A Sober Look at Spectral Learning

Han Zhao and Pascal Poupart

UNIVERSITY OF WATERLOO

June 17, 2014
Spectral Learning

What is spectral learning?

- New methods in machine learning to tackle mixture models and graphical models with latent variables.
Spectral Learning

What is spectral learning?

- New methods in machine learning to tackle mixture models and graphical models with latent variables.
- Dates back to Karl Pearson’s *method of moments* approach to solve mixture of Gaussians.
Spectral Learning

What is spectral learning?

- New methods in machine learning to tackle mixture models and graphical models with latent variables.
- Dates back to Karl Pearson’s method of moments approach to solve mixture of Gaussians.
- An alternative to the principle of maximum likelihood estimation and Bayesian inference.
Spectral Learning

What is spectral learning?

▶ New methods in machine learning to tackle mixture models and graphical models with latent variables.
▶ Dates back to Karl Pearson’s *method of moments* approach to solve mixture of Gaussians.
▶ An alternative to the principle of maximum likelihood estimation and Bayesian inference.
▶ Been widely applied to various models, including Hidden Markov Models [1, 2], mixture of Gaussians [3], Topic Models [4, 5, 6] and latent junction trees [7, 8], etc.
What is spectral learning?

- New methods in machine learning to tackle mixture models and graphical models with latent variables.
- Dates back to Karl Pearson’s *method of moments* approach to solve mixture of Gaussians.
- An alternative to the principle of maximum likelihood estimation and Bayesian inference.
- Been widely applied to various models, including Hidden Markov Models [1, 2], mixture of Gaussians [3], Topic Models [4, 5, 6] and latent junction trees [7, 8], etc.

Today I will focus on spectral algorithm for Hidden Markov Models.
Hidden Markov Model

- A discrete time stochastic process.
- Satisfies Markovian property.
- The state of the system at each time step is hidden, only the observation of the system is visible.
HMM can be defined as a triple $\langle T, O, \pi \rangle$:

- Transition matrix $T \in \mathbb{R}^{m \times m}$, $T_{ij} = \Pr(s_{t+1} = i \mid s_t = j)$.
- Observation matrix $O \in \mathbb{R}^{n \times m}$, $O_{ij} = \Pr(o_t = i \mid s_t = j)$.
- Initial distribution $\pi \in \mathbb{R}^m$, $\pi_i = \Pr(s_1 = i)$. 

---

HMM can be defined as a triple $\langle T, O, \pi \rangle$:

- Transition matrix $T \in \mathbb{R}^{m \times m}$, $T_{ij} = \Pr(s_{t+1} = i \mid s_t = j)$.
- Observation matrix $O \in \mathbb{R}^{n \times m}$, $O_{ij} = \Pr(o_t = i \mid s_t = j)$.
- Initial distribution $\pi \in \mathbb{R}^m$, $\pi_i = \Pr(s_1 = i)$. 

---
Given an HMM $\mathcal{H} = \langle T, O, \pi \rangle$, we are interested in two inference problems:

1. Marginal Inference (Estimation problem). Computing the marginal probability $\Pr(o_{1:t}) = \sum_{s_{1:t}} \Pr(o_{1:t}, s_{1:t}) = \sum_{s_{1:t}} \Pr(s_{1:t}) \Pr(o_{1:t} | s_{1:t})$.

2. MAP Inference (Decoding problem). Computing the sequence $s^*_{1:t}$ maximizing the posterior probability $s^*_{1:t} = \arg \max \Pr(s_{1:t} | o_{1:t})$.
HMM

Given an HMM $\mathcal{H} = \langle T, O, \pi \rangle$, we are interested in two inference problems:

1. **Marginal Inference (Estimation problem).** Computing the marginal probability

   $$
   \Pr(o_{1:t}) = \sum_{s_{1:t}} \Pr(o_{1:t}, s_{1:t}) = \sum_{s_{1:t}} \Pr(s_{1:t}) \Pr(o_{1:t} | s_{1:t})
   $$

   

   **Dynamic Programming**

2. **MAP Inference (Decoding problem).** Computing the sequence $s^*_{1:t}$ maximizing the posterior probability

   $$
   s^*_{1:t} = \arg \max_{s_{1:t}} \Pr(s_{1:t} | o_{1:t})
   $$

   **Viterbi Algorithm**

What about the learning problem?
HMM

Given an HMM $\mathcal{H} = \langle T, O, \pi \rangle$, we are interested in two inference problems:

