An Online Spectral Learning Algorithm for Partially Observable Nonlinear Dynamical Systems

Byron Boots and Geoffrey J. Gordon

AAAI 2011

Select Lab

Carnegie Mellon University
What is out there?

\[ o_{t-2} \rightarrow o_{t-1} \rightarrow o_t \rightarrow o_{t+1} \rightarrow o_{t+2} \rightarrow \ldots \]
What is out there? Dynamical Systems

Dynamical System = A recursive rule for updating state based on observations
we would like to learn a model of a dynamical system
Learning a Dynamical System

We would like to learn a model of a dynamical system.

today I will focus on **Spectral Learning Algorithms** for **Predictive State Representations**
Predictive State Representations (PSRs)

comprised of:

- set of actions $A$
- set of observations $O$
- initial state $x_1 \in \mathbb{R}^d$
- set of transition matrices $M_{ao} \in \mathbb{R}^{d \times d}$
- normalization vector $e \in \mathbb{R}^d$
Predictive State Representations (PSRs)

comprised of:

set of actions $A$
set of observations $O$
initial state $x_1 \in \mathbb{R}^d$
set of transition matrices $M_{ao} \in \mathbb{R}^{d \times d}$
normalization vector $e \in \mathbb{R}^d$

at each time step can predict:

$$P(o \mid x_t, do(a_t)) = e^\top M_{a_t,o} x_t$$
Predictive State Representations (PSRs)

comprised of:

- set of actions $A$
- set of observations $O$
- initial state $x_1 \in \mathbb{R}^d$
- set of transition matrices $M_{ao} \in \mathbb{R}^{d \times d}$
- normalization vector $e \in \mathbb{R}^d$

at each time step can predict:

$$P(o \mid x_t, do(a_t)) = e^\top M_{at,o} x_t$$
Predictive State Representations (PSRs)

comprised of:

- set of actions $A$
- set of observations $O$
- initial state $x_1 \in \mathbb{R}^d$
- set of transition matrices $M_{ao} \in \mathbb{R}^{d \times d}$
- normalization vector $e \in \mathbb{R}^d$

at each time step can predict:

$$P(o \mid x_t, do(a_t)) = e^\top M_{at,o} x_t$$
Predictive State Representations (PSRs)

comprised of:

- set of actions $A$
- set of observations $O$
- initial state $x_1 \in \mathbb{R}^d$
- set of transition matrices $M_{ao} \in \mathbb{R}^{d \times d}$
- normalization vector $e \in \mathbb{R}^d$

at each time step can predict:

$$P(o \mid x_t, do(a_t)) = e^\top M_{a_t,o} x_t$$

and update state:

$$x_{t+1} = M_{a_t,o_t} x_t / P(o_t \mid x_t, do(a_t))$$
Predictive State Representations (PSRs)

comprised of:

- set of actions $A$
- set of observations $O$
- initial state $x_1 \in \mathbb{R}^d$
- set of transition matrices $M_{ao} \in \mathbb{R}^{d \times d}$
- normalization vector $e \in \mathbb{R}^d$

at each time step can predict:

$$P(o \mid x_t, do(a_t)) = e^\top M_{a_t,o} x_t$$

and update state:

$$x_{t+1} = M_{a_t,o_t} x_t / P(o_t \mid x_t, do(a_t))$$
Predictive State Representations (PSRs)

- set of actions $A$
- set of observations $O$
- initial state $x_1 \in \mathbb{R}^d$
- set of transition matrices $M_{ao} \in \mathbb{R}^{d \times d}$
- normalization vector $e \in \mathbb{R}^d$

at each time step can predict:

$$P(o \mid x_t, do(a_t)) = e^\top M_{a_t,o} x_t$$

and update state:

$$x_{t+1} = M_{a_{t},o_{t}} x_t / P(o_{t} \mid x_t, do(a_t))$$
Predictive State Representations

parameters are only determined up to a similarity transform $S \in \mathbb{R}^{d \times d}$

if we replace

$$M_{ao} \rightarrow S^{-1} M_{ao} S$$
$$x_1 \rightarrow S^{-1} x_1$$
$$e \rightarrow S^\top e$$

the resulting PSR makes exactly the same predictions as the original one

\[ P(o \mid x_t, do(a_t)) = e^\top S S^{-1} M_{at,o} S S^{-1} x_t \]
PSRs Are Very Expressive

