Compressed Conditional Mean Embeddings for Model-Based RL

Guy Lever[1]

[1] University College London
Centre for Computational Statistics and Machine Learning

February 16, 2016

Co-authors

Joint work with:

- John Shawe-Taylor
- Ronnie Stafford
- Csaba Szepesvári

Overview

System for **model-based reinforcement learning**:

- Model MDP transition dynamics using "conditional mean embeddings": induces **finite MDP**
- Optimize policy: policy/value iteration solves finite MDP exactly

Related work: KBRL [2], "Kernel CMEs" [1], "Pseudo-MDPs" [3] use finite MDP induced by model. We address some drawbacks:

- **Compress** the finite MDP to **scale-up planning**
- Scale-up model learning: **fast**, **online** using sparse-greedy kernel matching pursuit
- Model represented in rich RKHS function class
- Bound value of learned policy in terms of model error

Experiments on quadrotor simulator

Overview

System for **model-based reinforcement learning**:

- Model MDP transition dynamics using "conditional mean embeddings": induces **finite MDP**
- Optimize policy: policy/value iteration solves finite MDP exactly

Related work: KBRL [2], "Kernel CMEs" [1], "Pseudo-MDPs" [3] use finite MDP induced by model. We address some drawbacks:

- **Compress** the finite MDP to **scale-up planning**
- Scale-up model learning: **fast**, **online** using sparse-greedy kernel matching pursuit
- Model represented in rich RKHS function class
- Bound value of learned policy in terms of model error

Experiments on quadrotor simulator

Reinforcement Learning Background

Reinforcement learning: agent sequentially interacts with unknown environment, receiving rewards
Formalized as MDP $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, r, P\}$

- $\mathcal{S}$ state space
- $\mathcal{A}$ action set
- $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ reward function (known)
- $P(s'|s, a)$ transition dynamics (Markovian, unknown)

Agent controls trajectory $s_1, a_1, s_2, ...,$ where $S_{i+1} \sim P(\cdot|s_i, a_i)$, using *policy* π where $A_t \sim \pi(\cdot|s_t)$, receives $r(s_t, a_t)$
Goal: find *policy* π^* maximizing cumulative reward:

$$J^{\pi} := \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(S_t, A_t); \pi\right]$$

Reinforcement Learning Background

Reinforcement learning: agent sequentially interacts with unknown environment, receiving rewards

Formalized as MDP $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, r, P\}$

- $\mathcal{S}$ state space
- $\mathcal{A}$ action set
- $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ reward function (known)
- $P(s'|s, a)$ transition dynamics (Markovian, unknown)

Agent controls trajectory $s_1, a_1, s_2, \ldots$, where $S_{i+1} \sim P(\cdot|s_i, a_i)$, using *policy* π where $A_t \sim \pi(\cdot|s_t)$, receives $r(s_t, a_t)$

Goal: find *policy* π^* maximizing cumulative reward:

$$J^\pi := \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(S_t, A_t); \pi\right]$$

Reinforcement Learning Background

Reinforcement learning: agent sequentially interacts with unknown environment, receiving rewards

Formalized as MDP $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, r, P\}$

- $\mathcal{S}$ state space
- $\mathcal{A}$ action set
- $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ reward function (known)
- $P(s'|s, a)$ transition dynamics (Markovian, unknown)

Agent controls trajectory $s_1, a_1, s_2, \ldots$, where $S_{i+1} \sim P(\cdot|s_i, a_i)$, using *policy* π where $A_t \sim \pi(\cdot|s_t)$, receives $r(s_t, a_t)$

Goal: find *policy* π^* maximizing cumulative reward:

$$J^\pi := \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(S_t, A_t); \pi\right]$$

Reinforcement Learning Background

Reinforcement learning: agent sequentially interacts with unknown environment, receiving rewards

Formalized as MDP $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, r, P\}$

- $\mathcal{S}$ state space
- $\mathcal{A}$ action set
- $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ reward function (known)
- $P(s'|s, a)$ transition dynamics (Markovian, unknown)

