# Weighted Spectral Learning and the Efficiency Sharpening algorithm

Michael Thon, Herbert Jaeger

Jacobs University Bremen

NIPS Workshop on Spectral Learning, Dec 10 2013

JACOBS
UNIVERSITY

# 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

JACOBS
UNIVERSITY

# Outline

1. Basic Theory
2. Weighted spectral learning
3. Efficiency sharpening

JACOBS
UNIVERSITY

$f : \Sigma^* \to \mathbb{R}$

$\Sigma$ – finite alphabet

$x, y, z \in \Sigma, \quad \overline{x}, \overline{y}, \overline{z} \in \Sigma^*, \quad \varepsilon$ – empty word

- Stochastic process
  - $f(\varepsilon) = 1$
  - $f(\overline{x}) = \sum_{z \in \Sigma} f(\overline{x}z)$
  - $f \geq 0$
- Controlled stochastic process
  - $\Sigma = \Sigma_I \times \Sigma_O$
  - $f(\varepsilon) = 1$
  - $\forall a \in \Sigma_I : \; f(\overline{x}) = \sum_{o \in \Sigma_o} f(\overline{x}ao)$
  - $f \geq 0$
- Stochastic language
  - $f(\varepsilon) = 0$
  - $\sum_{\overline{x} \in \Sigma^*} f(\overline{x}) = 1$
  - $f \geq 0$

JACOBS UNIVERSITY

# Sequential systems (SS) [Carlyle & Paz, 1971]

$f_{\overline{x}}(\overline{y}) := f(\overline{xy})$
$F = [f_{\overline{x}}(\overline{y})]$ – Hankel matrix$^\top$ (columns indexed by $\overline{x} \in \Sigma^*$)
$\mathcal{F} := \mathrm{span}\{f_{\overline{x}}\}$

Then

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

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

$$f(\overline{x}) = \tilde{\sigma}\, \underbrace{\tilde{\tau}_{x_n} \cdots \tilde{\tau}_{x_1} f_\varepsilon}_{f_{\overline{x}}}$$

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

JACOBS UNIVERSITY

# Properties of sequential systems $\mathcal{S}$

- Equivalence of sequential systems
  - $\mathcal{S} \cong \mathcal{S}'$: $f_{\mathcal{S}} = f_{\mathcal{S}'}$
  - $\mathcal{S}$ is minimal: no equivalent SS of smaller dimension
  - Can minimize any $\mathcal{S}$
  - For minimal SS, equivalence corresponds to a change of basis: for $\rho$ non-singular: $\rho\mathcal{S} = (\sigma\rho^{-1}, \{\rho\tau_z\rho^{-1}\}, \rho\omega_\varepsilon)$
- For minimal $\mathcal{S}$, 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
  - $\rho\mathcal{S}$ is interpretable, i.e., states $\omega_{\overline{x}} = \tau_{\overline{x}}\omega_\varepsilon = \begin{pmatrix} f_{\overline{x}}(\overline{y}_1) \\ \vdots \\ f_{\overline{x}}(\overline{y}_d) \end{pmatrix}$.

JACOBS UNIVERSITY

# 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 $\{m_{\overline{y}} = \sigma\tau_{\overline{y}}\}$ of projection functions.

JACOBS UNIVERSITY

# Learning sequential systems from data

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

**Recall:**

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

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

1. Map columns to $d$-dimensional representation via matrix $C$
2. Solve $\hat{\tau}_z C\hat{F} = C\hat{F}_z$ $\left(\text{ and } \hat{\sigma}C\hat{F} = [\hat{f}(\overline{x})],\ \hat{\omega}_\varepsilon = C[\hat{f}(\overline{y})]\right)$

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

and set

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

"learning equations"

[Kretzschmar, 2001]

JACOBS UNIVERSITY

# Learning sequential systems from data

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

**Recall:**

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

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

1. Map columns to $d$-dimensional representation via matrix $C$
2. Solve $\hat{\tau}_z C\hat{F} = C\hat{F}_z$ $\left(\text{ and } \hat{\sigma} C\hat{F} = [\hat{f}(\overline{x})],\ \hat{\omega}_\varepsilon = C[\hat{f}(\overline{y})]\right)$

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

and set

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

"learning equations"

[Kretzschmar, 2001]

JACOBS UNIVERSITY

# 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:

$$\text{iterate} \begin{cases} Q = (C\hat{F})^\dagger \\ C = (\hat{F}Q)^\dagger \end{cases} \qquad \text{[Zhao \& al., 2009]}$$

