_2, a_2, s_4, a_4, s_2, a_2, s_1, a_1, s_5, a_5, \ldots$

Traj 3 $[k = 5 \implies \mathbb{P}^{(5)}, \pi_5, p_5]$ $s_4, a_4, s_2, a_2, s_5, a_5, s_3, a_3, s_1, a_1, \ldots$

$\ldots$

Models and labels are unknown: We do not know the parameters $\mathbb{P}^{(k)}, f_k, p_k, \pi_k(\cdot \mid s)$ of any model, or the model label k_n for any trajectory n.

Goal: Cluster trajectories based on hidden model labels. This is essentially **unsupervised time-series clustering**.

Main Challenges

Lack of methods with provable guarantees.

- **Unsupervised:** Models and labels both unknown. Chicken and egg problem! Expectation-Maximization (EM) lacks guarantees.

Main Challenges

Lack of methods with provable guarantees.

- **Unsupervised:** Models and labels both unknown. Chicken and egg problem! Expectation-Maximization (EM) lacks guarantees.
- **Short trajectories, naive model estimates are crude:** Cluster using naive estimates $\hat{\mathbb{P}}_n(\cdot \mid s, a)$ from trajectories?

Main Challenges

Lack of methods with provable guarantees.

- **Unsupervised:** Models and labels both unknown. Chicken and egg problem! Expectation-Maximization (EM) lacks guarantees.
- **Short trajectories, naive model estimates are crude:** Cluster using naive estimates $\hat{\mathbb{P}}_n(\cdot \mid s, a)$ from trajectories? Too crude if trajectory length is much shorter than S.

Main Challenges

Lack of methods with provable guarantees.

- **Unsupervised:** Models and labels both unknown. Chicken and egg problem! Expectation-Maximization (EM) lacks guarantees.
- **Short trajectories, naive model estimates are crude:** Cluster using naive estimates $\hat{\mathbb{P}}_n(\cdot \mid s, a)$ from trajectories? Too crude if trajectory length is much shorter than S.
- **Time series without additive i.i.d noise:** Time series with martingale noise presents complications beyond additive i.i.d. noise.

Mixing Time Assumption

We essentially define the mixing time of the mixture here. This is more of a **notational definition**, outside of the implicit hope that $t_{mix} << S, A$.

Mixing Time Assumption

We essentially define the mixing time of the mixture here. This is more of a **notational definition**, outside of the implicit hope that $t_{mix} << S, A$.

Assumption (Mixing Time)

The K Markov chains on $\mathcal{S} \times \mathcal{A}$ induced by the behaviour policies π_k, each achieve mixing to a stationary distribution $d_k(s, a)$ with mixing time $t_{mix,k}$. Define the overall mixing time of the mixture of MDPs to be

$$t_{mix} := \max_k t_{mix,k}$$

Model Separation Assumption

For each pair of models, there should be at least one "visible" (s, a) pair that witnesses a difference between them. **If you can't "see a difference," you can't hope to cluster!**

Model Separation Assumption

For each pair of models, there should be at least one "visible" (s, a) pair that witnesses a difference between them. **If you can't "see a difference," you can't hope to cluster!**

Assumption (Model Separation)

There exist α, Δ so that for each pair k_1, k_2 of hidden labels, there exists a state action pair (s, a) (possibly depending on k_1, k_2) so that $d_{k_1}(s, a), d_{k_2}(s, a) \geq \alpha$ and $\|\mathbb{P}^{(k_1)}(\cdot \mid s, a) - \mathbb{P}^{(k_2)}(\cdot \mid s, a)\|_2 \geq \Delta$.

Main Result

Theorem (Informal)

With high probability, we can recover all labels exactly with K^2S trajectories of length $K^{3/2}t_{mix}$, up to logarithmic terms and instance-dependent constants.

Main Result

Theorem (Simplified)

There exist constants H_0, N_0 depending polynomially on $\frac{1}{\alpha}, \frac{1}{\Delta}, \frac{1}{\min_k f_k}, \log(1/\delta)$, we can recover all labels exactly with $n \geq K^2 S N_0$ trajectories of length $K^{3/2} H_0 t_{mix} \log n$ with probability at least $1 - \delta$.

Algorithm Outline

The algorithm is modular and broadly has 3 steps.

1. **Subspace Estimation**
2. **Clustering**
3. **Model Estimation and Classification**

Algorithm Outline

Each trajectory n corresponds to a very crude model estimate $\hat{\mathbb{P}}_n(\cdot \mid s, a)$. See the paper for many important subtleties.

1. **Subspace estimation:** Aggregate across estimates $\hat{\mathbb{P}}_n$ to obtain $(\mathbf{V}_{s,a}^T)_{K \times S}$, an estimate for the projector to $\text{span}_k\, \mathbb{P}^{(k)}(\cdot \mid s, a)$.
2. **Clustering:** Similarity-based clustering.

$$\text{dist}_1(m, n) = \max_{(s,a) \in \text{Freq}_\beta} \|\mathbf{V}_{s,a}^T \hat{\mathbb{P}}_m(\cdot \mid s, a) - \mathbf{V}_{s,a}^T \hat{\mathbb{P}}_n(\cdot \mid s, a)\|_2^2$$

Intuition

Estimates $\hat{\mathbb{P}}_n(\cdot \mid s, a)$ from trajectories are too crude when S is large.