Agent controls trajectory $s_1, a_1, s_2, \ldots$, where $S_{i+1} \sim P(\cdot|s_i, a_i)$, using *policy* π where $A_t \sim \pi(\cdot|s_t)$, receives $r(s_t, a_t)$

Goal: find *policy* π^* maximizing cumulative reward:

$$J^{\pi} := \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(S_t, A_t); \pi\right]$$

Reinforcement Learning Background (2)

Value function methods

$$V^{\pi}(s) := \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(S_t, A_t) \middle| S_1 = s; \pi\right]$$

policy/value iteration learns $V^*(s) = \sup_{\pi \in \Pi} V^{\pi}(s)$

Dynamics unknown $\Rightarrow$ **Model-based RL**

– **Sample efficient**, transfer model to new tasks...

Reinforcement Learning Background (2)

Value function methods

$V^\pi(s) := \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(S_t, A_t) \middle| S_1 = s; \pi\right]$

policy/value iteration learns $V^*(s) = \sup_{\pi\in\Pi} V^\pi(s)$

$$V^*(s) = \max_{a\in\mathcal{A}} \{r(s,a) + \gamma \mathbb{E}_{S'\sim P(\cdot|s,a)}[V^*(S')]\}$$

Dynamics unknown $\Rightarrow$ **Model-based RL**

– **Sample efficient**, transfer model to new tasks...

Reinforcement Learning Background (2)

Value function methods

$V^\pi(s) := \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(S_t, A_t) \middle| S_1 = s; \pi\right]$

policy/value iteration learns $V^*(s) = \sup_{\pi \in \Pi} V^\pi(s)$

$$V_{k+1}(s) \leftarrow \max_{a \in \mathcal{A}} \{r(s,a) + \gamma \mathbb{E}_{S' \sim P(\cdot|s,a)}[V_k(S')]\}$$

Dynamics unknown $\Rightarrow$ Model-based RL

– **Sample efficient**, transfer model to new tasks...

Reinforcement Learning Background (2)

Value function methods

$$V^\pi(s) := \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(S_t, A_t) \middle| S_1 = s; \pi\right]$$

policy/value iteration learns $V^*(s) = \sup_{\pi \in \Pi} V^\pi(s)$

$$V_{k+1}(s) \leftarrow \max_{a \in \mathcal{A}} \{r(s,a) + \gamma \mathbb{E}_{S' \sim P(\cdot|s,a)}[V_k(S')]\}$$

Dynamics unknown $\Rightarrow$ **Model-based RL**

– **Sample efficient**, transfer model to new tasks...

Reinforcement Learning Background (2)

Value function methods

$$V^\pi(s) := \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(S_t, A_t) \middle| S_1 = s; \pi\right]$$

policy/value iteration learns $V^*(s) = \sup_{\pi \in \Pi} V^\pi(s)$

$$V_{k+1}(s) \leftarrow \max_{a \in \mathcal{A}} \{r(s,a) + \gamma \mathbb{E}_{S' \sim P(\cdot|s,a)}[V_k(S')]\}$$

Dynamics unknown $\Rightarrow$ **Model-based RL**

– **Sample efficient**, transfer model to new tasks...

Model Dynamics with Kernel CMEs

What do we need from a model? Consider generalized linear value representation $V_k(s) \approx \langle w_k, \phi(s) \rangle_{\mathcal{F}}$

$$V_{k+1}(s) \leftarrow \max_{a \in \mathcal{A}} \{ r(s,a) + \gamma \mathbb{E}_{S' \sim P(\cdot|s,a)} [V_k(S')] \}$$

We need to learn "conditional mean embedding"

$$\mu(s,a) := \mathbb{E}_{S' \sim P(\cdot|s,a)} [\phi(S')]$$

No generative model, no density, no sampling

Incorporate approximate model $\hat{\mu} \approx \mu$ into policy/value iteration

