Discrete-time, discrete-valued observable operator models: a tutorial[1]

Herbert Jaeger
International University Bremen
Campus Ring 1, D-28759 Bremen
phone +49 421 200 3215
email: h.jaeger@iu-bremen.de

April 25, 2012

[1] This is a continued update from the printed 1998 version. See bottom of next page for change history

Abstract

This tutorial gives a basic yet rigorous introduction to *observable operator models* (OOMs). OOMs are a recently discovered class of models of stochastic processes. They are mathematically simple in that they require only concepts from elementary linear algebra. The linear algebra nature gives rise to an efficient, consistent, unbiased, constructive learning procedure for estimating models from empirical data. The tutorial describes in detail the mathematical foundations and the practical use of OOMs for identifying and predicting discrete-time, discrete-valued processes, both for output-only and input-output systems.

key words: stochastic time series, system identification, observable operator models

Zusammenfassung

Dies Tutorial bietet eine gründliche Einführung in *observable operator Modelle* (OOMs). OOMs sind eine kürzlich entdeckte Klasse von Modellen stochastischer Prozesse. Sie sind mit den Mitteln der elementaren linearen Algebra darzustellen. Die Einfachheit der Darstellung führt zu einem effizienten, konsistenten, erwartungstreuen, konstruktiven Lernverfahren für die Induktion von Modellen aus empirischen Daten. Das Tutorial beschreibt im Detail die mathematischen Grundlagen und die praktische Verwendung von OOMs für die Identifikation und die Vorhersage zeit- und wertdiskreter Prozesse, sowohl für reine Output-Systeme (Generatoren) als auch für Input-Output-Systeme.

Stichwörter: stochastische Zeitreihen, Systemidentifikation, Lernen, observable operator models

Last change: April 26, 2012. Change history: July 2003: additional section 10.8 which relates the input-output OOMs to the PSR models of controlled stochastic processes recently developed by Littman, Sutton, Singh et al. January 2004: correction of error in formula of online learning algorithm described in Section 10.8. April 2004: new point added to section 10.8. April 2012: corrected statement in Prop. 6(b), adding qualifier "minimal-

dimensional" (pointed out by M. Thon).

1 Introduction

In this introduction, I will briefly sketch the scope of applicability of OOMs (subsection 1.1), remark on the historical roots of OOMs (subsection 1.2), and explain the basic idea which gives rise to the notion of "observable operator" (subsection 1.3).

1.1 What OOMs are (not) good for

In robotics and many other disciplines, identifying and predicting stochastic systems is important. Consider the following examples:

1. A robot learns how the world responds to its actions, by constructing a probabilistic model of action-sensing dependencies.
2. A robot with redundant and unreliable kinematics in a noisy environment learns a stochastic forward model of how its body responds to motor signals.
3. A robot designer evaluates the robot's performance by analysing stochastic time series data generated by the robot in action.
4. An ethologist wants to detect informative patterns in the sequences of actions performed by an animal.
5. A neuroethologist seeks for dynamic correlations between spike trains of several neurons.

Numerous formal models of stochastic systems are available. They vary in many aspects. I discuss a few, pointing out where OOMs fall in each case.

Discrete vs. continuous: A system can be observed in discrete time increments or in continuous time. It can be observed in discrete categories or by real-valued measurements. For instance, modeling a sequence of observations of elementary behaviors would require a discrete-time, discrete-value model like a Markov chain, whereas the trajectories of a robot arm might most appropriately be captured by a continous-time, continuous-value stochastic differential equation.

This tutorial covers OOMs for discrete-time, discrete-value systems. This is the case where the theory and learning algorithm is fully developed. OOMs can also be constructed for continuous time and/or measurements [Jaeger, 1998b], but this extension is too immature for inclusion in a tutorial.

Expressiveness: Some stochastic models can capture a greater number of empirical phenomena than others. The former are more expressive than the latter. For instance, every system which can be captured by a Markov chain can also be correctly described by a hidden Markov model, but not vice versa. Hidden Markov models are more expressive than Markov chains in that they allow to describe memory effects. For the action sequences of amoeba a Markov chain might suffice, but for modeling human speech or an intelligent robot's actions, memory is crucial. The drawback of more expressive models is that they generally are more expensive to learn and to use.

OOMs are more expressive than hidden Markov models (HMMs), the most expressive class of models available so far for which an efficient learning procedure existed. In spite of their greater expressiveness, OOMs can be learnt easier than HMMs.

Input-output or output-only: A system's output may or may not be influenced by input fed into the system. For instance, if an animal or a robot learns a stochastic model of its environment, the robot's actions may be conceived as input into the environment system, while the sensory feedback is the environment's output. If, by contrast, an agent passively observes its environment, it builds an output-only model of the environment. Theoretically, output-only models are just a special case of input-output models (case of null input), but in practice there are substantial differences. Many stochastic models are formulated only for either one of the two cases.

Like HMMs, OOMs can be formulated for both output-only and input-output systems. The latter are a canonical extension of the former. Likewise, the learning procedure for IO-models is a straightforward extension of the learning procedure for output-only systems. This tutorial provides a detailed treatment of both cases.

Degree of stochasticity For many systems it is appropriate to cast them as "essentially deterministic plus noise". Examples abound in engineering and control. Such systems are often described by a deterministic model to which a noise term is added. Other processes, like throwing dice, are intrinsically random and have no practically relevant deterministic core. Appropriate models for such systems are e.g. Markov models or more complex stochastic automata. Robots and animals and ethologists are often faced with systems which lie somewhere in between the almost deterministic and the completely random ones. The appropriate degree of stochasticity for a model depends on how much information is available which is relevant for understanding the system. For instance, if one observes an animal for a short period of time, not knowing much about its neural machinery, and not recording most of the environmental factors which co-determine the animal's actions, one must resort to an intrinsically random model. If, by contrast, one gathers data over a long period of time and/or has substantial knowledge about the animal's internal information processing and/or monitors a lot of external factors, one might strive for an essentially deterministic model.

OOMs (like HMMs) are intrinsically random models. They do not explicitly address deterministic components in a system's behavior. Therefore, they are the best suited for intrinsically random systems, or for obtaining models in the face of missing information. If one wishes to obtain a "deterministic plus noise" type of model, one should not consider pure OOMs[1].

Stationary vs. non-stationary: A system might exhibit the same stochastic regularities over time, i.e. be stationary, or change its behavior due to learning, wear, change of environment or other factors (non-stationary system behavior). Mathematical models of stochastic systems very often can capture both cases. However, the empirical data required for learning stationary vs. nonstationary systems are different. A model of a stationary process is usually acquired from a single (or a few) long series of observations, whereas non-stationary models are obtained from many short sequences of observations. For instance, a

[1] Hybrid "deterministic plus OOM"-models are being investigated by Dieter Kutzera at GMD.

> stochastic model of a grasp movement (which is nonstationary) must be distilled from many trials of that movement, while a simple robot in a simple, stable arena will reveal itself in a long, stationary sequence of motions. Stationarity is relative to the chosen timescale: a long-range stationary history is usually made from many, stochastically repeated episodes of locally non-stationary character.
>
> OOMs can be formulated (and learnt) for stationary and non-stationary systems equally well. This tutorial covers both.

In sum, using OOMs is an option if one wishes to obtain expressive models of discrete-time, discrete-valued systems which are intrinsically stochastic or where available data are insufficient to learn deterministic-plus-noise models. If one has continuous-time, continuous-value systems, or if one wishes to extract a deterministic core from one's stochastic observations, other techniques are currently better suited than OOMs.

This profile indicates many interesting applications of OOMs in robotics, ethology and other "agent sciences".

1.2 Historical notes

This section provides a brief overview of mathematical research which can be considered ancestral to OOMs.

OOMs are a generalization of hidden Markov models (HMMs)[Bengio, 1999]. HMMs have been investigated mathematically long before they became a popular tool in speech processing [Rabiner, 1990] and control engineering [Elliott *et al.*, 1995]. A particular strand of research on HMMs is of specific interest for OOMs. This research was centered on the question to decide when two HMMs are equivalent, i.e. describe the same process [Gilbert, 1959]. The problem was tackled by framing HMMs within a more general class of stochastic processes, now termed *linearly dependent processes* (LDPs). Deciding the equivalence of HMMs amounts to characterising HMM-describable processes as LDPs. This line of research came to a successful conclusion in [Ito *et al.*, 1992], where equivalence of HMMs was characterised algebraically, and where a decision algorithm was provided. That article also gives an overview of the work done in this area.

It should be emphasized that linearly dependent processes are unrelated to linear systems in the standard sense, i.e. systems whose state sequences are generated by some linear operator (e.g., [Narendra, 1995]). The term, "linearly dependent processes", refers to certain linear relationships between conditional distributions that arise in the study of general stochastic processes. LDP's are thoroughly "nonlinear" in the standard sense of the word.

The class of LDPs has been characterized in various ways. The most concise description was developed in [Heller, 1965], using methods from category theory and algebra. This approach was taken up and elaborated in a recent comprehensive account on LDPs and the mathematical theory of HMMs, viewed as a subclass of LDPs [Ito, 1992].