JACOBS UNIVERSITY

# 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:

iterate $\begin{cases} Q = (C\hat{F})^\dagger \\ C = (\hat{F}Q)^\dagger \end{cases}$ [Zhao & al., 2009]

JACOBS UNIVERSITY

# Weighted spectral learning [Thon, in preparation]

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

1. Compute best weighted rank-$d$ approximation to $\hat{F}_{:}$
   - $\hat{F}_{:} \approx BA_{:}$, where $B, A_{:} = \underset{B,A_{:}}{\mathrm{argmin}} \|BA_{:} - \hat{F}_{:}\|_W$
   - $d$ 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 $A_* = \begin{bmatrix} A_{z_1} \\ \vdots \\ A_{z_n} \\ \hat{f}_{\varepsilon}^{\top} \end{bmatrix}$
   - $\hat{\tau}_* = \underset{\hat{\tau}_*,E,E_*}{\mathrm{argmin}} \left\{ \|E\|_W^2 + \|E_*\|_{W_*}^2 : \hat{\tau}_*(A+E) = (A_* + E_*) \right\}$

JACOBS UNIVERSITY

# Weighted spectral learning [Thon, in preparation]

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

1. Compute best weighted rank-$d$ approximation to $\hat{F}_{:}$
   - $\hat{F}_{:} \approx BA_{:}$, where $B, A_{:} = \underset{B,A_{:}}{\mathrm{argmin}} \|BA_{:} - \hat{F}_{:}\|_W$
   - $d$ 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 $A_* = \begin{bmatrix} A_{z_1} \\ \vdots \\ A_{z_n} \\ f_e^T \end{bmatrix}$
   - $\hat{\tau}_* = \underset{\hat{\tau}_*, E, E_*}{\mathrm{argmin}} \{\|E\|_W^2 + \|E_*\|_{W_*}^2 : \hat{\tau}_*(A+E) = (A_*+E_*)\}$

JACOBS UNIVERSITY

# Weighted spectral learning [Thon, in preparation]

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

1. Compute best weighted rank-$d$ approximation to $\hat{F}_{:}$
   - $\hat{F}_{:} \approx BA_{:}$, where $B, A_{:} = \underset{B,A_{:}}{\mathrm{argmin}} \|BA_{:} - \hat{F}_{:}\|_W$
   - $d$ 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 $A_* = \begin{bmatrix} A_{z_1} \\ \vdots \\ A_{z_n} \\ \hat{f}_{\varepsilon}^{\top} \end{bmatrix}$
   - $\hat{\tau}_* = \underset{\hat{\tau}_*,E,E_*}{\mathrm{argmin}} \left\{ \|E\|_W^2 + \|E_*\|_{W_*}^2 : \hat{\tau}_*(A+E) = (A_* + E_*) \right\}$

JACOBS UNIVERSITY

# 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:
  - $\hat{\tau}_* = \underset{\hat{\tau}_*, E, E_*}{\operatorname{argmin}} \left\{ \|E\|_W^2 + \|E_*\|_{W_*}^2 : \hat{\tau}_*(A+E) = (A_* + E_*) \right\}$
  - Note:

    $\hat{\tau}_*(A+E) = (A_* + E_*)$ implies $\operatorname{rank}\left(\left[\begin{smallmatrix} A+E \\ A_*+E_* \end{smallmatrix}\right]\right) = d$

    $\|E\|_W^2 + \|E_*\|_{W_*}^2 = \left\| \left[\begin{smallmatrix} A \\ A_* \end{smallmatrix}\right] - \left[\begin{smallmatrix} A+E \\ A_*+E_* \end{smallmatrix}\right] \right\|^2_{\left[\begin{smallmatrix} W \\ W_* \end{smallmatrix}\right]}$
  - Therefore

    $\left[\begin{smallmatrix} A+E \\ A_*+E_* \end{smallmatrix}\right]$ is the best $\left[\begin{smallmatrix} W \\ W_* \end{smallmatrix}\right]$-weighted rank-$d$ approximation to $\left[\begin{smallmatrix} A \\ A_* \end{smallmatrix}\right]$

    and

    $\hat{\tau}_* = U_* U^{-1}$, where $\left[\begin{smallmatrix} U \\ U_* \end{smallmatrix}\right] V = \left[\begin{smallmatrix} A+E \\ A_*+E_* \end{smallmatrix}\right]$.