Model Dynamics with Kernel CMEs

What do we need from a model? Consider generalized linear value representation $V_k(s) \approx \langle w_k, \phi(s) \rangle_{\mathcal{F}}$

$$V_{k+1}(s) \leftarrow \max_{a \in \mathcal{A}} \{ r(s,a) + \gamma \langle w_k, \mathbb{E}_{S' \sim P(\cdot|s,a)}[\phi(S')] \rangle_{\mathcal{F}} \}$$

We need to learn "conditional mean embedding"

$$\mu(s,a) := \mathbb{E}_{S' \sim P(\cdot|s,a)}[\phi(S')]$$

No generative model, no density, no sampling

Incorporate approximate model $\hat{\mu} \approx \mu$ into policy/value iteration

Model Dynamics with Kernel CMEs

What do we need from a model? Consider generalized linear value representation $V_k(s) \approx \langle w_k, \phi(s) \rangle_{\mathcal{F}}$

$$V_{k+1}(s) \leftarrow \max_{a \in \mathcal{A}} \{ r(s,a) + \gamma \langle w_k, \mathbb{E}_{S' \sim P(\cdot|s,a)}[\phi(S')] \rangle_{\mathcal{F}} \}$$

We need to learn **"conditional mean embedding"**

$$\mu(s,a) := \mathbb{E}_{S' \sim P(\cdot|s,a)}[\phi(S')]$$

No generative model, no density, no sampling

Incorporate approximate model $\hat{\mu} \approx \mu$ into policy/value iteration

Model Dynamics with Kernel CMEs

What do we need from a model? Consider generalized linear value representation $V_k(s) \approx \langle w_k, \phi(s) \rangle_{\mathcal{F}}$

$$V_{k+1}(s) \leftarrow \max_{a \in \mathcal{A}} \{ r(s,a) + \gamma \langle w_k, \mathbb{E}_{S' \sim P(\cdot|s,a)}[\phi(S')] \rangle_{\mathcal{F}} \}$$

We need to learn **"conditional mean embedding"**

$$\mu(s,a) := \mathbb{E}_{S' \sim P(\cdot|s,a)}[\phi(S')]$$

No generative model, no density, no sampling

Incorporate approximate model $\hat{\mu} \approx \mu$ into policy/value iteration

Learning CMEs

Given system data, $\mathcal{D} = \{(s_i, a_i), \phi(s'_i)\}_{i=1}^n$, find

$$\hat{\mu} = \underset{\mu:\mathcal{S}\times\mathcal{A}\to\mathcal{F}}{\operatorname{argmin}} \sum_{i=1}^{n} ||\phi(s'_i) - \mu(s_i, a_i)||^2_{\mathcal{F}}$$

e.g. kernel smoothing (KBRL), kernel least-squares

Induced Finite MDP:

Models have form $\hat{\mu}(s,a) := \sum_{j=1}^n \alpha_j(s,a)\phi(s'_j)$ so

$$\mathbb{E}[V(S')|s,a] \approx \sum_{i=1}^{n} \alpha_i(s,a)V(s'_i)$$

i.e. we only need to maintain V on samples

If $||\alpha(s,a)||_1 \leq 1$, $\alpha_i(s,a)$, plan exactly on finite (pseudo-)MDP...

Learning CMEs

Given system data, $\mathcal{D} = \{(s_i, a_i), \phi(s_i')\}_{i=1}^n$, find

$$\hat{\mu} = \underset{\mu:\mathcal{S}\times\mathcal{A}\to\mathcal{F}}{\operatorname{argmin}} \sum_{i=1}^{n} ||\phi(s_i') - \mu(s_i, a_i)||_{\mathcal{F}}^2$$

e.g. kernel smoothing (KBRL), kernel least-squares

Induced Finite MDP:

Models have form $\hat{\mu}(s, a) := \sum_{j=1}^n \alpha_j(s, a)\phi(s_j')$ so

$$\mathbb{E}[V(S')|s, a] \approx \sum_{i=1}^{n} \alpha_i(s, a)V(s_i')$$

i.e. we only need to maintain V on samples

If $||\alpha(s, a)||_1 \leq 1$, $\alpha_i(s, a)$, plan exactly on finite (pseudo-)MDP...