All of this work on HMMs and LDPs was mathematically oriented, and did not bear on the algorithmical question of learning models from data.

OOMs are just yet another characterization of LDPs. The advantage of the OOM characterization is its unprecedented mathematical simplicity, which gives rise to algorithmical efficiency.

1.3 Why they are called "observable operator models"

The name, "observable operator models", stems from the very way of how stochastic trajectories are mathematically modeled in this approach.

Traditionally, trajectories of discrete-time systems are seen as a sequence of states, which is generated by the repeated application of a single (possibly stochastic) operator T (fig. 1a). Metaphorically speaking, a trajectory is seen as a sequence of *locations* in state space, which are visited by the system due to the action of a time step operator.

In OOM theory, trajectories are perceived in a complementary fashion. From a set of operators (say, $\{T_A, T_B\}$), one operator is stochastically selected for application at every time step. The system trajectory is then identified with the sequence of operators. Thus, an observed piece of trajectory $\ldots ABAA \ldots$ would correspond to a concatenation of operators $\ldots T_A(T_A(T_B(T_A \ldots))) \ldots$ (fig. 1b). The fact that the observables are the operators themselves, led to the naming of this kind of stochastic models. An appropriate metaphor would be to view trajectories as sequences of *actions.*

Stochastic sequences of operators are a well-known object of mathematical investigation [Iosifescu and Theodorescu, 1969]. OOM theory grows out of the novel insight that the probability of selecting an operator can be computed *using the operator itself.*

The sections of this tutorial cover the following topics: (2) how a matrix representation of OOMs can be construed as a generalization of HMMs,

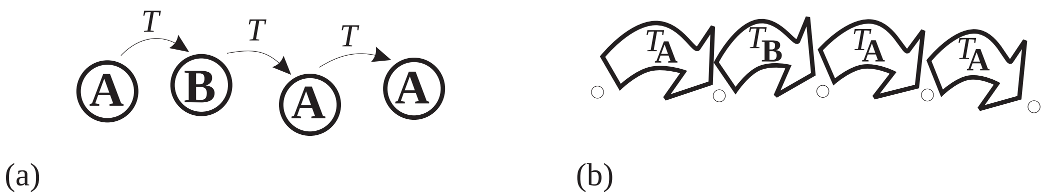

Figure 1: (a) The standard view of trajectories. A time step operator T yields a sequence ABAA of states. (b) The OOM view. Operators T_A, T_B are concatenated to yield a sequence ABAA of observables.

(3) how OOMs are used to generate and predict stochastic time series, (4) how an information-theoretic version of OOMs can be obtained from any stochastic process, (5) how these information-theoretic OOMs can be used to prove a fundamental theorem which reveals when two OOMs in matrix representation are equivalent, (6) that some low-dimensional OOMs can model processes which can be modeled either only by arbitrarily high-dimensional HMMs, or by none at all; and that one can model a conditional rise and fall of probabilities in processes timed by "probability oscillators", (7) what happens when instead of a memoryless Markov process, one "hides" a process with memory behind probabilistic observations in a "hidden LDP model", (8) how one can use the fundamental equivalence theorem to obtain OOMs whose state space dimensions can be interpreted as probabilities of certain future events, and (9.1) how these interpretable OOMs directly yield a procedure to estimate OOMs from data, a method which is constructive and gives unbiased estimates of defining parameters of a process, which is shown in (9.2), while (9.3) describes how to determine the model dimension which is appropriate for exploiting but not overfitting given data, and (9.4) indicates a variation of the OOM construction procedure for adaptive system identification. Section 10, which is very long, treats input-output OOMs in full detail, and (11), which is very brief, gives a conclusion.

2 From HMMs to OOMs

In this section, OOMs are introduced by generalization from HMMs. In this way it becomes immediately clear why the latter are a subclass of the former.

A basic HMM specifies the distribution of a discrete-time, stochastic process $(Y_t)_{t\in\mathbb{N}}$, where the random variables Y_t have finitely many outcomes from a set $\mathcal{O} = \{a_1, \ldots, a_n\}$. HMMs can be defined in several equivalent ways. In this article we will adhere to the definition which is customary in the speech recognition community and other application areas. The specification is done in two stages.

First, a Markov chain $(X_t)_{t\in\mathbb{N}}$ produces sequences of *hidden* states from a finite state set $\{s_1, \ldots, s_m\}$. Second, when the Markov chain is in state s_j at time t, it "emits" an observable outcome, with a time-invariant probability $P[Y_t = a_i \,|\, X_t = s_j]$.

Figure 2 presents an exemplary HMM.

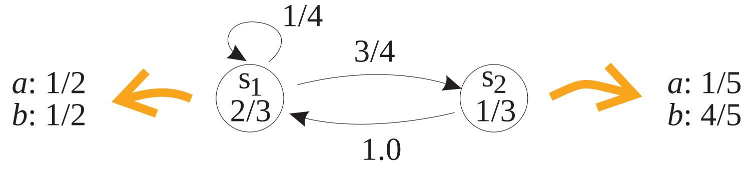

Figure 2: A HMM with two hidden states s_1, s_2 and two outcomes $\mathcal{O} = \{a, b\}$. Fine arrows indicate admissible hidden state transitions, with their corresponding probabilities marked besides. The initial state distribution $(2/3, 1/3)$ of the Markov chain is indicated inside the state circles. Emission probabilities $P[a_i \,|\, s_j]$ of outcomes are annotated besides bold grey arrows.

Formally, the state transition probabilities can be collected in a $m \times m$ matrix M which at place (i, j) contains the transition probability from state s_i to s_j (i.e., M is a Markov matrix, or *stochastic* matrix). For every $a \in \mathcal{O}$, we collect the emission probabilities $P[Y_t = a \mid X_t = s_j]$ in a diagonal *observation matrix* O_a of size $m \times m$. O_a contains, in its diagonal, the probabilities $P[Y_t = a \mid X_t = s_1], \ldots, P[Y_t = a \mid X_t = s_m]$. For the example from fig. 2, this gives

$$M = \begin{pmatrix} 1/4 & 3/4 \\ 1 & 0 \end{pmatrix}, \quad O_a = \begin{pmatrix} 1/2 & \\ & 1/5 \end{pmatrix}, \quad O_b = \begin{pmatrix} 1/2 & \\ & 4/5 \end{pmatrix}. \qquad (1)$$

In order to fully characterize a HMM, one also must supply an initial distribution $w_0 = (P[X_0 = s_1], \ldots, P[X_0 = s_m])^\top$ (superscript $\cdot^\top$ denotes transpose of vectors and matrices. Vectors are assumed to be column vectors throughout this article, unless noted otherwise). The process described by the HMM is stationary if w_0 is an invariant distribution of the Markov chain, i.e. if it satisfies

$$M^{\mathsf{T}} w_0 = w_0. \tag{2}$$

See [Doob, 1953] for details on Markov chains.

It is well-known that the matrices M and O_a ($a \in \mathcal{O}$) can be used to compute the probability of finite-length observable sequences. Let $\mathbf{1} = (1, \dots, 1)$ denote the m-dimensional row vector of units, and let $T_a := M^{\mathsf{T}} O_a$. Then the probability to observe the sequence $a_{i_0} \dots a_{i_k}$ among all possible sequences of length $k+1$ is equal to the number obtained by applying $T_{a_{i_0}}, \dots, T_{a_{i_k}}$ to w_0, and summing up the components of the resulting vector by multiplying it with $\mathbf{1}$:

$$P[a_{i_0} \dots a_{i_k}] = \mathbf{1} T_{a_{i_k}} \cdots T_{a_{i_0}} w_0. \tag{3}$$

The term $P[a_{i_0} \dots a_{i_k}]$ in (3) is a shorthand notation for $P[X_0 = a_{i_0}, \dots, X_k = a_{i_k}]$, which will be used throughout this article.

(3) is a matrix notation of the well-known "forward algorithm" for determining probabilities of observation sequences in HMMs. Proofs of (3) may be found e.g. in [Ito *et al.*, 1992] and [Ito, 1992].