# How to obtain weights

- Often $\hat{f}(\overline{x}) = \frac{\#(\overline{x})}{N}$
- Use $\hat{\mathrm{Var}}[\hat{f}(\overline{x})] = \frac{\hat{f}(\overline{x})(1-\hat{f}(\overline{x}))}{N-1} \approx \frac{\hat{f}(\overline{x})}{N}$
- $w_{\overline{x}} = \hat{\mathrm{Var}}[\hat{f}(\overline{x})]^{-1}$
- Can use row/column weights for $\hat{F}$:
  - Row weights $w_Y = [w_{\overline{y}}]_{\overline{y} \in Y}$
  - Column weights $w_X^\top = [w_{\overline{x}}]_{\overline{x} \in X}$
  - $W = w_Y w_X^\top$, i.e., $w_{\overline{xy}} = w_{\overline{x}} w_{\overline{y}}$
- Then:
  - Set $D_Y = \mathrm{diag}(w_Y)^{\frac{1}{2}}, \quad D_X = \mathrm{diag}(w_X)^{\frac{1}{2}}$
  - $\|M\|_W = \|D_Y M D_X\|$

JACOBS UNIVERSITY

# How to obtain weights

- Often $\hat{f}(\overline{x}) = \frac{\#(\overline{x})}{N}$
- Use $\hat{\mathrm{Var}}[\hat{f}(\overline{x})] = \frac{\hat{f}(\overline{x})(1-\hat{f}(\overline{x}))}{N-1} \approx \frac{\hat{f}(\overline{x})}{N}$
- $w_{\overline{x}} = \hat{\mathrm{Var}}[\hat{f}(\overline{x})]^{-1}$
- Can use row/column weights for $\hat{F}$:
  - Row weights $w_Y = [w_{\overline{y}}]_{\overline{y} \in Y}$
  - Column weights $w_X^\top = [w_{\overline{x}}]_{\overline{x} \in X}$
  - $W = w_Y w_X^\top$, i.e., $w_{\overline{xy}} = w_{\overline{x}} w_{\overline{y}}$
- Then:
  - Set $D_Y = \mathrm{diag}(w_Y)^{\frac{1}{2}}, \quad D_X = \mathrm{diag}(w_X)^{\frac{1}{2}}$
  - $\|M\|_W = \|D_Y M D_X\|$

JACOBS UNIVERSITY

# Simplified row/column weighted spectral algorithm

- $w_{\overline{x}} = \hat{f}(\overline{x})^{-1}$, $w_Y = [w_{\overline{y}}]_{\overline{y} \in Y}$ $w_X^\top = [w_{\overline{x}}]_{\overline{x} \in X}$
  $W = w_Y w_X^\top$, $D_Y = \mathrm{diag}(w_Y)^{\frac{1}{2}}$, $D_X = \mathrm{diag}(w_X)^{\frac{1}{2}}$

1. Compute best $W$-weighted rank-$d$ approximation to $\hat{F}$:

   $d$-truncated SVD $U_d S_d V_d^\top$ of $D_Y \hat{F} D_X$

   - Map columns to $d$-dimensional representation with $C = U_d^\top D_Y$
     - Set $A = C\hat{F}$ and $A_z = C\hat{F}_z$

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

$$\hat{\tau}_* = \begin{bmatrix} \hat{\tau}_{z_1} \\ \vdots \\ \hat{\tau}_{z_n} \\ \hat{\sigma} \end{bmatrix} \quad \text{and} \quad A_* = \begin{bmatrix} A_{z_1} \\ \vdots \\ A_{z_n} \\ \hat{f}_\varepsilon^\top \end{bmatrix}$$

JACOBS UNIVERSITY

# Simplified row/column weighted spectral algorithm

- Rows of $A = U_d^\top D_Y \hat{F}$ and $A_z = U_d^\top D_Y \hat{F}_z$ 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

$$\begin{bmatrix} \lambda & A \\ w_{z_1} & A_{z_1} \\ & \vdots \\ w_{z_n} & A_{z_n} \\ & \hat{f}_\varepsilon^\top \end{bmatrix} D_X \quad \text{by the } d\text{-truncated SVD} \quad \begin{bmatrix} U \\ U_{z_1} \\ \vdots \\ U_{z_n} \\ U_\sigma \end{bmatrix} S_d V_d^\top$$

- Set $\hat{\sigma} = U_\sigma U^{-1} w_A$, $\hat{\tau}_z = w_z^{-1} U_z U^{-1} w_A$, where $\hat{\tau} = \sum_z \hat{\tau}_z$, $\hat{\omega}_\varepsilon = \hat{\tau}\hat{\omega}_\varepsilon \; [= (U S_d V_d^\top)_1]$.

