Weighted Spectral Learning and the Efficiency Sharpening algorithm

Michael Thon, Herbert Jaeger

Jacobs University Bremen

NIPS Workshop on Spectral Learning, Dec 10 2013
Who we are

- Michael Thon
  - PhD student of
- Herbert Jaeger:
  - **Observable operator models** (OOM) [Jaeger, 1998] = “observable representation for HMMs”

- Group:
  - Algebraic structure of models
  - Relation to **predictive state representations** (PSR) and **stochastic multiplicity automata** (SMA)
  - **Statistically efficient** learning algorithms

- Not:
  - Extension of models
  - Application to real-world problems
Outline

1. Basic Theory
2. Weighted spectral learning
3. Efficiency sharpening
\( f : \Sigma^* \to \mathbb{R} \)

\( \Sigma \) – finite alphabet
\( x, y, z \in \Sigma, \quad \bar{x}, \bar{y}, \bar{z} \in \Sigma^*, \quad \varepsilon \) – empty word

- **Stochastic process**
  - \( f(\varepsilon) = 1 \)
  - \( f(\bar{x}) = \sum_{z \in \Sigma} f(\bar{x}z) \)
  - \( f \geq 0 \)

- **Controlled stochastic process**
  - \( \Sigma = \Sigma_I \times \Sigma_O \)
  - \( f(\varepsilon) = 1 \)
  - \( \forall a \in \Sigma_I : f(\bar{x}) = \sum_{o \in \Sigma_o} f(\bar{x}ao) \)
  - \( f \geq 0 \)

- **Stochastic language**
  - \( f(\varepsilon) = 0 \)
  - \( \sum_{\bar{x} \in \Sigma^*} f(\bar{x}) = 1 \)
  - \( f \geq 0 \)
Sequential systems (SS) [Carlyle & Paz, 1971]

\[ f_{\bar{x}}(y) := f(xy) \]
\[ F = [f_{\bar{x}}(\bar{y})] - \text{Hankel matrix}^\top \quad \text{(columns indexed by } \bar{x} \in \Sigma^*) \]
\[ \mathcal{F} := \text{span}\{f_{\bar{x}}\} \]

Then

- \( \tilde{\tau}_z : \mathcal{F} \rightarrow \mathcal{F}, \quad f_{\bar{x}} \mapsto f_{\bar{x}z} \)
- \( \tilde{\sigma} : \mathcal{F} \rightarrow \mathbb{R}, \quad f_{\bar{x}} \mapsto f(\bar{x}) \)

are well-defined linear operators on \( \mathcal{F} \), and

\[
f(\bar{x}) = \tilde{\sigma} \tilde{\tau}_{x_n} \cdots \tilde{\tau}_{x_1} f_\varepsilon \]

If \( \dim(\mathcal{F}) < \infty \), then – w.r.t. some basis of \( \mathcal{F} \) – we get a sequential system representation \( \mathcal{S} = \langle \sigma, \{\tau_z\}, \omega_\varepsilon \rangle \) for \( f \).
Properties of sequential systems $S$

- **Equivalence of sequential systems**
  - $S \cong S'$: $f_S = f_{S'}$
  - $S$ is **minimal**: no equivalent SS of smaller dimension
  - Can minimize any $S$
  - For minimal SS, equivalence corresponds to a change of basis:
    for $\rho$ non-singular: $\rho S = (\sigma \rho^{-1}, \{\rho \tau_z \rho^{-1}\}, \rho \omega_\varepsilon)$

- For minimal $S$, can find $\{\overline{y}_1, \ldots, \overline{y}_d\}$ s.t. $\{f_{\overline{y}_1}, \ldots, f_{\overline{y}_d}\}$ is a basis.

- Then
  - $\rho = \begin{pmatrix} \sigma \tau_{\overline{y}_1} \\ \vdots \\ \sigma \tau_{\overline{y}_d} \end{pmatrix}$ is non-singular
  - $\rho S$ is **interpretable**, i.e., states $\omega_x = \tau_x \omega_\varepsilon = \begin{pmatrix} f_x(\overline{y}_1) \\ \vdots \\ f_x(\overline{y}_d) \end{pmatrix}$.
OOMs, PSRs and SMA

A sequential system that represents a

- stochastic language
  - is a stochastic multiplicity automata (SMA)
  - generalizes probabilistic finite automata (PFA)

- stochastic process
  - is an observable operator model (OOM)
  - generalizes hidden Markov models (HMM)

- controlled stochastic process
  - is a transformed predictive state representation (TPSR)
  - is an input-output OOM (IO-OOM)
  - is a predictive state representation (PSR) if it is interpretable
  - generalizes partially observable Markov decision processes (POMDP)
  - Note: (T)PSRs consider set \( \{ m_y = \sigma \tau_y \} \) of projection functions.
Learning sequential systems from data

Given estimates $\hat{f}(x)$, find $S$ such that $f_S \approx f$.