M can be recovered from the operators T_a by observing that

$$M^{\mathsf{T}} = M^{\mathsf{T}} \cdot \mathbf{id} = M^{\mathsf{T}}(O_{a_1} + \cdots + O_{a_n}) = T_{a_1} + \cdots + T_{a_n}. \tag{4}$$

Eq. (3) shows that the distribution of the process (Y_t) is specified by the operators T_{a_i} and the vector w_0. Thus, the matrices T_{a_i} and w_0 contain the same information as the original HMM specification in terms of M, O_{a_i}, and w_0. I.e., one can rewrite a HMM as a structure $(\mathbb{R}^m, (T_a)_{a \in \mathcal{O}}, w_0)$, where $\mathbb{R}^m$ is the domain of the operators T_a. The HMM from the example, written in this way, becomes

$$\mathcal{M} = (\mathbb{R}^2, (T_a, T_b), w_0) = (\mathbb{R}^2, (\begin{pmatrix} 1/8 & 1/5 \\ 3/8 & 0 \end{pmatrix}, \begin{pmatrix} 1/8 & 4/5 \\ 3/8 & 0 \end{pmatrix}), (2/3, 1/3)^{\mathsf{T}}). \tag{5}$$

Now, one arrives at the definition of an OOM by (i) relaxing the requirement that M^{T} be the transpose of a stochastic matrix, to the weaker requirement that the columns of M^T each sum to 1, and by (ii) requiring from w_0 merely that it has a component sum of 1. That is, negative entries are now allowed in matrices and vectors, which are forbidden in stochastic matrices and probability vectors. Using the letter τ in OOMs in places where T appears in HMMs, and introducing $\mu = \sum_{a \in \mathcal{O}} \tau_a$ in analogy to (4), this yields:

Definition 1 *A m-dimensional OOM is a triple $\mathcal{A} = (\mathbb{R}^m, (\tau_a)_{a \in \mathcal{O}}, w_0)$, where $w_0 \in \mathbb{R}^m$ and $\tau_a : \mathbb{R}^m \mapsto \mathbb{R}^m$ are linear operators, satisfying*

1. $\mathbf{1} w_0 = 1$,
2. $\mu = \sum_{a \in \mathcal{O}} \tau_a$ *has column sums equal to 1,*
3. *for all sequences $a_{i_0} \dots a_{i_k}$ it holds that $\mathbf{1} \tau_{a_{i_k}} \cdots \tau_{a_{i_0}} w_0 \geq 0$.*

Conditions 1 and 2 reflect the relaxations (i) and (ii) mentioned previously, while condition 3 ensures that one obtains non-negative values when the OOM is used to compute probabilities. Unfortunately, condition 3 is useless for deciding or constructing OOMs. An alternative to condition 3, which is suitable for constructing OOMs, will be introduced in Section 6.

Since function concatenations of the form $\tau_{a_{i_k}} \circ \cdots \circ \tau_{a_{i_0}}$ will be used very often in the sequel, we introduce a shorthand notation for handling sequences of symbols. Following the conventions of formal language theory, we shall denote the empty sequence by ε (i.e., the sequence of length 0 which "consists" of no symbol at all), the set of all sequences of length k of symbols from $\mathcal{O}$, by $\mathcal{O}^k$; $\bigcup_{k \geq 1} \mathcal{O}^k$ by $\mathcal{O}^+$; and the set $\{\varepsilon\} \cup \mathcal{O}^+$ by $\mathcal{O}^*$. We shall write $\bar{a} \in \mathcal{O}^+$ to denote any finite sequence $a_0 \dots a_n$, and $\tau_{\bar{a}}$ to denote $\tau_{a_n} \circ \cdots \circ \tau_{a_0}$.

An OOM, as defined here, specifies a stochastic process, if one makes use of an analog of (3):

Proposition 1 *Let $\mathcal{A} = (\mathbb{R}^m, (\tau_a)_{a \in \mathcal{O}}, w_0)$ be an OOM according to the previous definition. Let $\Omega = \mathcal{O}^\infty$ be the set of all infinite sequences over $\mathcal{O}$, and $\mathfrak{A}$ be the σ-algebra generated by all finite-length initial events on Ω. Then, if one computes the probabilities of initial finite-length events in the following way:*

$$P_0[\bar{a}] := \mathbf{1} \tau_{\bar{a}} w_0, \tag{6}$$

the numerical function P_0 can be uniquely extended to a probability measure P on $(\Omega, \mathfrak{A})$, giving rise to a stochastic process $(\Omega, \mathfrak{A}, P, (X_t)_{t \in \mathbb{N}})$, where $X_n(a_1 a_2 \dots) = a_n$. If w_0 is an invariant vector of μ, i.e., if $\mu w_0 = w_0$, the process is stationary.

The proof is given in appendix A.

Since we introduced OOMs here by generalizing away from HMMs, it is clear that every process whose distribution can be specified by a HMM can also be characterized by an OOM.

I conclude this section with a remark on LDPs and OOMs. It is known that the distributions of LDPs can be characterized through matrix multiplications in a fashion which is very similar to (6) (cf. [Ito, 1992], theorem 1.8):

$$P[a_{i_0} \dots a_{i_k}] = \mathbf{1} Q I_{a_{i_k}} \dots Q I_{a_{i_0}} w_0. \tag{7}$$

The matrix Q does not depend on a, while the "projection matrices" I_a do. If one puts $Q = \mathbf{id}, I_a = \tau_a$, one easily sees that the class of processes which can be described by OOMs is the class of LDPs.

3 OOMs as generators and predictors

This section explains how to generate and predict the paths of a process $(X_t)_{t \in \mathbb{N}}$, whose distribution is specified by an OOM $\mathcal{A} = (\mathbb{R}^k, (\tau_a)_{a \in \mathcal{O}}, w_0)$. We describe the procedures mathematically and illustrate them with an example.

More precisely, the generation task consists in randomly producing, at times $t = 0, 1, 2, \dots$, outcomes $a_{i_0}, a_{i_1}, a_{i_2}, \dots$, such that (i) at time $t = 0$, the probability of producing b is equal to $\mathbf{1}\tau_b w_0$ according to (6), and (ii) at every time step $t > 0$, the probability of producing b (after $a_{i_0}, \dots, a_{i_{t-1}}$ have already been produced) is equal to $P[X_t = b \mid X_0 = a_{i_0}, \dots, X_{t-1} = a_{i_{t-1}}]$. Using (6), the latter amounts to calculating at time t, for every $b \in \mathcal{O}$, the conditional probability

$$\begin{aligned} & P[X_t = b \,|\, X_0 = a_{i_0}, \dots, X_{t-1} = a_{i_{t-1}}] \\ = & \quad \frac{P[X_0 = a_{i_0}, \dots, X_{t-1} = a_{i_{t-1}}, X_t = b]}{P[X_0 = a_{i_0}, \dots, X_{t-1} = a_{i_{t-1}}]} \\ = & \quad \mathbf{1}\tau_b \tau_{a_{i_{t-1}}} \cdots \tau_{a_{i_0}} w_0 / \mathbf{1}\tau_{a_{i_{t-1}}} \cdots \tau_{a_{i_0}} w_0 \\ = & \quad \mathbf{1}\tau_b \Big(\frac{\tau_{a_{i_{t-1}}} \cdots \tau_{a_{i_0}} w_0}{\mathbf{1}\tau_{a_{i_{t-1}}} \cdots \tau_{a_{i_0}} w_0} \Big) \\ =: & \quad \mathbf{1}\tau_b w_t, \end{aligned} \tag{8}$$

and producing at time t the outcome b with this conditional probability. Calculations of (8) can be carried out incrementally, if one observes that for $t \geq 1$, w_t can be computed from w_{t-1}:

$$w_t = \frac{\tau_{a_{t-1}} w_{t-1}}{\mathbf{1}\tau_{a_{t-1}} w_{t-1}}. \tag{9}$$

Note that all vectors w_t thus obtained have a component sum equal to 1.

Observe that the operation $\mathbf{1}\tau_b \cdot$ in (8) can be done effectively by pre-computing the vector $v_b := \mathbf{1}\tau_b$. Computing (8) then amounts to multiplying the (row) vector v_b with the (column) vector w_t, or, equivalently, it amounts to evaluating the inner product $< v_b, w_t >$.

The prediction task is completely analogous to the generation task. Given an initial realization $a_{i_0}, \ldots, a_{i_{t-1}}$ of the process up to time $t-1$, one has to calculate the probability by which an outcome b is going to occur at the next time step t. This is again an instance of (8), the only difference being that now the initial realization is not generated by oneself but is externally given.

Many-time-step probability predictions of collective outcomes can be calculated by evaluating inner products, too. Let the collective outcome $A = \{\bar{b}_1, \ldots, \bar{b}_n\}$ consist of n sequences of length $s+1$ of outcomes (i.e., outcome A is recorded when any of the sequences $\bar{b}_i$ occurs). Then, the probability that A is going to occur after an initial realization $\bar{a}$ of length $t-1$, can be computed as follows:

$$\begin{aligned}
&P[(X_t, \ldots, X_{t+s}) \in A \,|\, (X_0, \ldots, X_{t-1}) = \bar{a}] \\
&\quad = \sum_{\bar{b} \in A} P[(X_t, \ldots, X_{t+s}) = \bar{b} \,|\, (X_0, \ldots, X_{t-1}) = \bar{a}] \\
&\quad = \sum_{\bar{b} \in A} \mathbf{1}\tau_{\bar{b}} w_t \quad =: \quad \sum_{\bar{b} \in A} < v_{\bar{b}}, w_t > \\
&\quad = < \sum_{\bar{b} \in A} v_{\bar{b}}, w_t > \quad =: \quad < v_A, w_t > .
\end{aligned} \tag{10}$$

If one wants to calculate the future probability of a collective outcome A repeatedly, utilization of (10) reduces computational load considerably because the vector v_A needs to be (pre-)computed only once.

