Journal of Machine Learning Research 16 (2015) 103-147 Submitted 6/11; Revised 7/14; Published 1/15

Links Between Multiplicity Automata, Observable Operator Models and Predictive State Representations — a Unified Learning Framework

Michael Thon M.THON@JACOBS-UNIVERSITY.DE
Herbert Jaeger H.JAEGER@JACOBS-UNIVERSITY.DE
Jacobs University Bremen
28759 Bremen, Germany

Editor: Joelle Pineau

Abstract

Stochastic multiplicity automata (SMA) are weighted finite automata that generalize probabilistic automata. They have been used in the context of probabilistic grammatical inference. Observable operator models (OOMs) are a generalization of hidden Markov models, which in turn are models for discrete-valued stochastic processes and are used ubiquitously in the context of speech recognition and bio-sequence modeling. Predictive state representations (PSRs) extend OOMs to stochastic input-output systems and are employed in the context of agent modeling and planning.

We present SMA, OOMs, and PSRs under the common framework of sequential systems, which are an algebraic characterization of multiplicity automata, and examine the precise relationships between them. Furthermore, we establish a unified approach to learning such models from data. Many of the learning algorithms that have been proposed can be understood as variations of this basic learning scheme, and several turn out to be closely related to each other, or even equivalent.

Keywords: multiplicity automata, hidden Markov models, observable operator models, predictive state representations, spectral learning algorithms

1. Introduction

Multiplicity automata (MA) (Schützenberger, 1961) are weighted nondeterministic automata which generalize both finite and probabilistic automata. The discovery that MA are efficiently learnable (Bergadano and Varricchio, 1994; Ohnishi et al., 1994) in the exact learning model of Angluin (Angluin, 1987) sparked an interest in these, and several versions have been studied. One such version is stochastic multiplicity automata (SMA), which model rational stochastic languages and have been used in the context of probabilistic grammatical inference (Denis et al., 2006; Bailly et al., 2009). Independent of this line of research, hidden Markov models (HMMs) (see Rabiner, 1989) for discrete-valued stochastic processes have been extensively studied and are now a standard tool in many pattern recognition domains such as speech recognition, natural language processing and bio-sequence modeling. Observable operator models (OOMs) are a generalization of HMMs that was introduced by Jaeger (1998) following previous work on deciding the equivalence of HMMs (Ito et al., 1992). Finally, predictive state representations (PSRs) are mod-

els for stochastic input-output systems developed by Littman, Sutton, and Singh (2001) and inspired by OOMs. PSRs generalize partially observable Markov decision processes (POMDPs) (Kaelbling et al., 1998) and have been used in the context of agent modeling and planning (James et al., 2004; James and Singh, 2005; Wolfe and Singh, 2006; Boots et al., 2010). As it turns out, all of these models are instances of MA and thereby closely related, though this is not widely perceived, due in part to the disjoint scientific communities.

All of SMA, OOMs and PSRs model some form of probability distribution. A central task common to all cases is therefore to estimate a model from a given sample. This may also be referred to as learning, system identification or model induction depending on the context.

In this paper we present SMA, OOMs, and PSRs under a common framework and examine the precise relationships between them. Furthermore, we establish a unified approach to learning such models from data. Many of the learning algorithms that have been proposed can be understood as variations of this basic learning theme, and several turn out to be closely related or even equivalent.

In Section 2 we cover the essential theory for sequential systems (SSs) — a term coined by Carlyle and Paz (1971) for a purely algebraic characterization of MA. Though not new, we present this theory in a way that can be readily turned into algorithms, and with full proofs, because they give much insight and pave the way to the presented learning approach. The first result concerns the relationship between SSs and the objects that they describe, namely *formal series* $f : \Sigma^* \to K$ for $K = \mathbb{R}$ or $K = \mathbb{C}$ (see Section 1.1 for details). Any such function can be associated with a linear function space $\mathcal{F}$, and has a SS representation if and only if the space $\mathcal{F}$ is finite dimensional. In fact, a SS can be seen as a representation of f w.r.t. some basis of $\mathcal{F}$, and a change of basis will correspond to an equivalence transformation of SSs, where equivalence of two SSs means that they represent the same function. The remaining theory will be concerned with such transformations of SSs. It is shown how to transform any SS to an equivalent minimal SS, how to decide equivalence, how to normalize SSs and how to convert SSs into a so-called "interpretable" form.

In Section 3 we mention the relationship between MA and the more general class of weighted finite automata (WFA) and their extension to input-output systems called weighted finite-state transducers (WFST). We then present SMA, OOMs and PSRs as instances of SSs with specific additional constraints that model probabilistic languages, stochastic processes and controlled processes, respectively, via the formal series f that they describe. We only sketch the basic concepts and give pointers to relevant literature. The main emphasis is on exploring the relations among the various model classes. We show that SMA are related to OOMs in the same way that probabilistic finite automata are related to HMMs, and show how to trivially convert any HMM into an OOM. OOMs and PSRs share the notion of a "predictive state" for the modeled process, which can be either implicit (as in the case of OOMs) or explicit (as for PSRs). Any PSR is essentially an input-output (IO)-OOM, while any OOM can be converted to a PSR by making the state "interpretable". Finally, PSRs generalize POMDPs in the same way that OOMs generalize HMMs.

Section 4 on learning is the main technical contribution of this paper. We present a learning framework that covers the cases of SMA, OOMs and PSRs in a unified way. The

only difference for the model classes concerns the way that estimates are obtained from the sample data. To turn the learning framework into a concrete algorithm, several design choices need to be made. Depending on these, many algorithms that have been proposed in the literature are recovered. This unified viewpoint has several advantages. First of all, modifications and improvements made for a specific model class can be generalized to other learning algorithms. Additionally, the general learning framework allows us to identify the key points responsible for statistical efficiency and thereby indicates a clear path for improvements. In this section, we present generalized and simplified versions of two key OOM learning algorithms — error controlling (EC) and efficiency sharpening (ES) — and show that these are in fact closely related to spectral learning algorithms.

1.1 Notation

Let Σ^* be the set of words over a finite alphabet Σ, including the empty word ε. Symbols from the alphabet Σ will be denoted by normal variables as in $x, y \in \Sigma$, while words will be denoted by variables with a bar over them, e.g., $\overline{x}, \overline{y} \in \Sigma^*$. For $\overline{x}$ and $\overline{y}$ in Σ^*, let $\overline{xy}$ be the concatenation of words, and $|\overline{x}|$ denote the length of the word $\overline{x}$. Furthermore, let Σ^k denote the subset of words of length k. Let $\{\overline{x}_i \,|\, i \in \mathbb{N}\} = \Sigma^*$ be an enumeration of Σ^* such that $\overline{x}_0 = \varepsilon$. We will be interested in characterizing functions $f : \Sigma^* \to K$ for $K = \mathbb{R}$ or $K = \mathbb{C}$, since these can be used to describe probabilistic languages, stochastic processes and controlled processes (cf. Definitions 18, 20, and 28). These form a K-vector space which we denote by $K\langle\langle\Sigma\rangle\rangle$. For a given function $f : \Sigma^* \to K$, we define the *system matrix* F as the infinite matrix $F = [f(\overline{x}_j\overline{y}_i)]_{i,j\in\mathbb{N}}$. Note that this is the transpose of what is commonly known as the *Hankel matrix*. Furthermore, for a given function f we define the functions $f_{\overline{x}} : \Sigma^* \to K$ by setting $f_{\overline{x}}(\overline{y}) := f(\overline{xy})$ for any sequences $\overline{x}, \overline{y} \in \Sigma^*$. Note that these functions correspond to the columns of the system matrix F. Let $\mathcal{F} := \operatorname{span}\{f_{\overline{x}} \,|\, \overline{x} \in \Sigma^*\}$ be the linear space spanned by these functions / the columns of F. Clearly, $\mathcal{F} \subseteq K\langle\langle\Sigma\rangle\rangle$. We define $\operatorname{rank}(f) := \operatorname{rank}(F) = \operatorname{rank}(\mathcal{F})$.

A *d-dimensional sequential system (SS)* is a structure $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$, which consists of an *initial state vector* $\omega_\varepsilon \in K^d$, a matrix $\tau_z \in K^{d\times d}$ for each $z \in \Sigma$ and an *evaluation function* $\sigma : K^d \to K$. For $\overline{x} = x_1 \cdots x_n \in \Sigma^*$ let $\tau_{\overline{x}} = \tau_{x_n} \cdots \tau_{x_1}$, and let $\omega_{\overline{x}} = \tau_{\overline{x}}\omega_\varepsilon$, which we call a *state* of the SS $\mathcal{M}$. Let $\tau_\Sigma = \sum_{z\in\Sigma} \tau_z$.

If the function σ is linear, we call the sequential system *linear*. In this paper, we will be dealing only with the linear case, so σ will just be a row vector, i.e., $\sigma^\top \in K^d$.

For a given SS $\mathcal{M}$, we define its *(external) function* to be

$$f_\mathcal{M} : \Sigma^* \to K, \quad f_\mathcal{M}(\overline{x}) = \sigma\tau_{\overline{x}}\omega_\varepsilon \tag{1}$$

Finally, we define the *rank* of a SS $\mathcal{M}$ to be $\operatorname{rank}(\mathcal{M}) := \operatorname{rank}(f_\mathcal{M})$.

2. Basic Properties of Sequential Systems

In this section we present the basic theory for sequential systems. This goes back to Schützenberger (1961), to Carlyle and Paz (1971) who coined the term *sequential systems*, and to Fliess (1974) but has been presented in various forms also for OOMs (Jaeger, 2000b) and PSRs (Singh et al., 2004). Here, we present the theory in a concise, self-contained fashion that can readily be turned into algorithms.

We begin with a technical result that lies at the heart of the whole theory.

Proposition 1 *Let $f : \Sigma^* \to K$ be given. If* $\operatorname{rank}(f) = d < \infty$*, then there exist linear operators $\tilde{\tau}_z : \mathcal{F} \to \mathcal{F}$ for each $z \in \Sigma$ and a linear functional $\tilde{\sigma} : \mathcal{F} \to K$ that satisfy $\tilde{\tau}_z(f_{\overline{x}}) = f_{\overline{x}z}$ and $\tilde{\sigma}(f_{\overline{x}}) = f(\overline{x})$ for all $\overline{x} \in \Sigma^*$. Furthermore, $\tilde{\sigma}(\tilde{\tau}_{\overline{x}}(f_\varepsilon)) = f(\overline{x})$ for all $\overline{x} \in \Sigma^*$, where $\tilde{\tau}_{\overline{x}} = \tilde{\tau}_{x_n} \circ \cdots \circ \tilde{\tau}_{x_1}$.*

Proof Let $J \subset \mathbb{N}$ be an index set denoting a maximal set of linearly independent columns of the matrix F. Then clearly, $\mathcal{B} = \{f_{\overline{x}_j} \mid j \in J\}$ is a basis for $\mathcal{F}$. Define linear operators $\tilde{\tau}_z$ and a linear functional $\tilde{\sigma}$ by their action on these basis elements:

- $\tilde{\tau}_z(f_{\overline{x}_j}) := f_{\overline{x}_j z}$ for all $z \in \Sigma$,
- $\tilde{\sigma}(f_{\overline{x}_j}) := f_{\overline{x}_j}(\varepsilon) = f(\overline{x}_j)$.

We will show that then $\tilde{\tau}_z(f_{\overline{x}}) = f_{\overline{x}z}$ and $\tilde{\sigma}(f_{\overline{x}}) = f(\overline{x})$ for all $\overline{x} \in \Sigma^*$. For this, let $\overline{x} \in \Sigma^*$. Then $f_{\overline{x}} = \sum_{j\in J} \lambda_j f_{\overline{x}_j}$ for suitable coordinates λ_j, and $f_{\overline{x}z} = \sum_{j\in J} \lambda_j f_{\overline{x}_j z}$, since for any $\overline{y} \in \Sigma^*$, we have $f_{\overline{x}z}(\overline{y}) = f_{\overline{x}}(z\overline{y}) = \sum_{j\in J} \lambda_j f_{\overline{x}_j}(z\overline{y}) = \sum_{j\in J} \lambda_j f_{\overline{x}_j z}(\overline{y})$. Therefore, $\tilde{\tau}_z(f_{\overline{x}}) = \tilde{\tau}_z(\sum_{j\in J} \lambda_j f_{\overline{x}_j}) = \sum_{j\in J} \lambda_j f_{\overline{x}_j z} = f_{\overline{x}z}$, and $\tilde{\sigma}(f_{\overline{x}}) = \tilde{\sigma}(\sum_{j\in J} \lambda_j f_{\overline{x}_j}) = \sum_{j\in J} \lambda_j f_{\overline{x}_j}(\varepsilon) = f_{\overline{x}}(\varepsilon) = f(\overline{x})$.

Finally, $\tilde{\sigma}(\tilde{\tau}_{\overline{x}}(f_\varepsilon)) = \tilde{\sigma}(f_{\overline{x}}) = f(\overline{x})$ for all $\overline{x} \in \Sigma^*$. ■

The above proposition establishes a crucial property that makes this theory appealing, as it means that the functions $f = f_\varepsilon$, $f_{\overline{x}} = \tilde{\tau}_{\overline{x}}(f)$, the linear operators $\tilde{\tau}_z$ and the linear functional $\tilde{\sigma}$ have coordinate representations as vectors and matrices with respect to some basis $\mathcal{B}$ for $\mathcal{F}$. Note that this remains true even if $\operatorname{rank}(f) = \infty$, but the coordinate representations will then be infinite and of little practical use. The property $f(\overline{x}) = \tilde{\sigma}(\tilde{\tau}_{\overline{x}}(f_\varepsilon))$ (cf. Equation 1) means that the function f is fully described by the data $(\tilde{\sigma}, \{\tilde{\tau}_z\}, f_\varepsilon)$. If these are given in some coordinate representation, then we have a SS representation:

Proposition 2 *Let $f : \Sigma^* \to K$ be given. If* $\operatorname{rank}(f) = d < \infty$*, then there exists a d-dimensional SS $\mathcal{M}$ such that $f = f_{\mathcal{M}}$.*

Proof Let $\mathcal{B}$ be a basis for $\mathcal{F}$, and let $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ be the coordinate representations of $(\tilde{\sigma}, \{\tilde{\tau}_z\}, f_\varepsilon)$ with respect to $\mathcal{B}$, where we are using the definitions for $\tilde{\sigma}$ and $\tilde{\tau}_z$ from the above Proposition 1. Then for any $\overline{x} \in \Sigma^*$, we have $f(\overline{x}) = \tilde{\sigma}(\tilde{\tau}_{\overline{x}}(f_\varepsilon)) = \sigma\tau_{\overline{x}}\omega_\varepsilon = f_{\mathcal{M}}(\overline{x})$. ■

Note that for the SS $\mathcal{M}$ constructed in Proposition 2 as a coordinate representation with respect to some basis $\mathcal{B}$ of $\mathcal{F}$, the states $\omega_{\overline{x}} = \tau_{\overline{x}}\omega_\varepsilon$ will be the coordinate representations of the functions $f_{\overline{x}} = \tilde{\tau}_{\overline{x}}(f)$ with respect to the basis $\mathcal{B}$. Also note that due to Equation (1) we may evaluate $f(\overline{x})$ using the SS $\mathcal{M}$ without knowledge of the basis $\mathcal{B}$.

The above proposition suggests that two SS might describe the same function f if and only if they are representations for f with respect to different bases for $\mathcal{F}$. However, this is only correct for so-called *minimal* SS, as will be detailed out in the following.

Definition 3 *Two SSs $\mathcal{M}$ and $\mathcal{M}'$ are* equivalent*, denoted by $\mathcal{M} \cong \mathcal{M}'$, if they define the same function, i.e., if $f_{\mathcal{M}} = f_{\mathcal{M}'}$. It is clear that this notion is an equivalence relation on the set of all SSs.*

We now introduce concepts needed to characterize the equivalence on SS. We give such a characterization for *minimal* SS in Proposition 12. For this, we introduce the concept of minimal SS, give a criteria for minimality in Corollary 8 and a procedure in Algorithm 2 to construct an equivalent minimal SS.

Definition 4 *For a given SS $\mathcal{M}$ we call the linear spaces $W = \operatorname{span}\{\tau_{\overline{x}}\omega_\varepsilon \,|\, \overline{x} \in \Sigma^*\}$ the* state space *and $\tilde{W} = \operatorname{span}\{(\sigma\tau_{\overline{x}})^\top \,|\, \overline{x} \in \Sigma^*\}$ the* co-state space *of $\mathcal{M}$.*

Definition 5 *We call a d-dimensional SS $\mathcal{M}$* trimmed *if it has full state and co-state spaces, i.e., if $W = \tilde{W} = K^d$. We call a SS* minimal *if no equivalent model of lower dimension exists.*

It will turn out in Corollary 8 that a SS is minimal if and only if it is trimmed. But first, we show how to construct bases for the state (and co-state) space of a given SS.

Proposition 6 *The following procedure constructs a basis $\mathcal{B}$ for the state space W of a given d-dimensional SS in time $\mathcal{O}(\max\{d, |\Sigma|\}d^3)$ (the construction of a basis $\tilde{\mathcal{B}}$ for the co-state space $\tilde{W}$ is analogous):*

Algorithm 1: Compute a basis $\mathcal{B}$ for the state space W of a given SS

```
B ← {}, C ← {ω_ε}
while |C| > 0 do
    ω ← some element of C, C ← C \ {ω}
    if ω is linearly independent of B then
        B ← B ∪ {ω}
        C ← C ∪ {τ_z ω | z ∈ Σ}
```

Proof At any time during the run of the algorithm, $\mathcal{B}$ is a set of linearly independent vectors. Furthermore the set $\mathcal{C}$ of "candidate vectors" increases by $|\Sigma|$ elements each time a new vector is added to the set $\mathcal{B}$, but decreases by one element each run through the main loop. Therefore, the algorithm terminates after at most $d|\Sigma| + 1$ runs through the main loop, since there are at most d linearly independent vectors that can be added to $\mathcal{B}$. Next we examine the runtime of the algorithm. Checking ω for linear independence from $\mathcal{B}$ can be done by checking $P_{\mathcal{B}}\omega = \omega$ in time $\mathcal{O}(d^2)$ if the orthogonal projection matrix $P_{\mathcal{B}}$ onto $\operatorname{span}(\mathcal{B})$ is known. This check is performed at most $d|\Sigma|+1$ times, yielding a complexity of $\mathcal{O}(d^3|\Sigma|)$. Clearly, the matrix $P_{\mathcal{B}}$ must be updated every time a vector is added to $\mathcal{B}$, which is a $\mathcal{O}(d^3)$ operation that needs to performed at most d times, giving a total complexity of $\mathcal{O}(d^4)$. Finally, every time a vector ω is added to $\mathcal{B}$, the set $\mathcal{C}$ is increased by $|\Sigma|$ vectors, each of which requires time $\mathcal{O}(d^2)$ to be computed from ω, for a total time complexity of $\mathcal{O}(d^3|\Sigma|)$. Adding these together gives the claimed time complexity.

Finally, we show that the returned set $\mathcal{B}$ is indeed a basis of the state-space W. Observe that for all $\omega \in \mathcal{B}$ and for all $z \in \Sigma$, the vectors $\tau_z\omega$ have been added as "candidate vectors" to the set $\mathcal{C}$ at some point during the run of the algorithm — namely when ω was added to $\mathcal{B}$. Each of these vectors is checked in turn and is at that point either linearly dependent on $\mathcal{B}$, or added to $\mathcal{B}$. Therefore, these vectors $\tau_z\omega$ are all linearly dependent on the final set $\mathcal{B}$, i.e., $\tau_z(\mathcal{B}) \subseteq \operatorname{span}(\mathcal{B})$ for all $z \in \Sigma$. By linearity of τ_z this implies that also

$\tau_z(\mathrm{span}(\mathcal{B})) \subseteq \mathrm{span}(\mathcal{B})$ for all $z \in \Sigma$. So $\mathrm{span}(\mathcal{B})$ contains ω_ε and is closed under the action of τ_z for all $z \in \Sigma$, which implies that $\{\tau_{\overline{x}}\omega_\varepsilon \,|\, \overline{x} \in \Sigma^*\} \subseteq \mathrm{span}(\mathcal{B})$. But $\mathcal{B} \subset \{\tau_{\overline{x}}\omega_\varepsilon \,|\, \overline{x} \in \Sigma^*\}$ by construction of $\mathcal{B}$. Together, this implies $\mathrm{span}(\mathcal{B}) = \mathrm{span}(\{\tau_{\overline{x}}\omega_\varepsilon \,|\, \overline{x} \in \Sigma^*\}) = W$. ■

The above is a polynomial time algorithm for which we have explicitly stated the runtime complexity, since it is the workhorse for the operations of this section and dominates their runtimes. Note further that the computed bases are by construction of the form $\mathcal{B} = \{\tau_{\overline{x}_j}\omega_\varepsilon \,|\, j \in J\}$ and $\tilde{\mathcal{B}} = \{(\sigma\tau_{\overline{x}_i})^\top \,|\, i \in I\}$ for suitable index sets I, J and corresponding words $\overline{x}_i$ and $\overline{x}_j$ of length at most d, where d is the dimension of the SS. Also, the above procedure allows us to check whether a given SS is trimmed.

The following proposition is the core technical result needed to establish the connection between a SS being trimmed, having full rank, and being minimal.