JACOBS UNIVERSITY

# Simplified row/column weighted spectral algorithm

- Rows of $A = U_d^\top D_Y \hat{F}$ and $A_z = U_d^\top D_Y \hat{F}_z$ 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

$$\begin{bmatrix} \lambda & A \\ w_{z_1} & A_{z_1} \\ & \vdots \\ w_{z_n} & A_{z_n} \\ & \hat{f}_\varepsilon^\top \end{bmatrix} D_X \quad \text{by the } d\text{-truncated SVD} \quad \begin{bmatrix} U \\ U_{z_1} \\ \vdots \\ U_{z_n} \\ U_\sigma \end{bmatrix} S_d V_d^\top$$

- Set $\hat{\sigma} = U_\sigma U^{-1} w_A$, $\hat{\tau}_z = w_z^{-1} U_z U^{-1} w_A$, where $\hat{\tau} = \sum_z \hat{\tau}_z$, $\hat{\omega}_\varepsilon = \hat{\tau}\hat{\omega}_\varepsilon \; [= (U S_d V_d^\top)_1]$.

JACOBS UNIVERSITY

# Simplified row/column weighted spectral algorithm

- Rows of $A = U_d^\top D_Y \hat{F}$ and $A_z = U_d^\top D_Y \hat{F}_z$ 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

$$\begin{bmatrix} \lambda & A \\ w_{z_1} & A_{z_1} \\ & \vdots \\ w_{z_n} & A_{z_n} \\ & \hat{f}_\varepsilon^\top \end{bmatrix} D_X \quad \text{by the } d\text{-truncated SVD} \quad \begin{bmatrix} U \\ U_{z_1} \\ \vdots \\ U_{z_n} \\ U_\sigma \end{bmatrix} S_d V_d^\top$$

- Set

$$\begin{aligned} \hat{\sigma} &= U_\sigma U^{-1} w_A \\ \hat{\tau}_z &= w_z^{-1} U_z U^{-1} w_A \quad , \quad \text{where } \hat{\tau} = \textstyle\sum_z \hat{\tau}_z \\ \hat{\omega}_\varepsilon &= \hat{\tau} \hat{\omega}_\varepsilon \; [= (U S_d V_d^\top)_1] \end{aligned}$$