The generation procedure shall now be illustrated using the exemplary OOM $\mathcal{M}$ from (5). We first compute the vectors v_a, v_b:

$$
\begin{array}{rcccccl}
v_a & = & \mathbf{1}\tau_a & = & \mathbf{1}\begin{pmatrix} 1/8 & 1/5 \\ 3/8 & 0 \end{pmatrix} & = & (1/2, 1/5), \\
v_b & = & \mathbf{1}\tau_b & = & \mathbf{1}\begin{pmatrix} 1/8 & 4/5 \\ 3/8 & 0 \end{pmatrix} & = & (1/2, 4/5).
\end{array}
$$

Starting with $w_0 = (2/3, 1/3)$, we obtain probabilities $< v_a, w_0 > \ = 2/5, < v_b, w_0 > \ = 3/5$ of producing a vs. b at the first time step. We make a random decision for a vs. b, weighted according to these probabilities. Let's assume the dice fall for b. We now compute $w_1 = \tau_b w_0 / \mathbf{1}\tau_b w_0 = (7/12, 5/12)^{\mathsf{T}}$. For the next time step, we repeat these computations with w_1 in place of w_0, etc., etc.

4 From stochastic processes to OOMs

This section introduces OOMs again, but this time in a top-down fashion, starting from general stochastic processes. This alternative route clarifies the fundamental nature of observable operators. Furthermore, the insights obtained in this section will yield a short and instructive proof of the central theorem of OOM equivalence, to be presented in the next section. The material presented here is not required after the next section and may be skipped by readers with not so keen an interest in probability theory.

In Section 2, we have described OOMs as structures $(\mathbb{R}^m, (\tau_a)_{a \in \mathcal{O}}, w_0)$. In this section, we will arrive at isomorphic structures $(\mathfrak{G}, (\mathfrak{t}_a)_{a \in \mathcal{O}}, \mathfrak{g}_\varepsilon)$, where again $\mathfrak{G}$ is a vector space, $(\mathfrak{t}_a)_{a \in \mathcal{O}}$ is a family of linear operators on $\mathfrak{G}$, and $\mathfrak{g}_\varepsilon \in \mathfrak{G}$. However, the vector space $\mathfrak{G}$ is now a space of certain numerical prediction functions. In order to discriminate OOMs characterized on spaces $\mathfrak{G}$ from the "ordinary" OOMs, we shall call $(\mathfrak{G}, (\mathfrak{t}_a)_{a \in \mathcal{O}}, \mathfrak{g}_\varepsilon)$ an *predictor-space OOM.*

Let $(X_t)_{t \in \mathbb{N}}$, or for short, (X_t) be a discrete-time stochastic process with values in a finite set $\mathcal{O}$. Then, the distribution of (X_t) is uniquely characterized by the probabilities of finite initial subsequences, i.e. by all probabilities of the kind $P[\bar{a}]$, where $\bar{a} \in \mathcal{O}^+$.

We introduce a shorthand for conditional probabilities, by writing $P[\bar{a} \,|\, \bar{b}]$ for $P[(X_n, \ldots, X_{n+s}) = \bar{a} \,|\, (X_0, \ldots, X_{n-1}) = \bar{b}]$. We shall formally write the unconditional probabilities as conditional probabilities, too, with the empty condition ε, i.e. we use the notation $P[\bar{a} \,|\, \varepsilon] := P[(X_0 \ldots X_s) = \bar{a}] = P[\bar{a}]$.

Thus, the distribution of (X_t) is also uniquely characterized by its *conditional continuation probabilities*, i.e. by the conditional probabilities $P[\bar{a}\,|\,\bar{b}]$, where $\bar{a} \in \mathcal{O}^+, \bar{b} \in \mathcal{O}^*$.

For every $\bar{b} \in \mathcal{O}^*$, we collect all conditioned continuation probabilities of $\bar{b}$ into a numerical function

$$\begin{aligned} \mathfrak{g}_{\bar{b}} : \mathcal{O}^+ &\rightarrow \mathbb{R}, \\ \bar{a} &\mapsto P[\bar{a}\,|\,\bar{b}], \text{ if } P[\bar{b}] \neq 0 \\ &\mapsto 0, \text{ if } P[\bar{b}] = 0. \end{aligned} \tag{11}$$

The set $\{\mathfrak{g}_{\bar{b}}\,|\,\bar{b} \in \mathcal{O}^*\}$ uniquely characterizes the distribution of (X_t), too. Intuitively, a function $\mathfrak{g}_{\bar{b}}$ describes the future distribution of the process after an initial realization $\bar{b}$.

Let $\mathfrak{D}$ denote the set of all functions from $\mathcal{O}^+$ into the reals, i.e. the numerical functions on non-empty sequences. $\mathfrak{D}$ canonically becomes a real vector space if one defines scalar multiplication and vector addition as follows: for $\mathfrak{d}_1, \mathfrak{d}_2 \in \mathfrak{D}$, $\alpha, \beta \in \mathbb{R}$, $\bar{a} \in \mathcal{O}^+$ put $(\alpha\mathfrak{d}_1 + \beta\mathfrak{d}_2)(\bar{a}) := \alpha(\mathfrak{d}_1(\bar{a})) + \beta(\mathfrak{d}_2(\bar{a}))$.

Let $\mathfrak{G} = \langle\{\mathfrak{g}_{\bar{b}}\,|\,\bar{b} \in \mathcal{O}^*\}\rangle_{\mathfrak{D}}$ denote the linear subspace spanned in $\mathfrak{D}$ by the conditioned continuations. Intuitively, $\mathfrak{G}$ is the (linear closure of the) space of future distributions of the process (X_t).

Now we are halfway done with our construction of $(\mathfrak{G}, (\mathfrak{t}_a)_{a\in\mathcal{O}}, \mathfrak{g}_\varepsilon)$: we have constructed the vector space $\mathfrak{G}$, which corresponds to $\mathbb{R}^m$ in the "ordinary" OOMs from Section 2, and we have defined the initial vector $\mathfrak{g}_\varepsilon$, which is the counterpart of w_0. It remains for us to define the family of observable operators.

In order to specify a linear operator on a vector space, it suffices to specify the values the operator takes on a basis of the vector space. Choose $\mathcal{O}_0^* \subseteq \mathcal{O}^*$ such that the set $\{\mathfrak{g}_{\bar{b}}\,|\,\bar{b} \in \mathcal{O}_0^*\}$ is a basis of $\mathfrak{G}$. Define, for every $a \in \mathcal{O}$, a linear function $\mathfrak{t}_a : \mathfrak{G} \rightarrow \mathfrak{G}$ by putting

$$\mathfrak{t}_a(\mathfrak{g}_{\bar{b}}) := P[a\,|\,\bar{b}]\mathfrak{g}_{\bar{b}a} \tag{12}$$

for all $\bar{b} \in \mathcal{O}_0^*$ ($\bar{b}a$ denotes the concatenation of the sequence $\bar{b}$ with a). It turns out that (12) carries over from basis elements $\bar{b} \in \mathcal{O}_0^*$ to all $\bar{b} \in \mathcal{O}^*$:

Proposition 2 *For all $\bar{b} \in \mathcal{O}^*$, $a \in \mathcal{O}$, the linear operator $\mathfrak{t}_a$ satisfies the condition*

$$\mathfrak{t}_a(\mathfrak{g}_{\bar{b}}) = P[a\,|\,\bar{b}]\mathfrak{g}_{\bar{b}a}. \tag{13}$$

The proof is given in appendix B. Intuitively, the operator $\mathfrak{t}_a$ describes the change of knowledge about a process due to an incoming observation of a. More precisely, assume that the process has initially been observed up to time n. That is, an initial observation $\bar{b} = b_0 \ldots b_n$ has been made. Our knowledge about the state of the process at this moment is tantamount to the predictor function $\mathfrak{g}_{\bar{b}}$. Then assume that at time $n+1$ an outcome a is observed. After that, our knowledge about the process state is then expressed by $\mathfrak{g}_{\bar{b}a}$. But this is (up to scaling by $P[a\,|\,\bar{b}]$) just the result of applying $\mathfrak{t}_a$ to the old state, $\mathfrak{g}_{\bar{b}}$.

The operators $(\mathfrak{t}_a)_{a\in\mathcal{O}}$ are the analog of the observable operators $(\tau_a)_{a\in\mathcal{O}}$ in OOMs and can likewise be used to compute probabilities of finite sequences:

Proposition 3 *Let $\{\mathfrak{g}_{\bar{b}}\,|\,\bar{b} \in \mathcal{O}_0^*\}$ be a basis of $\mathfrak{G}$. Let $\bar{a} := a_{i_0} \ldots a_{i_k}$ be an initial realization of (X_t) of length $k+1$. Let $\sum_{i=1,\ldots,n} \alpha_i \mathfrak{g}_{\bar{b}_i} = \mathfrak{t}_{a_{i_k}} \ldots \mathfrak{t}_{a_{i_0}} \mathfrak{g}_\varepsilon$ be the linear combination of $\mathfrak{t}_{a_{i_k}} \ldots \mathfrak{t}_{a_{i_0}} \mathfrak{g}_\varepsilon$ from basis vectors. Then it holds that*

$$P[\bar{a}] = \sum_{i=1,\ldots,n} \alpha_i. \tag{14}$$

Note that (14) is valid for any basis $\{\mathfrak{g}_{\bar{b}}\,|\,\bar{b} \in \mathcal{O}_0^*\}$. The proof can be found in appendix C. (14) corresponds exactly to (6), since left-multiplication of a vector with $\mathbf{1}$ amounts to summing the vector components, which in turn are the coefficients of that vector w.r.t. a vector space basis.