Proposition 7 *For a d-dimensional SS $\mathcal{M}$, let $\{\tau_{\overline{x}_j}\omega_\varepsilon \,|\, j \in J\}$ and $\{(\sigma\tau_{\overline{x}_i})^\top \,|\, i \in I\}$ be bases for W and $\tilde{W}$ respectively, and define $F^{I,J} = [f_{\mathcal{M}}(\overline{x}_j\overline{x}_i)]_{(i,j)\in I\times J}$, then* $\mathrm{rank}(\mathcal{M}) = \mathrm{rank}(F^{I,J}) \leq \min\{|I|, |J|\} \leq d$. *Furthermore, if $|I| = d$ or $|J| = d$ then* $\mathrm{rank}(\mathcal{M}) = \min\{|I|, |J|\}$.

Proof Define $\Pi = ((\sigma\tau_{\overline{x}_k})^\top)_{k\in\mathbb{N}}^\top$ and $\Phi = (\tau_{\overline{x}_k}\omega_\varepsilon)_{k\in\mathbb{N}}$, as well as $\Pi_I = ((\sigma\tau_{\overline{x}_i})^\top)_{i\in I}^\top \in K^{|I|\times d}$ and $\Phi_J = (\tau_{\overline{x}_j}\omega_\varepsilon)_{j\in J} \in K^{d\times|J|}$. The rows of Π_I are a basis for the row space of Π and the columns of Φ_J are a basis for the column space of Φ. Now $F = \Pi\Phi$ and $F^{I,J} = \Pi_I\Phi_J$. Therefore $\mathrm{rank}(\mathcal{M}) := \mathrm{rank}(F) = \mathrm{rank}(\Pi\Phi) = \mathrm{rank}(\Pi_I\Phi) = \mathrm{rank}(\Pi_I\Phi_J) = \mathrm{rank}(F^{I,J})$. Moreover, $\mathrm{rank}(\Pi_I) = |I|$ and $\mathrm{rank}(\Phi_J) = |J|$ imply that $\mathrm{rank}(\Pi_I\Phi_J) \leq \min\{|I|, |J|\} \leq d$ as well as $\mathrm{rank}(\Pi_I\Phi_J) = |J|$ if $|I| = d$ and $\mathrm{rank}(\Pi_I\Phi_J) = |I|$ if $|J| = d$. ■

From this, we obtain the following result, which allows us to check a d-dimensional SS for minimality by checking whether the SS is trimmed, i.e., by constructing bases for the state and co-state space and checking if these have dimension d.

Corollary 8 *Let $\mathcal{M}$ be a d-dimensional SS. The following are equivalent:*

(i) *$\mathcal{M}$ is trimmed*

(ii) $\mathrm{rank}(\mathcal{M}) = d$

(iii) *$\mathcal{M}$ is minimal*

Proof If $\mathcal{M}$ has full rank, i.e., $\mathrm{rank}(\mathcal{M}) = d$, then $\mathcal{M}$ must be minimal, as any lower-dimensional SS must have a lower rank and therefore cannot be equivalent. Conversely, if $\mathcal{M}$ is minimal, then we must have $\mathrm{rank}(\mathcal{M}) = d$, since by Proposition 2 there exists a $\mathrm{rank}(\mathcal{M})$-dimensional equivalent SS. By Proposition 7 — and using the notation from the proposition — we see that $\mathrm{rank}(\mathcal{M}) = d \Leftrightarrow |I| = |J| = d$, i.e., if and only if $\mathcal{M}$ is trimmed. ■

Next, we define the transformation of a SS by linear maps ρ and ρ'. Such transformations will serve as the basic operation on SS for all conversion operations.

Definition 9 *For a d-dimensional SS $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ and any matrices $\rho \in K^{n\times d}$ and $\rho' \in K^{d\times n}$, we define the n-dimensional SS $\rho\mathcal{M}\rho' := (\sigma\rho', \{\rho\tau_z\rho'\}, \rho\omega_\varepsilon)$.*

If ρ is non-singular, and $\rho' = \rho^{-1}$, then this transformation will yield an equivalent *conjugated* SS. If the SS is minimal, then this corresponds to a change of basis for the underlying function space $\mathcal{F}$.

Lemma 10 *Let $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ be a d-dimensional SS, and $\rho \in \mathbb{R}^{d\times d}$ be non-singular. Then $\mathcal{M} \cong \rho\mathcal{M}\rho^{-1}$. We will call $\rho\mathcal{M}\rho^{-1}$ a* conjugate *of $\mathcal{M}$.*

Proof $\forall \overline{x} \in \Sigma^* : f_{\rho\mathcal{M}\rho^{-1}}(\overline{x}) = (\sigma\rho^{-1})(\rho\tau_{x_n}\rho^{-1})\cdots(\rho\tau_{x_1}\rho^{-1})(\rho\omega_\varepsilon) = \sigma\tau_{\overline{x}}\omega_\varepsilon = f_{\mathcal{M}}(\overline{x})$. ■

We already know how to check for minimality. We now show how to convert a given SS to an equivalent minimal SS using the introduced transformations on SSs.

Proposition 11 *For a given SS $\mathcal{M}$, the following procedure constructs an equivalent minimal SS $\mathcal{M}''$:*

Algorithm 2: Minimization of a SS $\mathcal{M}$

1 *Construct a basis $\{\tau_{\overline{x}_j}\omega_\varepsilon \mid j \in J\}$ for the state space W of $\mathcal{M}$*
Set $\Phi = (\tau_{\overline{x}_j}\omega_\varepsilon)_{j\in J}$.
Set $\mathcal{M}' = \Phi^\dagger\mathcal{M}\Phi$, where $\Phi^\dagger$ denotes the Moore-Penrose pseudoinverse of Φ.
2 *Construct a basis $\{(\sigma'\tau'_{\overline{x}_i})^\top \mid i \in I'\}$ for the co-state space $\tilde{W}'$ of $\mathcal{M}'$.*
Set $\Pi' = ((\sigma'\tau'_{\overline{x}_i})^\top)^\top_{i\in I'}$.
Set $\mathcal{M}'' = \Pi'\mathcal{M}'\Pi'^\dagger$.

Proof Note that by construction the columns of Φ and $\Pi'^\top$ form bases for the spaces W and $\tilde{W}'$ respectively. Therefore, $\Phi^\dagger\Phi = id$ and $\Phi\Phi^\dagger|_W = id$, as well as $(\Pi'^\top)^\dagger\Pi'^\top = id$ and $\Pi'^\top(\Pi'^\top)^\dagger|_{\tilde{W}'} = id$. We can simply check equivalence, i.e., that for any $\overline{x} \in \Sigma^*$,

$$\begin{aligned}
f_{\mathcal{M}''}(\overline{x}) &= \sigma''\tau''_{x_n}\cdots\tau''_{x_1}\omega''_\varepsilon \\
&= \sigma'\Pi'^\dagger\Pi'\tau'_{x_n}\Pi'^\dagger\cdots\Pi'\tau'_{x_1}\Pi'^\dagger\Pi'\omega'_\varepsilon \\
&= \omega'^\top_\varepsilon\Pi'^\top(\Pi'^\top)^\dagger\tau'^\top_{x_1}\Pi'^\top\cdots(\Pi'^\top)^\dagger\tau'^\top_{x_n}\Pi'^\top(\Pi'^\top)^\dagger\sigma'^\top \\
&= \sigma'\tau'_{x_n}\cdots\tau'_{x_1}\omega'_\varepsilon \\
&= \sigma\Phi\Phi^\dagger\tau_{x_n}\Phi\cdots\Phi^\dagger\tau_{x_1}\Phi\Phi^\dagger\omega_\varepsilon \\
&= \sigma\tau_{\overline{x}}\omega_\varepsilon = f_{\mathcal{M}}(\overline{x}).
\end{aligned}$$

Next, consider $(\tau'_{\overline{x}_j}\omega'_\varepsilon)_{j\in J} = (\Phi^\dagger\tau_{\overline{x}_j}\omega_\varepsilon)_{j\in J} = \Phi^\dagger\Phi = id$. This implies that $\mathcal{M}'$ has full state space W' and that $\{\tau'_{\overline{x}_j}\omega'_\varepsilon \mid j \in J\}$ is a basis for W', since the dimension d' of $\mathcal{M}'$ is $|J|$ by construction. By Proposition 7, $|J| = d'$ implies $\mathrm{rank}(\mathcal{M}') = \min(|I'|, |J|) = |I'|$. By construction $|I'| = d''$ where d'' is the dimension of $\mathcal{M}''$. Furthermore, $\mathrm{rank}(\mathcal{M}') = \mathrm{rank}(\mathcal{M}'')$ since $\mathcal{M}' \cong \mathcal{M}''$ so by Corollary 8 $\mathcal{M}''$ is minimal. ■

As we can convert any SS to an equivalent minimal SS using the above Algorithm 2, it will be sufficient to characterize equivalence only for minimal SS. This is done by the following result.

Proposition 12 *Let $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ and $\mathcal{M}' = (\sigma', \{\tau'_z\}, \omega'_\varepsilon)$ be minimal d-dimensional SS. The following are equivalent:*

(i) $\mathcal{M} \cong \mathcal{M}'$

(ii) $\mathcal{M}' = \rho\mathcal{M}\rho^{-1}$ *for some non-singular* $\rho \in K^{d\times d}$

(iii) $\Pi\Phi = \Pi'\Phi'$, $\Pi\omega_\varepsilon = \Pi'\omega'_\varepsilon$, $\sigma\Phi = \sigma'\Phi'$ *and* $\forall z \in \Sigma : \Pi\tau_z\Phi = \Pi'\tau'_z\Phi'$, *where* $\{\tau_{\overline{x}_j}\omega_\varepsilon \mid j \in J\}$ *and* $\{(\sigma\tau_{\overline{x}_i})^\top \mid i \in I\}$ *are bases for the state and co-state spaces W and $\tilde{W}$ of $\mathcal{M}$ respectively, and* $\Pi = ((\sigma\tau_{\overline{x}_i})^\top)_{i\in I}^\top$, $\Phi = (\tau_{\overline{x}_j}\omega_\varepsilon)_{j\in J}$, $\Pi' = ((\sigma'\tau'_{\overline{x}_i})^\top)_{i\in I}^\top$, *and* $\Phi' = (\tau'_{\overline{x}_j}\omega'_\varepsilon)_{j\in J}$.

Proof Lemma 10 establishes $(ii) \Rightarrow (i)$. For $(i) \Rightarrow (iii)$ note that $f_\mathcal{M} = f_{\mathcal{M}'}$ implies that $\Pi\tau_{\bar{z}}\Phi = [f(\overline{x}_j\bar{z}\overline{x}_i)]_{i,j\in I\times J} = \Pi'\tau'_{\bar{z}}\Phi'$ for all $\bar{z} \in \Sigma^*$, as well as $\Pi\omega_\varepsilon = (f(\overline{x}_i))_{i\in I}^\top = \Pi'\omega'_\varepsilon$ and $\sigma\Phi = (f(\overline{x}_j))_{j\in J} = \sigma'\Phi'$. Finally, to see $(iii) \Rightarrow (ii)$, note that Π and Φ have full rank, since $\mathcal{M}$ is minimal, so Π' and Φ' must also have full rank. Let $\rho = \Pi'^{-1}\Pi = \Phi'\Phi^{-1}$, then $\rho^{-1} = \Phi\Phi'^{-1}$. We can now easily check that $\mathcal{M}' = \rho\mathcal{M}\rho^{-1}$. ■

Note that this allows us to decide equivalence for any given SS $\mathcal{M}$ and $\mathcal{M}'$ by first converting them to equivalent minimal SS $\tilde{\mathcal{M}}$ and $\tilde{\mathcal{M}}'$ respectively using Algorithm 2, and then checking for equivalence by criteria (iii) from the above Proposition 12. The required bases for the state and co-state spaces of $\tilde{\mathcal{M}}$ and $\tilde{\mathcal{M}}'$ can be computed by Algorithm 1.

The following proposition shows that any SS can be transformed into an equivalent SS where σ and ω_ε can be essentially any desired vectors. This implies that it is no restriction to assume some fixed form for σ, as is sometimes done. For instance, in the case of OOMs often $\sigma = (1, \ldots, 1)$ is used, while for MA often $\sigma = (1, 0, \ldots, 0)$ is assumed.

Proposition 13 *Let $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ be a d-dimensional SS, and let $\sigma'^\top, \omega'_\varepsilon \in K^d$ such that $\sigma'\omega'_\varepsilon = \sigma\omega_\varepsilon$. Then there exists a non-singular linear map ρ such that $\rho\mathcal{M}\rho^{-1} = (\sigma', \{\tau'_z\}, \omega'_\varepsilon)$.*

Proof Extend $\{\sigma^\top\}$ to an orthogonal basis $\{\sigma^\top, v_2, \ldots, v_d\}$ of K^d, and $\{\sigma'^\top\}$ to an orthogonal basis $\{\sigma'^\top, v'_2, \ldots, v'_d\}$ of K^d. We distinguish two cases:

If $c := \sigma\omega_\varepsilon = \sigma'\omega'_\varepsilon \neq 0$, then $\rho_1 = (\omega_\varepsilon, v_2, \ldots, v_d)^{-1}$ and $\rho_2 = (\omega'_\varepsilon, v'_2, \ldots, v'_d)$ are non-singular. Let $\rho = \rho_2\rho_1$. We can easily see that $\rho_2\rho_1\omega_\varepsilon = \rho_2 e_1 = \omega'_\varepsilon$ and $\sigma\rho^{-1} = \sigma\rho_1^{-1}\rho_2^{-1} = c \cdot e_1^\top\rho_2^{-1} = \sigma'$, since $\sigma'\rho_2 = c \cdot e_1^\top$.

If $\sigma\omega_\varepsilon = \sigma'\omega'_\varepsilon = 0$, then (perhaps after reordering v_i and v'_i) $\rho_1 = (\frac{\sigma^\top}{\sigma\sigma^\top}, \omega_\varepsilon, v_3, \ldots, v_d)^{-1}$ and $\rho_2 = (\frac{\sigma'^\top}{\sigma'\sigma'^\top}, \omega'_\varepsilon, v'_3, \ldots, v'_d)$ are non-singular. Let $\rho = \rho_2\rho_1$. We can again check that $\rho_2\rho_1\omega_\varepsilon = \rho_2 e_2 = \omega'_\varepsilon$ and $\sigma\rho^{-1} = \sigma\rho_1^{-1}\rho_2^{-1} = e_1\rho_2^{-1} = \sigma'$, since $\sigma'\rho_2 = e_1$. ■

Finally, we introduce a special property called *interpretability* that a SS can have. This concept has led to some confusion in the past — especially regarding the relationship between OOMs and PSRs. This is due to the fact that it has been defined differently for OOMs, IO-OOMs and PSRs, as will be discussed later. Another source of confusion is that interpretability has been regarded as a crucial property for learning, which is however only

true for the the very early learning algorithms. Here we give a definition of interpretability that works for all models, and we will defer the discussion of the different uses to the later sections.

Definition 14 *A d-dimensional SS $\mathcal{M}$ is said to be* interpretable w.r.t. the sets $Y_1, \ldots, Y_d \subset \Sigma^*$ *if the states $\omega_{\overline{x}}$ take the form $\omega_{\overline{x}} = [f_{\mathcal{M}}(\overline{x}Y_1), \ldots, f_{\mathcal{M}}(\overline{x}Y_d)]^\top$ for all $\overline{x} \in \Sigma^*$, where $f_{\mathcal{M}}(\overline{x}Y) = \sum_{\overline{y}\in Y} f_{\mathcal{M}}(\overline{xy})$.*

The following proposition and algorithm show how to *make a SS interpretable*, i.e., how to convert any given SS into an equivalent interpretable form.

Proposition 15 *Let $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ be a d-dimensional minimal SS, and $Y_1, \ldots, Y_d \subset \Sigma^*$. If $\rho = [(\sigma\tau_{Y_1})^\top, \ldots, (\sigma\tau_{Y_d})^\top]^\top$ is non-singular, where $\tau_Y = \sum_{\overline{y}\in Y} \tau_{\overline{y}}$, then $\mathcal{M}' := \rho\mathcal{M}\rho^{-1} \cong \mathcal{M}$ and $\mathcal{M}'$ is interpretable w.r.t. $Y_1, \ldots, Y_d$.*

Proof $\forall \overline{x} \in \Sigma^* : \omega'_{\overline{x}} = \rho\omega_{\overline{x}} = [\sigma\tau_{Y_1}\tau_{\overline{x}}\omega_\varepsilon, \ldots, \sigma\tau_{Y_d}\tau_{\overline{x}}\omega_\varepsilon]^\top = [f_{\mathcal{M}}(\overline{x}Y_1), \ldots, f_{\mathcal{M}}(\overline{x}Y_d)]^\top$. ■

Corollary 16 *For a SS $\mathcal{M}$, the following algorithm returns an equivalent interpretable SS.*

Algorithm 3: Make a SS $\mathcal{M}$ of rank d interpretable

1 *Minimize $\mathcal{M}$, i.e., find an equivalent minimal SS $\mathcal{M}'$ using Algorithm 2.*
2 *Construct a basis $\{(\sigma'\tau'_{\overline{x}_i})^\top \mid i \in I\}$ of the co-state space $\tilde{W}'$ of $\mathcal{M}'$ using Algorithm 1*
Select sets $Y_k = \{\overline{x}_{i_k}\}$ where $\{i_1, \ldots, i_d\} = I$.
Set $\rho = [(\sigma'\tau'_{Y_1})^\top, \ldots, (\sigma'\tau'_{Y_d})^\top]^\top$.
3 *Return $\rho\mathcal{M}'\rho^{-1}$.*

Proof The above algorithm indeed returns an equivalent SS that is interpretable w.r.t. the selected sets Y_k, since $\mathcal{M}'$ is minimal and therefore ρ is non-singular by construction. ■

3. Versions of Sequential Systems

In this section we first show that SS are an algebraic characterization of multiplicity automata (MA), and we mention the relationship to the more general class of weighted finite automata (WFA) and its extension to weighted finite-state transducers (WFST). We then define stochastic multiplicity automata (SMA), observable operator models (OOMs) and predictive state representations (PSRs), which are known to generalize probabilistic finite automata (PFA), hidden Markov models (HMMs) and partially observable Markov decision processes (POMDPs), respectively. We show that these are all instances of SSs that are used to model different kinds of objects. Furthermore, we examine the relations between these models. An overview is given in Figure 1.

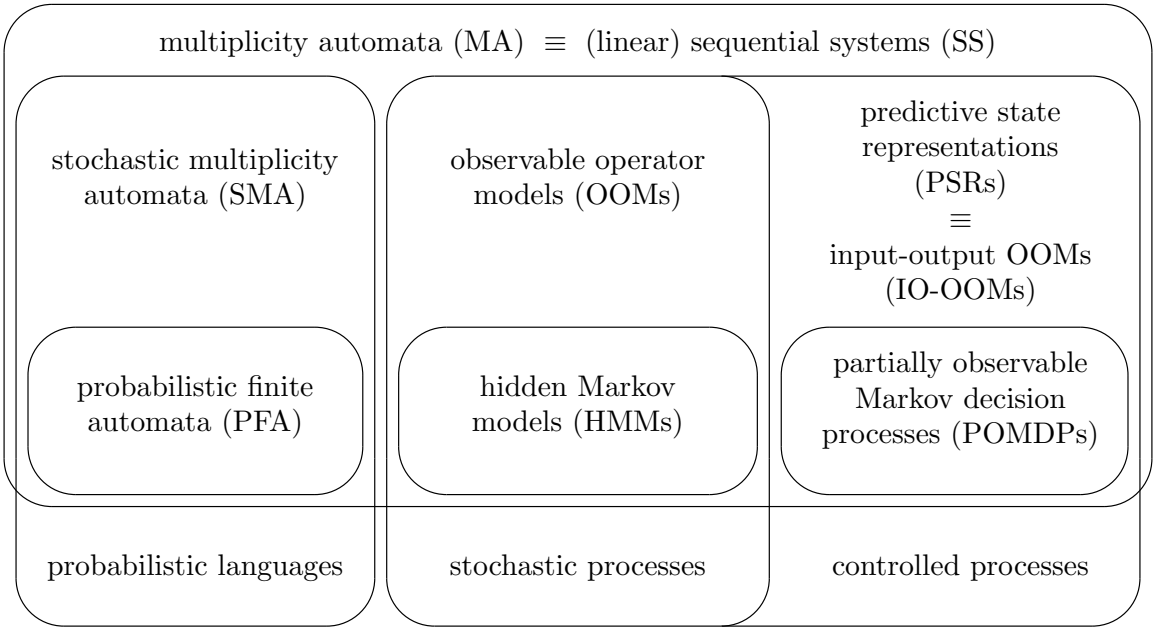

Figure 1: SMA, OOMs and PSRs are versions of SSs that model probabilistic languages, stochastic processes and controlled processes respectively, and strictly generalize PFA, HMMs and POMDPs respectively.

3.1 Multiplicity Automata and Weighted Automata

The above definition of linear finite dimensional SS is an equivalent algebraic way of looking at a type of automata that were introduced by Schützenberger (1961) and are most commonly known as *multiplicity automata* (Salomaa and Soittola, 1978; Berstel and Reutenauer, 1988). We will give a very brief introduction.