Learning CMEs

Given system data, $\mathcal{D} = \{(s_i, a_i), \phi(s_i')\}_{i=1}^n$, find

$$\hat{\mu} = \underset{\mu:\mathcal{S}\times\mathcal{A}\to\mathcal{F}}{\operatorname{argmin}} \sum_{i=1}^{n} ||\phi(s_i') - \mu(s_i, a_i)||_{\mathcal{F}}^2$$

e.g. kernel smoothing (KBRL), kernel least-squares

Induced Finite MDP:

Models have form $\hat{\mu}(s, a) := \sum_{j=1}^n \alpha_j(s, a)\phi(s_j')$ so

$$\mathbb{E}[V(S')|s, a] \approx \sum_{i=1}^{n} \alpha_i(s, a) V(s_i')$$

i.e. we only need to maintain V on samples

If $||\alpha(s, a)||_1 \leq 1$, $\alpha_i(s, a)$, plan exactly on finite (pseudo-)MDP...

Learning CMEs

Given system data, $\mathcal{D} = \{(s_i, a_i), \phi(s_i')\}_{i=1}^n$, find

$$\hat{\mu} = \underset{\mu:\mathcal{S}\times\mathcal{A}\to\mathcal{F}}{\operatorname{argmin}} \sum_{i=1}^{n} ||\phi(s_i') - \mu(s_i, a_i)||_{\mathcal{F}}^2$$

e.g. kernel smoothing (KBRL), kernel least-squares

Induced Finite MDP:

Models have form $\hat{\mu}(s, a) := \sum_{j=1}^n \alpha_j(s, a)\phi(s_j')$ so

$$\mathbb{E}[V(S')|s, a] \approx \sum_{i=1}^{n} \alpha_i(s, a) V(s_i')$$

i.e. we only need to maintain V on samples

If $||\alpha(s, a)||_1 \leq 1$, $\alpha_i(s, a)$, plan exactly on finite (pseudo-)MDP...

Learning CMEs

Given system data, $\mathcal{D} = \{(s_i, a_i), \phi(s_i')\}_{i=1}^n$, find

$$\hat{\mu} = \underset{\mu:\mathcal{S}\times\mathcal{A}\to\mathcal{F}}{\operatorname{argmin}} \sum_{i=1}^{n} ||\phi(s_i') - \mu(s_i, a_i)||_{\mathcal{F}}^2$$

e.g. kernel smoothing (KBRL), kernel least-squares

Induced Finite MDP:

Models have form $\hat{\mu}(s, a) := \sum_{j=1}^n \alpha_j(s, a)\phi(s_j')$ so

$$\mathbb{E}[V(S')|s, a] \approx \sum_{i=1}^{n} \alpha_i(s, a) V(s_i')$$

i.e. we only need to maintain V on samples

If $||\alpha(s, a)||_1 \leq 1$, $\alpha_i(s, a)$, plan exactly on finite (pseudo-)MDP...

Algorithm: Online Policy Optimization with CME model

Repeat:

1: Update data $\mathcal{D} = \{(s_i, a_i, s_i')\}_{i=1}^n$

2: Update dynamics model $\hat{\mu}(s, a) = \sum_{j=1}^n \alpha_j(s, a)\phi(s_j')$

3: Value iteration with approximate model: for $s \in \{s_1', \ldots, s_n'\}$

$$V_{k+1}(s) \leftarrow \max_{a \in \mathcal{A}}\{r(s, a) + \gamma \mathbb{E}_{S' \sim P(\cdot|s,a)}[V_k(S')]\}$$

$$\approx \max_{a \in \mathcal{A}}\{r(s, a) + \gamma \sum_{j=1}^n \alpha_j(s, a) V_k(s_j')\}$$