Due to (14), the distribution of the process (X_t) is uniquely characterized by the observable operators $(\mathfrak{t}_a)_{a\in\mathcal{O}}$. Conversely, these operators are uniquely defined by the distribution of (X_t). I.e., the following definition makes sense:

Definition 2 *Let $(X_t)_{t\in\mathbb{N}}$ be a stochastic process with values in a finite set $\mathcal{O}$. The structure $(\mathfrak{G}, (\mathfrak{t}_a)_{a\in\mathcal{O}}, \mathfrak{g}_\varepsilon)$ is called* the *predictor-space observable operator model of the process. The vector space dimension of $\mathfrak{G}$ is called the dimension of the process and is denoted by dim(X_t).*

I remarked in the introduction that stochastic processes have previously been characterized in terms of vector spaces. Although the vector spaces were constructed in other ways than $\mathfrak{G}$, they lead to equivalent notions of process dimension. [Heller, 1965] called finite-dimensional (in our sense) stochastic processes *finitary*; in [Ito *et al.*, 1992] the process dimension (if finite) was called *minimum effective degree of freedom*.

(13) clarifies the fundamental character of observable operators: $\mathfrak{t}_a$ describes how the knowledge about the process's future changes through an observation of a. The power of the observable operator idea lies in the fact that these operators turn out to be linear (proposition 2). I have only treated the discrete time, discrete value case here. However, predictor-space OOMs can be defined in a similar way also for continuous-time, arbitrary-valued processes (sketch in [Jaeger, 1998b]). It turns out that in those cases, the resulting observable operators are linear, too. In a nutshell, the change of knowledge about a process due to incoming observations is a fundamentally linear phenomenon.

In the remainder of this section, I describe how the dimension of a process is related to the dimensions of ordinary OOMs of that process.

Proposition 4 *1. If (X_t) is a process with finite dimension m, then an m-dimensional ordinary OOM of this process exists.*

2. A process (X_t) whose distribution is described by a k-dimensional OOM $\mathcal{A} = (\mathbb{R}^k, (\tau_a)_{a \in \mathcal{O}}, w_0)$ has a dimension $m \leq k$.

The proof is in the appendix. Thus, if a process (X_t) has dimension m, and we have a k-dimensional OOM $\mathcal{A}$ of (X_t), we know that a m-dimensional OOM $\mathcal{A}'$ exists which is equivalent to $\mathcal{A}$ in the sense of specifying the same distribution. Furthermore, $\mathcal{A}'$ is minimal-dimensional in its equivalence class. A minimal-dimensional OOM $\mathcal{A}'$ can be constructively obtained from $\mathcal{A}$ in several ways, all of which amount to an implicit construction of the predictor-space OOM of the process specified by $\mathcal{A}$. Since the learning algorithm presented in later sections can be used for this construction, too, I do not present a dedicated procedure for obtaining minimal-dimensional OOMs here.

5 Equivalence of OOMs

Given two OOMs $\mathcal{A} = (\mathbb{R}^k, (\tau_a)_{a \in \mathcal{O}}, w_0), \mathcal{B} = (\mathbb{R}^l, (\tau'_a)_{a \in \mathcal{O}}, w'_0)$, when are they *equivalent* in the sense that they describe the same distribution? This question can be answered using the insights gained in the previous section.

First, construct minimal-dimensional OOMs $\mathcal{A}', \mathcal{B}'$ which are equivalent to $\mathcal{A}$ and $\mathcal{B}$, respectively. If the dimensions of $\mathcal{A}', \mathcal{B}'$ are not equal, then $\mathcal{A}$ and $\mathcal{B}$ are not equivalent. We can therefore assume that the two OOMs whose equivalence we wish to ascertain have the same (and minimal) dimension. Then, the answer to our question is given in the following proposition:

Proposition 5 *Two minimal-dimensional OOMs* $\mathcal{A} = (\mathbb{R}^m, (\tau_a)_{a\in\mathcal{O}}, w_0)$, $\mathcal{B} = (\mathbb{R}^m, (\tau'_a)_{a\in\mathcal{O}}, w'_0)$ *are equivalent iff there exists a bijective linear map* $\varrho : \mathbb{R}^m \to \mathbb{R}^m$*, satisfying the following conditions:*

1. $\varrho(w_0) = w'_0$*,*

2. $\tau'_a = \varrho\tau_a\varrho^{-1}$ *for all* $a \in \mathcal{O}$*,*

3. $\mathbf{1}v = \mathbf{1}\varrho v$ *for all (column) vectors* $v \in \mathbb{R}^m$*.*

Sketch of proof: $\Leftarrow$: trivial. $\Rightarrow$: We have done all the hard work in the previous section! Let $\sigma_{\mathcal{A}}, \sigma_{\mathcal{B}}$ be the canonical projections from $\mathcal{A}, \mathcal{B}$ on the predictor-space OOM of the process specified by $\mathcal{A}$ (and hence by $\mathcal{B}$). Observe that $\sigma_{\mathcal{A}}, \sigma_{\mathcal{B}}$ are bijective linear maps which preserve the component sum of vectors. Define $\varrho := \sigma_{\mathcal{B}}^{-}1 \circ \sigma_{\mathcal{A}}$. Then, *(1)* follows from $\sigma_{\mathcal{A}}(w_0) = \sigma_{\mathcal{B}}(w'_0) = \mathfrak{g}_\varepsilon$, *(2)* follows from $\forall \bar{c} \in \mathcal{O}^+ : \quad \sigma(\tau_{\bar{c}} w_0) = \sigma(\tau'_{\bar{c}} w_0) = P[\bar{c}]\mathfrak{g}_{\bar{c}}$, and *(3)* from the fact that $\sigma_{\mathcal{A}}, \sigma_{\mathcal{B}}$ preserve component sum of vectors.

6 A non-HMM linearly dependent process

The question of when a LDP can be captured by a HMM has been fully answered in the literature (original result in [Heller, 1965], refinements in [Ito, 1992]), and examples of non-HMM LDPs have been given. I briefly restate the results, and then describe an example of such a process which is simpler than the examples given in the literature. The aim is to provide an intuitive insight in which sense the class of LDPs is “larger” than the class of processes which can be captured by HMMs.

Characterizing HMMs as LDPs heavily draws on the theory of convex cones and non-negative matrices. I first introduce some concepts, following the notation of a standard textbook [Berman and Plemmons, 1979].

With a set $S \subseteq \mathbb{R}^n$ we associate the set S^G, the *set generated by* S, which consists of all finite nonnegative linear combinations of elements of S. A set

$K \subseteq \mathbb{R}^n$ is defined to be a *convex cone* if $K = K^G$. A convex cone K^G is called *n-polyhedral* if K has n elements. A cone K is *pointed* if for every nonzero $v \in K$, the vector $-v$ is not in K. A cone is *proper* if it is pointed, closed, and its interior is not empty.

Using these concepts, the following theorem in $(a1)$, $(a2)$ gives two conditions which individually are equivalent to condition 3 in definition 1, and (b) refines condition $(a1)$ for determining when an OOM is equivalent to a HMM. Finally, (c) states necessary conditions which every τ_a in an OOM must satisfy.

Proposition 6 *(a1) Let $\mathcal{A} = (\mathbb{R}^m, (\tau_a)_{a\in\mathcal{O}}, w_0)$ be a structure consisting of linear maps $(\tau_a)_{a\in\mathcal{O}}$ on $\mathbb{R}^m$ and a vector $w_0 \in \mathbb{R}^m$. Let $\mu := \sum_{a\in\mathcal{O}} \tau_a$. Assume that the first two conditions from definition 1 hold, i.e. $\mathbf{1}w_0 = 1$ and μ has column sums equal to 1. Then $\mathcal{A}$ is an OOM if and only if there exist convex cones $(K_a)_{a\in\mathcal{O}}$ satisfying the following conditions:*

1. *$\mathbf{1}v \geq 0$ for all $v \in K_a$ (where $a \in \mathcal{O}$),*
2. *$w_0 \in (\bigcup_{a\in\mathcal{O}} K_a)^G$,*
3. *$\forall a, b \in \mathcal{O} :\ \tau_b K_a \subseteq K_b$.*

(a2) Using the same assumptions as before, $\mathcal{A}$ is an OOM if and only if there exists a convex cone K satisfying the following conditions:

1. *$\mathbf{1}v \geq 0$ for all $v \in K$,*
2. *$w_0 \in K$,*
3. *$\forall a \in \mathcal{O} :\ \tau_a K \subseteq K$.*

(b) Assume that $\mathcal{A}$ is a minimal-dimensional OOM. Then there exists a hidden Markov model equivalent to $\mathcal{A}$ if and only if a convex cone K according to condition (a2) exists which is n-polyhedral for some n. n can be selected such that it is not greater than the minimal state number for HMMs equivalent to $\mathcal{A}$.