Definition 17 *A* K-multiplicity automaton *(MA) is a structure* $\langle \Sigma, Q, \varphi, \iota, \tau \rangle$*, where* Σ *is an* alphabet*,* Q *is a finite set of* states*,* $\varphi : Q \times \Sigma \times Q \to K$ *is the* state transition function*,* $\iota : Q \to K$ *is the* initialization function*, and* $\tau : Q \to K$ *is the* termination function*. The state transition function is extended to words by setting* $\forall \overline{x} \in \Sigma^*, z \in \Sigma$: $\varphi(q, \overline{x}z, q') = \sum_{s \in Q} \varphi(q, \overline{x}, s)\varphi(s, z, q')$*, and* $\varphi(q, \varepsilon, q') = 1$ *if* $q = q'$ *and 0 otherwise. A multiplicity automaton* $\mathcal{M}$ *then defines a function*

$$f_{\mathcal{M}} : \Sigma^* \to K, \quad f_{\mathcal{M}}(\overline{x}) = \sum_{q,q' \in Q} \iota(q)\varphi(q, \overline{x}, q')\tau(q').$$

The formal equivalence of MA to linear finite-dimensional SS is easily seen by rewriting the definition of MA in terms of matrix multiplication: Set $\omega_\varepsilon = [\iota(q_i)]_i$, $\tau_z = [\varphi(q_j, z, q_i)]_{i,j}$, and $\sigma = [\tau(q_j)]_j^\top$. Then we have $\tau_{\overline{x}z} = [\varphi(q_j, \overline{x}z, q_i)]_{i,j} = [\sum_{q_k \in Q} \varphi(q_j, \overline{x}, q_k)\varphi(q_k, z, q_i)]_{i,j} = [\varphi(q_k, z, q_i)]_{i,k}[\varphi(q_j, \overline{x}, q_k)]_{k,j} = \tau_z\tau_{\overline{x}}$ and similarly $f_{\mathcal{M}}(\overline{x}) = \sigma\tau_{\overline{x}}\omega_\varepsilon$. However, the above definition of MA makes it apparent how MA are an extension of non-deterministic finite

automata (NFA) to WFA that add weights to the initial and terminal states as well as the state transitions. The weight of a path from an initial state to a termination state is then given by the product of the corresponding weights (hence the name *multiplicity* automata), while the value $f_{\mathcal{M}}(\overline{x})$ is computed by summing the weights of all paths compatible with $\overline{x}$.

At this point we should mention that MA as defined here are merely a special case of WFA. The difference is that for MA we consider weights from a field K (here $K = \mathbb{R}$ or $K = \mathbb{C}$), while for WFA the weights are only required to come from an algebraic structure K called a semiring. There exists a large body of theory for WFA that generalizes the theory of SS that we have presented in Section 2, which can be found in the recent textbook by Droste et al. (2009). Note that while MA and WFA are formally closely related, there is a difference in the way they are viewed and used. For instance, WFA are often considered over the semiring $\mathbb{R}^+$ with weights given the interpretation of transition probabilities, which are then called probabilistic finite automata (PFA). Such PFA are graphical models, and the states Q are latent states. For $\mathbb{R}$-MA, however, the weights are allowed to be negative, and the weights as well as the states Q become abstract notions. In other words, PFA (and WFA in general) are typically used when the states and transition structure carry some meaning, while MA are typically used as an abstract tool to characterize functions $f : \Sigma^* \to K$. This difference in perspective is reflected in the relationship of PFA to SMA, HMM to OOM and POMDP to PSR described in the remainder of this Section 3. Note that PFA are a special case of MA, as $\mathbb{R}^+ \subset \mathbb{R}$. In fact, there exist functions $f : \Sigma^* \to \mathbb{R}^+$ that can be described by a MA, but not by a PFA, i.e., MA are strictly more general than PFA. This sequence of increasing generalization starting with finite automata (FA) can be summarized as follows:

$$\text{FA} \subset \text{NFA} \subset \text{PFA} \subset \text{MA} \equiv \text{SS} \subset \text{WFA}.$$

Furthermore, there exists a natural extension of WFA to input-output systems that are called weighted finite-state transducers (WFST). Here, the alphabet Σ is split as $\Sigma = \Sigma_I \times \Sigma_O$, where Σ_I is regarded as input alphabet and Σ_O as output alphabet. The function $f_{\mathcal{M}} : \Sigma_I^* \times \Sigma_O^* \to K$ is then viewed as describing a relation between Σ_I and Σ_O. Again, K is in general only required to be a semiring, but a typical choice is $K = \mathbb{R}^+$ with the interpretation of state transition probabilities, yielding a latent variable model called probabilistic finite-state transducers (PFST). WFST are a flexible class of models that have been shown to unify several common approaches used in the the context of language and speech processing; a survey is given by Mohri et al. (2002). Furthermore, IO-OOMs and thereby PSRs (cf. Section 3.2 and Section 3.3) are in fact WFST with weights in $K = \mathbb{R}$, although they are not usually viewed this way, as WFST are typically seen as latent variable models, while IO-OOMs and PSRs are not. However, since PFST are MA, as long as the desired application merely requires the characterization of the function $f_{\mathcal{M}} : \Sigma_I^* \times \Sigma_O^* \to \mathbb{R}^+$, the SS learning algorithms described in Section 4 can be applied to the case of WFST as well, as has been done recently by Balle et al. (2011).

Note that in the context of MA one is usually interested in characterizing functions $f : \Sigma^* \to K$, which are also called *formal series* in general and *recognizable series* if they are computed by a MA. However, a MA $\mathcal{M}$ can also be used to recognize a language $L \subseteq \Sigma^*$ by setting $L_{\mathcal{M}} = \{\overline{x} \in \Sigma^* \mid f_{\mathcal{M}}(\overline{x}) \subseteq J\}$ for some subset $J \subseteq K$, e.g., $J = \{k \in K : k > \kappa\}$

for some threshold parameter $\kappa \geq 0$. The class of languages recognizable by MA is known to be strictly more general than the class of regular languages (Cortes and Mohri, 2000).

MA have received a lot of attention in the context of learning theory following the discovery of efficient learning algorithms (Bergadano and Varricchio, 1994; Ohnishi et al., 1994) in an extended version of the exact learning model of Angluin (1987). This led to further results on the learnability of several classes of DNF formulae (Bergadano et al., 1996), the class of polynomials over finite fields, decision trees and others (Beimel et al., 1996, 2000).

3.1.1 Stochastic Multiplicity Automata and Stochastic Languages

Additionally, MA have been applied in the context of probabilistic grammatical inference (Denis et al., 2006; Bailly et al., 2009), which is of particular interest to us because of the close relationship of these approaches to OOMs and PSRs — as we shall see.

Definition 18 *A function $f : \Sigma^* \to \mathbb{R}$ that satisfies $0 \leq f \leq 1$ and $f(\Sigma^*) = \sum_{\overline{x} \in \Sigma^*} f(\overline{x}) = 1$ is called a* stochastic language*,* probabilistic language *or just* distribution *over Σ^*. A distribution $f_{\mathcal{M}}$ on Σ^* that is defined by some MA $\mathcal{M}$ is called a* rational stochastic language*, and a MA that defines such a distribution is called a* stochastic MA *(SMA).*

Denis and Esposito (2008) give a comprehensive overview of rational stochastic languages over various fields K, their relationships and relations to subclasses such as the important class of probabilistic regular languages.

Definition 19 *A* probabilistic (finite) automaton *(PFA) is a SMA with the following restrictions: (i) ι, τ, φ have values in $[0, 1]$, and (ii) $\iota(Q) = 1$ and $\forall q \in Q : \tau(q) + \varphi(q, \Sigma, Q) = 1$, where $\iota(Q) = \sum_{q \in Q} \iota(q)$ and $\varphi(q, \Sigma, Q) = \sum_{x \in \Sigma} \sum_{q' \in Q} \varphi(q, x, q')$. The stochastic languages that can be represented by PFA are called* probabilistic regular languages*.*

PFA are closely related to hidden Markov models (HMMs), and the relationship has been detailed out by Dupont et al. (2005). It is however less well known that SMA are closely related to observable operator models — a class of models for stochastic processes that generalize HMM in a similar way that SMA generalize PFA.

We point out two results that are relevant in the context of modeling probabilistic languages by MA. First of all, it is known that it is an NP-hard problem to compute the maximum likelihood estimate of parameters of a PFA with known structure from a given training set of words (Abe and Warmuth, 1992). In practice, algorithms based on expectation maximization (EM) (Dempster et al., 1977) are used which compute locally optimal models instead. In contrast to this, the algebraic theory for SSs allows for powerful learning algorithms (see Section 4) that often outperform EM-trained PFA or HMMs (Rosencrantz et al., 2004; Jaeger et al., 2006a). However, these learning algorithms may return MA that are arbitrarily close to SMA but fail to represent stochastic languages. It is in fact undecidable whether a MA represents a stochastic language (Denis and Esposito, 2004).

3.2 Observable Operator Models and Stochastic Processes

Observable operator models were introduced by Jaeger (1997) as a concise algebraic characterization of stochastic processes (see also Jaeger, 1998, 2000b; Jaeger et al., 2006b). These

models are closely related to other algebraic characterizations of stochastic processes (Heller, 1965; Ito, 1992; Upper, 1997) that were studied in the context of deciding the equivalence for HMMs (Gilbert, 1959), which came to a successful conclusion by framing HMMs in the more general class of *linearly dependent processes* by Ito et al. (1992).

Definition 20 *A* (discrete-valued) stochastic process *is a function* $f : \Sigma^* \to [0,1]$ *that satisfies (i)* $f(\varepsilon) = 1$ *and (ii)* $\forall \overline{x} \in \Sigma^* : f(\overline{x}) = \sum_{x \in \Sigma} f(\overline{x}x)$*. Such a function* f *defines the probabilities of initial observation sequences. An* observable operator model *(OOM) is a linear SS* $\mathcal{M}$ *such that* $f_{\mathcal{M}}$ *is a stochastic process. A stochastic process that can be modeled by a finite dimensional OOM is called a* linearly dependent process*.*

One of the interesting features of OOMs is their notion of "state" of a (stochastic) process. The idea that goes back to Zadeh (1969) is that a system state is really nothing more than the information that is required to predict the future. In the case of OOMs, the states $\omega_{\overline{x}}$ not only carry enough information to predict the future, they *are* (in a certain sense) just future predictions.

To see this, recall that the states $\omega_{\overline{x}}$ of a SS are coordinate representations of the functions $f_{\overline{x}}$ w.r.t. some unknown basis $\mathcal{B}$ of the function space $\mathcal{F}$. In the case of OOMs, these functions take on the meaning that $f_{\overline{x}}(\overline{y}) = P(\overline{x}\overline{y})$, i.e., they give the probability of observing the sequence $\overline{x}$ followed by $\overline{y}$. These functions are therefore called *future prediction functions* in the context of OOMs. The operators $\{\tau_z\}$ are then state update operators that update a state $\omega_{\overline{x}}$ (corresponding to the future prediction function $f_{\overline{x}}$ after an initial observation of $\overline{x}$) according to the new observation z to the new state $\omega_{\overline{x}z}$ (corresponding to the future prediction function $f_{\overline{x}z}$ after an initial observation of $\overline{x}z$) — hence the name "observable operators" (Jaeger, 1998).

For convenience, these functions $f_{\overline{x}}$, as well as the corresponding states $\omega_{\overline{x}}$, are often normalized to $f_{\overline{x}}/f(\overline{x})$ and $\omega_{\overline{x}}/\sigma\omega_{\overline{x}}$ respectively, since $f_{\overline{x}}(\overline{y})/f(\overline{x}) = \sigma\tau_{\overline{y}}\omega_{\overline{x}}/\sigma\omega_{\overline{x}} = P(\overline{y}|\overline{x})$, the probability of observing $\overline{y}$ given that $\overline{x}$ has been observed. Therefore, an OOM started in the normalized state $\omega_{\overline{x}}/\sigma\omega_{\overline{x}}$ represents a stochastic process started after an initial observation of $\overline{x}$. This corresponds to the notion of a *residual automaton* in the context of SMA, which is obtained by starting a SMA in the (normalized) state $\omega_{\overline{x}}/\sum_{\overline{z} \in \Sigma^*} \sigma\tau_{\overline{z}}\omega_{\overline{x}}$ and then represents a *residual language* (Denis and Esposito, 2004).

3.2.1 Relation to Hidden Markov Models

Any HMM can be trivially converted into an OOM. A hidden Markov model (HMM) consists of an unobserved Markov process X_t that takes values in a finite set of states $Q = \{s_1, \ldots, s_n\}$, and is governed by a stochastic state transition matrix $T = [P(X_{t+1} = s_j \,|\, X_t = s_i)]_{i,j}$. At each time step an observation Y_t from Σ is made according to the emission vector $E_z = [P(Y_t = z \,|\, X_t = s_i)]_i$. Finally, an initial state vector $\pi = [P(X_0 = s_i)]_i$ is needed to fully specify the distribution of the stochastic process Y_t (Rabiner, 1989).

Proposition 21 *(Jaeger, 2000b) A given HMM* $(T, \{E_z\}_{z \in \Sigma}, \pi)$ *with* N *states is equivalent to the OOM* $(\sigma, \{\tau_z\}, \omega_\varepsilon)$ *defined by* $\sigma = (1, \ldots, 1)$*,* $\tau_z = T^\top \mathrm{diag}(E_z)$ *and* $\omega_\varepsilon = \pi$*. The rank of the OOM is less than or equal to* N*.*

Moreover, there are examples of OOMs of finite rank that cannot be modeled by any HMM with a finite number of states. A prototypical example is the so-called "probability clock" (Jaeger, 1998). It is an open question how to find a "close" HMM for a given OOM. While OOMs can be seen as a generalization of HMMs, one should keep in mind that there is a fundamental difference in the notion of the state of the process. The state vector in the case of a HMM is a stochastic vector that expresses the belief about the underlying hidden state, while for an OOM it is a coordinate representation of the corresponding future prediction function. However, under certain conditions it is possible to recover HMM-like hidden states from an OOM (Hsu et al., 2009; Anandkumar et al., 2012).

3.2.2 Relationship to Stochastic Multiplicity Automata

The main difference between OOMs and SMA is that OOMs model stochastic processes, while SMA model distributions on words. However, we can use a stochastic process to model a distribution on words if we introduce a termination symbol \$.

Definition 22 *An OOM $\mathcal{M}$ over the alphabet $\Sigma_\$ = \Sigma \cup \{\$\}$ is* terminating *if $f_\mathcal{M}(\Sigma^*\$) := \sum_{\overline{x}\in\Sigma^*} \sigma\tau_\$\tau_{\overline{x}}\omega_\varepsilon = 1$.*

Proposition 23 *An OOM $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ over the alphabet Σ can be extended to a terminating OOM $\mathcal{M}' = (\sigma, \{\tau'_z\}, \omega_\varepsilon)$ over the alphabet $\Sigma_\$ = \Sigma\cup\{\$\}$ by setting $\tau'_z = (1-p)\tau_z$ and $\tau'_\$ = p\tau_\Sigma$ for some fixed termination probability $p \in (0,1)$, where $\tau_\Sigma = \sum_{z\in\Sigma}\tau_z$.*

Proof We first show that $\mathcal{M}'$ describes a stochastic process. Clearly, $f_{\mathcal{M}'} \geq 0$ and $f_{\mathcal{M}'}(\varepsilon) = \sigma\omega_\varepsilon = 1$. To show property (ii), take any $\overline{x} \in \Sigma^*_\$$ and note that by linearity $\tau'_{\overline{x}}\omega_\varepsilon = \sum_k \lambda_k \tau_{\overline{x}_k}\omega_\varepsilon$ for suitable $\lambda_k \in \mathbb{R}$ and sequences $\overline{x}_k \in \Sigma^*$ (this is obtained by replacing all occurrences of $\tau'_\$$ by $p\sum_{z\in\Sigma}\tau_z$). Then $\sum_{z\in\Sigma_\$} f_{\mathcal{M}'}(\overline{x}z) = \sigma(\sum_{z\in\Sigma_\$}\tau'_z)\tau'_{\overline{x}}\omega_\varepsilon = \sigma\tau_\Sigma\tau'_{\overline{x}}\omega_\varepsilon = \sum_k \lambda_k\sigma\tau_\Sigma\tau_{\overline{x}_k}\omega_\varepsilon = \sum_k \lambda_k\sigma\tau_{\overline{x}_k}\omega_\varepsilon = \sigma\tau'_{\overline{x}}\omega_\varepsilon = f_{\mathcal{M}'}(\overline{x})$. Furthermore, $f_{\mathcal{M}'}(\Sigma^*\$) = \sum_{\overline{x}\in\Sigma^*}\sigma\tau'_\$\tau'_{\overline{x}}\omega_\varepsilon = \sum_{l=0}^\infty\sum_{\overline{x}\in\Sigma^l}\sigma p\tau_\Sigma(1-p)^l\tau_{\overline{x}}\omega_\varepsilon = \sum_{l=0}^\infty p(1-p)^l = 1$. ■

Definition 24 *A terminating OOM $\mathcal{M}$ over the alphabet $\Sigma\cup\{\$\}$ and a SMA $\mathcal{A}$ over the alphabet Σ are* related*, if $f_\mathcal{M}(\overline{x}\$) = f_\mathcal{A}(\overline{x})$ for all $\overline{x}\in\Sigma^*$.*

Lemma 25 *If $\mathcal{A} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ is a minimal d-dimensional SMA, then $\tau_{\Sigma^*} = \sum_{k=0}^\infty \tau_\Sigma^k$ exists and is equal to $(I_d - \tau_\Sigma)^{-1}$, where $\tau_\Sigma = \sum_{z\in\Sigma}\tau_z$.*

Proof We will show that the spectral radius[1] $\rho(\tau_\Sigma)$ satisfies $\rho(\tau_\Sigma) < 1$, which implies the lemma. Assume $\rho(\tau_\Sigma) \geq 1$, i.e., there exists some $\lambda \in \mathbb{C}, |\lambda| \geq 1$ and $v \in \mathbb{C}^d$ such that $\tau_\Sigma v = \lambda v$. As $\mathcal{A}$ is minimal, we may find sequences $\overline{x}_j, \overline{x}_i \in \Sigma^*$ such that $\Pi = ((\sigma\tau_{\overline{x}_i})^\top)_{i\in I}^\top$ and $\Phi = (\tau_{\overline{x}_j}\omega_\varepsilon)_{j\in J}$ with $|I| = |J| = d$ are non-singular using Algorithm 1. Then $v = \Phi a$ for some $a \in \mathbb{C}^d$, and $\Pi\tau_\Sigma^k\Phi a = \lambda^k\Pi\Phi a$ for any $k \in \mathbb{N}$. Now the SMA property $f_\mathcal{A}(\Sigma^*) = \sum_{k=0}^\infty \sigma\tau_\Sigma^k\omega_\varepsilon = 1$ implies that $\Pi\tau_\Sigma^k\Phi \to 0$ as $k\to\infty$, while the right hand side $\lambda^k\Pi\Phi a$ does not (note $\Pi\Phi a \neq 0$), which is a contradiction. ■

1. For $A \in \mathbb{C}^{n\times n}$ with eigenvalues $\lambda_1, \ldots, \lambda_k$, the spectral radius is defined as $\rho(A) := \max_i |\lambda_i|$.

Proposition 26 *Let $\mathcal{A} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ be a minimal d-dimensional SMA. Then $\mathcal{M} = (\sigma', \{\tau'_z\}, \omega'_\varepsilon)$ is a related $(d+1)$-dimensional terminating OOM over the alphabet $\Sigma_\$ = \Sigma \cup \{\$\}$, if*

- $\sigma' = [\sigma \sum_{k=0}^{\infty} \tau_\Sigma^k, 1] = [\sigma(I_d - \tau_\Sigma)^{-1}, 1]$,
- $\tau'_z = \left[\begin{smallmatrix} \tau_z & 0 \\ 0 & 0 \end{smallmatrix}\right], \quad \tau'_\$ = \left[\begin{smallmatrix} 0 & 0 \\ \sigma & 1 \end{smallmatrix}\right]$, *and*
- $\omega'_\varepsilon = \left[\begin{smallmatrix} \omega_\varepsilon \\ 0 \end{smallmatrix}\right]$.

Proof We can simply check that for all $\overline{z} \in \Sigma_\$^*$

$$f_\mathcal{M}(\overline{z}) = \sigma' \tau'_{\overline{z}} \omega'_\varepsilon = \begin{cases} \sigma(\sum_{k=0}^{\infty} \tau_\Sigma^k) \tau_{\overline{z}} \omega_\varepsilon & \text{if } \overline{z} \in \Sigma^*, \\ \sigma \tau_{\overline{x}} \omega_\varepsilon & \text{if } \overline{z} = \overline{x}\$ \ldots \$ \text{ for some } \overline{x} \in \Sigma^*, \\ 0 & \text{otherwise.} \end{cases}$$

This implies $f_\mathcal{M} \geq 0$, $f_\mathcal{M}(\overline{x}\$) = f_\mathcal{A}(\overline{x})$ for all $\overline{x} \in \Sigma^*$ ($\mathcal{M}$ and $\mathcal{A}$ are related), as well as $f_\mathcal{M}(\Sigma^*\$) = f_\mathcal{A}(\Sigma^*) = 1$ ($\mathcal{M}$ is terminating if it is an OOM). Furthermore, $\sigma' \omega'_\varepsilon = f_\mathcal{A}(\Sigma^*) = 1$ and $\sigma' \tau'_{\Sigma_\$} = [\sigma \sum_{k=0}^{\infty} \tau_\Sigma^k \tau_\Sigma + \sigma, 1] = \sigma'$, which imply property (i) and (ii) for a stochastic process respectively. ■

Proposition 27 *Conversely, let $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ be a d-dimensional terminating OOM over the alphabet $\Sigma \cup \{\$\}$. Then $\mathcal{A} = (\sigma\tau_\$, \{\tau_z\}, \omega_\varepsilon)$ is a related d-dimensional SMA over the alphabet Σ.*

Proof Clearly, $f_\mathcal{A}(\overline{x}) = f_\mathcal{M}(\overline{x}\$) \geq 0$ for all $\overline{x} \in \Sigma^*$ and $f_\mathcal{A}(\Sigma^*) = f_\mathcal{M}(\Sigma^*\$) = 1$. ■

3.2.3 Historical Remarks

Note that our definition of OOMs given in Definition 20 differs slightly from the definition typically found in the literature.