Recall:

$$\tilde{\tau}_z f_x = f_{xz}$$

- Gather estimates into Hankel matrix for sets $X, Y \subset \Sigma^*$:
  $$\hat{F} = \left[ \hat{f}(xy) \right]_{y \in Y, x \in X}, \quad \hat{F}_z = \left[ \hat{f}(xz) \right]_{y \in Y, x \in X}$$

1. Map columns to $d$-dimensional representation via matrix $C$

2. Solve $\hat{\tau}_z C\hat{F} = C\hat{F}_z$

   \[
   \text{and } \hat{\sigma} C\hat{F} = [\hat{f}(x)], \quad \hat{\omega}_\varepsilon = C[\hat{f}(y)]
   \]

   i.e., find $Q$ such that $C\hat{F}Q$ is invertible, e.g., $Q = (C\hat{F})^\dagger$

   and set

   $$\hat{\tau}_z = C\hat{F}_z Q (C\hat{F}Q)^{-1}$$

   $$\hat{\sigma} = [\hat{f}(x)] Q (C\hat{F}Q)^{-1}$$

   $$\hat{\omega}_\varepsilon = C[\hat{f}(y)]$$

   “learning equations”

   [Kretzschmar, 2001]
Learning sequential systems from data

Given estimates $\hat{f}(\overline{x})$, find $S$ such that $f_S \approx f$.

Recall:

$\tilde{\tau}_z f_{\overline{x}} = f_{\overline{x}z}$

- Gather estimates into Hankel matrix for sets $X, Y \subset \Sigma^*$:
  \[
  \hat{F} = \begin{bmatrix} \hat{f}(\overline{xy}) \end{bmatrix}_{\overline{y} \in Y, \overline{x} \in X}, \quad \hat{F}_z = \begin{bmatrix} \hat{f}(\overline{xz}y) \end{bmatrix}_{\overline{y} \in Y, \overline{x} \in X}
  \]

1. Map columns to $d$-dimensional representation via matrix $C$

2. Solve $\hat{\tau}_z C \hat{F} = C \hat{F}_z$ (and $\hat{\sigma} C \hat{F} = [\hat{f}(\overline{x})], \quad \hat{\omega}_\varepsilon = C[\hat{f}(\overline{y})]$)

i.e., find $Q$ such that $C \hat{F} Q$ is invertible, e.g., $Q = (C \hat{F})^\dagger$

\[
\begin{align*}
\hat{\sigma} &= [\hat{f}(\overline{x})]Q(C \hat{F} Q)^{-1} \\
\hat{\tau}_z &= C \hat{F}_z Q(C \hat{F} Q)^{-1} \\
\hat{\omega}_\varepsilon &= C[\hat{f}(\overline{y})]
\end{align*}
\]

“learning equations”

[Kretzschmar, 2001]
Spectral learning [Rosencrantz & al., 2004]

1. Find best rank-$d$ approximation to $\hat{F}$ [via $d$-truncated SVD]:
   \[ U_d S_d V_d^\top \approx \hat{F} \]
   - Map columns to $d$-dimensional representation via $C = U_d^\top$.
   - May select $d$ via threshold on singular values.

2. Select $Q = (C\hat{F})^\dagger = V_d (S_d)^\dagger$, i.e., solve learning equations in least squares sense.

Note:
- Can alternatively compute SVD of $\hat{F}_i = [\hat{F} \hat{F}_{z_1} \ldots \hat{F}_{z_n}]$.
- This turns out to be equivalent to the “error controlling” algorithm:
  \[
  \begin{cases}
  Q = (C\hat{F})^\dagger \\
  C = (\hat{F}Q)^\dagger 
  \end{cases}
  \] [Zhao & al., 2009]
Spectral learning [Rosencrantz & al., 2004]

1. Find best rank-\(d\) approximation to \(\hat{F}\) [via \(d\)-truncated SVD]:
\[
U_d S_d V_d^\top \approx \hat{F}
\]
- Map columns to \(d\)-dimensional representation via \(C = U_d^\top\).
- May select \(d\) via threshold on singular values.