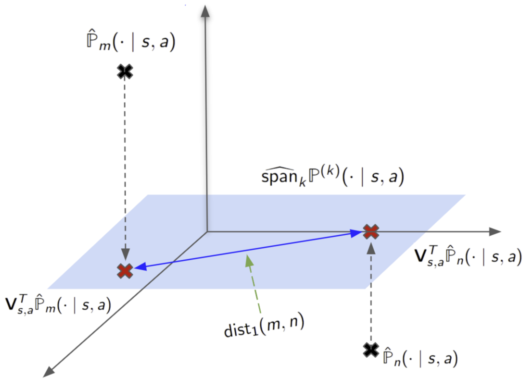

Figure: Project crude estimates to a previously estimated subspace

Algorithm Outline

Each trajectory n gives a very crude model estimate $\hat{\mathbb{P}}_n(\cdot \mid s, a)$. See the paper for many important subtleties.

1. **Subspace estimation:** Aggregate across estimates $\hat{\mathbb{P}}_n$ to obtain $(\mathbf{V}_{s,a}^T)_{K \times S}$, an estimate for the projector to $\widehat{\text{span}}_k \mathbb{P}^{(k)}(\cdot \mid s, a)$.
2. **Clustering:** Similarity-based clustering.

$$\text{dist}_1(m, n) = \max_{(s,a) \in \text{Freq}_\beta} \|\mathbf{V}_{s,a}^T \hat{\mathbb{P}}_m(\cdot \mid s, a) - \mathbf{V}_{s,a}^T \hat{\mathbb{P}}_n(\cdot \mid s, a)\|_2^2$$

3. **Model Estimation and Classification:** Estimate a model $\mathbb{P}^{(k)}(\cdot \mid s, a)$ from each cluster. Use the models to classify any new trajectories, refine using the EM algorithm.

Practical Implementation and Experiments

- **Determining K and Hyperparameters:** We provide theory-informed heuristics for determining K and the hyperparameters that we use.

Practical Implementation and Experiments

- **Determining K and Hyperparameters:** We provide theory-informed heuristics for determining K and the hyperparameters that we use.
- **Beyond just models:** One can also use this algorithm with estimators of objects other than models, like occupancy measures and rewards.

Practical Implementation and Experiments

- **Determining K and Hyperparameters:** We provide theory-informed heuristics for determining K and the hyperparameters that we use.
- **Beyond just models:** One can also use this algorithm with estimators of objects other than models, like occupancy measures and rewards.
- **Subspace estimation is crucial:** We demonstrate that using random K-dimensional subspaces or no subspaces works much worse than our method.

Practical Implementation and Experiments

- **Determining K and Hyperparameters:** We provide theory-informed heuristics for determining K and the hyperparameters that we use.
- **Beyond just models:** One can also use this algorithm with estimators of objects other than models, like occupancy measures and rewards.
- **Subspace estimation is crucial:** We demonstrate that using random K-dimensional subspaces or no subspaces works much worse than our method.
- **Evaluation:** We match clusters to labels using the Hungarian algorithm, and report the proportion of mislabelled trajectories.

End-To-End Performance (Gridworld)

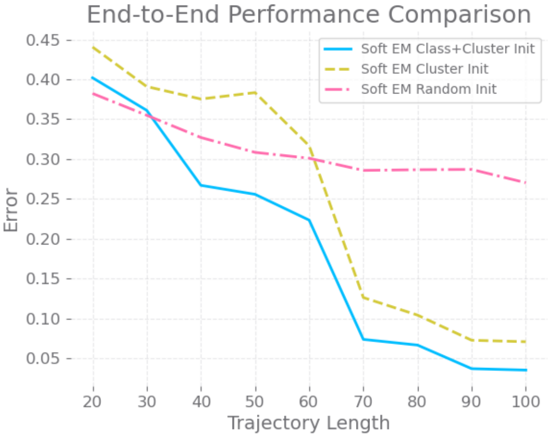

Figure: Gridworld, $K = 2, N = 1000$

End-To-End Performance (Last.FM)

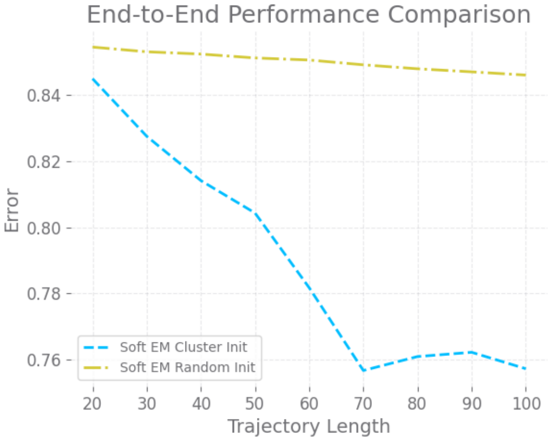

Figure: Last.FM data, $K = 10, N = 750$

Future work

- Computational improvements using matrix sketching.
- Continuous state and action spaces.
- Other controlled process, for example, linear dynamical system with control input.