1. Marginal Inference (Estimation problem). Computing the marginal probability

$$\Pr(o_{1:t}) = \sum_{s_{1:t}} Pr(o_{1:t}, s_{1:t}) = \sum_{s_{1:t}} Pr(s_{1:t}) Pr(o_{1:t}|s_{1:t})$$

2. MAP Inference (Decoding problem). Computing the sequence $s_{1:t}^*$ maximizing the posterior probability

$$s_{1:t}^* = \arg \max_{s_{1:t}} Pr(s_{1:t} | o_{1:t})$$
HMM

Given an HMM $\mathcal{H} = \langle T, O, \pi \rangle$, we are interested in two inference problems:

1. **Marginal Inference (Estimation problem).** Computing the marginal probability

   $$\Pr(o_{1:t}) = \sum_{s_{1:t}} \Pr(o_{1:t}, s_{1:t}) = \sum_{s_{1:t}} \Pr(s_{1:t}) \Pr(o_{1:t} | s_{1:t})$$

   **Dynamic Programming !**

2. **MAP Inference (Decoding problem).** Computing the sequence $s_{1:t}^*$ maximizing the posterior probability

   $$s_{1:t}^* = \arg \max_{s_{1:t}} \Pr(s_{1:t} | o_{1:t})$$

   **Viterbi Algorithm !**
HMM

Given an HMM $\mathcal{H} = \langle T, O, \pi \rangle$, we are interested in two inference problems:

1. Marginal Inference (Estimation problem). Computing the marginal probability

   $$\Pr(o_{1:t}) = \sum_{s_{1:t}} \Pr(o_{1:t}, s_{1:t}) = \sum_{s_{1:t}} \Pr(s_{1:t}) \Pr(o_{1:t} | s_{1:t})$$

   **Dynamic Programming**!

2. MAP Inference (Decoding problem). Computing the sequence $s_{1:t}^*$ maximizing the posterior probability

   $$s_{1:t}^* = \arg \max_{s_{1:t}} \Pr(s_{1:t} | o_{1:t})$$

   **Viterbi Algorithm**!
HMM

Given an HMM $\mathcal{H} = \langle T, O, \pi \rangle$, we are interested in two inference problems:

1. Marginal Inference (Estimation problem). Computing the marginal probability

$$\Pr(o_{1:t}) = \sum_{s_{1:t}} Pr(o_{1:t}, s_{1:t}) = \sum_{s_{1:t}} Pr(s_{1:t}) Pr(o_{1:t} \mid s_{1:t})$$

Dynamic Programming !

2. MAP Inference (Decoding problem). Computing the sequence $s^*_{1:t}$ maximizing the posterior probability

$$s^*_{1:t} = \arg \max_{s_{1:t}} \Pr(s_{1:t} \mid o_{1:t})$$

Viterbi Algorithm !

What about the learning problem?
Let $\mathcal{H} = \langle T, O, \pi \rangle$ be an HMM, define the following observable operators:

$$A_x \triangleq T \text{diag}(O_{x,1}, \ldots, O_{x,m}), \quad \forall x \in [n]$$

$\mathcal{H} = \langle \pi, A_x \rangle, \forall x \in [n]$ is an equivalent parameterization of HMM.
HMM Reparametrization

Let $\mathcal{H} = \langle T, O, \pi \rangle$ be an HMM, define the following observable operators:

$$A_x \triangleq T \text{diag}(O,x_1, \ldots, O,x_m), \quad \forall x \in [n]$$

$\mathcal{H} = \langle \pi, A_x \rangle, \forall x \in [n]$ is an equivalent parameterization of HMM.
HMM Reparametrization

Let $\mathcal{H} = \langle T, O, \pi \rangle$ be an HMM, define the following observable operators:

$$A_x \triangleq T \text{diag}(O_{x,1}, \ldots, O_{x,m}), \quad \forall x \in [n]$$

$\mathcal{H} = \langle \pi, A_x \rangle, \forall x \in [n]$ is an equivalent parameterization of HMM.

$$A_x[i,j] = \Pr(s_{t+1} = i | s_t = j) \times \Pr(o_t = x | s_t = j) = \Pr(s_{t+1} = i, o_t = x | s_t = j).$$
HMM Reparametrization

We can express the marginal probability in terms of observable operators:

\[
Pr(o_{1:t}) = \sum_{s_{1:t+1}} Pr(o_{1:t}, s_{1:t+1})
\]
HMM Reparametrization