4: Act greedily $\pi_K(s) = \text{argmax}_{a \in \mathcal{A}}\{r(s, a) + \gamma \sum_{j=1}^n \alpha_j(s, a) V_K(s_j')\}$

Advantages: value iteration converges; avoid approx. dynamic programming; good performance bounds

Problems: planning scales poorly $O(|\mathcal{A}|kn^2)$; model learning can be slow

Algorithm: Online Policy Optimization with CME model

Repeat:

1: Update data $\mathcal{D} = \{(s_i, a_i, s_i')\}_{i=1}^n$

2: Update dynamics model $\hat{\mu}(s,a) = \sum_{j=1}^n \alpha_j(s,a)\phi(s_j')$

3: Value iteration with approximate model: for $s \in \{s_1', ..., s_n'\}$

$$V_{k+1}(s) \leftarrow \max_{a \in \mathcal{A}} \{r(s,a) + \gamma \mathbb{E}_{S' \sim P(\cdot|s,a)}[V_k(S')]\}$$
$$\approx \max_{a \in \mathcal{A}} \{r(s,a) + \gamma \sum_{j=1}^n \alpha_j(s,a) V_k(s_j')\}$$

4: Act greedily $\pi_K(s) = \text{argmax}_{a \in \mathcal{A}} \{r(s,a) + \gamma \sum_{j=1}^n \alpha_j(s,a) V_K(s_j')\}$

Advantages: value iteration converges; avoid approx. dynamic programming; good performance bounds

Problems: planning scales poorly $O(|\mathcal{A}|kn^2)$; model learning can be slow

Fast Planning with a Compressed Model

Compress the induced MDP without losing performance:
Maintain a "δ-lossy compression" $\mathcal{C} = \{c_1, ..., c_m\}$ of $\{s'_1, ..., s'_n\}$ s.t.

$$\max_{1 \le j \le n} \min_{b:||b||_1 \le 1} || \sum_{i=1}^{m} b_i \phi(c_i) - \phi(s'_j) ||_{\mathcal{F}} \le \delta$$

Represent $\hat{\mu}$ on $\mathcal{C}$: $\hat{\mu}(s,a) = \sum_{j=1}^{m} \alpha_i(s,a)\phi(c_j)$:

planning $O(|\mathcal{A}|km^2)$

Algorithm: augmentCompressionSet$(\mathcal{C}, \delta, s)$

Input: Initial compression set $\mathcal{C} = c_1, ..., c_m$, candidate $s \in \mathcal{S}$, tolerance δ

if $\min_{\mathbf{b} \in \mathbb{R}^m, ||\mathbf{b}||_1 \le 1} || \sum_{i=1}^{m} b_i \phi(c_i) - \phi(s) ||_{\mathcal{F}} > \delta$ **then**

  Augment: $\mathcal{C} \leftarrow \mathcal{C} \cup s$

end if

minimization is a Lasso

Fast Planning with a Compressed Model

Compress the induced MDP without losing performance:
Maintain a "δ-lossy compression" $\mathcal{C} = \{c_1, ..., c_m\}$ of $\{s'_1, ..., s'_n\}$ s.t.

$$\max_{1 \leq j \leq n} \min_{b:||b||_1 \leq 1} || \sum_{i=1}^{m} b_i \phi(c_i) - \phi(s'_j)||_{\mathcal{F}} \leq \delta$$

Represent $\hat{\mu}$ on $\mathcal{C}$: $\hat{\mu}(s, a) = \sum_{j=1}^{m} \alpha_i(s, a)\phi(c_j)$:
planning $O(|\mathcal{A}|km^2)$