First of all, the property (ii) for a stochastic process means that an OOM must satisfy $\sigma\tau_\Sigma\omega_{\overline{x}} = \sigma\omega_{\overline{x}}$ for all $\overline{x} \in \Sigma^*$, which implies (ii)' $\sigma\tau = \sigma$ if the OOM is minimal, but not in general. The property (ii)' is however often stated as part of the definition for OOMs. Our above Definition 20 is therefore slightly more relaxed than the standard definition in the case of non-minimal models, but this has no practical consequences.

Furthermore, for purely historical reasons, OOMs are sometimes required to satisfy $\sigma = (1, \ldots, 1)$, which is mainly an issue of normalization (cf. Proposition 13). However, this in turn has led to a more restrictive definition of interpretability for OOMs, since due to property (i) of stochastic processes, an OOM that satisfies $\sigma = (1, \ldots, 1)$ can only be interpretable with respect to sets Y_k, if $1 = \sigma\omega_\varepsilon = (1, \ldots, 1) \cdot [f_\mathcal{M}(Y_i)]_i^\top = \sum_k \sum_{\overline{y} \in Y_k} P(\overline{y})$. This is typically assured by requiring the sets Y_k to partition Σ^l for some l. One can relax this restriction on the sets Y_k for the definition of interpretability — as we have done in Definition 14 — if one is willing to drop the normalization requirement $\sigma = (1, \ldots, 1)$ as well.

Nevertheless, even though the normalization requirement $\sigma = (1, \ldots, 1)$ is superfluous, several of the OOM learning algorithms have been designed to yield OOMs normalized such that $\sigma = (1, \ldots, 1)$ — oftentimes unnecessarily complicating the algorithms — and some proofs have made use of this normalization as well. Later in Section 4 we present simplified and generalized versions of the EC and ES learning algorithms by removing this normalization restriction from the algorithms and proofs.

3.3 Predictive State Representations and Controlled Processes

Following the development of OOMs for stochastic processes, extensions to the case of controlled processes — stochastic processes that depend on an external input at each time step — were proposed by Jaeger (1998) as *input-output* OOMs, by Littman et al. (2001) as *predictive state representations* and as a further variant as *transformed PSRs* by Rosencrantz et al. (2004). All approaches are (in the linear case) equivalent and can be easily understood in the framework of linear SSs.

Definition 28 *A* (discrete-valued) controlled (stochastic) process *with input from Σ_I and output in Σ_O is a function $p : \Sigma^* \to [0, 1]$ that satisfies (i) $p(\varepsilon) = 1$ and (ii) $\forall \overline{x} \in \Sigma^*, a \in \Sigma_I : p(\overline{x}) = \sum_{o \in \Sigma_O} p(\overline{x}ao)$, where $\Sigma = (\Sigma_I \times \Sigma_O)$ and $ao = (a, o)$. We define $p(\overline{y}|\overline{x}) = p(\overline{x}\overline{y})/p(\overline{x})$ for $p(\overline{x}) \neq 0$ and zero otherwise. An* input-output OOM *(IO-OOM) is just a SS that models a controlled process.*

Note that the values of p are not probabilities. One may interpret $p(a_1 o_1 \ldots a_n o_n)$ as $P(o_1 \ldots o_n | a_1 \ldots a_n)$, i.e., as the conditional probability of observing the outputs $o_1 \ldots o_n$ given the inputs $a_1 \ldots a_n$. However, one must take care, as the sequence of inputs may depend on the observed outputs as well. This is explained in more detail in Section 4.1.

Definition 29 *Let p be a controlled process with* predictive states $\dot{\omega}_{\overline{h}}$ *defined as $\dot{\omega}_{\overline{h}} = [p(\overline{q}_1|\overline{h}), \ldots, p(\overline{q}_d|\overline{h})]^\top \in \mathbb{R}^d$ for $\overline{h} \in \Sigma^*$ and some fixed set of sequences $\overline{q}_i \in \Sigma^*$. If $\dot{\omega}_{\overline{h}}$ is a sufficient statistic for any history $\overline{h} \in \Sigma^*$, i.e., for every $\overline{x} \in \Sigma^*$ there is a function $m_{\overline{x}} : \mathbb{R}^d \to [0, 1]$ such that $p(\overline{x}|\overline{h}) = m_{\overline{x}}(\dot{\omega}_{\overline{h}})$ for all $\overline{h} \in \Sigma^*$, then the sequences $\{\overline{q}_1, \ldots, \overline{q}_d\}$ are called* core tests*, which together with the* initial state $\dot{\omega}_\varepsilon$ *and* projection functions $m_{\overline{x}}$ *form a d-dimensional* predictive state representation *(PSR) for p. If the projection functions are linear functionals (i.e., just row vectors in $\mathbb{R}^d$), then the PSR is called* linear*.*

Note that PSRs share the notion of "state" with OOMs in that the state consists of the information required to predict the future, but PSRs additionally require the entries of the state vectors $\dot{\omega}_{\overline{h}}$ to be "predictions" $p(\overline{q}_i|\overline{h})$ for the core tests $\overline{q}_i$. Such states are therefore called *predictive states*.

We will only consider linear PSRs for controlled processes here, and show that these are essentially SS for controlled processes (i.e., IO-OOMs) that are additionally interpretable with respect to singleton sets (core tests). Note that there has been some confusion about the precise relationship between PSRs and IO-OOMs, which we address in Sections 3.3.2 and 3.3.3 below.

Proposition 30 *Let a d-dimensional linear PSR consisting of core tests $\overline{q}_i$, projection functions $m_{\overline{x}}$ and an initial state $\dot{\omega}_\varepsilon$ for a controlled process p be given. Then an equivalent SS $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ is obtained by setting*

$$\omega_\varepsilon = \dot{\omega}_\varepsilon, \quad \tau_z = [(m_{z\overline{q}_1})^\top, \ldots, (m_{z\overline{q}_d})^\top]^\top \quad \text{and} \quad \sigma = \sum_{o \in \Sigma_O} m_{ao} \text{ for any } a \in \Sigma_I.$$

Furthermore, $\mathcal{M}$ will be interpretable w.r.t. the sets $\{\overline{q}_i\}$.

Proof First note that $\sigma\dot{\omega}_{\overline{x}} = \sum_{o\in\Sigma_O} m_{ao}\dot{\omega}_{\overline{x}} = \sum_{o\in\Sigma_O} p(ao|\overline{x}) = 1$ for all $\overline{x} \in \Sigma^*$ such that $p(\overline{x}) \neq 0$ because p is a controlled process. Next, we prove that (*) $\omega_{\overline{x}} = p(\overline{x})\dot{\omega}_{\overline{x}}$ and (**) $f_{\mathcal{M}}(\overline{x}) = p(\overline{x})$ by induction on the length l of $\overline{x}$:

- For $l = 0$ we have $\omega_\varepsilon = p(\varepsilon)\dot{\omega}_\varepsilon$ and $f_{\mathcal{M}}(\varepsilon) = \sigma\omega_\varepsilon = \sigma\dot{\omega}_\varepsilon = 1 = p(\varepsilon)$.
- Assume (*) and (**) are true for all $\overline{x} \in \Sigma^l$. Let $\overline{x}z \in \Sigma^{l+1}$. Then (*) $\omega_{\overline{x}z} = \tau_z\omega_{\overline{x}} = \tau_z\dot{\omega}_{\overline{x}}p(\overline{x}) = [p(z\overline{q}_i|\overline{x})]_i^\top p(\overline{x}) = [p(\overline{q}_i|\overline{x}z)]_i^\top p(z|\overline{x})p(\overline{x}) = \dot{\omega}_{\overline{x}z}p(\overline{x}z)$ and (**) $f_{\mathcal{M}}(\overline{x}z) = \sigma\omega_{\overline{x}z} = \sigma\dot{\omega}_{\overline{x}z}p(\overline{x}z) = p(\overline{x}z)$.

Note that property (*) says that $\omega_{\overline{x}} = p(\overline{x})\dot{\omega}_{\overline{x}} = [p(\overline{x}\overline{q}_1), \ldots, p(\overline{x}\overline{q}_d)]^\top$ for all $\overline{x}$, i.e., that $\mathcal{M}$ is interpretable w.r.t. the sets $\{\overline{q}_i\}$. ■

Proposition 31 *Conversely, let $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ be a SS for a controlled process p. Then an equivalent PSR is obtained by making the SS interpretable with respect to singleton sets $\{\overline{y}_i\}$ for appropriate sequences $\overline{y}_i \in \Sigma^*$ (e.g., using Algorithm 3). We can then use these as core tests for the PSR, and set $m_{\overline{x}} = \sigma\tau_{\overline{x}}$ for all $\overline{x} \in \Sigma^*$.*

Proof We assume that the SS has been made interpretable w.r.t. the sequences $\overline{y}_1, \ldots, \overline{y}_d$. Then the normalized states $\dot{\omega}_{\overline{h}} = \omega_{\overline{h}}/\sigma\omega_{\overline{h}}$ have the form $\dot{\omega}_{\overline{h}} = [p(\overline{y}_1|\overline{h}), \ldots, p(\overline{y}_d|\overline{h})]^\top$. Furthermore, for all $\overline{h} \in \Sigma^*$: $m_{\overline{x}}\dot{\omega}_{\overline{h}} = \sigma\tau_{\overline{x}}\dot{\omega}_{\overline{h}} = \sigma\tau_{\overline{x}}\tau_{\overline{h}}\omega_\varepsilon/\sigma\tau_{\overline{h}}\omega_\varepsilon = p(\overline{x}|\overline{h})$, as desired. ■

Corollary 32 *A linear PSR can be specified by the parameters $(\{m_{ao}\}, \{M_{ao}\}, \omega_\varepsilon^\top)$ for $ao \in \Sigma_I \times \Sigma_O$, where $M_{ao} = \tau_{ao}^\top$ and $m_{ao} = (\sigma\tau_{ao})^\top$, and defines a controlled process via*

$$p(a_1o_1 \cdots a_no_n) = \omega_\varepsilon^\top M_{a_1o_1} \cdots M_{a_{n-1}o_{n-1}} m_{a_no_n}.$$

This is the usual way of specifying a PSR.

Note that transformed PSRs (TPSRs) are just PSRs that model controlled processes in the form of Corollary 32 without any further requirements (i.e., without the requirement that the states need to be interpretable). These are readily converted to SSs by setting $\sigma = (\sum_{o\in\Sigma_O} m_{ao})^\top$ for any $a \in \Sigma_I$ and using the equations from the Corollary 32 otherwise. Note that this may not give equivalent models if the PSR does not model a controlled process.

3.3.1 Relation to Partially Observable Markov Decision Processes

Finally, we note how to convert POMDPs into SSs (which can then be further converted to PSRs by making the SS interpretable, as described above). A POMDP with d states $Q = \{s_1, \ldots, s_d\}$ for a controlled process with input alphabet Σ_I and output alphabet Σ_O consists of an initial belief state $b \in \mathbb{R}^d$ whose i-th element is the probability of the model starting in state s_i, a state transition matrix $T_a \in \mathbb{R}^{d \times d}$ for each action $a \in A$ such that the i,j-th entry of T_a is the probability of transitioning to state s_i from state s_j if action a is taken, and a vector $O_{ao} \in \mathbb{R}^d$ for each action-observation pair $ao \in (\Sigma_I \times \Sigma_O)$ whose i-th entry is the probability of observing o after arriving in state s_i by taking action a (Kaelbling et al., 1998).

Setting $O'_{ao} = \operatorname{diag}(O_{ao})$ we can summarize the belief-state update procedure for the POMDP concisely by stating that a POMDP models a controlled stochastic process p via the equation

$$p(a_1 o_1 \cdots a_n o_n) = (1, \ldots, 1)(O'_{a_n o_n} T_{a_n}) \cdots (O'_{a_1 o_1} T_{a_1}) b.$$

Clearly, setting $\sigma = (1, \ldots, 1)$, $\tau_{ao} = O'_{ao} T_a$ and $\omega_\varepsilon = b$ yields an equivalent SS.

3.3.2 IO-OOMs, Interpretable IO-OOMs, PSRs and TPSRs

We have shown above that IO-OOMs, PSRs and TPSRs are equivalent models in the sense that they model the same class of controlled processes and that they can be readily converted into one another. Furthermore, TPSRs are essentially IO-OOMs (except that the evaluation functional σ is replaced by the set $\{m_{ao}\}$ of evaluation functionals), while PSRs are TPSRs (and therefore essentially IO-OOMs) with predictive states, which corresponds to IO-OOMs being interpretable w.r.t. singleton sets (core tests). This is summarized in Table 1.

SSs for controlled processes with...	single evaluation functional σ	set of evaluation functionals $\{m_{ao}\}$
abstract, uninterpretable states	IO-OOMs	TPSRs
predictive states	IO-OOMs that are interpretable w.r.t. singleton sets	PSRs

Table 1: The differences between IO-OOMs, PSRs and TPSRs

Note that we have written "IO-OOMs that are interpretable w.r.t. singleton sets" instead of simply "interpretable IO-OOMs" for a reason. This is because interpretability was originally defined for IO-OOMs in a more restrictive way (cf. Section 3.3.3). It has been shown that not every IO-OOM has an equivalent "interpretable IO-OOM" (in the original sense) (Singh et al., 2004), i.e., that "interpretable IO-OOMs" are less general than IO-OOMs and PSRs. At the same time it was believed that some notion of interpretability would be crucial for the learnability of such models, which is however not the case, as we shall see in Section 4. Together, this has led to the false impression that PSRs are more general than IO-OOMs.

As the original notion of interpretability for IO-OOMs has turned out to be overly restrictive, we propose to employ the notion of interpretability that we have introduced here for SSs as the "correct" notion for IO-OOMs, and consider the original notion as deprecated.

3.3.3 Historical Remarks

The same remarks that we have made above in Section 3.2.3 for OOMs also apply to IO-OOMs. Namely, IO-OOMs were originally required to satisfy (ii)': $\forall a \in \Sigma_I : \sigma \sum_{o \in \Sigma_O} \tau_{ao} = \sigma$ instead of the the property (ii) for a controlled process. This is equivalent for minimal models, but slightly more restrictive in general. However, as every SS can be minimized, this has no practical consequences.

Furthermore, IO-OOMs were originally typically required to satisfy $\sigma = (1, \ldots, 1)$, which is again merely a matter of normalization. However, an IO-OOM that satisfies $\sigma = (1, \ldots, 1)$ can only be interpretable with respect to the sets Y_k, if $1 = \sigma\omega_\varepsilon = (1, \ldots, 1) \cdot [f_\mathcal{M}(Y_i)]_i^\top = \sum_k \sum_{\overline{y} \in Y_k} p(\overline{y})$. It turns out that this can be assured by requiring the sets Y_k to partition $\Sigma_O^l \times \{a_1\} \times \cdots \times \{a_l\}$ for some l and a fixed sequence $a_1 \ldots a_l$ of inputs called a *characterization frame*. This restriction on the choice of sets Y_k therefore became part of the original definition of interpretability for IO-OOMs.

Unfortunately, unlike the case for OOMs, the resulting original notion of interpretability for IO-OOMs has turned out to be a severe limitation (Singh et al., 2004).

However, one may use the more general notion of interpretability given in Definition 14 for IO-OOMs instead, if one is willing to drop the (unnecessary) normalization requirement $\sigma = (1, \ldots 1)$.

3.4 Extensions

In this section we have presented SMAs, OOMs and PSRs as versions of linear sequential systems — or more generally weighted finite automata — that model probabilistic languages, stochastic processes and controlled processes respectively, as is summarized in Figure 1. For completeness, we wish to briefly mention some extensions of these basic model types that have been studied, but which are beyond the scope of this paper.

First of all, various non-linear SSs exists. For instance, several versions of *quantum finite automata* have been studied (Kondacs and Watrous, 1997; Moore and Crutchfield, 2000). One form are SSs $(\sigma, \{\tau_x\}, \omega_\varepsilon \in \mathbb{C}P^d)$ where the operators τ_x are unitary and $\sigma(\tau_{\overline{x}}\omega_\varepsilon) = ||\pi\tau_{\overline{x}}\omega_\varepsilon||^2$ for some projection π and the Fubini-Study metric $||\cdot||$ (Moore and Crutchfield, 2000). A similar type of OOMs exist which are called *norm-OOMs*. These are SSs $(\sigma, \{\tau_x\}, \omega_\varepsilon \in \mathbb{R}^d)$ such that $\sum_{x \in \Sigma} \tau_x^\top \tau_x = I$ and $\sigma(\tau_{\overline{x}}\omega_\varepsilon) = ||\tau_x\omega_\varepsilon||^2$. Such norm-OOMs describe stochastic processes and can always be converted into an equivalent OOM (Zhao and Jaeger, 2010). Recently, *quadratic weighted automata* have been proposed by Bailly (2011), where a SS $\mathcal{M}$ is learnt for $\sqrt{f}$ and a product SS $\mathcal{M} \otimes \mathcal{M}$ is constructed that satisfies $f_{\mathcal{M}\otimes\mathcal{M}} = f_\mathcal{M}^2 \approx f$. All of these approaches avoid the "negative probabilities problem", where the estimated model $f_\mathcal{M}$ may violate the requirement $f_\mathcal{M} \geq 0$. Non-linear versions of PSRs have also been investigated, which have been shown to in some cases yield representations for deterministic dynamical systems that are exponentially smaller than a minimal OOM representation (Rudary and Singh, 2003).

Furthermore, OOMs and PSRs are models for discrete-valued stochastic (controlled) processes. Many real-world processes of interest are, however, continuous-valued. A continuous version of OOMs exists that extends semi-continuous HMMs (Jaeger, 2000a), and WFST have been similarly extended to allow for continuous inputs (Recasens and Quattoni, 2013). Multivariate continuous inputs and outputs are handled using features of observations by reduced-rank HMMs (Siddiqi et al., 2010). So called predictive linear Gaussian models (PLGs), which are based on PSRs, closely resemble linear dynamical system models (Rudary et al., 2005; Wingate and Singh, 2006a,b; Rudary and Singh, 2006, 2008) and are further generalized by exponential family PSRs (Wingate and Singh, 2008b,a). A generalization of OOMs using Hilbert space embeddings was introduced by Song et al. (2010). This has been further refined and extended to include features and can now be employed — among other things — for controlled processes and to planning in reinforcement learning tasks (Boots and Gordon, 2010; Boots et al., 2010, 2013).

4. Learning

In this section we present a general approach to learning SSs from data. We show how several of the learning algorithms that have been proposed for SMA, OOMs and PSRs can be understood in this framework, and that in fact many of the proposed learning algorithms are closely related.

We begin by establishing a result that lies at the heart of the learning algorithms, which was formulated by Kretzschmar (2001) for the case of OOMs. Assuming a function $f_{\mathcal{M}}$ can be described by some minimal SS $\mathcal{M}$, it allows us to reconstruct an equivalent SS $\mathcal{M}'$ from data given in the form of finitely many function values of $f_{\mathcal{M}}$ — as long as these are given exactly and we know the rank d of the underlying model $\mathcal{M}$. We will therefore refer to the Equations (2) as the *learning equations*.

Proposition 33 *For a minimal d-dimensional SS $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$, let $\{\tau_{\overline{x}_j}\omega_\varepsilon \,|\, j \in J\}$ and $\{(\sigma\tau_{\overline{x}_i}^\top)^\top \,|\, i \in I\}$ be finite sets that span the state space W and the co-state space $\tilde{W}$ respectively. Define $F^{I,J} = [f_{\mathcal{M}}(\overline{x}_j\overline{x}_i)]_{(i,j)\in I\times J}$ and $F_z^{I,J} = [f_{\mathcal{M}}(\overline{x}_j z\overline{x}_i)]_{(i,j)\in I\times J}$. Furthermore, define $F^{I,0} = [f_{\mathcal{M}}(\overline{x}_i)]_{i\in I}$ and $F^{0,J} = [f_{\mathcal{M}}(\overline{x}_j)]_{j\in J}^\top$. Let $C \in \mathbb{R}^{d\times|I|}$ and $Q \in \mathbb{R}^{|J|\times d}$ be rank d matrices such that $CF^{I,J}Q$ is invertible. Then the SS $\mathcal{M}' = (\sigma', \{\tau_z'\}, \omega_\varepsilon')$ defined as follows is equivalent to $\mathcal{M}$:*

$$\begin{aligned} \sigma' &= F^{0,J}Q(CF^{I,J}Q)^{-1}, \\ \tau_z' &= CF_z^{I,J}Q(CF^{I,J}Q)^{-1}, \\ \omega_\varepsilon' &= CF^{I,0}. \end{aligned} \tag{2}$$

Furthermore, $CF^{I,J} = (\omega'_{\overline{x}_j})_{j\in J}$ and $CF_z^{I,J} = (\omega'_{\overline{x}_j z})_{j\in J}$, where $\omega'_{\overline{x}} = \tau'_{\overline{x}}\omega'_\varepsilon$ are states of the SS $\mathcal{M}'$.