We can express the marginal probability in terms of observable operators:

\[
\Pr(o_{1:t}) = \sum_{s_{1:t+1}} \Pr(o_{1:t}, s_{1:t+1}) \\
= \sum_{s_{1:t+1}} \left[ \Pr(s_{t+1} | s_t) \Pr(o_t | s_t) \right] \cdots \left[ \Pr(s_2 | s_1) \Pr(o_1 | s_1) \right] \Pr(s_1)
\]
HMM Reparametrization

We can express the marginal probability in terms of observable operators:

\[
\Pr(o_{1:t}) = \sum_{s_{1:t+1}} \Pr(o_{1:t}, s_{1:t+1}) = \sum_{s_{1:t+1}} \left[ \Pr(s_{t+1}|s_t) \Pr(o_t|s_t) \right] \cdots \left[ \Pr(s_2|s_1) \Pr(o_1|s_1) \right] \Pr(s_1) = \sum_{s_{1:t+1}} A_{o_t}[s_{t+1}, s_t] \cdots A_{o_1}[s_2, s_1] \pi_{s_1}
\]

Goal of Learning: Estimate the observable operators from sequence of observations.
HMM Reparametrization

We can express the marginal probability in terms of observable operators:

\[
\Pr(o_{1:t}) = \sum_{s_{1:t+1}} \Pr(o_{1:t}, s_{1:t+1}) \\
= \sum_{s_{1:t+1}} [\Pr(s_{t+1}|s_t) \Pr(o_t|s_t)] \cdots [\Pr(s_2|s_1) \Pr(o_1|s_1)] \Pr(s_1) \\
= \sum_{s_{1:t+1}} A_{o_t}[s_{t+1}, s_t] \cdots A_{o_1}[s_2, s_1] \pi_{s_1} \\
= 1^T A_{o_t} \cdots A_{o_1} \pi
\]
HMM Reparametrization

We can express the marginal probability in terms of observable operators:

\[
Pr(o_{1:t}) = \sum_{s_{1:t+1}} Pr(o_{1:t}, s_{1:t+1}) \\
= \sum_{s_{1:t+1}} [Pr(s_{t+1} | s_t) Pr(o_t | s_t)] \cdots [Pr(s_2 | s_1) Pr(o_1 | s_1)] Pr(s_1) \\
= \sum_{s_{1:t+1}} A_{ot}[s_{t+1}, s_t] \cdots A_{o1}[s_2, s_1] \pi_{s_1} \\
= 1^T A_{ot} \cdots A_{o1} \pi
\]

Goal of Learning: Estimate the observable operators from sequence of observations.
Spectral Learning for HMM [1]

Assumption 1: $\pi > 0$ element-wise, and $T$ and $O$ are full rank ($\text{rank}(T) = \text{rank}(O) = m$). Define the first three order moments of the observations:

$$P_1[i] = \Pr(x_1) = i$$

$$P_{2,1}[i, j] = \Pr(x_2 = i, x_1 = j)$$

$$P_{3,x,1}[i, j] = \Pr(x_3 = i, x_2 = x, x_1 = j), \forall x \in [n]$$
Spectral Learning for HMM [1]

Assumption 1: $\pi > 0$ element-wise, and $T$ and $O$ are full rank $(\text{rank}(T) = \text{rank}(O) = m)$. Define the first three order moments of the observations:

$$P_1[i] = \Pr(x_1) = i$$

$$P_{2,1}[i, j] = \Pr(x_2 = i, x_1 = j)$$

$$P_{3,x,1}[i, j] = \Pr(x_3 = i, x_2 = x, x_1 = j), \forall x \in [n]$$

Let $U \in \mathbb{R}^{n \times m}$ be the left singular matrix of $P_{2,1}$, define the following observable operators:

$$b_1 = U^T P_1$$

$$b_\infty = (P_{2,1}^T U)^+ P_1$$

$$B_x = (U^T P_{3,x,1})(U^T P_{2,1})^+, \forall x \in [n]$$

where $M^+$ denotes the Moore-Penrose pseudoinverse of matrix $M$. 
Theorem (Observable HMM Representation [1])

Assume the HMM obeys assumption 1, then

1. $b_1 = (U^T O)\pi$
2. $b_\infty^T = 1^T (U^T O)^{-1}$
3. $B_x = (U^T O)A_x (U^T O)^{-1}$ \quad $\forall x \in [n]$
4. $\Pr(o_{1:t}) = b_\infty^T B_{x_t} \cdots B_{x_1} b_1$
Spectral Learning for HMM [1]