(c) Let $\mathcal{A}$ be a minimal-dimensional OOM, and τ_a be one of its observable operators, and K be a cone according to (a2). Then (i) the spectral radius $\varrho(\tau_a)$ of τ_a is an eigenvalue of τ_a, (ii) the degree of $\varrho(\tau_a)$ is greater or equal to the degree of any other eigenvalue λ with $\mid lambda \mid = \varrho(\tau_a)$, and (iii) an

eigenvector of corresponding to $\varrho(\tau_a)$ lies in K. (The degree of an eigenvalue λ of a matrix is the size of the largest diagonal block in the Jordan canonical form of the matrix, which contains λ).

Notes on the proof. The proof of parts (*a1*) and (*b*) go back to [Heller, 1965] and have been reformulated in [Ito, 1992][2]. The equivalence of (*a1*) with (*a2*) is an easy exercise. The conditions collected in (*c*) are equivalent to the statement that $\tau_a K \subseteq K$ for some proper cone K (proof in theorems 3.2 and 3.5 in [Berman and Plemmons, 1979]). It is easily seen that for a minimal-dimensional OOM, the cone K required in (*a2*) is proper. Thus, (*c*) is a direct implication of (*a2*).

Proposition 6 has two simple but interesting implications: (i) every two-dimensional OOM is equivalent to a HMM (since all cones in two dimensions are polyhedral); (ii) every non-negative OOM (i.e., matrices τ_a have only non-negative entries) is equivalent to a HMM (since non-negative matrices map the positive orthant, which is a polyhedral cone, on itself).

Part (*c*) is sometimes useful to rule out a structure $\mathcal{A}$ as an OOM, by showing that some τ_a fails to satisfy the conditions given. Unfortunately, even if every τ_a of a structure $\mathcal{A}$ satisfies the conditions in (*c*), $\mathcal{A}$ need not be a valid OOM. Imperfect as it is, however, (*c*) is the strongest result available at this moment in the direction of characterising OOMs.

Proposition 6 is particularly useful to *build* OOMs from scratch, starting with a cone K and constructing observable operators satisfying $\tau_a K \subseteq K$. Note, however, that the theorem provides no means to *decide*, for a given structure $\mathcal{A}$, whether $\mathcal{A}$ is a valid OOM, since the theorem is non-constructive w.r.t. K.

More specifically, part (*b*) yields a construction of OOMs which are not equivalent to any HMM. This will be demonstrated in the remainder of this section.

Let $\tau_\varphi : \mathbb{R}^3 \to \mathbb{R}^3$ be the linear mapping which right-rotates $\mathbb{R}^3$ by an angle φ around the first unit vector $e_1 = (1, 0, 0)$. Select some angle φ which is not a rational multiple of 2π. Then, put $\tau_a := \alpha\tau_\varphi$, where $0 < \alpha < 1$. Any 3-dimensional process described by a 3-dimensional OOM containing such a τ_a is not equivalent to any HMM: let K_a be a convex cone corresponding to a according to proposition 6(*a1*). Due to condition *3*, K_a must satisfy $\tau_a K_a \subseteq K_a$. Since τ_a rotates any set of vectors around $(1, 0, 0)$ by φ, this

[2] Heller and Ito use a different definition for HMMs, which yields a different version of the minimality statement in part (*b*)

implies that K_a is rotation symmetric around $(1,0,0)$ by φ. Since φ is a non-rational multiple of 2π, K_a cannot be polyhedral. According to (b), this implies that an OOM which features this τ_a cannot be equivalent to any HMM.

I describe now such an OOM with $\mathcal{O} = \{a, b\}$. The operator τ_a is fixed, according to the previous considerations, by selecting $\alpha = 0.5$ and $\varphi = 1.0$. For τ_b, we take an operator which projects every $v \in \mathbb{R}^3$ on a multiple of $(.75, 0, .25)$, such that $\mu = \tau_a + \tau_b$ has column vectors with component sums equal to 1 (cf. definition 1(2)). The circular convex cone K whose border is obtained from rotating $(.75, 0, .25)$ around $(1, 0, 0)$ obviously satisfies the conditions in proposition 6(*a2*). Thus, we obtain a valid OOM provided that we select $w_0 \in K$. Using abbreviations $s := \sin(1.0), c := \cos(1.0)$, the matrices read as follows

$$\begin{aligned}
\tau_a &= 0.5 \begin{pmatrix} 1 & 0 & 0 \\ 0 & c & s \\ 0 & -s & c \end{pmatrix} \\
\tau_b &= \begin{pmatrix} .75 \cdot .5 & .75(1 - .5c + .5s) & .75(1 - .5s - .5c) \\ 0 & 0 & 0 \\ .25 \cdot .5 & .25(1 - .5c + .5s) & .25(1 - .5s - .5c) \end{pmatrix}.
\end{aligned} \qquad (15)$$

As starting vector w_0 we take $(.75, 0, .25)$, to obtain an OOM $\mathcal{C} = (\mathbb{R}^3, (\tau_a, \tau_b), w_0)$. I will briefly describe the phenomenology of the process generated by $\mathcal{C}$. The first observation is that every occurrence of b "resets" the process to the initial state w_0. Thus, we only have to understand what happens after uninterrupted sequences of a's. I.e., we should look at the conditional probabilities $P[\cdot\,|\,\varepsilon], P[\cdot\,|\,a], P[\cdot\,|\,aa], \ldots$, i.e., at $P[\cdot\,|\,a^t]$, where $t = 0, 1, 2 \ldots$. Figure 3 gives a plot of $P[a\,|\,a^t]$. The process could amply be called a "probability clock", or "probability oscillator"!

Rotational operators can be exploited for "timing" effects. In our example, for instance, if the process would be started in the state according to $t = 4$ in fig. 3, there would be a high chance for two initial a's to occur, with a rapid drop in probability for a third or fourth. Such non-exponential-decay duration patterns for identical sequences are difficult to achieve with HMMs. Essentially, HMMs offer two possibilities for identical sequences: (i) recurrent transitions into a state where a is emitted, (ii) transitions along sequences of states, each of which can emit a. Option (i) is cumbersome because recurrent transitions imply exponential decay of state, which is unsuitable for

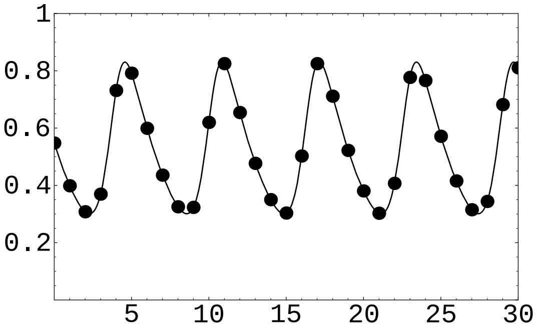

Figure 3: The rise and fall of probability to obtain a from the "probability clock" $\mathcal{C}$. Horizontal axis represents time steps t, vertical axis represents the probabilities $P[a \mid a^t] = \mathbf{1}\tau_a(\tau_a^t w_0 / \mathbf{1}\tau_a^t w_0)$, which are rendered as dots. The solid line connecting the dots is given by $f(t) = \mathbf{1}\tau_a(\tau_t w_0 / \mathbf{1}\tau_t w_0)$, where τ_t is the rotation by angle t described in the text.

many empirical processes; option (ii) blows up model size. A more detailed discussion of this problem in HMMs can be found in [Rabiner, 1990].

If one defines a three-dimensional OOM $\mathcal{C}'$ in a similar manner, but with a rational fraction of 2π as an angle of rotation, one obtains a process which can be modeled by a HMM. It follows from proposition $6(b)$ that the minimal HMM state number in this case is at least k, where k is the smallest integer such that $k\varphi$ is a multiple of 2π. Thus, the smallest HMM equivalent to a suitably chosen 3-dimensional OOM can have an arbitrarily great number of states. In a similar vein, any HMM which would give a reasonable approximation to the process depicted in fig. 3, would require at least 6 states, since in this process it takes approximately 6 time steps for one full rotation.

7 Hidden LDP models

The basic metaphor of a HMM is one of a hidden stochastic process which is indirectly observed by probabilistic measurements, viz., by emitted outcomes. In this section I describe what happens when instead of a Markov chain, the hidden process is a linearly dependent process.

Assume that one has a finite set $\mathcal{S} = \{s_1, \ldots, s_k\}$ of "hidden" outcomes, and a hidden m-dimensional linearly dependent process $(X_t)_{t\in\mathbb{N}}$ specified by an OOM $\mathcal{H} = (\mathbb{R}^m, (\tau_s)_{s\in\mathcal{S}}, w_0)$. When the hidden process produces s_i, an observable outcome a_j from a finite set $\mathcal{O} = \{a_1, \ldots, a_l\}$ is emitted with a fixed probability $p_{s_i a_j} := P[Y_t = a_j \,|\, X_t = s_i]$.