Proof Let $\Pi = ((\sigma\tau_{\overline{x}_i})^\top)_{i\in I}^\top$, $\Phi = (\tau_{\overline{x}_j}\omega_\varepsilon)_{j\in J}$. Then $F^{I,J} = \Pi\Phi$, $F_z^{I,J} = \Pi\tau_z\Phi$, $F^{I,0} = \Pi\omega_\varepsilon$ and $F^{0,J} = \sigma\Phi$. We can then simply calculate $\tau_z' = C\Pi\tau_z\Phi Q(C\Pi\Phi Q)^{-1} = C\Pi\tau_z(C\Pi)^{-1}$, as well as $\omega_\varepsilon' = C\Pi\omega_\varepsilon$ and $\sigma' = \sigma\Phi Q(C\Pi\Phi Q)^{-1} = \sigma(C\Pi)^{-1}$. That is, we have shown that $\mathcal{M}' = \rho\mathcal{M}\rho^{-1}$ for the non-singular transformation $\rho = C\Pi$. Furthermore, $CF^{I,J} = C\Pi\Phi =$

$\rho\Phi = (\rho\tau_{\overline{x}_j}\omega_\varepsilon)_{j\in J} = (\tau'_{\overline{x}_j}\omega'_\varepsilon)_{j\in J}$, and analogously for $CF_z^{I,J}$. ■

The matrices C and Q that appear in the learning Equations (2) are indeed arbitrary (provided that $CF^{I,J}Q$ has the correct dimension d and full rank), as long as the function values $f_{\mathcal{M}}(\overline{x})$ are given exactly. However, if one only has access to estimates $\hat{f}(\overline{x})$, then the selection of C and Q plays a crucial role in obtaining good model estimates, as will be further discussed in Section 4.4.

Furthermore, note that we generally do not know a priori which sets of words to consider such that $\{\tau_{\overline{x}_j}\omega_\varepsilon \mid j \in J\}$ and $\{(\sigma\tau_{\overline{x}_i}^\top)^\top \mid i \in I\}$ span the state and co-state spaces W and $\tilde{W}$ of $\mathcal{M}$. Proposition 6 guarantees that it suffices to consider all words of length at most d, but the rank d of $\mathcal{M}$ is generally unknown as well. Selecting appropriate sets of words $\overline{x}_i$ and $\overline{x}_j$ and an appropriate model dimension d are therefore crucial and non-trivial steps in learning models from data.

We can turn the above Proposition 33 into a generic learning procedure for SSs:

Algorithm 4: General procedure for learning a SS from data

1. Obtain estimates $\hat{f}(\overline{x})$ of the function values $f(\overline{x})$ for words $\overline{x} \in \Sigma^*$.
2. Choose finite sets $\{\overline{x}_j \mid j \in J\}, \{\overline{x}_i \mid i \in I\} \subset \Sigma^*$, which we call sets of *indicative* and *characteristic* words respectively. Then assemble the estimates $\hat{f}(\overline{x})$ into estimates of the matrices $\hat{F}^{I,J}$, $\hat{F}_z^{I,J}$, $\hat{F}^{I,0}$ and $\hat{F}^{0,J}$.
3. Find a reasonable target dimension d for the model to be learnt.
4. Choose $C \in \mathbb{R}^{d\times|I|}$ and $Q \in \mathbb{R}^{|J|\times d}$ called the *characterizer* and *indicator*, such that $C\hat{F}^{I,J}Q$ is invertible.
5. Apply the learning Equations (2) to obtain a model estimate $\hat{\mathcal{M}}$.

At this point we should clarify what is meant here by learning a model from data. For general MA the goal is often to reconstruct an automaton from as few *membership queries* — obtaining the value $f(\overline{x})$ for some $\overline{x} \in \Sigma^*$ — and *equivalence queries* — proposing a function h and receiving a *counterexample* $\overline{x}$ such that $h(\overline{x}) \neq f(\overline{x})$ if $h \neq f$ — as possible. This is an extended version of the exact learning model of Angluin (1987). However, in the case of SMA, OOMs and PSRs, the external function represents a distribution. Therefore, in these cases it is usual to assume that we observe samples from this distribution and wish to estimate model parameters from the given samples such that the estimated model best describes the underlying distribution — "best" in a sense that depends on the context and the approach taken by a specific learning algorithm.

We should also mention one common problem when learning SMAs, OOMs and PSRs from data. Namely, even if the function $f_{\mathcal{M}}$ in question can be described by a SMA, OOM or PSR model $\mathcal{M}$, the learnt model $\hat{\mathcal{M}}$ will only be an approximation to $\mathcal{M}$ and will describe a function $f_{\hat{\mathcal{M}}}$ that may not satisfy the properties of a probabilistic language, stochastic process or controlled process, respectively, i.e., the learnt model $\hat{\mathcal{M}}$ may *not* be a SMA, OOM or PSR. What typically happens is that the learnt model $\hat{\mathcal{M}}$ will predict "negative probabilities" for certain sequences $\overline{x}$. Moreover, it is an undecidable problem whether a

given SS $\hat{\mathcal{M}}$ satisfies $f_{\hat{\mathcal{M}}} \geq 0$, and therefore, whether it is a SMA — a result that carries over to OOMs and PSRs as well (Wiewiora, 2008). In practice, there are three basic ways to deal with this "negative probabilities problem": First of all, one can resort to alternative models as described in Section 3.4 that preclude the problem by design. For the particular case of quadratic weighted automata the learning procedure presented here still applies (Bailly, 2011), but in general one will need alternative learning algorithms. Secondly, one may attempt to learn a restricted class of SS such as PFA, HMMs or POMDPs by enforcing additional constraints on the parameters of the SS. This can be achieved either by adding a set of convex constraints to a generalized version of the spectral learning method presented in Section 4.4.2 (Balle et al., 2012), or by an additional conversion step (Anandkumar et al., 2012), which however may fail. Finally, one may work with such an "invalid" SS model by employing a simple and effective heuristic as described by Jaeger et al. (2006b, Appendix J) to normalize all model predictions to fall into the desired range.

Finally, we will briefly remark on the runtime characteristics of the above learning procedure. Steps 1 and 2 can be accomplished in time $\mathcal{O}(N)$, where N is the size of the training data, for most strategies mentioned in Section 4.2 by employing a suffix tree or similar representation of the training data. For a given target dimension d, Step 4, when solved via the EC (Section 4.4.3) or spectral algorithms (Section 4.4.3), requires the $\mathcal{O}(d|I||J|)$ computation of a d-truncated singular value decomposition (SVD) of $\hat{F}^{I,J}$, while the ES algorithm (Section 4.4.4) requires $\mathcal{O}(d^2 l \max\{|I|, |J|\})$ operations to compute C, where l is the (generally very small) average length of characteristic and indicative words, and $\mathcal{O}(d|I||J|)$ operations to compute Q — per iteration (but one typically uses a constant number of iterations), which therefore amounts to a run-time of $\mathcal{O}(d|I||J|)$ as well. Solving the learning Equations (2) for Step 5 essentially requires the computation of the operators $\hat{\tau}_z$, which costs $\mathcal{O}(d|I||J||\Sigma|)$ operations. So for a known target dimension d, the above learning procedure typically requires $\mathcal{O}(N + d|I||J||\Sigma|)$ operations. Step 3 can be solved by computing a $d_{\max}$-truncated SVD of $\hat{F}^{I,J}$ for some upper bound $d_{\max} < \min\{|I|, |J|\}$ on the target dimension, which incurs a runtime costs of $\mathcal{O}(d_{\max}|I||J|)$, or by using cross-validation, which requires repeatedly performing, for various choices of d, Steps 4 and 5 as well as evaluations on test data of size T, which we assume to be constant, incurring a runtime cost of $\mathcal{O}(d \log(d)|I||J||\Sigma|)$, where d is the finally selected model dimension.

In the following, we will discuss the steps of the learning procedure in more detail.

4.1 Obtaining Estimates $\hat{f}(\overline{x})$

This step clearly depends on the context we are dealing with. Recall that in the context of SMA, the functions we are considering are distributions on words, while in the context of OOMs and PSRs they represent stochastic processes and controlled processes respectively. The following Remarks 34 to 36 summarize how to obtain these estimates in the different scenarios of probabilistic languages, stochastic processes and controlled processes, respectively.

Remark 34 *Let $f : \Sigma^* \to [0, 1]$ be a distribution on Σ^*, and let $S = (s_1, s_2, \ldots, s_N)$ be a collection of N samples from f. Then $\hat{f}(\overline{x}) = \frac{\#(\overline{x})}{N}$, where $\#(\overline{x})$ denotes the number of occurrences of $\overline{x}$ in the sample S, is a consistent estimator for $f(\overline{x})$.*

In the case of stochastic processes, one typically observes few (or even just one) long initial realization of the process. In this case it is still possible to obtain the desired estimates if the stochastic process is stationary and ergodic[2] by invoking the ergodic theorem and using time-averages as estimates. The same idea is commonly used in the case of controlled processes as well and called *suffix-history* method in the PSR community.

Remark 35 *Let $f : \Sigma^* \to [0,1]$ be a stationary and ergodic stochastic process, and let $\bar{s} = s_1 s_2 \dots s_N$ be a finite initial realization of length N from this process. Then*

$$\hat{f}(\overline{x}) = \frac{\#(\overline{x})}{N - |\overline{x}| + 1},$$

where $\#(\overline{x})$ denotes the number of occurrences of $\overline{x}$ in the sequence $\bar{s}$ is a consistent estimator for $f(\overline{x})$.

In the case of controlled processes the situation is more complicated. It is important to have a good understanding of the meaning of the value $f(\overline{x})$ when f is a controlled process and $\overline{x} = a_1 o_1 \dots a_n o_n \in (\Sigma_I \times \Sigma_O)^n$ is some input-output sequence. Intuitively, this is the probability of the system output $o_1 \dots o_n$ conditioned on the system input $a_1 \dots a_n$. This is sometimes written as $f(a_1 o_1 \dots a_n o_n) = P(o_1 \dots o_n \,|\, a_1 \dots a_n)$ even though this notation is misleading, as it suggests that $P(o_1 \dots o_n \,|\, a_1 \dots a_n) = \frac{P(a_1 o_1 \dots a_n o_n)}{P(a_1 \Sigma_O \dots a_n \Sigma_O)}$, which is false (Bowling et al., 2006). To clarify this, consider the stochastic process that is specified by the controlled process f *together with some system input specification.* This stochastic process is governed by probabilities of the form

$$P(a_1 o_1 \dots a_n o_n) = \prod_{k=1}^{n} P(o_k \,|\, a_1 o_1 \dots a_k) \cdot \prod_{k=1}^{n} P(a_k \,|\, a_1 o_1 \dots a_{k-1} o_{k-1}).$$

The second factor in the equation models the system input and is sometimes called the *input policy* π, while the first factor models the system output and is just the controlled process f. Therefore, for $\overline{x} = a_1 o_1 \dots a_n o_n$,

$$f(\overline{x}) = P(o_1 \dots o_n \,|\, a_1 \dots a_n) = \prod_{k=1}^{n} P(o_k \,|\, a_1 o_1 \dots a_k) = \frac{P(\overline{x})}{\pi(\overline{x})}. \tag{3}$$

Note that for the special case of a *blind* input policy π — one that does not depend on the observed output, i.e., that satisfies $P(a_k \,|\, a_1 o_1 \dots a_{k-1} o_{k-1}) = P(a_k \,|\, a_1 \dots a_{k-1})$ for all $\overline{x}$ — we in fact do have $\pi(\overline{x}) = P(a_1 \Sigma_O \dots a_n \Sigma_O)$.

From the above Equation (3), the following estimates are derived (Bowling et al., 2006):

Remark 36 *Let $f : \Sigma^* \to [0,1]$ be a controlled process, and let $\bar{s} = a_1 o_1 \dots a_N o_N$ be a finite initial sample from f according to some input policy π, such that the resulting stochastic process is stationary and ergodic. Then*

$$\hat{f}(\overline{x}) = \prod_{k=1}^{n} \frac{\#(a_1 o_1 \dots a_k o_k)}{\#(a_1 o_1 \dots a_k)}$$

2. A stationary ergodic process is a stochastic process where the statistical properties do not change with time (stationarity) and where these can be estimated as time-averages from a single long sample (ergodicity). For details, see for example the textbook by Gray (1988)

is a consistent estimator for $f(\overline{x})$. If the input policy π is known, then

$$\hat{f}(\overline{x}) = \frac{\#(\overline{x})}{N - |\overline{x}| + 1} \cdot \frac{1}{\pi(\overline{x})}$$

is also a consistent estimator which may be used instead. Again, $\#(\overline{x})$ denotes the number of occurrences of $\overline{x}$ in the sequence $\bar{s}$.

None of the above estimates exploits the rich structure of the matrix F. If required, some of the convex constraints that the matrix F must satisfy can be ensured by applying an additional normalization step to the estimated matrix $\hat{F}$, as done by McCracken and Bowling (2006). These convex constraints — including a convex relaxation of the rank constraint — may also be used to infer missing values if some entries $\hat{f}(\overline{x})$ cannot be obtained directly, which becomes relevant in the context of learning more general (e.g., non-stochastic) weighted automata (Balle and Mohri, 2012), or to infer sequence alignment when learning WFST from unaligned input-output sequences (Bailly et al., 2013).

4.2 Choosing Indicative and Characteristic Words

Choosing indicative and characteristic words $\{\overline{x}_j \,|\, j \in J\}, \{\overline{x}_i \,|\, i \in I\} \subset \Sigma^*$ is equivalent to selecting which columns J and rows I of the system matrix F to estimate. Clearly, it is only possible to obtain a correct estimate for f if I and J are selected such that $\text{rank}(F) = d = \text{rank}(F^{I,J})$. It is however unclear how to satisfy this if the true rank is unknown or even impossible if $\text{rank}(F) = \infty$ — as may often be the case for real-world examples. Determining an appropriate rank for the model will be discussed in the following section.

One approach is, however, to attempt to select minimal sets of indicative and characteristic words such that $\text{rank}(F) = \text{rank}(F^{I,J})$. Such minimal sets are called sets of *core histories* and *core tests* in the context of PSRs, and their selection is called the *discovery problem*. This problem is easily solved by Algorithm 1 once a (minimal) SS model for f is known. For the case where only function values of f are available, an iterative procedure has been proposed (James and Singh, 2004) that, starting with the empty words, adds in each iteration all length-one extensions of previously found core histories and tests, but retains only a minimal set needed to span $\hat{F}^{I,J}$. Since any noisy matrix is typically non-singular, some notion of numerical linear independence is used to decide which words to retain in each step. It is important to note that there exist simple examples of finite rank where this iterative procedure fails to deliver sets of core histories and tests (James and Singh, 2004), i.e., it does not in general solve the discovery problem. A similar algorithm called DEES has been proposed in the context of learning SMA (Denis et al., 2006). The algorithms for learning MA in the exact learning framework also work by finding a minimal set of indicative and characteristic words, but there it is assumed that the function f may be queried exactly, and furthermore equivalence queries are employed to find additional core tests and histories (Ohnishi et al., 1994; Bergadano and Varricchio, 1994; Beimel et al., 2000).

It is important to note that there is no requirement to find minimal or even small sets of indicative and characteristic words, i.e., one does not need to solve the discovery problem when learning SS models from data (and once a SS model has been learnt, the problem is easily solved by Algorithm 1). In fact, using small such sets means that less of the available

training data will enter the model estimation, i.e., the available data will be under-exploited. It is therefore desirable to use (much) larger sets of indicative and characteristic words than strictly needed.

An approach which is in some sense complementary is to use all sequences of a given length l. By Proposition 6 one can ensure $\text{rank}(F^{I,J}) = d$ by choosing $l \geq d$. However, this is highly impractical, since the size of $\hat{F}^{I,J}$ grows exponentially with l. Also, many of the estimates in $\hat{F}^{I,J}$ will be based on very few — if any — occurrences in the available training data. Nevertheless, choosing a length $l \ll d$ and utilizing as indicative as well as characteristic words all words of length l that occur at least once in the training data often gives good results (Zhao et al., 2009a).

A further approach is to select as indicative and characteristic words all those that actually occur in the data and therefore allow data-based estimates (Bailly et al., 2009). However, it is reasonable to disallow indicative (resp. characteristic) words that are suffixes (resp. prefixes) of some other indicative (resp. characteristic) word if they always occur at the same positions in the training data, as these would just lead to identical columns (resp. rows) in the estimated matrices that are based on the same parts of the training data (Jaeger et al., 2006b). Moreover, one may select only the words that occur most frequently in the data (Balle et al., 2014). These approaches yield a choice of indicative and characteristic words that is matched to the available training data and can be computed in time $\mathcal{O}(N)$ where N is the size of the training data by using a suffix tree or similar representation of the training data.

Finally, it is also possible to group words into sets of words (as is also done in Definition 14) that we call *events*, and to use indicative and characteristic events in place of words. This corresponds to adding the respective columns and rows in the matrices $\hat{F}^{I,J}, \hat{F}_z^{I,J}$, etc. and can be formally accomplished by a special selection of the indicator and characterizer matrices Q and C. Finding good indicative and characteristic events was the strategy adopted by early OOM learning algorithms (Jaeger, 2000b). A further generalization of this idea of considering events in place of words is proposed by Wingate et al. (2007). Using such events may carry an additional advantage if the estimation of $\hat{f}(Y)$ from the available data can be performed more efficiently or accurately than computing $\hat{f}(Y) = \sum_{\overline{x} \in Y} \hat{f}(\overline{x})$.

4.3 Determining the Model Rank

We should note that the goal of this step may be stated in two different ways. First of all, we may be interested in estimating the true rank of the external function f and use this as the model rank. On the other hand, we may rather be interested in choosing any model rank that allows for a good approximation of the external function f from the available data. These goals are related, as one can only hope to estimate an exact model if the model rank is at least $\text{rank}(f)$. However, they are not the same, and it depends on the context which approach is most appropriate. For instance, if it is known that the external function f must have a small finite rank, which may even carry some meaning, it may be desirable (and well-defined) to estimate this true rank from the data. On the other hand, when dealing with real-world systems of possibly infinite rank, and faced with generally limited training data, it may not even make sense to speak of the correct model rank. In such cases one will typically use the second approach, which is really an instance of the bias-variance dilemma.

4.3.1 Estimating the True Rank

For suitably chosen indicative and characteristic words, one can expect to have $\text{rank}(f) = \text{rank}(F^{I,J})$. However, since one only has access to an estimate $\hat{F}^{I,J}$ of this matrix, a typical approach is to determine what is known as the *numerical rank* (or *effective rank* or *pseudorank*). We give a brief description following Hansen (1998).

The *numerical ε-rank* r_ε of a matrix A may be defined as the smallest rank of any matrix that can be obtained from A by a small perturbation E of size at most ε:

$$r_\varepsilon(A) = \min_{||E|| \leq \varepsilon} \text{rank}(A + E).$$

In terms of the singular values $\sigma_1 \geq \cdots \geq \sigma_K$ of A this means that r_ε satisfies $\sigma_{r_\varepsilon} > \varepsilon \geq \sigma_{r_\varepsilon+1}$ if the size of the perturbation E is measured by the spectral norm $||\cdot||_2$, or alternatively that r_ε is the smallest k such that $\sum_{i=k+1}^{K} \sigma_i^2 \leq \varepsilon^2$ if the Frobenius norm $||\cdot||_F$ is used instead. Both criteria can be used to determine r_ε.

Assuming that A is only an estimate of an underlying matrix $\tilde{A}$, it makes sense to choose ε to be of the same order as the expected size of the error, i.e., $\varepsilon \approx E[||A - \tilde{A}||]$. The numerical rank of A is then $r_\varepsilon(A)$ for some reasonable choice of ε. Note that the notion of numerical rank makes sense if the errors on matrix entries of A are of comparable magnitudes and can be reasonably quantified, and if there is a significant *gap* between σ_{r_ε} and $\sigma_{r_\varepsilon+1}$. Otherwise, the numerical rank is somewhat arbitrary. It is furthermore important to note that the numerical rank measures how many dimensions can be significantly distinguished from noise. It is therefore only a lower bound for the true rank of the underlying matrix.

The main difficulty in determining the numerical rank of the matrix $\hat{F}^{I,J}$ therefore lies in finding a suitable ε. This may be approached by obtaining estimates for or bounds on the variances of the individual matrix entries (Jaeger, 1998; James and Singh, 2004), which may, however, differ widely across $\hat{F}^{I,J}$. These approaches will therefore lead to very conservative estimates of the rank. Still, these estimates will be consistent, i.e., will converge to the true rank in the limit of infinite training data.

Independent of such error estimates it may be reasonable to assume that there will be a relative "gap" between σ_{d+1} and σ_d in the singular value spectrum of $\hat{F}^{I,J}$ around the true rank $d = \text{rank}(F^{I,J})$. A recently proposed method searches for such a gap starting from σ_{r_ε}, where the numerical rank r_ε of $\hat{F}^{I,J}$ is used as a lower bound for the true rank (Bailly et al., 2009).

4.3.2 Finding a Suitable Model Rank

Intuitively speaking, the model rank should be chosen sufficiently large to be able to represent the complexity of the data, but not too large, as otherwise overfitting results.