Predictive State Representations

Reduced-Rank HMMs & Reduced Rank POMDPS

HMMs & POMDPS

for fixed latent dimension $d$
Learning PSRs

can use fast, statistically consistent, spectral methods to learn PSR parameters
A General Principle

- **Compression**
  - Data about past (many samples)
  - Compress

- **Expansion**
  - Data about future (many samples)
  - Expand

- **State**
  - Predict
  - Bottleneck
A General Principle

If bottleneck = rank constraint, then get a spectral method.
Why Spectral Methods?

There are many ways to learn a dynamical system

- Maximum Likelihood via Expectation Maximization, Gradient Descent, ...
- Bayesian inference via Gibbs, Metropolis Hastings, ...

In contrast to these methods, spectral learning algorithms give

- **No local optima:**
  - Huge gain in computational efficiency
- Slight loss in statistical efficiency
Spectral Learning for PSRs

moments of directly observable features

\[ \Sigma_{T,AO,H} \] “trivariance” tensor of features of the future, present, and past

\[ \Sigma_{T,H} \] covariance matrix of features of the future and past

\[ \Sigma_{AO,AO} \] covariance matrix of features present
Spectral Learning for PSRs

moments of directly observable features

$\Sigma_{T, AO, H}$ “trivariance” tensor of features of the future, present, and past

$\Sigma_{T, H}$ covariance matrix of features of the future and past

$\Sigma_{AO, AO}$ covariance matrix of features present

$U$ left $d$ singular vectors of $\Sigma_{T, H}$
Spectral Learning for PSRs

moments of directly observable features

\[ \Sigma_{T, AO,H} \] “trivariance” tensor of features of the future, present, and past

\[ \Sigma_{T,H} \] covariance matrix of features of the future and past

\[ \Sigma_{AO,AO} \] covariance matrix of features present

\[ U \] left \( d \) singular vectors of \( \Sigma_{T,H} \)

\[ S^{-1} M_{ao} S := \Sigma_{T, AO, H} \times_1 U^\top \times_2 \phi(a_o)^\top (\Sigma_{AO,AO})^{-1} \times_3 (\Sigma_{T,H} U)^\dagger \]

the other parameters can be found analogously
Spectral Learning for PSRs

Spectral Learning Algorithm:
• Estimate $\Sigma_{T,AO,H}$, $\Sigma_{T,H}$, and $\Sigma_{AO,AO}$ from data
• Find $\hat{U}$ by SVD
• Plug in to recover PSR parameters

• Learning is Statistically Consistent
• Only requires Linear Algebra

For details, see:

Infinite Features

• Can extend the learning algorithm to infinite feature spaces
  ‣ Kernels

• Learning algorithm that we have seen is linear algebra
  ‣ works just fine in an arbitrary RKHS
  ‣ Can rewrite all of the formulas in terms of Gram matrices
  ‣ Uses kernel SVD instead of SVD

Result: Hilbert Space Embeddings of Dynamical Systems

• handles near arbitrary observation distributions
• good prediction performance

For details, see:

An Experiment

Batch Methods

- **Bottleneck**: SVD of Gram or Covariance matrix
  - \( G: (# \text{ time steps})^2 \)
  - \( C: (# \text{ features} \times \text{window length}) \times (# \text{ time steps}) \)

- E.g., 1 hr video, 24 fps, 300×300, features of past and future are all pixels in 2 s windows
  - \( G: (3600 \times 24) \times (3600 \times 24) \approx 10^{10} \)
Making it Fast