Algorithm: `augmentCompressionSet`$(\mathcal{C}, \delta, s)$

Input: Initial compression set $\mathcal{C} = c_1, ..., c_m$, candidate $s \in \mathcal{S}$, tolerance δ
if $\min_{b \in \mathbb{R}^m, ||b||_1 \leq 1} || \sum_{i=1}^{m} b_i \phi(c_i) - \phi(s)||_{\mathcal{F}} > \delta$ **then**
  Augment: $\mathcal{C} \leftarrow \mathcal{C} \cup s$
end if

minimization is a Lasso

Fast Planning with a Compressed Model

Compress the induced MDP without losing performance:
Maintain a "δ-lossy compression" $\mathcal{C} = \{c_1, ..., c_m\}$ of $\{s'_1, ..., s'_n\}$ s.t.

$$\max_{1 \le j \le n} \min_{b:||b||_1 \le 1} || \sum_{i=1}^{m} b_i \phi(c_i) - \phi(s'_j)||_{\mathcal{F}} \le \delta$$

Represent $\hat{\mu}$ on $\mathcal{C}$: $\hat{\mu}(s,a) = \sum_{j=1}^{m} \alpha_i(s,a)\phi(c_j)$:

planning $O(|\mathcal{A}|km^2)$

Algorithm: `augmentCompressionSet`$(\mathcal{C}, \delta, s)$

Input: Initial compression set $\mathcal{C} = c_1, ..., c_m$, candidate $s \in \mathcal{S}$, tolerance δ

if $\min_{\mathbf{b} \in \mathbb{R}^m, ||\mathbf{b}||_1 \le 1} || \sum_{i=1}^{m} b_i \phi(c_i) - \phi(s)||_{\mathcal{F}} > \delta$ **then**

  Augment: $\mathcal{C} \leftarrow \mathcal{C} \cup s$

end if

minimization is a Lasso

Performance guarantees for Compressed Model

Theorem

Bound for value iteration with CME $\hat{\mu}$: for any $\tilde{V}^* \in \mathcal{F}$

$$||V^{\pi_k} - V^*||_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\Big(\gamma^k ||V^{\pi_1} - V^{\pi_0}||_\infty + 2||V^* - \tilde{V}^*||_\infty + \sup_{s,a} ||\mathbb{E}_{S' \sim P(\cdot|s,a)}[\phi(S')] - \hat{\mu}(s,a)||_{\mathcal{F}} ||\tilde{V}^*||_{\mathcal{F}}\Big) =: B_k(\hat{\mu})$$

Theorem

If $\hat{\mu}$ defined on $\{s'_1, ..., s'_n\}$ and C a δ-lossy compression, then there exists $\bar{\mu}$ represented on C such that after k value iterations using $\bar{\mu}$

$$||V^{\pi_k} - V^*||_\infty \leq B_k(\hat{\mu}) + \frac{2\gamma\delta}{(1-\gamma)^2}||\tilde{V}^*||_{\mathcal{F}}$$

Performance guarantees for Compressed Model

Theorem

Bound for value iteration with CME $\hat{\mu}$: for any $\tilde{V}^* \in \mathcal{F}$

$$
\begin{aligned}
||V^{\pi_k} - V^*||_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\Big(&\gamma^k ||V^{\pi_1} - V^{\pi_0}||_\infty + 2||V^* - \tilde{V}^*||_\infty \\
&+ \sup_{s,a} ||\mathbb{E}_{S' \sim P(\cdot|s,a)}[\phi(S')] - \hat{\mu}(s,a)||_{\mathcal{F}} ||\tilde{V}^*||_{\mathcal{F}}\Big) \\
&=: B_k(\hat{\mu})
\end{aligned}
$$

Theorem

If $\hat{\mu}$ defined on $\{s'_1, \ldots, s'_n\}$ and C a δ-lossy compression, then there exists $\bar{\mu}$ represented on C such that after k value iterations using $\bar{\mu}$

$$
||V^{\pi_k} - V^*||_\infty \leq B_k(\hat{\mu}) + \frac{2\gamma\delta}{(1-\gamma)^2}||\tilde{V}^*||_{\mathcal{F}}
$$