One standard approach is to use cross-validation. For this, one needs to split the available data into training and test data. One then estimates models of various ranks from the training data and evaluates these on the test data, for instance by calculating the log likelihood of the test data under the models. Finally, one chooses the model rank that gives the best performance. Care must be taken when estimating models for controlled or stochastic processes from one long training sequence $\bar{s}$, as this sequence cannot be partitioned arbitrarily into training and test sets, and the distribution over future observations given a

history of observations at some time t may differ from the initial distribution. Additionally, performing cross-validation is computationally intense.

In comparison, the above methods based on calculating the numerical rank of $\hat{F}^{I,J}$ are elegant algebraic approaches to the problem. Recall that the numerical rank will reflect the number of dimensions present in the training data that can be distinguished from noise. It is therefore reasonable to postulate that the numerical rank of $\hat{F}^{I,J}$ might be a well-suited choice for the model dimension.

Interestingly, though, there is some evidence that at least the EC and spectral learning procedures described in the following section do not seem to suffer much from overfitting (Zhao et al., 2009a). In practical applications it may therefore be viable to simply pre-select a high model dimension.

Deeper insight into this crucial part of the learning procedure is unfortunately lacking. Further research into this question is therefore needed.

4.4 Selecting the Characterizer and Indicator

The effect of the characterizer C and indicator Q is to reduce the available data in $\hat{F}^{I,J}$, $\hat{F}_z^{I,J}$, $\hat{F}^{I,0}$ and $\hat{F}^{0,J}$ to a d-dimensional representation, where d is the chosen target dimension for the model to be learnt.

Assuming that $d = \text{rank}(F) = \text{rank}(CF^{I,J}Q)$, the matrices $CF^{I,J}Q, CF_z^{I,J}Q, CF^{I,0}$, and $F^{0,J}Q$ together contain the same information as F and are sufficient to reconstruct a SS model for f via the learning Equations (2). The requirement that $CF^{I,J}Q$ must have full rank d therefore ensures that no information is lost.

In fact — provided that $CF^{I,J}Q$ has full rank d — really any choice of characterizer and indicator *may* be used and will lead to a consistent model estimation, i.e., a correct model will be obtained in the limit of infinite training data. Hamilton et al. (2013) show that for certain dynamical systems a random choice of characterizer C does indeed work well.

However, in general the choice of characterizer C and indicator Q is central to achieving statistical efficiency, i.e., making efficient use of the available training data. This step lies at the heart of the learning procedure, and in fact much research — even if not explicitly stated — can be seen as optimizing this step of the learning algorithm.

4.4.1 By Selection / Grouping of Rows and Columns of $\hat{F}$

It is important to note that the choice of indicative and characteristic words discussed in Section 4.2 can be viewed equivalently as a special choice of characterizer and indicator. To see this, assume one could estimate the entire matrix $\hat{F}$ from data. Then any selection of rows I and columns J from $\hat{F}$ can be achieved by characterizer and indicator matrices C, Q of the form $C = C'C^I$ and $Q = Q^J Q'$, where C^I and Q^I are appropriate binary matrices with a single one entry in the corresponding columns or rows, and zeros otherwise, such that $C^I \hat{F} Q^J = \hat{F}^{I,J}$. This can easily be extended to account for groupings of words into events by allowing several one entries per column / row of C^I, Q^J respectively.

One advantage of this point of view is that this immediately justifies grouping of words into events, as suggested in Section 4.2. But more importantly, this highlights that choosing indicative and characteristic words as described in Section 4.2 is in fact a restricted approach to the more general problem of finding appropriate characterizer and indicator matrices. We

argue that a good choice of characterizer and indicator is the key to achieving high statistical efficiency of the learning procedure and that therefore the (pre-)selection of indicative and characteristic words should be guided by trying to retain as much information from the available training data as possible. In other words, the (pre-)selection of indicative and characteristic words in Section 4.2 is primarily a practical necessity that should rather be seen as discarding rows and columns from $\hat{F}$ that carry only little or no information.

4.4.2 Spectral Methods

Recall that the j-th columns of the matrices F and F_z correspond to the functions $f_{\overline{x}_j}$ and $f_{\overline{x}_j z}$, and that the operator τ_z of any minimal model $\mathcal{M}$ for f — regarded as a linear operator $\tilde{\tau}_z$ on the space $\mathcal{F}$ — satisfies $\tilde{\tau}_z(f_{\overline{x}_j}) = f_{\overline{x}_j z}$ (cf. Proposition 1). The matrix τ_z is just a representation of this operator with respect to some basis of $\mathcal{F}$. We can therefore regard the columns of F and F_z as argument-value pairs for the operator $\tilde{\tau}_z$, from which we can recover $\tilde{\tau}_z$. To obtain a matrix representation τ_z, we need to fix some basis for the column space $\mathcal{F}$, which corresponds to mapping the columns of F and F_z to $\mathbb{R}^d$ — this is accomplished by the characterizer C.

We are only given estimates $\hat{F}^{I,J}$ and $\hat{F}_z^{I,J}$. The idea of the spectral methods is to find an estimate of the column space $\hat{\mathcal{F}}$ by projecting the columns of $\hat{F}^{I,J}$ and $\hat{F}_z^{I,J}$ to a best rank d representation (best in the least squares sense). This is accomplished by the d-truncated SVD. We then estimate the matrices $\hat{\tau}_z$ via least squares linear regression from the so obtained argument-value pairs. Note that the column space $\mathcal{F}$ is already spanned by the columns of $F^{I,J}$ — if I and J are chosen appropriately — and we may therefore base the estimate of the principal subspace $\hat{\mathcal{F}}$ on the estimate $\hat{F}^{I,J}$ only. Formally, this means:

Algorithm 5: Spectral method for computing characterizer C and indicator Q

1 Compute $U_d S_d V_d^\top$, the d-truncated SVD of $\hat{F}^{I,J}$.
2 Set $C = U_d^\top$ and $Q = (C\hat{F}^{I,J})^\dagger = V_d S_d^\dagger$.

Note that $U_d S_d V_d^\top$ indeed gives the best rank d approximation to $\hat{F}^{I,J}$ with respect the Frobenius norm by the Eckart-Young theorem (Eckart and Young, 1936). However, the matrix $F^{I,J}_{\hat{\mathcal{M}}}$ reconstructed via the so learnt model $\hat{\mathcal{M}}$ — which will clearly have rank at most d — will in general *not* be a best rank d approximation to $\hat{F}^{I,J}$. This is due to the fact that constructing $F^{I,J}_{\hat{M}}$ from the model $\hat{\mathcal{M}}$ enforces additional structure. Interestingly, we have observed that the reconstructed matrix $F^{I,J}_{\hat{\mathcal{M}}}$ is often a better approximation to the *true* matrix $F^{I,J}$ than either of $\hat{F}^{I,J}$ and its best rank d approximation.

This spectral approach is often referred to as principal component analysis (PCA). However, PCA typically involves mean-centering the data first. PCA projects the data onto a d-dimensional *affine* subspace that contains the data mean, while here we know that the data $\hat{F}^{I,J}$ lie approximately on a true subspace (even though they do not have zero mean). Mean-centering the data is therefore inappropriate in this context — nevertheless, it it sometimes done anyway (Bailly et al., 2009). To avoid confusion, we refer to learning algorithms based on this idea simply as spectral learning algorithms (Rosencrantz et al., 2004; Hsu et al., 2009; Bailly et al., 2009; Siddiqi et al., 2010; Boots and Gordon, 2010; Bailly, 2011; Balle et al., 2011, 2014). Furthermore, an online version of this spectral

learning algorithm has been developed by Boots and Gordon (2011), whereas a modification that combines the subspace estimation step (determining the characterizer C) and linear regression step (solving the learning Equations 2) into a single optimization problem is given by Balle et al. (2012).

Clearly, these methods are motivated by trying to find a model $\hat{\mathcal{M}}$ of rank d such that its external function $f_{\hat{\mathcal{M}}}$ best approximates the estimated external function $\hat{f}$. To make this precise, one needs to define a distance measure on functions in $\mathbb{R}\langle\langle\Sigma\rangle\rangle$. In the case of stochastic languages the functions all lie in the Hilbert space $l_2(\Sigma^*)$ and the metric of this function space may be used. For stochastic processes, a natural choice may be the cross-entropy. This will be related to finding a maximum-likelihood estimate of model parameters from data. So far, none of these questions has been resolved. However, *sample complexity* results that fall into the probably approximately correct (PAC) learning framework (Valiant, 1984) are available for several spectral learning algorithms (Hsu et al., 2009; Bailly et al., 2009; Siddiqi et al., 2010; Bailly, 2011). These give bounds on the number or size N of samples that are required to obtain a model estimate $\mathcal{M}$ that is approximately correct (i.e., such that $|f_{\mathcal{M}} - f| < \varepsilon$ for a given ε and a specified distance measure) with probability at least $1 - \delta$ for a given δ. Typically, the required size N is shown to be polynomial in the PAC parameters $1/\epsilon$ and $1/\delta$, as well as other parameters that depend on f such as the alphabet size $|\Sigma|$ and the rank of f.

Finally, we mention a shortcoming of the spectral methods as they are commonly used. They implicitly assume that the variances of the estimates $\hat{f}(\overline{x}_j\overline{x}_i)$ are all of the same order. This, however, is clearly not the case, which suggests that replacing the SVD computation by a weighted low-rank matrix approximation (Markovsky and Huffel, 2007a) and the linear regression of the learning Equations (2) by weighted total least squares (Markovsky and Huffel, 2007b) may give better results, as long as weights that reflect the precision of the estimates $\hat{f}(\overline{x})$ can be estimated reliably from the available data. In fact, if the variances $\mathrm{Var}(\hat{f}(\overline{x}_j\overline{x}_i))$ can be estimated and — even approximately — factored as $\mathrm{Var}(\hat{f}(\overline{x}_j\overline{x}_i)) = v_j w_i > 0$, then this leads to a simple row and column weighted spectral learning method:

Algorithm 6: Row and column weighted spectral learning

1 Let $D_I = [\mathrm{diag}(w_i)_{i\in I}]^{-\frac{1}{2}}$ and $D_J = [\mathrm{diag}(v_j)_{j\in J}]^{-\frac{1}{2}}$ be suitable row and column weight matrices
2 Let $\tilde{F}^{I,J} = D_I \hat{F}^{I,J} D_J$ and $\tilde{F}_z^{I,J} = D_I \hat{F}_z^{I,J} D_J$
3 Let $\tilde{U}_d \tilde{S}_d \tilde{V}_d^\top$ be the d-truncated SVD of $\tilde{F}^{I,J}$
4 Let $C = \tilde{U}_d^\top D_I$ and $Q = D_J (C\tilde{F}^{I,J} D_J)^\dagger = D_J \tilde{V}_d \tilde{S}_d^\dagger$.

We mention this particular row and column weighted approach here, as it is simple, effective, and we will show that it is closely related to the ES approach described in Section 4.4.4.

4.4.3 The EC Algorithm

The error controlling (EC) approach selects characterizer and indicator matrices C and Q that minimize an error bound for the *relative approximation error* of the estimated model parameters (Zhao et al., 2009a). This algorithm was originally formulated for OOMs only, and made use of the normalization $\sigma = (1, \ldots, 1)$ that is often used in the context of OOMs.

This in turn imposed additional restrictions on the admissible selections of indicative and characteristic words. Here, we present a more general and yet simplified EC approach that eliminates these restrictions and applies to learning SMA, OOMs, IO-OOMs and PSRs alike.

To formalize this, first assume we have fixed C and Q, and derived estimated operators $\hat{\tau}_z$ and correct operators τ_z from the estimates $\hat{F}^{I,J}$, $\hat{F}_z^{I,J}$ and the correct matrices $F^{I,J}$, $F_z^{I,J}$ respectively using the learning Equations (2). Note that these depend on the choice of C and Q. To write things more concisely, denote the matrix obtained by stacking the τ_z operators by $\tau_* = [\tau_{z_1}; \ldots; \tau_{z_l}]$ (using MATLAB notation), where $\Sigma = \{z_1, \ldots, z_l\}$, and $\hat{\tau}_* = [\hat{\tau}_{z_1}; \ldots; \hat{\tau}_{z_l}]$. Similarly, construct the matrices $F_*^{I,J}$ and $\hat{F}_*^{I,J}$ by stacking the $F_z^{I,J}$ and $\hat{F}_z^{I,J}$ respectively.

Proposition 37 *For a given choice of C and Q, and using the above definitions, the estimate $\hat{\tau}_*$ has a* relative approximation error

$$\frac{\|\tau_* - \hat{\tau}_*\|_F}{\|\tau_*\|_F} \leq \kappa \left(\|F^{I,J} - \hat{F}^{I,J}\|_F + \frac{\sqrt{l}}{\rho(\tau_\Sigma)} \|F_*^{I,J} - \hat{F}_*^{I,J}\|_F \right),$$

where $\rho(\tau_\Sigma)$ is the spectral radius of the matrix τ_Σ, which is independent of the choice of C and Q, and $\kappa = \|C\|_F \|Q(C\hat{F}^{I,J}Q)^{-1}\|_F$.

This is a slightly improved and more general version of the central Proposition 3 presented in (Zhao et al., 2009a). For completeness, the proof is given in the appendix.

The EC algorithm then selects C, Q in such a way that the quantity κ is minimized, which is equivalent to the optimization problem

$$(C, Q) = \underset{(C,Q)}{\operatorname{argmin}} \{\|C\|_F \|Q\|_F : C\hat{F}^{I,J}Q = I_d\}, \tag{4}$$

since every (C, Q) that minimizes κ gives a solution (C, Q') to Equation (4) by substituting $Q' = Q(C\hat{F}^{I,J}Q)^{-1}$ and noting $(C\hat{F}^{I,J}Q') = I_d$. This optimization problem can be solved efficiently by the following iterative procedure (Zhao et al., 2009a):

Algorithm 7: The C, Q optimization resulting from the EC approach

initialize $C \in \mathbb{R}^{d \times |I|}$ randomly
repeat
  $Q = (C\hat{F}^{I,J})^\dagger, \quad C = (\hat{F}^{I,J}Q)^\dagger$
until *convergence of* $\|C\|_F \|Q\|_F$

Although not previously realized, this turns out to be related to a well-known EM-based algorithm for principal component analysis for which it is known that the rows of C (upon convergence) will span the space of the first d principle components of $\hat{F}^{I,J}$ (Roweis, 1998). We can use this relationship to gain the following insight.

Proposition 38 *Assuming the model rank d is chosen such that the singular values σ_i of $\hat{F}^{I,J}$ satisfy $\sigma_d > \sigma_{d+1}$, the EC algorithm as presented here and the spectral method presented in the previous section will lead to equivalent models.*

Proof Note that the condition $\sigma_d > \sigma_{d+1}$ merely says that $\operatorname{rank}(\hat{F}^{I,J}) \geq d$ and that the d-dimensional principal subspace of $\hat{F}^{I,J}$ is unique. Let C and $Q = (C\hat{F}^{I,J})^\dagger$ be the characterizer and indicator obtained by the spectral method, and let C' and $Q' = (C'\hat{F}^{I,J})^\dagger$ be the result of the above iterative procedure after convergence. Then the rows of C and C' will each span the same d-dimensional space (Roweis, 1998). This means that $C = \rho C'$ for some non-singular $\rho \in \mathbb{R}^{d\times d}$, and therefore $Q = (\rho C'\hat{F}^{I,J})^\dagger = (C'\hat{F}^{I,J})^\dagger\rho^{-1} = Q'\rho^{-1}$. By Proposition 12 the learning Equations (2) will result in equivalent models. ■

In fact, the above optimization problem can also be solved non-iteratively by a d-truncated SVD. This is a new result for which we give the full proof in the appendix:

Proposition 39 *Let $U_d S_d V_d^\top \approx \hat{F}^{I,J}$ be the d-truncated SVD of $\hat{F}^{I,J}$. Then $C^* = S_d^{-\frac{1}{2}} U_d^\top$ and $Q^* = (C^*\hat{F}^{I,J})^\dagger = V_d S_d^{-\frac{1}{2}}$ are a solution to the optimization problem in Equation (4) — provided a solution exists at all, i.e.,* $\operatorname{rank}(\hat{F}^{I,J}) \geq d$.

Clearly, this solution (C^*, Q^*) will again yield an equivalent model. Finally, we note that other versions of bounds on the relative approximation error than given in Proposition 37 may be considered instead, which can lead to choices of C and Q that give non-equivalent models. The performance of these seems to be comparable, though (Zhao et al., 2009b).

4.4.4 Efficiency Sharpening

The ES algorithm has previously been worked out only for the case of stationary stochastic processes and "traditional" OOMs where $\sigma = (1, \ldots, 1)$. Here we give an account of the ES principle that is more general than in the original work, and we establish connections to the spectral algorithms. The basic ES principle as we present it here may also be applied to learning SMA, IO-OOMs and PSRs from data. However, the concrete ES algorithm presented in Algorithm 8 makes use of several variance approximations and resulting simplifications that are only valid for the estimators from Remark 35 for the case of stationary stochastic processes.

The idea of the efficiency sharpening (ES) (Jaeger et al., 2006b) learning algorithm is to view the learning Equations (2) as a model estimator parameterized by C (and Q), and to select C such that the resulting estimator has minimum variance while still being consistent. Furthermore, this optimal choice of C is derived from knowledge of a model $\mathcal{M}$ for f, or in practice from a previous estimate thereof. To make this approach tractable, some simplifying assumptions are made.

First, a simplified version of the learning Equations (2) is used, where the indicator is taken to be $Q = (CF^{I,J})^\dagger$. This leads to operator estimates

$$\hat{\tau}_z = C\hat{F}_z^{I,J}(C\hat{F}^{I,J})^\dagger.$$

Jaeger et al. (2006b) now argue that due to the (pseudo)inversion, the variance of $\hat{\tau}_z$ is dominated by the variance of the factor $C\hat{F}^{I,J}$. The variance of a matrix is here taken w.r.t. the Frobenius norm. The ES algorithm therefore strives to find an admissible C such that the variance of $C\hat{F}^{I,J}$ is minimized — assuming knowledge of a model $\mathcal{M}$ for

f. A characterizer C is admissible if $CF^{I,J}Q$ is invertible. This is solved by the following proposition, which we state here in a more general form than in the original work (Jaeger et al., 2006b):

Proposition 40 *Let $\mathcal{M} = (\sigma, \{\tau_z\}, \omega_\varepsilon)$ be a d-dimensional minimal SS for a function $f : \Sigma^* \to \mathbb{R}$, and assume that $\hat{f}(\overline{x})$ are unbiased and uncorrelated estimators for all $\overline{x} \in \Sigma^*$. Define*

$$C^* = \Pi^\top D_I^2, \quad \text{where } \Pi^\top = ((\sigma\tau_{\overline{x}_i})^\top)_{i\in I}, \text{ and } D_I^2 = [\mathrm{diag}(\sum_{j\in J} \mathrm{Var}[\hat{f}(\overline{x}_j\overline{x}_i)])_{i\in I}]^\dagger.$$

Then $\mathrm{Var}[C\hat{F}^{I,J}]$ is minimized by the characterizer $C^* + 0$ among all characterizers of the form $C^* + G$ that satisfy $G\Pi = 0$.

The proof is given in the appendix, however, some explanatory remarks are in order. First of all, the assumptions that the estimates $\hat{f}(\overline{x})$ are unbiased and uncorrelated are reasonable, yet not strictly correct, meaning that the characterizer C^* will only approximate the theoretically optimal characterizer.

Next, we need a technical lemma to understand why it suffices to consider only characterizers of the form $(C^* + G)$ for some G satisfying $G\Pi = 0$:

Lemma 41 *If C^* has full row rank, then any admissible characterizer C can be written as $\rho(C^* + G)$ for some non-singular $\rho \in \mathbb{R}^{d\times d}$ and G such that $G\Pi = 0$.*

Proof Let C be some admissible characterizer. Then $C\Pi \in \mathbb{R}^{d\times d}$ must be invertible. Also, $C^*\Pi = (D_I\Pi)^\top(D_I\Pi)$ will be invertible if C^* has full row rank. Choosing $\rho = (C\Pi)(C^*\Pi)^{-1}$ and $G = \rho^{-1}(C - \rho C^*)$ we can easily verify that $C = \rho(C^* + G)$ and $G\Pi = 0$. ■

Note that the characterizers $C^* + G$ and $\rho(C^* + G)$ will lead to equivalent models via the learning Equations (2). Therefore, if the characterizer C^* is best among the class of characterizers $C^* + G$ where $G\Pi = 0$ then it is also the overall best choice.

Furthermore, the condition that C^* must have full row rank can be assured by (i) choosing indicative and characteristic sequences and the modeling dimension d accordingly, so that $d = \mathrm{rank}(\mathcal{M}) = \mathrm{rank}(F^{I,J}) = \mathrm{rank}(\Pi)$ and (ii) assuming that the variance of the estimators $\hat{f}(\overline{x})$ is non-zero, ensuring that D_I is invertible — which will typically be the case in practice.

Finally, to compute C^* via Proposition 40, we need to know the variances of the estimators $\hat{f}(\overline{x})$ occurring in D_I. Instead, we will replace D_I by an approximation that can be computed directly from the model $\mathcal{M}$. The approximation we present here is only valid for the case of stationary stochastic processes, but may be modified to cover the case of probabilistic languages as well.