• Two techniques
  ‣ online learning
  ‣ random projections

• Neither one new, but combination with spectral learning for PSRs is, and makes huge difference in practice
Online Learning

\[ U \text{ left } d \text{ singular vectors of } \Sigma_{\mathcal{T}, \mathcal{H}} \]

\[ S^{-1} M_{ao} S := \Sigma_{\mathcal{T}, \mathcal{AO}, \mathcal{H}} \times_1 U^\top \times_2 \phi(ao)^\top (\Sigma_{\mathcal{AO}, \mathcal{AO}})^{-1} \times_3 (\Sigma_{\mathcal{T}, \mathcal{H}} U)^\dagger \]
Online Learning

\[ U \text{ left } d \text{ singular vectors of } \Sigma_{\mathcal{T},\mathcal{H}} \]

\[ S^{-1} M_{ao} S := \Sigma_{\mathcal{T},\mathcal{A}O,\mathcal{H}} \times_1 U^\top \times_2 \phi(a_o)^\top (\Sigma_{\mathcal{A}O,\mathcal{A}O})^{-1} \times_3 (\Sigma_{\mathcal{T},\mathcal{H}} U)^\dagger \]

- With each new observation, rank-1 update of:
  - SVD (Brand)
  - inverse (Sherman-Morrison)

- \( n \) features; latent dimension \( d \); \( T \) steps
  - space = \( O(nd) \): may fit in cache!
  - time = \( O(nd^2 T) \): bounded time per example
Random Projections

\[ U \text{ left } d \text{ singular vectors of } \Sigma_{\mathcal{T},\mathcal{H}} \]

\[ S^{-1}M_{ao}S := \Sigma_{\mathcal{T},\mathcal{AO},\mathcal{H}} \times_1 U^\top \times_2 \phi(ao)^\top (\Sigma_{\mathcal{AO},\mathcal{AO}})^{-1} \times_3 (\Sigma_{\mathcal{T},\mathcal{H}}U)^\dagger \]

- With each new observation, **rank-1 update** of:
  - SVD (Brand)
  - inverse (Sherman-Morrison)

- \( n \) features; latent dimension \( d \); \( T \) steps
  - space = \( O(nd) \): may fit in cache!
  - time = \( O(nd^2T) \): bounded time per example

- **Problem**: no rank-1 update of kernel SVD!
  - can use random projections [Rahimi & Recht, 2007]
For each model, we performed filtering for different time steps and finally, windows is learned also with sequences of data.

We trained a 2-dimensional embedded HMM using different filtering extents to predict future IMU readings.

Figure 4 resulted in a more balanced dataset with 10 minutes of data while the slot car circled the track controlled by a constant policy. The goal of this experiment was to evaluate the effectiveness of different models.

For this experiment, we grouped the latter five audio clips as Human Speech, and the other five as Nondominant sounds. Six classes of labeled audio clips were present in the data, one being Human Speech.

Our final experiment concerns an audio classification task. Since the original data had a disproportionately large amount of Human Speech samples, we used Parzen windows to group the latter five classes into a single class of Nondominant sounds.

Figure 5 shows the results of our experiments, demonstrating that embedded HMMs have higher accuracy and avoid local minima. For each model size, we performed 100 random restarts for training and testing.

Mean Observation, HMM, Embedded, and Online with Random Features scoring functions were used for classification.

The graph indicates that embedded HMMs perform better in terms of accuracy and robustness. Embedded HMMs were trained using multidimensional EM to avoid local minima. Stable Subspace ID and spectral learning were used, along with Gaussian RBF kernels with different bandwidths set with the ‘median trick’.
Online Learning Example

Conference Room

Table
Online Learning Example

Conference Room

- online+random: 100k features, 11k frames, limit = avail. data
- offline: 2k frames, compressed & subsampled, compute-limited
Online Learning Example

Conference Room

- online+random: 100k features, 11k frames, limit = avail. data
- offline: 2k frames, compressed & subsampled, compute-limited

100 steps
Online Learning Example

Conference Room

- online+random: 100k features, 11k frames, limit = avail. data
- offline: 2k frames, compressed & subsampled, compute-limited
Online Learning Example

Conference Room

- online+random: 100k features, 11k frames, limit = avail. data
- offline: 2k frames, compressed & subsampled, compute-limited

600 steps
Online Learning Example

Conference Room

- online+random: 100k features, 11k frames, limit = avail. data
- offline: 2k frames, compressed & subsampled, compute-limited

600 steps

Table

final embedding
(colors = 3rd dim)
• We present spectral learning algorithms for PSR models of partially observable nonlinear dynamical systems.

• We show how to update parameters of the estimated PSR model given new data
  
  ▸ efficient online spectral learning algorithm

• We show how to use random projections to approximate kernel-based learning algorithms