JACOBS UNIVERSITY

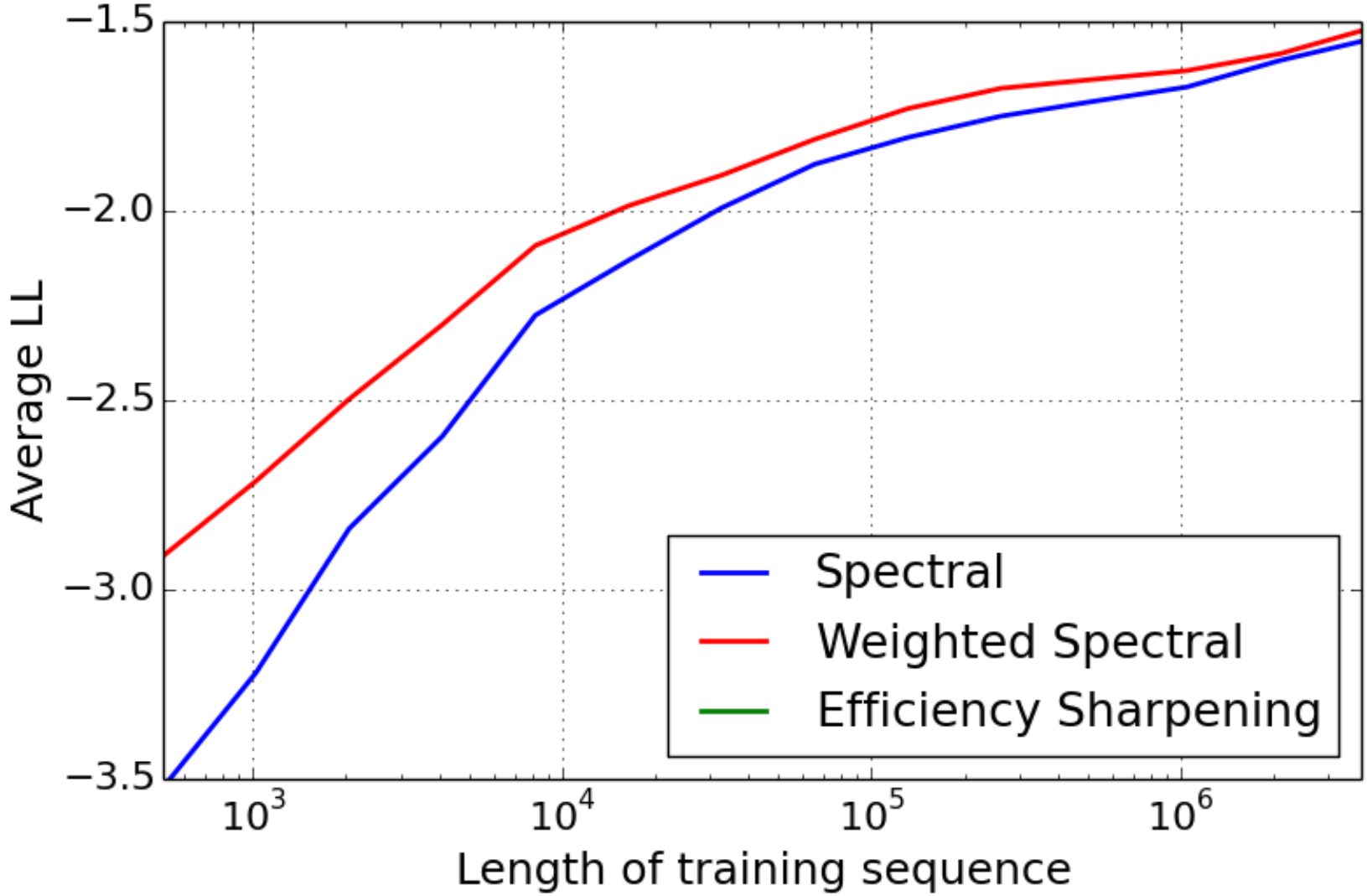
# A simple demo

- 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

JACOBS UNIVERSITY

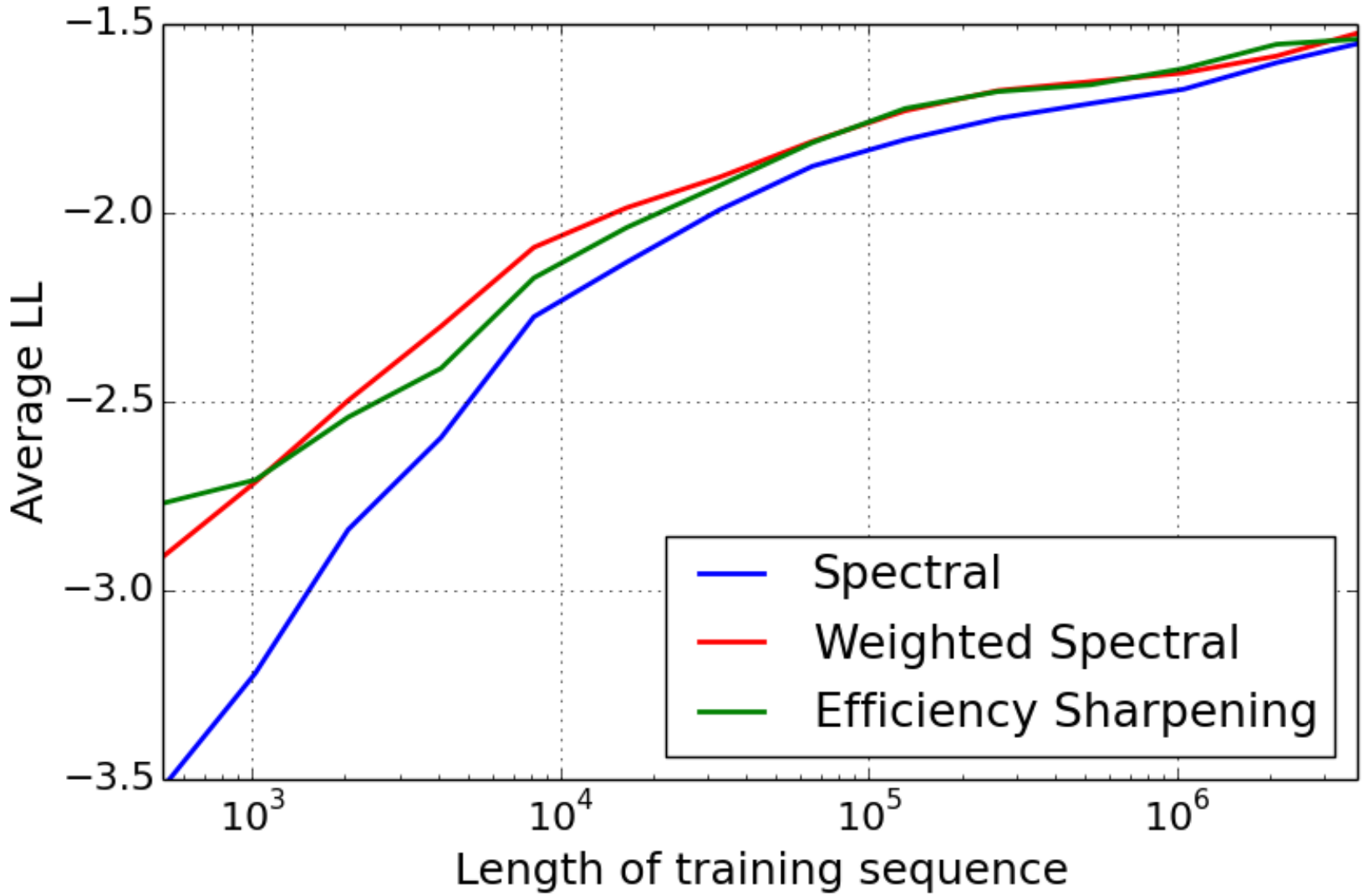
# Results

Quality of learnt models for various training sequence lengths



# Results

Quality of learnt models for various training sequence lengths



# Efficiency sharpening [Jaeger & al., 2006]

- Assume:
  - A model $\mathcal{S} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ for $f$ is known
  - [ $\mathrm{Var}[\hat{f}(\overline{x})] \approx K \cdot f(\overline{x})$ ]
  - [ $f$ 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:
  - $\boxed{C = \Pi^\top D_Y^2,}$ where $\Pi = [\sigma\tau_{\overline{y}}]_{\overline{y}\in Y}$, $D_Y = \mathrm{diag}\left([f_{\mathcal{S}}(\overline{y})]_{\overline{y}\in Y}\right)^{-\frac{1}{2}}$
- Select $Q$ to perform weighted regression
- Since $\mathcal{S}$ is not known, use iterative procedure:
  - Compute $C$ from estimate $\hat{\mathcal{S}}$
  - Set $\boxed{Q = D_X(C\hat{F}D_X)^\dagger,}$ for $D_X = \mathrm{diag}\left(\left[\hat{f}(\overline{x})\right]_{\overline{x}\in X}\right)^{-\frac{1}{2}}$
  - Compute new estimate $\hat{\mathcal{S}}$ via learning equations

JACOBS UNIVERSITY