Consider the estimators $\hat{f}(\overline{x})$ as in Remarks 34 and 35. It is reasonable to assume that the counts $\#(\overline{x})$ follow a binomial distribution, i.e., $\#(\overline{x}) \sim b_{N,p}$, where N is the length of the training sequence $\bar{s}$ and $p = f(\overline{x})$. This gives $\mathrm{Var}[\hat{f}(\overline{x})] = f(\overline{x})(1 - f(\overline{x}))/N$, which we may further approximate by $f(\overline{x})/N$, as in practice the values of $f(\overline{x})$ will typically be

small for most sequences $\overline{x}$. Also, the division by N is superfluous, as it cancels via the learning Equations (2). Using the approximation $\mathrm{Var}[\hat{f}(\overline{x})] \approx f(\overline{x})$, one can approximate

$$D_I^2 \approx \tilde{D}_I^2 := [\mathrm{diag}(\sum_{j \in J} f(\overline{x}_j \overline{x}_i))_{i \in I}]^\dagger = [\mathrm{diag}(\Pi \tau_{\overline{x}_J} \omega_\varepsilon)]^\dagger,$$

where $\tau_{\overline{x}_J} = \sum_{j \in J} \tau_{\overline{x}_j}$. The approximation

$$C^* \approx C^r := \Pi^\top \tilde{D}_I^2$$

is the characterizer that is actually used in the ES algorithm.

In the case of a stationary stochastic process and a choice of indicative words that partition Σ^l or $\Sigma^{\leq l}$ for some l one will have $\tau_{\overline{x}_J} \omega_\varepsilon = \omega_\varepsilon$, and therefore $\tilde{D}_I^2 = [\mathrm{diag}(\Pi \omega_\varepsilon)]^\dagger$. In this case, the columns $c_i = (\sigma \tau_{\overline{x}_i})^\top / \sigma \tau_{\overline{x}_i} \omega_\varepsilon$ of C^r can be seen as the normalized states $\omega^r_{\overline{x}_i{}^r} / \omega_\varepsilon^\top \omega^r_{\overline{x}_i{}^r}$ for the reversed words $\overline{x}_i{}^r$ under the *reversed* model $\mathcal{M}^\top = (\omega_\varepsilon^\top, \{\tau_z^\top\}, \sigma^\top)$, where $\omega^r_{\overline{x}_i{}^r} = \tau^\top_{(\overline{x}_i)_1} \cdots \tau^\top_{(\overline{x}_i)_k} \sigma^\top$. This is essentially the original version given by Jaeger et al. (2006b), and the reason why this characterizer was called the *reverse characterizer*. This make-up of C^r from states of the reversed process is also instrumental for the practical algorithms given by Jaeger et al. (2006b).

Additionally, the ES algorithm further exploits the interpretation of columns of $CF^{I,J}$ and $CF_z^{I,J}$ as model states $\omega_{\overline{x}_j}$ and $\omega_{\overline{x}_j z}$ as given in Proposition 33. These columns give argument-value pairs from which the operators τ_z can be deduced — as we have seen before. However, it is argued that in the face of estimates $\hat{F}^{I,J}$ and $\hat{F}_z^{I,J}$ the j-th columns should be weighted by $(\sum_{i \in I} \hat{f}(\overline{x}_j \overline{x}_i))^{-\frac{1}{2}}$ prior to performing linear regression to better reflect the weight of evidence that each column estimate is based on.

In practice a true model $\mathcal{M}$ is unknown. Therefore, the ES algorithm employs the following iterative procedure (again, our treatment here is more general than the original account by Jaeger et al. (2006b)):

Algorithm 8: The ES algorithm (for the case of stochastic processes)

1 Select some initial model estimate $\hat{\mathcal{M}}$ (e.g., via the learning Equations 2 using a random choice of C and Q).

repeat

2 Using the current model estimate $\hat{\mathcal{M}}$, compute $C = \hat{\Pi}^\top D_I^2$, where $\hat{\Pi}^\top = ((\hat{\sigma} \hat{\tau}_{\overline{x}_i})^\top)_{i \in I}$ and $D_I^2 = [\mathrm{diag}(\hat{\Pi} \sum_{j \in J} \hat{\tau}_{\overline{x}_j} \hat{\omega}_\varepsilon)_{i \in I}]^\dagger$.

3 Let $Q = D_J (C \hat{F}^{I,J} D_J)^\dagger$, where $D_J = [\mathrm{diag}(\sum_{i \in I} \hat{f}(\overline{x}_j \overline{x}_i))_{j \in J}]^{\dagger \frac{1}{2}}$.

4 Obtain a new model estimate $\hat{\mathcal{M}}$ via the learning Equations (2).

until *some fixed number of iterations, or some performance criteria of the estimated models stops increasing.*

Note that this procedure constructs a sequence of estimators along with a sequence of model estimates. The rationale of such ES algorithms is that the sequence of estimators increases in statistical efficiency, hence the name *efficiency sharpening* algorithms. The ES iterations come with no convergence guarantees. Nevertheless, this procedure has been found in practice to converge in very few iterations (3 – 5 typically suffice), and the results are of a similar quality as obtained by spectral algorithms (comparisons in Zhao et al., 2009a,b).

The ES algorithm is closely related to the row and column weighted spectral algorithm presented in Section 4.4.2. Precisely:

Proposition 42 *Assume $F^{I,J}$ of rank d is determined by some underlying minimal model $\mathcal{M} = (\omega_\varepsilon^\top, \{\tau_z^\top\}, \sigma^\top)$ of rank d and let $\Pi^\top = ((\sigma\tau_{\overline{x}_i})^\top)_{i\in I}$, $D_I = [\mathrm{diag}(\sum_{j\in J} f(\overline{x}_j\overline{x}_i))_{i\in I}]^{\dagger\frac{1}{2}}$ and $D_J = [\mathrm{diag}(\sum_{i\in I} f(\overline{x}_j\overline{x}_i))_{j\in J}]^{\dagger\frac{1}{2}}$. Let $C^r = \Pi^\top D_I^2$ be the reverse characterizer, and let $C' = \tilde{U}_d^\top D_I$ be the characterizer obtained by the weighted spectral method, where $\tilde{U}_d\tilde{S}_d\tilde{V}_d^\top$ is the d-truncated SVD of $D_I F^{I,J} D_J$. Then $C^r = \rho C'$ for some non-singular transformation ρ.*

Proof First, $\tilde{U}_d\tilde{S}_d\tilde{V}_d^\top = D_I F^{I,J} D_J$, since $F^{I,J}$ is assumed to have rank d. Now observe that $\tilde{U}_d\tilde{S}_d\tilde{V}_d^\top = D_I F^{I,J} D_J = D_I\Pi\Phi D_J$, where $\Phi = (\tau_{\overline{x}_j}\omega_\varepsilon)_{j\in J}$, and therefore the columns of $D_I F^{I,J}$, $\tilde{U}_d$ and $D_I\Pi$ all span $\mathrm{im}(D_I F^{I,J})$. So $C' = \tilde{U}_d^\top D_I$ and $C^r = (D_I\Pi)^\top D_I = \Pi^\top D_I^2$ have the same row space, and we can therefore find such a transformation ρ. ■

This means that the reverse characterizer C^r also gives a representation of the principal subspace of the weighted matrix $D_I F^{I,J}$. The main difference to the weighted spectral method described in Section 4.4.2 is that C^r is derived algebraically from an underlying model estimate, while the weighted spectral method estimates the principle subspace from the weighted data matrix $\hat{D}_I\hat{F}^{I,J}$ with weights $\hat{D}_I$ that also need to be determined from the data, e.g., $\hat{D}_I = [\mathrm{diag}(\sum_{j\in J} \hat{f}(\overline{x}_j\overline{x}_i))_{i\in I}]^{\dagger\frac{1}{2}}$.

5. Conclusion

We have shown that OOMs, PSRs and SMA are closely related instances of MA, and we have presented a unified learning framework for estimating such models from data that subsumes many of the existing learning algorithms. In presenting the learning framework, we have isolated the key design choices that need to be made to obtain a concrete learning algorithm. For each design choice we have surveyed the approaches that have been taken in the past and have tried to give some guidance.

We briefly summarize the choices that need to be made to obtain a concrete learning algorithm. First of all, estimates of the system matrices $\hat{F}^{I,J}$ and $\hat{F}_z^{I,J}$ must be obtained from the available training data. Individual entries may be estimated by the formulas given in Section 4.1. However, it is of much greater importance to decide which entries need to be estimated, that is, which rows I and columns J should be selected. This is discussed in Section 4.2. While many of the existing algorithms attempt to choose as few rows and columns to estimate as possible, we argue that this leads to poor statistical efficiency, and that the selection should ideally be matched to the available training data. Next, one must select a suitable model dimension d. This may be achieved by an algebraic criterion, as described in Section 4.3.1, or by cross-validation. It is also possible to treat this as a learning parameter that can be hand-tuned by the modeler. We note that it is generally neither necessary nor advisable to set the target dimension to the correct rank of the underlying system, as the optimal choice depends on the available training data. Finally, the estimated system matrices $\hat{F}^{I,J}$ and $\hat{F}_z^{I,J}$ need to be "compressed" to $d \times d$ matrices by suitable characterizer and indicator matrices C and Q. A good selection of C and Q is vital

to obtaining high statistical efficiency, and this is treated in detail in Section 4.4. We show that several of the proposed approaches to selecting C and Q can be seen as variations of a spectral learning algorithm presented in Section 4.4.2.

We conclude with a remark on implementing such a learning algorithm in practice. Clearly, the main limiting factor is the size of the matrices $\hat{F}^{I,J}$ and $\hat{F}_z^{I,J}$, as these may become very large. However, it is possible to obtain an efficient sparse representation of these matrices by employing a suffix tree representation of the training data (Zhao et al., 2009b,a; Jaeger et al., 2006b). Furthermore, if one uses the method described in Section 4.4.4 one can avoid evaluating these matrices explicitly and instead calculate $C\hat{F}^{I,J}$ and $C\hat{F}_z^{I,J}$ directly (Jaeger et al., 2006b).

Acknowledgements

We gratefully acknowledge the funding by the German Research Foundation (DFG) under the project JA 1210/5-1. We would also like to thank the anonymous reviewers for their constructive and very helpful comments.

Appendix

Proof [of Proposition 37](adapted from Zhao et al., 2009a) Let $C_* = \mathrm{diag}(C, \ldots, C)$ (l copies of C). Using the introduced notation the learning Equations (2) can be written concisely to obtain:

$$\begin{aligned}
\tau_* &= \left(C_*\hat{F}_*^{I,J}Q + C_*(F_*^{I,J} - \hat{F}_*^{I,J})Q\right)\left(C\hat{F}^{I,J}Q + C(F^{I,J} - \hat{F}^{I,J})Q\right)^{-1} \\
&= \left(C_*\hat{F}_*^{I,J}Q + C_*(F_*^{I,J} - \hat{F}_*^{I,J})Q\right)(C\hat{F}^{I,J}Q)^{-1}\left(I_d + C(F^{I,J} - \hat{F}^{I,J})Q(C\hat{F}^{I,J}Q)^{-1}\right)^{-1} \\
&= \left(\hat{\tau}_* + \left(C_*(F_*^{I,J} - \hat{F}_*^{I,J})Q\right)(C\hat{F}^{I,J}Q)^{-1}\right)\left(I_d + C(F^{I,J} - \hat{F}^{I,J})Q(C\hat{F}^{I,J}Q)^{-1}\right)^{-1},
\end{aligned}$$

which implies $\tau_* + \tau_* C(F^{I,J} - \hat{F}^{I,J})Q(C\hat{F}^{I,J}Q)^{-1} = \hat{\tau}_* + (C_*(F_*^{I,J} - \hat{F}_*^{I,J})Q)(C\hat{F}^{I,J}Q)^{-1}$. By rearranging, taking Frobenius norms and using the triangle inequality and submultiplicativity, we obtain

$$\|\tau_* - \hat{\tau}_*\|_F \leq \|C\|_F \|Q(C\hat{F}^{I,J}Q)^{-1}\|_F \left(\|\tau_*\|_F \|F^{I,J} - \hat{F}^{I,J}\|_F + \frac{\|C_*\|_F}{\|C\|_F}\|F_*^{I,J} - \hat{F}_*^{I,J}\|_F\right).$$

Now $\frac{\|C_*\|_F}{\|C\|_F} = \sqrt{l}$, and $\|\tau_*\|_F^2 = \sum_{z\in\Sigma}\|\tau_z\|_F^2 \geq \|\tau_\Sigma\|_F^2 \geq \rho(\tau_\Sigma)^2$, where $\tau_\Sigma = \sum_{z\in\Sigma}\tau_z$, and the result follows. ■

Note that in the original paper the inequality $\|\tau_*\|_F \geq \frac{1}{\sqrt{l}}$ was used instead, which depended on the columns of τ_* summing to 1. This was in turn insured by adding additional restrictions on the choice of characteristic words and characterizer C. These are now no longer needed.

Lemma 43 *Let $D = \mathrm{diag}(d_1, \ldots, d_n)$ and $S = \mathrm{diag}(s_1, \ldots, s_n)$ satisfying $d_1 \geq \cdots \geq d_n \geq 0$ and $0 \leq s_1 \leq \cdots \leq s_n$, and let U be an orthogonal $n \times n$ matrix, i.e., $U^\top U = UU^\top = I$. Then $\|DUS\|_F \geq \|DS\|_F$.*

Proof $\|DUS\|_F^2 = \sum_{i,j=1}^n (d_i u_{ij} s_j)^2$. Furthermore, $U^\top U = UU^\top = I_n$ implies that $(*)$ $\sum_{i=1}^n u_{i,j}^2 = 1$ for all j and $\sum_{j=1}^n u_{i,j}^2 = 1$ for all i. We will show the slightly stronger claim that $\|DUS\|_F^2 \geq \|DS\|_F^2$ for any matrix U satisfying $(*)$, which allows us to assume w.l.o.g. that $u_{i,j} \geq 0$ for all entries in U, since only the squared entries $u_{i,j}^2$ appear in the expressions for $\|DUS\|_F^2$ and $(*)$. So from now on we assume that U merely satisfies $(*)$ and that all entries in U are non-negative.

First note that if U is lower triangular, then $(*)$ implies that $U = I_n$: $\sum_{i=1}^n u_{i,n}^2 = 1$ implies that $u_{n,n}^2 = 1$ and $u_{i,n}^2 = 0$ for $i < n$. Then $\sum_{j=1}^n u_{n,j}^2 = 1$ implies that $u_{n,j}^2 = 0$ for $j < n$, since $u_{n,n}^2 = 1$. That is, $U = \left[\begin{smallmatrix} U_{n-1} & 0 \\ 0 & 1 \end{smallmatrix}\right]$, and the condition $(*)$ must therefore hold for U_{n-1} as well. By induction on n, $U = I_n$. In this case $\|DUS\|_F^2 = \|DS\|_F^2$.

So assume U is not lower triangular. Consider a row-wise ordering of matrix positions, i.e., define $\mathrm{ord}(i,j) = (i-1)n + j$, and let $(i',j') = \operatorname*{argmin}_{(i,j)} \{\mathrm{ord}(i,j) : j > i, u_{i,j} \neq 0\}$, i.e., i' is the first row of U to contain a non-zero element above the diagonal, and j' is the column index of the first such entry within the i'-th row. We call $\mathrm{ord}(i',j')$ the *order* of U, and say that a lower triangular matrix has infinite order.

Now consider the i'-th column of U. By the choice of i' we must have $\sum_{i=1}^{i'-1} u_{i,i'}^2 = 0$, and therefore $\sum_{i=i'+1}^n u_{i,i'}^2 = 1 - u_{i',i'}^2 = \sum_{j=1}^n u_{i',j}^2 - u_{i',i'}^2 \geq u_{i',j'}^2$. We can therefore find a vector v such that $v_i = 0$ for $i < i'$, $v_{i'} = -u_{i',j'}^2$, and $0 \leq v_i \leq u_{i,i'}^2$ as well as $\sum_{i=i'+1}^n v_i = u_{i',j'}^2$ for $i = i'+1, \ldots, n$. Let $U^2 = [u_{i,j}^2]_{i,j=1\ldots n}$ be the matrix of element-wise squares of entries in U, and let $\tilde{U}^2$ be obtained by subtracting the vector v from the i'-th column of U^2 and adding v to the j'-th column of U^2. Let $\tilde{U}$ be the matrix of element-wise square roots of entries in $\tilde{U}^2$.

We can easily check that all entries in $\tilde{U}^2$ are non-negative, so that this is well-defined. Also $\tilde{U}$ satisfies $(*)$, since $\sum_{i=1}^n v_i = 0$ by construction, and adding such a vector to one column of $\tilde{U}^2$ and subtracting from another does not change the row and column sums. Furthermore,

$$\begin{aligned}
\|DUS\|_F^2 - \|D\tilde{U}S\|_F^2 &= \sum_{i=1}^n \left(d_i^2 v_i s_{i'}^2 - d_i^2 v_i s_{j'}^2\right) \\
&= d_{i'}^2 v_{i'} (s_{i'}^2 - s_{j'}^2) + \sum_{i=i'+1}^n d_i^2 v_i (s_{i'}^2 - s_{j'}^2) \\
&= (s_{i'}^2 - s_{j'}^2) \left(d_{i'}^2 v_{i'} + \sum_{i=i'+1}^n d_i^2 v_i \right).
\end{aligned}$$

Now $s_{i'}^2 - s_{j'}^2 \leq 0$ since $j' > i'$, and $\sum_{i=i'+1}^n d_i^2 v_i \leq d_{i'}^2 \sum_{i=i'+1}^n v_i = d_{i'}^2 u_{i',j'}^2$, while $d_{i'}^2 v_{i'} = -d_{i'}^2 u_{i',j'}^2$, so $(d_{i'}^2 v_{i'} + \sum_{i=i'+1}^n d_i^2 v_i) \leq 0$. This shows that $\|DUS\|_F^2 \geq \|D\tilde{U}S\|_F^2$. And finally, the order of $\tilde{U}$ is larger than the order of U, as we have eliminated the non-zero element of lowest order above the diagonal in U, and in turn have introduced only non-zero elements above the diagonal of higher order (in rows below the i'-th), or none at all.

By iterating this construction we arrive at a lower triangular matrix U^* with non-negative entries that satisfies $(*)$ and $\|DUS\|_F^2 \geq \|DU^*S\|_F^2 = \|DS\|_F^2$. ■

Proof [Proof of Proposition 39] Assume $r = \text{rank}(\hat{F}^{I,J}) \geq d$ and let $USV^\top = \hat{F}^{I,J}$ be the full SVD of $\hat{F}^{I,J}$. We can simply verify that indeed $(C^*\hat{F}^{I,J})^\dagger = (S_d^{-\frac{1}{2}}U_d^\top USV^\top)^\dagger = (S_d^{-\frac{1}{2}}V_d^\top)^\dagger = V_d S_d^{-\frac{1}{2}}$, which implies that $C^*\hat{F}^{I,J}Q^* = C^*\hat{F}^{I,J}(C^*\hat{F}^{I,J})^\dagger = I_d$, as required. Furthermore, $\|C^*\|_F\|Q^*\|_F = \|S_d^{-\frac{1}{2}}\|_F^2 = \sum_{i=1}^d \sigma_i^{-1}$, where the σ_i are the singular values of $\hat{F}^{I,J}$, which are also the diagonal elements of S. We will show that this is indeed the minimum of $\|C\|_F\|Q\|_F$ subject to $C\hat{F}^{I,J}Q = I_d$.

Using the substitution $C = C'U^\top$ and $Q = VQ'$, we can see that minimizing $\|C\|_F\|Q\|_F$ subject to $C\hat{F}^{I,J}Q = I_d$ is equivalent to minimizing $\|C'\|_F\|Q'\|_F$ subject to $C'SQ' = I_d$ and that this will have the same minimal value. Let C'_r, Q'_r and S_r be truncated versions of C', Q' and S that consist of the first r columns, rows or rows and columns, respectively. Then minimizing $\|C'_r\|_F\|Q'_r\|_F$ subject to $C'_rS_rQ'_r = I_d$ is equivalent and has the same minimal value, because $C'SQ' = C'_rS_rQ'_r$ (since $\sigma_i = 0$ for $i > r$) and the additional columns in C' and rows in Q' are best set to zero.

Assume now that C'_r and $Q'_r = (C'_rS_r)^\dagger$ minimize $\|C'_r\|_F\|Q'_r\|_F$ subject to $C'_rS_rQ'_r = I_d$. We can select $Q'_r = (C'_rS_r)^\dagger$, as this minimizes $\|C'_r\|_F\|Q'_r\|_F$ subject to $C'_rS_rQ'_r = I_d$ for a given C'_r. It remains to show that $\|C'_r\|_F\|Q'_r\|_F \geq \|C^*\|_F\|Q^*\|_F = \sum_{i=1}^d \sigma_i^{-1}$.