Performance guarantees for Compressed Model

Theorem

Bound for value iteration with CME $\hat{\mu}$: for any $\tilde{V}^* \in \mathcal{F}$

$$\begin{aligned}||V^{\pi_k} - V^*||_\infty &\leq \frac{2\gamma}{(1-\gamma)^2}\Big(\gamma^k ||V^{\pi_1} - V^{\pi_0}||_\infty + 2||V^* - \tilde{V}^*||_\infty \\ &+ \sup_{s,a} ||\mathbb{E}_{S' \sim P(\cdot|s,a)}[\phi(S')] - \hat{\mu}(s,a)||_{\mathcal{F}} ||\tilde{V}^*||_{\mathcal{F}}\Big) \\ &=: B_k(\hat{\mu})\end{aligned}$$

Theorem

If $\hat{\mu}$ defined on $\{s'_1, \ldots, s'_n\}$ and C a δ-lossy compression, then there exists $\bar{\mu}$ represented on C such that after k value iterations using $\bar{\mu}$

$$||V^{\pi_k} - V^*||_\infty \leq B_k(\hat{\mu}) + \frac{2\gamma\delta}{(1-\gamma)^2}||\tilde{V}^*||_{\mathcal{F}}$$

Performance guarantees for Compressed Model

Theorem

Bound for value iteration with CME $\hat{\mu}$: for any $\tilde{V}^* \in \mathcal{F}$

$$||V^{\pi_k} - V^*||_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\Big(\gamma^k ||V^{\pi_1} - V^{\pi_0}||_\infty + 2||V^* - \tilde{V}^*||_\infty$$
$$+ \sup_{s,a} ||\mathbb{E}_{S' \sim P(\cdot|s,a)}[\phi(S')] - \hat{\mu}(s,a)||_\mathcal{F} ||\tilde{V}^*||_\mathcal{F}\Big)$$
$$=: B_k(\hat{\mu})$$

Theorem

If $\hat{\mu}$ defined on $\{s'_1, \ldots, s'_n\}$ and $\mathcal{C}$ a δ-lossy compression, then there exists $\bar{\mu}$ represented on $\mathcal{C}$ such that after k value iterations using $\bar{\mu}$

$$||V^{\pi_k} - V^*||_\infty \leq B_k(\hat{\mu}) + \frac{2\gamma\delta}{(1-\gamma)^2}||\tilde{V}^*||_\mathcal{F}$$

Performance guarantees for Compressed Model

Theorem

Bound for value iteration with CME $\hat{\mu}$: for any $\tilde{V}^* \in \mathcal{F}$

$$\begin{aligned} ||V^{\pi_k} - V^*||_\infty &\leq \frac{2\gamma}{(1-\gamma)^2}\Big(\gamma^k ||V^{\pi_1} - V^{\pi_0}||_\infty + 2||V^* - \tilde{V}^*||_\infty \\ &+ \sup_{s,a} ||\mathbb{E}_{S' \sim P(\cdot|s,a)}[\phi(S')] - \hat{\mu}(s,a)||_{\mathcal{F}} ||\tilde{V}^*||_{\mathcal{F}}\Big) \\ &=: B_k(\hat{\mu}) \end{aligned}$$

Theorem

If $\hat{\mu}$ defined on $\{s'_1, \dots, s'_n\}$ and $\mathcal{C}$ a δ-lossy compression, then there exists $\bar{\mu}$ represented on $\mathcal{C}$ such that after k value iterations using $\bar{\mu}$

$$||V^{\pi_k} - V^*||_\infty \leq B_k(\hat{\mu}) + \frac{2\gamma\delta}{(1-\gamma)^2} ||\tilde{V}^*||_{\mathcal{F}}$$

Fast Model Learning with Matching Pursuit

Optimize the model: Kernel smoothing, Kernel least-squares?