2. Select \(Q = (C\hat{F})^\dagger = V_d (S_d)^\dagger\),
   i.e., solve learning equations in least squares sense.

Note:
- Can alternatively compute SVD of \(\hat{F} = [\hat{F} \hat{F}_{z_1} \ldots \hat{F}_{z_n}]\)
- This turns out to be equivalent to the “error controlling” algorithm:
  
  \[
  \begin{cases}
    Q = (C\hat{F})^\dagger \\
    C = (\hat{F}Q)^\dagger
  \end{cases}
  \]
  [Zhao & al., 2009]
Weighted spectral learning [Thon, in preparation]

- Take into account the precision of the estimates $\hat{f}(\bar{x})$
  - Weights $w_{\bar{x}} = \text{Var}[\hat{f}(\bar{x})]^{-1}$

1. Compute best weighted rank-$d$ approximation to $\hat{F}$:
   - $\hat{F} \approx BA$, where $B, A = \arg\min_{B, A} \|BA - \hat{F}\|_W$
   - $d$ columns of $B$ span column space of $\hat{F}$
   - Columns of $A = [A \ A_{z_1} \ldots A_{z_n}]$ give coordinates
   - Solve iteratively by fixing one $(A_i, B)$ and solving for other [MLPCA]

2. Solve learning equations by weighted regression or TLS
   - define $\hat{\tau}_* = \begin{bmatrix} \hat{\tau}_{z_1} \\ \vdots \\ \hat{\tau}_{z_n} \\ \hat{\sigma} \end{bmatrix}$ and $A_* = \begin{bmatrix} A_{z_1} \\ \vdots \\ A_{z_n} \end{bmatrix}$
   - $\hat{\tau}_* = \arg\min_{\hat{\tau}_*, E, E_*} \{ \|E\|_W^2 + \|E_*\|_{W_*}^2 : \hat{\tau}_*(A + E) = (A_* + E_*) \}$
Weighted spectral learning [Thon, in preparation]

- Take into account the precision of the estimates \( \hat{f}(\mathbf{x}) \)
  - Weights \( w_\mathbf{x} = \text{Var}[\hat{f}(\mathbf{x})]^{-1} \)

1. Compute best weighted rank-\(d\) approximation to \( \hat{F}_\mathbf{x} \):
   - \( \hat{F}_\mathbf{x} \approx BA_\mathbf{x} \), where \( B, A_\mathbf{x} = \text{argmin}_{B, A_\mathbf{x}} ||BA_\mathbf{x} - \hat{F}_\mathbf{x}||_W \)
   - \( d \) columns of \( B \) span column space of \( \hat{F}_\mathbf{x} \)
   - Columns of \( A_\mathbf{x} = [A \ A_{z1} \ldots A_{zn}] \) give coordinates
   - Solve iteratively by fixing one \((A_\mathbf{x}, B)\) and solving for other [MLPCA]

2. Solve learning equations by weighted regression or TLS

   - Define \( \hat{\tau}_\mathbf{x} = \begin{bmatrix} \hat{\tau}_{z1} \\ \vdots \\ \hat{\tau}_{zn} \end{bmatrix} \) and \( A_\mathbf{x} = \begin{bmatrix} A_{z1} \\ \vdots \\ A_{zn} \end{bmatrix} \)
   - \( \hat{\tau}_\mathbf{x} = \text{argmin}_{\hat{\tau}_\mathbf{x}, E, E_\mathbf{x}} \left\{ \|E\|_W^2 + \|E_\mathbf{x}\|_W^2 : \hat{\tau}_\mathbf{x} (A + E) = (A_\mathbf{x} + E_\mathbf{x}) \right\} \)
Weighted spectral learning [Thon, in preparation]

- Take into account the precision of the estimates $\hat{f}(x)$
  - Weights $w_x = \text{Var}[\hat{f}(x)]^{-1}$

1. Compute best weighted rank-$d$ approximation to $\hat{F}$:
   - $\hat{F} \approx BA$, where $B, A = \text{argmin}_{B,A} ||BA - \hat{F}||_W$
   - $d$ columns of $B$ span column space of $\hat{F}$
   - Columns of $A = [A \ A_{z_1} \ldots \ A_{z_n}]$ give coordinates
   - Solve iteratively by fixing one $(A, B)$ and solving for other [MLPCA]