# 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 $X \subset \Sigma^*$:
  - $$X = \left\{ \overline{x} \in \Sigma^* : \begin{array}{l} l_{\min} \leq |\overline{x}| \leq l_{\max}, \\ \#(\overline{x}) > c_{\min}, \\ \overline{x} \text{ has unique continuation statistics} \end{array} \right\}$$

JACOBS UNIVERSITY

# 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

JACOBS
UNIVERSITY

# 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!

JACOBS UNIVERSITY

# References


[Carlyle & Paz, 1971] Realizations by stochastic finite automata. *Journal of Computer and System Sciences*, 5(1):26–40

[Jaeger, 1998] Discrete-time, discrete-valued observable operator models: a tutorial. *Technical Report 42, GMD Sankt Augustin, Germany*

[Kretzschmar, 2001] Learning symbol sequences with observable operator models. *Technical Report 161, GMD Sankt Augustin, Germany*

[Rosencrantz & al., 2004] Learning low dimensional predictive representations. *ICML 2004*, 695–702

[Jaeger & al., 2006] Learning observable operator models via the ES algorithm. *In: New Directions in Statistical Signal Processing: From Systems to Brains*

[Markovsky & Van Huffel, 2007] Overview of total least-squares methods. *Signal Processing*, 87(10):2283–2302

[Zhao & al., 2009] A bound on modeling error in observable operator models and an associated learning algorithm. *Neural Computation*, 21(9):2687–2712

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


JACOBS UNIVERSITY