Theorem (Observable HMM Representation [1])

Assume the HMM obeys assumption 1, then

1. \( b_1 = (U^T O)\pi \)
2. \( b_\infty^T = 1^T (U^T O)^{-1} \)
3. \( B_x = (U^T O)A_x(U^T O)^{-1} \quad \forall x \in [n] \)
4. \( \Pr(o_{1:t}) = b_\infty^T B_{x_t} \cdots B_{x_1} b_1 \)

\( b_1, b_\infty \) and \( B_x \) only depend on first three order moments of observations, free of hidden states!
Spectral Learning for HMM [1]

Main result of Spectral Learning algorithm for HMM:

**Theorem (Sample Complexity)**

There exists a constant \( C > 0 \) such that the following holds. Pick any \( 0 < \epsilon, \eta < 1 \) and \( t \geq 1 \). Assume the HMM obeys assumption 1, and

\[
N \geq C \cdot \frac{t^2}{\epsilon^2} \cdot \left( \frac{m \cdot \log(1/\epsilon)}{\sigma_m(O)^2 \sigma_m(P_{2,1})^4} + \frac{m \cdot n_0(\epsilon) \cdot \log(1/\epsilon)}{\sigma_m(O)^2 \sigma_m(P_{2,1})^2} \right)
\]

With probability at least \( 1 - \eta \), the model returned by the spectral learning algorithm for HMM satisfies

\[
\sum_{x_1, \ldots, x_t} |\Pr(x_1:t) - \hat{\Pr}(x_1:t)| \leq \epsilon
\]

where \( n_0(\epsilon) = \mathcal{O}(\epsilon^{1/(1-s)}) \), \( s > 1 \) a constant.
Compared with EM

Expectation-Maximization [9]:

- Local search heuristic algorithm based on the principle of Maximum Likelihood Estimation

For a given $t \geq 1$, and $0 < \epsilon, \eta < 1$, spectral learning algorithm:
Compared with EM

Expectation-Maximization [9]:

- Local search heuristic algorithm based on the principle of Maximum Likelihood Estimation
- Local optima problem.

For a given $t \geq 1$, and $0 < \epsilon, \eta < 1$, spectral learning algorithm:
Compared with EM

Expectation-Maximization [9]:

- Local search heuristic algorithm based on the principle of Maximum Likelihood Estimation
- Local optima problem.
- No consistency guarantees.

For a given $t \geq 1$, and $0 < \epsilon, \eta < 1$, spectral learning algorithm:
Compared with EM

Expectation-Maximization [9]:

- Local search heuristic algorithm based on the principle of Maximum Likelihood Estimation
- Local optima problem.
- No consistency guarantees.

For a given \( t \geq 1 \), and \( 0 < \epsilon, \eta < 1 \), spectral learning algorithm:

- A finite sample complexity to be consistent in terms of \( L_1 \) error on marginal probability.
Compared with EM

Expectation-Maximization [9]:

- Local search heuristic algorithm based on the principle of Maximum Likelihood Estimation
- Local optima problem.
- No consistency guarantees.

For a given \( t \geq 1 \), and \( 0 < \epsilon, \eta < 1 \), spectral learning algorithm:

- A finite sample complexity to be consistent in terms of \( L_1 \) error on marginal probability.
- No local optima since it only solves an SVD without any local search.
EM v.s. Spectral algorithm

Two synthetic experiments:

<table>
<thead>
<tr>
<th></th>
<th>SmallSyn</th>
<th>LargeSyn</th>
</tr>
</thead>
<tbody>
<tr>
<td># states</td>
<td>4</td>
<td>50</td>
</tr>
<tr>
<td># observations</td>
<td>8</td>
<td>100</td>
</tr>
<tr>
<td>test set size</td>
<td>4096</td>
<td>10,000</td>
</tr>
<tr>
<td>length of test sequence</td>
<td>4</td>
<td>50</td>
</tr>
</tbody>
</table>

Measure: normalized $L_1$ prediction error on test data set

$$L_1 = \sum_{x_{1:t} \in T} | \Pr(x_{1:t}) - \hat{\Pr}(x_{1:t}) |^{\frac{1}{t}}$$

where $T$ is the test set.
EM v.s. Spectral algorithm

**SmallSyn**

- **LearnHMM**
- **EM**

**LargeSyn**

- **LearnHMM**
- **EM**
EM v.s. Spectral algorithm

Negative probability problem with spectral learning algorithm:

- Size of training data.

\[
\text{Proportion of negative probabilities: } \frac{|\{ \hat{P}(x_1:t) < 0 | x_1:t \in T \}|}{|T|}
\]
EM v.s. Spectral algorithm

Negative probability problem with spectral learning algorithm:

- Size of training data.
- Estimation of rank hyperparameter.
**EM v.s. Spectral algorithm**

Negative probability problem with spectral learning algorithm:

- Size of training data.
- Estimation of rank hyperparameter.
- Length of test sequence.