2. Solve learning equations by weighted regression or TLS
   - define $\hat{\tau}_* = \begin{bmatrix} \hat{\tau}_{z_1} \\ \vdots \\ \hat{\tau}_{z_n} \\ \hat{\sigma} \end{bmatrix}$ and $A_* = \begin{bmatrix} A_{z_1} \\ \vdots \\ A_{z_n} \end{bmatrix}$
   - $\hat{\tau}_* = \text{argmin}_{\hat{\tau}_*, E, E_*} \{ ||E||_W^2 + ||E_*||_{W_*}^2 : \hat{\tau}_*(A + E) = (A_* + E_*) \}$
Remarks on weighted TLS

- See “Overview of TLS methods” [Markovsky & Van Huffel, 2007]

- Can also take into account structure of \( \hat{F} \):

- TLS and best low-rank matrix approximation are closely related:
  
  \[
  \hat{\tau}^* = \text{argmin}_{\hat{\tau}^*, E, E_*} \left\{ \| E \|^2_W + \| E_* \|^2_{W_*} : \hat{\tau}^* (A + E) = (A_* + E_*) \right\}
  \]

  \[
  \hat{\tau}^* (A + E) = (A_* + E_*) \text{ implies } \text{rank} \left( \begin{bmatrix} A + E \\ A_* + E_* \end{bmatrix} \right) = d
  \]

  \[
  \| E \|^2_W + \| E_* \|^2_{W_*} = \| \begin{bmatrix} A \\ A_* + E_* \end{bmatrix} - \begin{bmatrix} A + E \\ A_* + E_* \end{bmatrix} \|^2_{W_{W_*}}
  \]

  \[
  \text{Therefore } \begin{bmatrix} A + E \\ A_* + E_* \end{bmatrix} \text{ is the best } \begin{bmatrix} W \\ W_* \end{bmatrix}-\text{weighted rank-}d \text{ approximation to } \begin{bmatrix} A \\ A_* \end{bmatrix}
  \]

  and

  \[
  \hat{\tau}^* = U_* U^{-1}, \text{ where } \begin{bmatrix} U \\ U_* \end{bmatrix} V = \begin{bmatrix} A + E \\ A_* + E_* \end{bmatrix}
  \]
How to obtain weights

- Often $\hat{f}(x) = \frac{\#(x)}{N}$

- Use $\hat{\text{Var}}[\hat{f}(x)] = \frac{\hat{f}(x)(1-\hat{f}(x))}{N-1} \approx \frac{\hat{f}(x)}{N}$

- $w_x = \hat{\text{Var}}[\hat{f}(x)]^{-1}$

- Can use row/column weights for $\hat{F}$:
  - Row weights $w_Y = [w_y]_{y \in Y}$
  - Column weights $w_X^\top = [w_x]_{x \in X}$
  - $W = w_Y w_X^\top$, i.e., $w_{xy} = w_x w_y$

- Then:
  - Set $D_Y = \text{diag}(w_Y)^{\frac{1}{2}}$, $D_X = \text{diag}(w_X)^{\frac{1}{2}}$
  - $\|M\|_W = \|D_Y M D_X\|$
How to obtain weights

- Often $\hat{f}(\bar{x}) = \frac{\#(\bar{x})}{N}$

- Use $\hat{\text{Var}}[\hat{f}(\bar{x})] = \frac{\hat{f}(\bar{x})(1-\hat{f}(\bar{x}))}{N-1} \approx \frac{\hat{f}(\bar{x})}{N}$

- $w_{\bar{x}} = \hat{\text{Var}}[\hat{f}(\bar{x})]^{-1}$

- Can use row/column weights for $\hat{F}$:
  - Row weights $w_Y = [w_y]_{y \in Y}$
  - Column weights $w_X^\top = [w_{\bar{x}}]_{\bar{x} \in X}$
  - $W = w_Y w_X^\top$, i.e., $w_{\bar{x}y} = w_{\bar{x}} w_y$

- Then:
  - Set $D_Y = \text{diag}(w_Y)^{\frac{1}{2}}$, $D_X = \text{diag}(w_X)^{\frac{1}{2}}$
  - $\|M\|_W = \|D_Y M D_X\|$
Simplified row/column weighted spectral algorithm

- \( w_x = \hat{f}(\bar{x})^{-1} \), \( w_y = [w_y]_{y \in Y} \)
- \( w_X = [w_x]_{x \in X} \)
- \( W = w_Y w_X^\top \), \( D_Y = \text{diag}(w_Y)^{\frac{1}{2}} \), \( D_X = \text{diag}(w_X)^{\frac{1}{2}} \)