It turns out that the observable process (Y_t) is also a LDP, and has dimension m at most. This has already been noted in a special case (unit emission probabilities) in [Heller, 1965] (proposition 2.3). Here I explain why it is generally true.

Consider the combined process $(Z_t) := (X_t, Y_t)$ with values in $\mathcal{S} \times \mathcal{O}$. It is easy to see that the distribution of (Z_t) is characterized by the OOM $(\mathbb{R}^m, (\tau_{(s,a)})_{(s,a)\in\mathcal{S}\times\mathcal{O}}, w_0)$, where $\tau_{(s,a)} := p_{sa}\tau_s$. Define

$$\tau_a := \sum_{s\in\mathcal{S}} \tau_{(s,a)} = \sum_{s\in\mathcal{S}} p_{sa}\tau_s. \tag{16}$$

By a straightforward computation similar to the one in (10), it can be concluded that (Y_t) is modeled by the m-dimensional OOM $\mathcal{A} = (\mathbb{R}^m, (\tau_a)_{a\in\mathcal{O}}, w_0)$. Therefore, $\dim(Y_t) \leq m$.

Loosely speaking, this means that the class of linearly dependent processes is closed under uncertain observation.

In many applications of HMMs, one wishes to gain information about most likely hidden state sequences, given a sequence of observable outcomes. From a mathematical point of view, this is justified because it can be shown (at least in some cases of interest) that the hidden Markov chain (X_t) is uniquely determined in distribution by the observable process (Y_t) ([Jaeger, 1997a], proposition 18). However, if one allows the hidden process to be a LDP, this is not longer warranted. For a given observable process (Y_t), there exist many *non-equivalent* hidden LDPs which can outwardly produce (Y_t). Different assumptions about the emission probabilities lead to different hidden processes. Therefore, the notion of "most likely hidden state sequence" is no longer well-defined. This observation is likely to have some methodological importance (especially for speech recognition) and shall now be described in more detail.

Let the observable process (Y_t) have an OOM $\mathcal{A} = (\mathbb{R}^m, (\tau_a)_{a\in\mathcal{O}}, w_0)$. We wish to find hidden LDPs $\mathcal{H} = (\mathbb{R}^m, (\tau_s)_{s\in\mathcal{S}}, w_0)$, with appropriate emission probabilities p_{sa}, such that (Y_t) is modeled by $\mathcal{H}$ and the emission probabilities, as described above. To this end, we first treat the case that $\mathcal{O} = \{a_1, \ldots, a_n\}$ and $\mathcal{S} = \{s_1, \ldots, s_n\}$ each have n elements.

Choose emission probabilities arbitrarily, except for the constraint that the $n \times n$ matrix $(p_{s_i a_j})$ be nonsingular. According to (16), the to-be-found observable operators τ_s of the hidden process must satisfy the following system of linear equations:

$$\begin{array}{rcl} \tau_{a_1} & = & p_{s_1 a_1}\tau_{s_1} + \cdots + p_{s_n a_1}\tau_{s_n} \\ & \cdots & \\ \tau_{a_n} & = & p_{s_1 a_n}\tau_{s_1} + \cdots + p_{s_n a_n}\tau_{s_n}. \end{array} \tag{17}$$

This system is solvable for the matrices τ_{s_i} since $(p_{s_i a_j})$ is nonsingular. The structure $\mathcal{H} = (\mathbb{R}^m, (\tau_s)_{s\in\mathcal{S}}, w_0)$ thus obtained may or may not be a valid OOM (condition *3* from definition 1 need not hold). However, if one chooses $(p_{s_i a_j}) = \mathbf{id}$, one gets $\mathcal{H} = \mathcal{A}$, i.e. a valid OOM. By an argument of continuity, one can expect at least that if one chooses $(p_{s_i a_j})$ in some ε-neighborhood of $\mathbf{id}$, $\mathcal{H}$ will be a valid OOM. If this indeed holds for $\mathcal{H}$, one has found a hidden LDP model for the process (Y_t) with emission probabilities $p_{s_i a_j}$.

The following example demonstrates that indeed one can find hidden LDPs that are markedly different from each other, yet give rise to the same observable process. We take for $\mathcal{A}$ the exemplary HMM $\mathcal{M}$ from Section 2, eq. (5). It was obtained from a two-state Markov chain by the following emission probabilities (cf. (1)):

$$(p^{\mathrm{MARKOV}}_{s_i a_j}) = \begin{pmatrix} 1/2 & 1/2 \\ 1/5 & 4/5 \end{pmatrix}.$$

Now, if we choose the following quite different emission probabilities:

$$(p^{\mathrm{LDP}}_{s_i a_j}) = \begin{pmatrix} 3/4 & 1/4 \\ 0 & 1 \end{pmatrix},$$

solving (17) yields

$$\mathcal{H} = (\mathbb{R}^2, (\begin{pmatrix} 1/6 & 4/15 \\ 1/2 & 0 \end{pmatrix}, \begin{pmatrix} 1/12 & 11/15 \\ 1/4 & 0 \end{pmatrix}), (2/3, 1/3)^{\mathsf{T}}),$$

which is a valid OOM, since the observable operators and the starting vector are non-negative, which ensures condition *3* from definition 1. We have thus obtained two quite different hidden processes (namely, the original Markov chain and the process specified by $\mathcal{H}$) which give rise to the same observable process (namely, the process specified by $\mathcal{M}$), by virtue of different emission probabilities.

A similar multitude of hidden processes also occurs in cases where $\mathcal{O} = \{a_1, \ldots, a_l\}$ and $\mathcal{S} = \{s_1, \ldots, s_k\}$ have different numbers of elements. To see this, assume that one has a hidden OOM $\mathcal{H} = (\mathbb{R}^m, (\tau_s)_{s \in \mathcal{S}}, w_0)$ which gives rise to an observable LPD characterized by $\mathcal{A} = (\mathbb{R}^m, (\tau_a)_{a \in \mathcal{O}}, w_0)$, by virtue of emission probabilities $(p_{s_i a_j})_{1 \leq i \leq k, 1 \leq j \leq l}$. According to the previous arguments, it is typically possible to find an OOM $\mathcal{H}' = (\mathbb{R}^m, (\tau_{s'})_{s' \in \mathcal{S}'}, w_0)$ with $\mathcal{S}' = \{s'_1, \ldots, s'_k\}$, which gives rise to the process described by $\mathcal{H}$ by virtue of emission probabilities $(p'_{s'_h s_i})_{1 \leq i,h \leq k} \neq \mathbf{id}$. It is easy to see that $\mathcal{H}'$ gives rise to the observable process described by $\mathcal{A}$ by virtue of emission probabilities $(p_{s'_h a_j})_{1 \leq h \leq k, 1 \leq j \leq l} = (p'_{s'_h s_i})(p_{s_i a_j})$.

8 Interpretable OOMs

Some minimal-dimension OOMs have a remarkable property: their state space dimensions can be interpreted as probabilities of certain future outcomes. These *interpretable* OOMs will be described in this section.

First some terminology. Let $(X_t)_{t \in \mathbb{N}}$ be an m-dimensional LDP. For some suitably large k, let $\mathcal{O}^k = A_1 \cup \cdots \cup A_m$ be a partition of the set of sequences of length k into m disjoint nonempty subsets. The collective outcomes A_i are called *characteristic events* if some sequences $\bar{b}_1, \ldots, \bar{b}_m$ exist such that the $m \times m$ matrix

$$(P[A_i \,|\, \bar{b_j}])_{i,j} \tag{18}$$

is nonsingular (where $P[A_i \,|\, \bar{b_j}]$ denotes $\sum_{\bar{a} \in A_i} P[\bar{a} \,|\, \bar{b_j}]$). Every LDP has characteristic events:

Proposition 7 *Let $(X_t)_{t \in \mathbb{N}}$ be an m-dimensional LDP. Then there exists some $k \geq 1$ and a partition $\mathcal{O}^k = A_1 \cup \cdots \cup A_m$ of $\mathcal{O}^k$ into characteristic events.*

The proof is given in the appendix. Let $\mathcal{A} = (\mathbb{R}^m, (\tau_a)_{a \in \mathcal{O}}, w_0)$ be an m-dimensional OOM of the process (X_t). Using the characteristic events $A_1, \ldots, A_m$, we shall now construct from $\mathcal{A}$ an equivalent, *interpretable* OOM $\mathcal{A}(A_1, \ldots, A_m)$, which has the property that the m state vector components represent the probabilities of the m characteristic events to occur.