Let $LDR^\top = C'_rS_r$ be the SVD of C'_rS_r. Then $C'_r = LDR^\top S_r^{-1}$, and $Q'_r = (C'_rS_r)^\dagger = RD^\dagger L^\top$. Let $d_1, \ldots, d_d$ be the diagonal elements of D and let D_r be the $r \times r$ matrix obtained by extending D with zero rows. Then

$$\|C'_r\|_F^2 = \|LDR^\top S_r^{-1}\|_F^2 = \|D_rR^\top S_r^{-1}\|_F^2 \overset{\text{Lemma 43}}{\geq} \|D_rS_r^{-1}\|_F^2 = \sum_{i=1}^d d_i^2\sigma_i^{-2},$$

$$\|Q'_r\|_F^2 = \|RD^\dagger L^\top\|_F^2 = \|D^\dagger\|_F^2 = \sum_{i=1}^d d_i^{-2}.$$

Multiplying these expressions and substituting $d_i^2 = a_i^2\sigma_i$, we obtain

$$\begin{aligned}
\|C'_r\|_F^2\|Q'_r\|_F^2 &= \left(\sum_{i=1}^d a_i^2\sigma_i^{-1}\right)\left(\sum_{i=1}^d a_i^{-2}\sigma_i^{-1}\right) \\
&= \sum_{i=1}^d \sigma_i^{-2} + \sum_{\substack{i,j=1 \\ i<j}}^d \left(\frac{a_i^2}{a_j^2} + \frac{a_j^2}{a_i^2}\right)\sigma_i^{-1}\sigma_j^{-1} \\
&= \sum_{i=1}^d \sigma_i^{-2} + \sum_{\substack{i,j=1 \\ i<j}}^d \left(\left(\frac{a_i}{a_j} - \frac{a_j}{a_i}\right)^2 + 2\right)\sigma_i^{-1}\sigma_j^{-1} \\
&\geq \left(\sum_{i=1}^d \sigma_i^{-1}\right)^2,
\end{aligned}$$

since this expression is clearly minimal when $a_i = 1$ for all i. So we can conclude that $\|C'_r\|_F\|Q'_r\|_F \geq \sum_{i=1}^d \sigma_i^{-1}$. Therefore, C^* and Q^* are in fact a minimal solution to the optimization problem (4). ■

Proof [of Proposition 40] First, we calculate:

$$\begin{aligned}
\mathrm{Var}[C\hat{F}^{I,J}] &\overset{(*)}{=} E\left[||C\hat{F}^{I,J} - CF^{I,J}||_F^2\right] \\
&= \sum_{j\in J}\sum_{k=1}^{d} E\left[\left(\sum_{i\in I} c_{ki}\hat{f}(\overline{x}_j\overline{x}_i) - \sum_{i\in I} c_{ki} f(\overline{x}_j\overline{x}_i)\right)^2\right] \\
&\overset{(*)}{=} \sum_{j\in J}\sum_{k=1}^{d} \mathrm{Var}\left[\sum_{i\in I} c_{ki}\hat{f}(\overline{x}_j\overline{x}_i)\right] \\
&\overset{(**)}{=} \sum_{j\in J}\sum_{k=1}^{d}\sum_{i\in I} c_{ki}^2 \mathrm{Var}[\hat{f}(\overline{x}_j\overline{x}_i)] \\
&= \sum_{i\in I} \|(C)_i\|_F^2 \sum_{j\in J} \mathrm{Var}[\hat{f}(\overline{x}_j\overline{x}_i)] = \sum_{i\in I} v_i \|(C)_i\|_F^2,
\end{aligned}$$

where $(C)_i$ is the i-th column of C, and $v_i = \sum_{j\in J} \mathrm{Var}[\hat{f}(\overline{x}_j\overline{x}_i)]$. Note that we have used unbiasedness in $(*)$ and uncorrelatedness in $(**)$.

Our goal is now to minimize $J(G) = \mathrm{Var}[(C^* + G)\hat{F}^{I,J}] = \sum_{i\in I} v_i ||(C^* + G)_i||_F^2$ subject to the constraints $h_{k,l}(G) = [G\Pi]_{k,l} = 0$ for $k, l = 1 \ldots d$. Note that if $v_i = 0$ for some i, then the i-th column of G does not influence the value of $J(G)$, and we may w.l.o.g. fix $(G)_i = 0$ and replace the equality constraints by $\tilde{h}_{k,l}(G) = [GDD^\dagger\Pi]_{k,l} = 0$, where $D = \mathrm{diag}[(v_i)_{i\in I}]$. This is a convex quadratic programming problem, therefore $G = 0$ will be a solution if and only if it satisfies the KKT conditions

$$\begin{aligned}
&\frac{\partial J}{\partial G}(G) + \sum_{k,l=1}^{d} \lambda_{k,l} \frac{\partial h_{k,l}}{\partial G}(G) = 0, \text{ and} \\
&\forall k, l = 1 \ldots d : \tilde{h}_{k,l}(G) = 0,
\end{aligned}$$

for some Lagrange multipliers $\lambda_{k,l} \in \mathbb{R}$. Clearly, the latter condition $\tilde{h}_{k,l}(G) = 0$ is satisfied for all k, l by $G = 0$. We can calculate $\sum_{k,l=1}^{d} \lambda_{k,l} \frac{\partial \tilde{h}_{k,l}}{\partial G}(G) = \lambda\Pi^\top D^\dagger D$, where $\lambda \in \mathbb{R}^{d\times d}, [\lambda]_{k,l} = \lambda_{k,l}$, as well as $\frac{\partial J}{\partial G}(G) = 2(C^* + G)D = 2(\Pi^\top D_I^2 + G)D$. The first condition is then satisfied by $G = 0$ with $\lambda = -2I$, since $\Pi^\top D_I^2 D = \Pi^\top D^\dagger D$ by definition of D_I. ■

References

Naoki Abe and Manfred K. Warmuth. On the computational complexity of approximating distributions by probabilistic automata. *Machine Learning*, 9:205–260, 1992.

Animashree Anandkumar, Daniel Hsu, and Sham M. Kakade. A method of moments for mixture models and hidden markov models. In Shie Mannor, Nathan Srebro, and Robert C. Williamson, editors, *Proceedings of the 25th Annual Conference on Learning*

Theory (COLT 2012), volume 23 of *JMLR Workshop & Conference Proceedings*, pages 33.1–33.34, 2012.

Dana Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1987.

Raphaël Bailly. Quadratic weighted automata: Spectral algorithm and likelihood maximization. In Chun-Nan Hsu and Wee Sun Lee, editors, *Proceedings of the 3rd Asian Conference on Machine Learning (ACML 2011)*, volume 20 of *JMLR Workshop & Conference Proceedings*, pages 147–163, 2011.

Raphaël Bailly, François Denis, and Liva Ralivola. Grammatical inference as a principal component analysis problem. In Andrea Pohoreckyj Danyluk, Léon Bottou, and Michael L. Littman, editors, *Proceedings of the 26th International Conference on Machine Learning (ICML 2009)*, volume 382 of *ACM Proceedings*, pages 33–40, 2009.

Raphael Bailly, Xavier Carreras, and Ariadna Quattoni. Unsupervised spectral learning of finite state transducers. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26 (NIPS 2013)*, pages 800–808. Curran Associates, Inc., 2013.

Borja Balle and Mehryar Mohri. Spectral learning of general weighted automata via constrained matrix completion. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, pages 2168–2176, 2012.

Borja Balle, Ariadna Quattoni, and Xavier Carreras. A spectral learning algorithm for finite state transducers. In Dimitrios Gunopulos, Thomas Hofmann, Donato Malerba, and Michalis Vazirgiannis, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference (ECML/PKDD 2011), Proceedings, Part I*, volume 6911 of *Lecture Notes in Computer Science*, pages 156–171. Springer, 2011.

Borja Balle, Ariadna Quattoni, and Xavier Carreras. Local loss optimization in operator models: A new insight into spectral learning. In *Proceedings of the 29th International Conference on Machine Learning (ICML 2012)*. icml.cc / Omnipress, 2012.

Borja Balle, Xavier Carreras, Franco M. Luque, and Ariadna Quattoni. Spectral learning of weighted automata – a forward-backward perspective. *Machine Learning*, 96(1-2):33–63, 2014.

Amos Beimel, Francesco Bergadano, Nader H. Bshouty, Eyal Kushilevitz, and Stefano Varricchio. On the applications of multiplicity automata in learning. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science (FOCS 1996)*, pages 349–358. IEEE Computer Society, 1996.

Amos Beimel, Francesco Bergadano, Nader H. Bshouty, Eyal Kushilevitz, and Stefano Varricchio. Learning functions represented as multiplicity automata. *Journal of the ACM*, 47(3):506–530, 2000.

Francesco Bergadano and Stefano Varricchio. Learning behaviors of automata from multiplicity and equivalence queries. In Maurizio A. Bonuccelli, Pierluigi Crescenzi, and Rossella Petreschi, editors, *Proceedings of the 2nd Italian conference on Algorithms and Complexity (CIAC 1994)*, volume 778 of *Lecture Notes in Computer Science*, pages 54–62. Springer, 1994.

Francesco Bergadano, Dario Catalano, and Stefano Varricchio. Learning sat-k-DNF formulas from membership queries. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC 1996)*, pages 126–130. ACM, 1996.

Jean Berstel, Jr. and Christophe Reutenauer. *Rational Series and Their Languages*, volume 12 of *EATCS Monographs on Theoretical Computer Science*. Springer, 1988.

Byron Boots and Geoffrey J. Gordon. Predictive state temporal difference learning. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23 (NIPS 2010)*, pages 271–279. MIT Press, 2010.

Byron Boots and Geoffrey J. Gordon. An online spectral learning algorithm for partially observable nonlinear dynamical systems. In Wolfram Burgard and Dan Roth, editors, *Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI 2011)*. AAAI Press, 2011.

Byron Boots, Sajid M. Siddiqi, and Geoffrey J. Gordon. Closing the learning-planning loop with predictive state representations. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, pages 1369–1370. IFAAMAS, 2010.

Byron Boots, Geoffrey J. Gordon, and Arthur Gretton. Hilbert space embeddings of predictive state representations. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI 2013)*, pages 92–101. AUAI Press, 2013.

Michael Bowling, Peter McCracken, Michael James, James Neufeld, and Dana F. Wilkinson. Learning predictive state representations using non-blind policies. In William W. Cohen and Andrew Moore, editors, *Proceedings of the 23rd International Conference on Machine Learning (ICML 2006)*, volume 148 of *ACM Proceedings*, pages 129–136, 2006.

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

Corinna Cortes and Mehryar Mohri. Context-free recognition with weighted automata. *Grammars*, 3(2/3):133–150, 2000.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood estimation from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39 (1):1–38, 1977.

François Denis and Yann Esposito. Learning classes of probabilistic automata. In *Proceedings of the 17th Annual Conference on Learning Theory (COLT 2004)*, volume 3120 of *Lecture Notes in Computer Science*, pages 124–139. Springer, 2004.

François Denis and Yann Esposito. On rational stochastic languages. *Fundamenta Informaticae*, 86(1):41–77, 2008.

François Denis, Yann Esposito, and Amaury Habrard. Learning rational stochastic languages. In Gábor Lugosi and Hans-Ulrich Simon, editors, *Proceedings of the 19th Annual Conference on Learning Theory (COLT 2006)*, volume 4005 of *Lecture Notes in Computer Science*, pages 274–288. Springer, 2006.

Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of Weighted Automata*. Springer, 2009.

Pierre Dupont, François Denis, and Yann Esposito. Links between probabilistic automata and hidden Markov models: probability distributions, learning models and induction algorithms. *Pattern Recognition*, 38(9):1349–1371, 2005.

Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

Michel Fliess. Matrices de Hankel. *Journal de Mathématiques Pures et Appliquées*, 53: 197–222, 1974.

Edgar J. Gilbert. On the identifiability problem for functions of finite Markov chains. *The Annals of Mathematical Statistics*, 30(3):688–697, 1959.

Robert M. Gray. *Probability, Random Processes, and Ergodic Properties*. Springer, 1988.

William L. Hamilton, Mahdi M. Fard, and Joelle Pineau. Modelling sparse dynamical systems with compressed predictive state representations. In Sanjoy Dasgupta and David Mcallester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML 2013)*, volume 28 of *JMLR Workshop & Conference Proceedings*, pages 178–186, 2013.

Per Christian Hansen. *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1998.

Alex Heller. On stochastic processes derived from Markov chains. *The Annals of Mathematical Statistics*, 36(4):1286–1291, 1965.

Daniel Hsu, Sham M. Kakade, and Tong Zhang. A spectral algorithm for learning hidden Markov models. In *Proceedings of the 22nd Annual Conference on Learning Theory (COLT 2009)*, 2009.

Hisashi Ito. *An Algebraic Study of Discrete Stochastic Systems*. Unpublished doctoral dissertation, University of Tokyo, Bunkyo-ku, Tokyo, 1992.

Hisashi Ito, Shun ichi Amari, and Kingo Kobayashi. Identifiability of hidden Markov information sources and their minimum degrees of freedom. *IEEE Transactions on Information Theory*, 38(2):324–333, 1992.

Herbert Jaeger. Observable operator models and conditioned continuation representations. Arbeitspapiere der GMD 1043, GMD Forschungszentrum Informationstechnik, Sankt Augustin, Germany, 1997.

Herbert Jaeger. Discrete-time, discrete-valued observable operator models: a tutorial. Technical Report 42, GMD Forschungszentrum Informationstechnik, Sankt Augustin, Germany, 1998.

Herbert Jaeger. Modeling and learning continuous-valued stochastic processes with OOMs. GMD Report 102, GMD Forschungszentrum Informationstechnik, Sankt Augustin, Germany, 2000a.

Herbert Jaeger. Observable operator models for discrete stochastic time series. *Neural Computation*, 12(6):1371–1398, 2000b.

Herbert Jaeger, MingJie Zhao, and Andreas Kolling. Efficient estimation of ooms. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18 (NIPS 2005)*, pages 555–562. MIT Press, 2006a.

Herbert Jaeger, MingJie Zhao, Klaus Kretzschmar, Tobias Oberstein, Dan Popovici, and Andreas Kolling. Learning observable operator models via the ES algorithm. In Simon Haykin, José C. Príncipe, Terrence J. Sejnowski, and John McWhirter, editors, *New Directions in Statistical Signal Processing: From Systems to Brains*, Neural Information Processing, chapter 14, pages 417–464. MIT Press, Cambridge, MA, USA, 2006b.

Michael R. James and Satinder P. Singh. Learning and discovery of predictive state representations in dynamical systems with reset. In Carla E. Brodley, editor, *Proceedings of the 21st International Conference on Machine Learning (ICML 2004)*, volume 69 of *ACM Proceedings*, pages 53–60, 2004.

Michael R. James and Satinder P. Singh. Planning in models that combine memory with predictive representations of state. In Manuela M. Veloso and Subbarao Kambhampati, editors, *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 2005)*, pages 987–992. AAAI Press, 2005.

Michael R. James, Satinder Singh, and Michael L. Littman. Planning with predictive state representations. In *Proceedings of the 3rd International Conference on Machine Learning and Applications (ICMLA 2004)*, pages 304–311. IEEE Computer Society, 2004.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2):99–134, 1998.

Attila Kondacs and John Watrous. On the power of quantum finite state automata. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science (FOCS 1996)*, pages 66–75. IEEE Computer Society, 1997.

Klaus Kretzschmar. Learning symbol sequences with observable operator models. GMD Report 161, GMD Forschungszentrum Informationstechnik, Sankt Augustin, Germany, 2001.

Michael L. Littman, Richard S. Sutton, and Satinder P. Singh. Predictive representations of state. In Thomas G. Dietterich, Suzanna Becker, and Zoubin Ghahramani, editors, *Advances in Neural Information Processing Systems 14 (NIPS 2001)*, pages 1555–1561. MIT Press, 2001.

Ivan Markovsky and Sabine Van Huffel. Left vs right representations for solving weighted low-rank approximation problems. *Linear Algebra and its Applications*, 422(2-3):540–552, 2007a.

Ivan Markovsky and Sabine Van Huffel. Overview of total least-squares methods. *Signal Processing*, 87(10):2283–2302, 2007b.

Peter McCracken and Michael H. Bowling. Online discovery and learning of predictive state representations. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18 (NIPS 2005)*. MIT Press, 2006.

Mehryar Mohri, Fernando Pereira, and Michael Riley. Weighted finite-state transducers in speech recognition. *Computer Speech & Language*, 16(1):69–88, 2002.

Cristopher Moore and James P. Crutchfield. Quantum automata and quantum grammars. *Theoretical Computer Science*, 237(1-2):275–306, 2000.

Hiroyuki Ohnishi, Hiroyuki Seki, and Tadao Kasami. A polynomial time learning algorithm for recognizable series. *IEICE Transactions on Information and Systems*, E77-D(10): 1077–1085, 1994.

Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Adrià Recasens and Ariadna Quattoni. Spectral learning of sequence taggers over continuous sequences. In Hendrik Blockeel, Kristian Kersting, Siegfried Nijssen, and Filip Zelezný, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference (ECML/PKDD 2013), Proceedings, Part I*, volume 8188 of *Lecture Notes in Computer Science*, pages 289–304. Springer, 2013.

Matthew Rosencrantz, Geoffrey J. Gordon, and Sebastian Thrun. Learning low dimensional predictive representations. In Carla E. Brodley, editor, *Proceedings of the 21st International Conference on Machine Learning (ICML 2004)*, volume 69 of *ACM Proceedings*, pages 695–702, 2004.

Sam Roweis. EM algorithms for PCA and SPCA. In Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, editors, *Advances in Neural Information Processing Systems 10 (NIPS 1997)*, pages 626–632. MIT Press, 1998.

Matthew Rudary and Satinder P. Singh. Predictive linear-Gaussian models of controlled stochastic dynamical systems. In William W. Cohen and Andrew Moore, editors, *Proceedings of the 23rd International Conference on Machine Learning (ICML 2006)*, volume 148 of *ACM Proceedings*, pages 777–784, 2006.

Matthew Rudary and Satinder P. Singh. Predictive linear-Gaussian models of stochastic dynamical systems with vector-value actions and observations. In *Proceedings of the 10th International Symposium on Artificial Intelligence and Mathematics (ISAIM 2008)*, 2008.

Matthew Rudary, Satinder P. Singh, and David Wingate. Predictive linear-Gaussian models of stochastic dynamical systems. In Fahiem Bacchus and Tommi Jaakkola, editors, *Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence (UAI 2005)*, pages 501–508. AUAI Press, 2005.

Matthew R. Rudary and Satinder Singh. A nonlinear predictive state representation. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, pages 855–862. MIT Press, 2003.

Arto Salomaa and Matti Soittola. *Automata-Theoretic Aspects of Formal Power Series*. Texts and Monographs in Computer Science. Springer, 1978.

Marcel Paul Schützenberger. On the definition of a family of automata. *Information and Control*, 4(2-3):245–270, 1961.

Sajid M. Siddiqi, Byron Boots, and Geoffrey J. Gordon. Reduced-rank hidden markov models. In Yee Whye Teh and D. Mike Titterington, editors, *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, volume 9 of *JMLR Workshop & Conference Proceedings*, pages 741–748, 2010.

Satinder Singh, Michael R. James, and Matthew R. Rudary. Predictive state representations: A new theory for modeling dynamical systems. In Joseph Halpern, editor, *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence (UAI 2004)*, pages 512–519. AUAI Press, 2004.

Le Song, Byron Boots, Sajid M. Siddiqi, Geoffrey J. Gordon, and Alex J. Smola. Hilbert space embeddings of hidden Markov models. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML 2010)*, pages 991–998. Omnipress, 2010.

Daniel R. Upper. *Theory and Algorithms for Hidden Markov Models and Generalized Hidden Markov Models*. PhD thesis, University of California at Berkeley, 1997.

Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

Eric W. Wiewiora. *Modeling Probability Distributions with Predictive State Representations*. PhD thesis, University of California, San Diego, 2008.

David Wingate and Satinder P. Singh. Kernel predictive linear Gaussian models for nonlinear stochastic dynamical systems. In William W. Cohen and Andrew Moore, editors, *Proceedings of the 23rd International Conference on Machine Learning (ICML 2006)*, volume 148 of *ACM Proceedings*, pages 1017–1024, 2006a.

David Wingate and Satinder P. Singh. Mixtures of predictive linear Gaussian models for nonlinear, stochastic dynamical systems. In Anthony Cohn, editor, *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI 2006)*. AAAI Press, 2006b.

David Wingate and Satinder P. Singh. Exponential family predictive representations of state. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20 (NIPS 2007)*, pages 1617–1624. MIT Press, 2008a.

David Wingate and Satinder P. Singh. Efficiently learning linear-linear exponential family predictive representations of state. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*, volume 307 of *ACM Proceedings*, pages 1176–1183, 2008b.

David Wingate, Vishal Soni, Britton Wolfe, and Satinder P. Singh. Relational knowledge with predictive state representations. In Manuela M. Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 2035–2040. AAAI Press, 2007.

Britton Wolfe and Satinder P. Singh. Predictive state representations with options. In William W. Cohen and Andrew Moore, editors, *Proceedings of the 23rd International Conference on Machine Learning (ICML 2006)*, volume 148 of *ACM Proceedings*, pages 1025–1032, 2006.

Lotfi Asker Zadeh. The concept of system, aggregate, and state in system theory. In Lotfi Asker Zadeh and Elijah Polak, editors, *System Theory*, volume 8 of *Inter-University Electronics Series*, pages 3–42. McGraw-Hill, New York, 1969.

MingJie Zhao and Herbert Jaeger. Norm observable operator models. *Neural Computation*, 22(7):1927–1959, 2010.

MingJie Zhao, Herbert Jaeger, and Michael Thon. A bound on modeling error in observable operator models and an associated learning algorithm. *Neural Computation*, 21(9):2687–2712, 2009a.

MingJie Zhao, Herbert Jaeger, and Michael Thon. Making the error-controlling algorithm of observable operator models constructive. *Neural Computation*, 21(12):3460–3486, 2009b.