1. Compute best \( W \)-weighted rank-\( d \) approximation to \( \hat{F} \):
   - \( d \)-truncated SVD \( U_d S_d V_d^\top \) of \( D_Y \hat{F} D_X \)

2. Map columns to \( d \)-dimensional representation with \( C = U_d D_Y \)
   - Set \( A = C \hat{F} \) and \( A_z = C \hat{F}_z \)

2. Solve \( \hat{\tau}_* A \approx A_* \) by weighted TLS, where

\[
\hat{\tau}_* = \begin{bmatrix}
\hat{\tau}_{z_1} \\
\vdots \\
\hat{\tau}_{z_n} \\
\hat{\sigma}
\end{bmatrix}
\quad \text{and} \quad
A_* = \begin{bmatrix}
A_{z_1} \\
\vdots \\
A_{z_n} \\
\hat{f}_\epsilon^\top
\end{bmatrix}
\]
Simplified row/column weighted spectral algorithm

- Rows of $A = U_d^T D_Y \hat{F}$ and $A_z = U_d^T D_Y \hat{F}_z$ are already weighted
- Column weights $w_X$ for $A$ and $A_z$
- Multiply weights for $A_z$ by $w_z = \hat{f}(z)$
- Multiply $A$ by weight $\lambda$, e.g., $\lambda = 1$

2. For $\hat{\tau} A \approx A_*$, compute best rank-$d$ approximation to

$$\begin{bmatrix}
\lambda & A \\
 w_{z_1} & A_{z_1} \\
 \vdots \\
w_{z_n} & A_{z_n} \\
 \hat{f}^T_{\epsilon}
\end{bmatrix} D_X$$

by the $d$-truncated SVD

$$\begin{bmatrix}
U \\
U_{z_1} \\
\vdots \\
U_{z_n} \\
U_{\sigma}
\end{bmatrix} S_d V_d^T$$

Set

$$\hat{\sigma} = U_{\sigma} U^{-1} w_A$$
$$\hat{\tau}_z = w_z^{-1} U_z U^{-1} w_A$$
$$\hat{\omega}_\epsilon = \hat{\tau} \hat{\omega}_\epsilon \left[ = (US_d V_d^T)_1 \right]$$

where $\hat{\tau} = \sum_z \hat{\tau}_z$. 

Michael Thon (Jacobs University)
Weights and ES algorithm

NIPS Spectral Learning 14 / 19
Simplified row/column weighted spectral algorithm

- Rows of $A = U_d^T D Y \hat{F}$ and $A_z = U_d^T D Y \hat{F}_z$ are already weighted.
- Column weights $w_X$ for $A$ and $A_z$.
- Multiply weights for $A_z$ by $w_z = \hat{f}(z)$.
- Multiply $A$ by weight $\lambda$, e.g., $\lambda = 1$.

For $\hat{\tau} A \approx A_\tau$, compute best rank-$d$ approximation to

$$\begin{bmatrix} \lambda & A \\ w_{z_1} A_{z_1} & \vdots \\ w_{z_n} A_{z_n} & \hat{f}_\epsilon^T \end{bmatrix} D_X$$

by the $d$-truncated SVD

$$\begin{bmatrix} U \\ U_{z_1} \\ \vdots \\ U_{z_n} \\ U_\sigma \end{bmatrix} S_d V_d^T$$

Set

- $\hat{\sigma} = U_\sigma U^{-1} w_A$
- $\hat{\tau}_z = w_z^{-1} U_z U^{-1} w_A$
- $\hat{\omega}_\epsilon = \hat{\tau} \hat{\omega}_\epsilon$ [= $(US_d V_d^T)_1$]
Simplified row/column weighted spectral algorithm

- Rows of \( A = U_d^T D_Y \hat{F} \) and \( A_z = U_d^T D_Y \hat{F}_z \) are already weighted.
- Column weights \( w_X \) for \( A \) and \( A_z \).
- Multiply weights for \( A_z \) by \( w_z = \hat{f}(z) \).
- Multiply \( A \) by weight \( \lambda \), e.g., \( \lambda = 1 \).