<table>
<thead>
<tr>
<th>Training Size</th>
<th>Rank Hyperparameter</th>
<th>Proportion of Negative Probabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000</td>
<td>1</td>
<td>0.05</td>
</tr>
<tr>
<td>50000</td>
<td>1.5</td>
<td>0.1</td>
</tr>
<tr>
<td>100000</td>
<td>2</td>
<td>0.15</td>
</tr>
<tr>
<td>500000</td>
<td>2.5</td>
<td>0.2</td>
</tr>
<tr>
<td>1000000</td>
<td>3</td>
<td>0.25</td>
</tr>
</tbody>
</table>

**SmallSyn**

<table>
<thead>
<tr>
<th>Training Size</th>
<th>Rank Hyperparameter</th>
<th>Proportion of Negative Probabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000</td>
<td>1</td>
<td>0.05</td>
</tr>
<tr>
<td>50000</td>
<td>1.5</td>
<td>0.1</td>
</tr>
<tr>
<td>100000</td>
<td>2</td>
<td>0.15</td>
</tr>
<tr>
<td>500000</td>
<td>2.5</td>
<td>0.2</td>
</tr>
<tr>
<td>1000000</td>
<td>3</td>
<td>0.25</td>
</tr>
</tbody>
</table>

**LargeSyn**

<table>
<thead>
<tr>
<th>Training Size</th>
<th>Rank Hyperparameter</th>
<th>Proportion of Negative Probabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000</td>
<td>1</td>
<td>0.05</td>
</tr>
<tr>
<td>50000</td>
<td>1.5</td>
<td>0.1</td>
</tr>
<tr>
<td>100000</td>
<td>2</td>
<td>0.15</td>
</tr>
<tr>
<td>500000</td>
<td>2.5</td>
<td>0.2</td>
</tr>
<tr>
<td>1000000</td>
<td>3</td>
<td>0.25</td>
</tr>
</tbody>
</table>
EM v.s. Spectral algorithm

Negative probability problem with spectral learning algorithm:

- Size of training data.
- Estimation of rank hyperparameter.
- Length of test sequence.

Proportion of negative probabilities:

$$\text{NEG\_PROP} = \frac{\left\{ \hat{P}(x_{1:t}) < 0 \mid x_{1:t} \in \mathcal{T} \right\}}{\left| \mathcal{T} \right|}$$

---

SmallSyn

- Training Size = 10000
- Training Size = 50000
- Training Size = 100000
- Training Size = 500000
- Training Size = 1000000

LargeSyn

- Training Size = 10000
- Training Size = 50000
- Training Size = 100000
- Training Size = 500000
- Training Size = 1000000
Compared with EM

Why EM succeeds in practice?
If the log-likelihood function of model parameter tends to concave/quasi-concave when the sample size goes to infinity?
Compared with EM

Why EM succeeds in practice?
If the log-likelihood function of model parameter tends to concave/quasi-concave when the sample size goes to infinity?

1. Local search algorithms, for example, EM algorithm in our case, will converge to global optima, hence obtain the maximum likelihood estimator [10].
Compared with EM

Why EM succeeds in practice?
If the log-likelihood function of model parameter tends to concave/quasi-concave when the sample size goes to infinity?

1. Local search algorithms, for example, EM algorithm in our case, will converge to global optima, hence obtain the maximum likelihood estimator [10].

2. Consistency. Sequence of MLE converges in probability to the true model parameter (suppose the model is identifiable by parameter) [11].
Compared with EM

Why EM succeeds in practice?

If the log-likelihood function of model parameter tends to concave/quasi-concave when the sample size goes to infinity?

1. Local search algorithms, for example, EM algorithm in our case, will converge to global optima, hence obtain the maximum likelihood estimator [10].

2. Consistency. Sequence of MLE converges in probability to the true model parameter (suppose the model is identifiable by parameter) [11].