We use the (vector-valued) kernel regressor: put kernel K on input space $\mathcal{S} \times \mathcal{A}$,

$$\hat{\mu}(s,a) := \sum_{i=1}^{n} \sum_{j=1}^{m} K((s,a),(s_i,a_i)) W_{ij} \phi(c_j),$$

Learn W by sparse-greedy kernel matching pursuit:

- fast, online and W row sparse.
- models dynamics in rich kernel-defined RKHS $\mathcal{H}_K$

Project: $\alpha(s,a) = \text{argmin}\, \beta : ||\beta||_1 \leq 1 \{ || \sum_{j=1}^{m} \beta_j(s,a)\phi(s'_j) - \sum_{i=1}^{n} \sum_{j=1}^{m} K((s,a),(s_i,a_i)) W_{ij} \phi(c'_j) ||_{\mathcal{F}}^2 \}$ (Lasso)

Fast Model Learning with Matching Pursuit

Optimize the model: Kernel smoothing, Kernel least-squares?
We use the (vector-valued) kernel regressor: put kernel K on input space $\mathcal{S} \times \mathcal{A}$,

$$\hat{\mu}(s, a) := \sum_{i=1}^{n} \sum_{j=1}^{m} K((s, a), (s_i, a_i)) W_{ij} \phi(c_j),$$

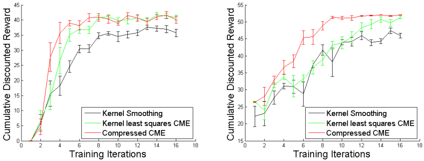
Learn W by **sparse-greedy** kernel matching pursuit:

- **fast**, **online** and W row sparse.
- models dynamics in rich kernel-defined RKHS $\mathcal{H}_K$

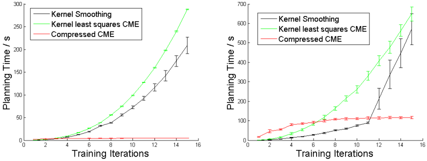
Project: $\alpha(s, a) = \text{argmin}\, \beta : ||\beta||_1 \leq 1\{|| \sum_{j=1}^{m} \beta_j(s, a)\phi(s_j') - \sum_{i=1}^{n} \sum_{j=1}^{m} K((s, a), (s_i, a_i)) W_{ij} \phi(c_j')||_{\mathcal{F}}^2\}$ (Lasso)

Experiments

Mountain Car and Cart-Pole benchmark MDPs: rewards



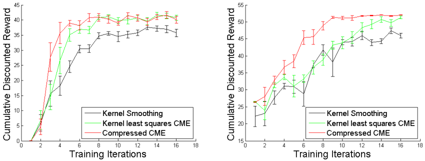
faster planning using compact data-defined representation



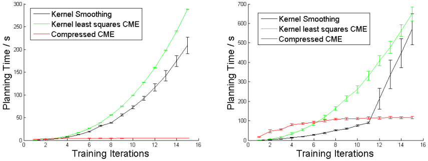
Simulated quadrotor experiments, $dim(\mathcal{S}) = 13$.

Experiments

Mountain Car and Cart-Pole benchmark MDPs: rewards



faster planning using compact data-defined representation



Simulated quadrotor experiments, $dim(\mathcal{S}) = 13$.

Conclusions

A system for general reinforcement learning:

- Learn system transition dynamics using CME
- Compress the model for fast planning
- Rich, data-dependent, RKHS model class
- Optimize policy with value/policy iteration on induced finite MDP
- Performance guarantee

Future work

- Represent $\mu(s, a)$ using neural nets
- Connection to subgoals

References

S. Grunewalder, G. Lever, L. Baldessarre, M. Pontil, A. Gretton
Modelling transition dynamics in MDPs with RKHS embeddings.
ICML 2012.

D. Ormoneit, S. Sen
Kernel-based reinforcement learning.
Machine Learning 2002.

H. Yao, Cs. Szepesvári, B.A. Pires and X. Zhang
Pseudo-MDPs and Factored Linear Action Models.
IEEE ADPRL, 2014.