For \( \hat{\tau}_* A \approx A_* \), compute best rank-\( d \) approximation to

\[
\begin{bmatrix}
\lambda & A \\
 w_{z_1} A_{z_1} & \\
 \vdots & \\
w_{z_n} A_{z_n} & \hat{f}_*^T \\
\end{bmatrix}
\]

by the \( d \)-truncated SVD

\[
D_X \quad \text{by the } d\text{-truncated SVD}
\]

\[
\begin{bmatrix}
U \\
U_{z_1} \\
\vdots \\
U_{z_n} \\
U_\sigma \\
\end{bmatrix}
\]

\[
S_d V_d^T
\]

Set

\[
\hat{\sigma} = U_\sigma U^{-1} w_A \\
\hat{\tau}_z = w_z^{-1} U_z U^{-1} w_A \\
\hat{\omega}_\varepsilon = \hat{\tau} \hat{\omega}_\varepsilon = (U S_d V_d^T)_1
\]

where \( \hat{\tau} = \sum_z \hat{\tau}_z \).
A simple demo

- Use “bible.txt” from the large Canterbury Corpus, reduced to $|\Sigma| = 27$

- Split into training sequence of length 3,831,102 and test sequence of length $2^{16}$

- Train OOMs (stochastic process models) using
  - Spectral algorithm
  - Simplified row/column weighted spectral algorithm
  - Efficiency sharpening algorithm

- Settings:
  - $X = Y = \Sigma^2$ (all words of length 2) excluding words that do not occur
  - Target dimension $d$ optimized via cross-validation

- Evaluate via average log-likelihood on test sequence
Results

Quality of learnt models for various training sequence lengths

-1.5
-2.0
-2.5
-3.0
-3.5

Average LL

Length of training sequence

$10^3$ $10^4$ $10^5$ $10^6$

Spectral
Weighted Spectral
Efficiency Sharpening
Results

Quality of learnt models for various training sequence lengths

- Average LL vs Length of training sequence
  - Spectral
  - Weighted Spectral
  - Efficiency Sharpening

Michael Thon (Jacobs University)
Efficiency sharpening [Jaeger & al., 2006]

- Assume:
  - A model $S = (\sigma, \\{\tau_z\}, \omega_\varepsilon)$ for $f$ is known
  - $\text{Var}[\hat{f}(\bar{x})] \approx K \cdot f(\bar{x})$
  - $f$ is a stationary and ergodic stochastic process
  - $X = \Sigma^l$ for some length $l$

- View learning equations as model estimator parameterized by $C$
- Select $C$ such that the variance of this estimator is minimized:
  - $C = \Pi^T D_Y^2$, where $\Pi = [\sigma \tau_{\bar{y}}]_{\bar{y} \in Y}$, $D_Y = \text{diag} \left( [f_S(\bar{y})]_{\bar{y} \in Y} \right)^{-\frac{1}{2}}$

- Select $Q$ to perform weighted regression
- Since $S$ is not known, use iterative procedure:
  - Compute $C$ from estimate $\hat{S}$
  - Set $Q = D_X (C \hat{F} D_X)^\dagger$, for $D_X = \text{diag} \left( [\hat{f}(\bar{x})]_{\bar{x} \in X} \right)^{-\frac{1}{2}}$
  - Compute new estimate $\hat{S}$ via learning equations
Efficiency sharpening

- Estimates the principal subspace of the Hankel matrix $\hat{F}$ from a previous model estimate.
- Uses row and column weights.
- Gives good results after few iterations.
- Can avoid computation of $\hat{F}$. Instead, $C\hat{F}$ and $C\hat{F}_z$ can be approximated from a suffix tree representation of the input data.
- Allows to effectively use $Y = \Sigma^*$ and optimize $X \subset \Sigma^*$:

$$X = \left\{ \bar{x} \in \Sigma^* \mid \begin{array}{l} l_{\text{min}} \leq |\bar{x}| \leq l_{\text{max}}, \\
\#(\bar{x}) > c_{\text{min}}, \\
\bar{x} \text{ has unique continuation statistics} \end{array} \right\}$$
(IO)-OOMs, (T)PSRs and SMA are SS and share the same theory

- Weights can improve the spectral learning algorithms

- Efficiency sharpening estimates the principal subspace of the Hankel matrix and weights from a previous model estimate
Conclusion

- (IO)-OOMs, (T)PSRs and SMA are SS and share the same theory
- Weights can improve the spectral learning algorithms
- Efficiency sharpening estimates the principal subspace of the Hankel matrix and weights from a previous model estimate

Thank you!
References


[Thon, in preparation] PhD Thesis *Jacobs University Bremen, Germany*