3. Asymptotic normality. The distribution of MLE tends to be a Gaussian distribution with mean the true parameter and covariance matrix equal to the inverse the Fisher information matrix, i.e., more and more concentrated [11].
Compared with EM

Why EM succeeds in practice?
If the log-likelihood function of model parameter tends to concave/quasi-concave when the sample size goes to infinity?

1. Local search algorithms, for example, EM algorithm in our case, will converge to global optima, hence obtain the maximum likelihood estimator [10].

2. Consistency. Sequence of MLE converges in probability to the true model parameter (suppose the model is identifiable by parameter) [11].

3. Asymptotic normality. The distribution of MLE tends to be a Gaussian distribution with mean the true parameter and covariance matrix equal to the inverse the Fisher information matrix, i.e., more and more concentrated [11].

Synthetic Experiment

Is our conjecture true in HMM? An HMM with one single parameter for visualization:

\[
\mathcal{H} = \langle T = \begin{pmatrix} \theta & 1 - \theta \\ 1 - \theta & \theta \end{pmatrix}, O = \begin{pmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{pmatrix}, \pi = (0.5, 0.5) \rangle
\]

Beta distribution with uniform distribution as prior.
Exact Bayesian updating with more and more observations.
Synthetic Experiment

The graph shows the (normalized) likelihood of θ for 10 observations. The likelihood peaks at some value of θ, indicating the most probable parameter value given the data.
Synthetic Experiment
Synthetic Experiment

![Graph showing normalized likelihoods for 10, 20, and 30 observations.](image-url)
Synthetic Experiment

The graph shows the (normalized) likelihood of different observations. The x-axis represents the parameter θ, and the y-axis represents the likelihood. There are four curves, each representing different numbers of observations: 10, 20, 30, and 40 observations. The curves indicate how the likelihood changes with different values of θ for each number of observations.
Synthetic Experiment

(normalized) likelihood

θ

10 observations
20 observations
30 observations
40 observations
50 observations
Synthetic Experiment

The plot shows the normalized likelihood as a function of \( \theta \) for different numbers of observations. The curves represent:

- 10 observations
- 20 observations
- 30 observations
- 40 observations
- 50 observations
- 60 observations

The likelihood peaks as the number of observations increases, indicating a stronger signal in the data.
Synthetic Experiment

(normalized) likelihood

θ

10 observations
20 observations
30 observations
40 observations
50 observations
60 observations
70 observations
Synthetic Experiment
Synthetic Experiment

(normalized) likelihood

θ

10 observations
20 observations
30 observations
40 observations
50 observations
60 observations
70 observations
80 observations
90 observations
Synthetic Experiment
Synthetic Experiment

Another small synthetic experiment: HMM with 2 states, 2 observations and 4 free parameters.
Synthetic Experiment

Another small synthetic experiment: HMM with 2 states, 2 observations and 4 free parameters.

Log-likelihood Comparison

- EM log-likelihood
- True log-likelihood

Training Size

Log-likelihood
Conclusions

Spectral learning for HMM

Pros:

1. Additive $L_1$ error bound with finite sample complexity.

Cons:
Conclusions

Spectral learning for HMM

Pros:

1. Additive $L_1$ error bound with finite sample complexity.
2. No local optima.

Cons:
Conclusions

Spectral learning for HMM

Pros:
1. Additive $L_1$ error bound with finite sample complexity.
2. No local optima.

Cons:
1. Negative probability.
Conclusions

Spectral learning for HMM

Pros:
1. Additive $L_1$ error bound with finite sample complexity.
2. No local optima.

Cons:
1. Negative probability.
2. Not most statistically efficient.
Conclusions

Spectral learning for HMM

Pros:
1. Additive $L_1$ error bound with finite sample complexity.
2. No local optima.

Cons:
1. Negative probability.
2. Not most statistically efficient.
3. Slow to converge.
Conclusions

EM for HMM

Pros:

1. Fast to converge.

Cons:
Conclusions

EM for HMM
Pro:
1. Fast to converge.
2. Statistically efficient.

Cons:
Conclusions

EM for HMM

Pros:

1. Fast to converge.
2. Statistically efficient.
3. Optimization based approach.

Cons:
Conclusions

EM for HMM

Pros:
1. Fast to converge.
2. Statistically efficient.
3. Optimization based approach.

Cons:
1. Local search heuristics, no provable guarantee for global optima.
Conclusions

EM for HMM
Pros:
1. Fast to converge.
2. Statistically efficient.
3. Optimization based approach.
Cons:
1. Local search heuristics, no provable guarantee for global optima.
2. Stuck in local optima for non-convex optimization.