More precisely, if during the generation procedure described in Section 3, $\mathcal{A}(A_1, \ldots, A_m)$ is in state $w_t = (w_t^1, \ldots, w_t^m)$ at time t, the probability $P[(X_{t+1}, \ldots, X_{t+k}) \in A_i \mid w_t]$ that the collective outcome A_i is generated in the next k time steps, is equal to w_t^i. In shorthand notation:

$$P[A_i \,|\, w_t] = w_t^i. \tag{19}$$

We shall use proposition 5 to obtain $\mathcal{A}(A_1, \ldots, A_m)$. Define $\tau_{A_i} := \sum_{\bar{a} \in A_i} \tau_{\bar{a}}$. Define a mapping $\varrho : \mathbb{R}^m \to \mathbb{R}^m$ by

$$\varrho(x) := (\mathbf{1}\tau_{A_1} x, \ldots, \mathbf{1}\tau_{A_m} x). \tag{20}$$

The mapping ϱ is obviously linear. It is also bijective, since the matrix $(P[A_i \,|\, \bar{b}_j]) = (\mathbf{1}\tau_{A_i} x_j)$, where $x_j := \tau_{\bar{b}_j} w_0 / \mathbf{1}\tau_{\bar{b}_j} w_0$, is nonsingular. Furthermore, ϱ preserves component sums of vectors, since for $i = 1, \ldots, m$ it holds that $\mathbf{1}x_j = 1 = \mathbf{1}(P[A_1 \,|\, x_j], \ldots, P[A_m \,|\, x_j]) = \mathbf{1}(\mathbf{1}\tau_{A_1} x, \ldots, \mathbf{1}\tau_{A_m} x) = \mathbf{1}\varrho(x_j)$ (note that a linear map preserves component sums if it preserves component sums of basis vectors). Hence, ϱ satisfies the conditions of proposition 5. We therefore obtain an OOM equivalent to $\mathcal{A}$ by putting

$$\mathcal{A}(A_1, \ldots, A_m) = (\mathbb{R}^m, (\varrho \tau_a \varrho^{-1})_{a \in \mathcal{O}}, \varrho w_0) =: (\mathbb{R}^m, (\tau'_a)_{a \in \mathcal{O}}, w'_0). \tag{21}$$

In $\mathcal{A}(A_1, \ldots, A_m)$, equation (19) holds. To see this, let w'_t be a state vector obtained in a generation run of $\mathcal{A}(A_1, \ldots, A_m)$ at time t. Then conclude $w'_t = \varrho\varrho^{-1} w'_t = (\mathbf{1}\tau_{A_1}(\varrho^{-1} w'_t), \ldots, \mathbf{1}\tau_{A_m}(\varrho^{-1} w'_t)) = (P[A_1 \,|\, \varrho^{-1} w'_t], \ldots, P[A_m \,|\, \varrho^{-1} w'_t])$ (computed in $\mathcal{A}$) $= (P[A_1 \,|\, w'_t], \ldots, P[A_m \,|\, w'_t])$ (computed in $\mathcal{A}(A_1, \ldots, A_m)$).

The $m \times m$ matrix corresponding to ϱ can easily be obtained from the original OOM $\mathcal{A}$ by observing that

$$\varrho = (\mathbf{1}\tau_{A_i} e_j), \tag{22}$$

where e_i is the i-th unit vector.

The following fact lies at the heart of the learning algorithm presented in the next section:

Proposition 8 *In an interpretable OOM $\mathcal{A}(A_1, \ldots, A_m)$ it holds that*

1. $w_0 = (P[A_1], \ldots, P[A_m])$,

2. $\tau_{\bar{b}} w_0 = (P[\bar{b}A_1], \ldots, P[\bar{b}A_m])$.

The proof is trivial.

The state dynamics of interpretable OOM can be graphically represented in a standardized fashion, which allows to visually compare the dynamics of different processes. Any state vector w_t occurring in a generation run of an interpretable OOM is a probability vector. It lies in the non-negative hyperplane $H^{\geq 0} := \{(v^1, \ldots, v^m) \in \mathbb{R}^m \,|\, v^1 + \cdots + v^m = 1, v^i \geq 0 \text{ for } i = 1, \ldots, m\}$. Therefore, if one wishes to depict a state sequence $w_0, w_1, w_2 \ldots$, one only needs to render the bounded area $H^{\geq 0}$. Specifically, in the case $m = 3$, $H^{\geq 0}$ is the triangular surface shown in fig. 4(a). We can use it as the drawing plane, putting the point $(1/3, 1/3, 1/3)$ in the origin. For our orientation, we include the contours of $H^{\geq 0}$ into the graphical representation. This is an equilateral triangle whose edges have length $\sqrt{2}$. If $w = (w^1, w^2, w^3) \in H^{\geq 0}$ is a state vector, its components can be recovered from its position within this triangle, by exploiting $w^i = \sqrt{2/3} d_i$, where the d_i are the distances to the edges of the triangle. A similar graphical representation of states was first introduced in [Smallwood and Sondik, 1973] for HMMs.

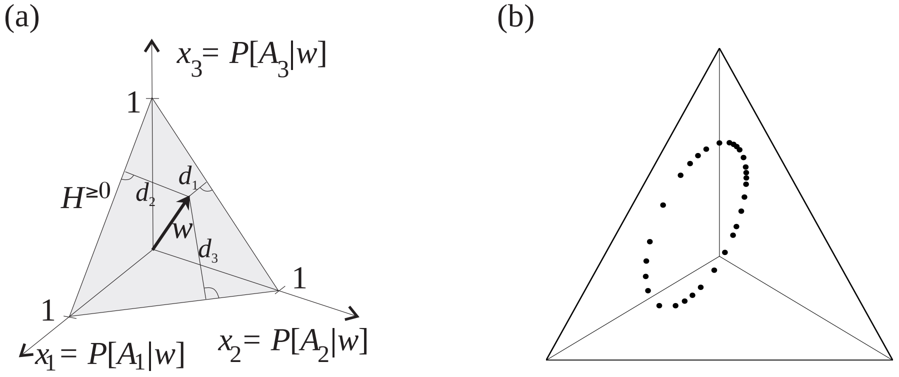

Figure 4: (a) The positioning of $H^{\geq 0}$ within state space. (b) State sequence of the probability clock, corresponding to fig. 3. For details compare text.

When one wishes to graphically represent states of higher-dimensional, interpretable OOMs (i.e. where $m > 3$), one can join some of the characteristic events, until three merged events are left. State vectors can then be plotted in a way similar to the one just outlined.

To see an instance of interpretable OOMs, consider the "probability clock" example from (15). With $k = 2$, the following partition (among others) of $\{a,b\}^2$ yields characteristic events: $A_1 := \{aa\}, A_2 := \{ab\}, A_3 := \{ba, bb\}$. Using (22), one can calculate the matrix ϱ, which we omit here, and compute the interpretable OOM $\mathcal{C}(A_1, A_2, A_3)$ using (21). These are the observable operators τ_a and τ_b thus obtained:

$$\tau_a = \begin{pmatrix} 0.645 & -0.395 & 0.125 \\ 0.355 & 0.395 & -0.125 \\ 0 & 1 & 0 \end{pmatrix}, \quad \tau_b = \begin{pmatrix} 0 & 0 & 0.218 \\ 0 & 0 & 0.329 \\ 0 & 0 & 0.452 \end{pmatrix}. \tag{23}$$

Fig. 4(b) shows a 30-step state sequence obtained by iterated applications of τ_a of this interpretable equivalent of the “probability clock”. This sequence corresponds to fig. 3.

9 Learning OOMs

This section describes a constructive algorithm for learning OOMs from data. The central idea behind this algorithm is proposition 8, which allows to estimate state vectors from data by simple frequency counts. From these estimated state vectors, to-be-learnt observable operators can be obtained through elementary linear algebra constructions.

There are two standard situations where one wants to learn a model of a stochastic system: (i) from a single long data sequence (or few of them) one wishes to learn a model of a stationary process, and (ii) from many short sequences one wants to induce a model of a non-stationary process. OOMs can model both stationary and nonstationary processes, depending on the initial state vector w_0 (cf. proposition 1). The learning algorithms presented here are applicable in both cases. For the sake of notational convenience, we will however only treat the stationary case. I.e., we assume that a (long) path $S = a_0 a_1 \cdots a_N$ of a stationary process (X_t) is given. The task is to construct an OOM which describes a process $(\tilde{X}_t)$ such that $(\tilde{X}_t) \approx (X_t)$ in a certain sense.

Subsection 9.1 treats the case where $\dim(X_t) = m$ is finite and known, i.e. (X_t) is generated by an (unknown) m-dimensional OOM $\mathcal{A}$. An almost surely consistent estimation procedure for $\mathcal{A}$ is presented. In subsection 9.2, this technique is refined, endowing the learning algorithm with further desirable properties, among them unbiasedness. In subsection 9.3, a heuristic for determining the appropriate model dimension is described. Subsection

9.4 sketches an online version of the learning algorithm for adaptive system identification.

9.1 The basic algorithm

This subsection has three parts. First, the basic learning algorithm is described, and it is explained mathematically why it works. Second, its usage is demonstrated with a simplistic toy example. Third, it is applied to the "probability clock" process introduced in sections 6 and 8.

We shall address the following learning task. Assume that a sequence $S = a_0 a_1 \cdots a_N$ is given, and that S is a path of an unknown stationary LDP (X_t). We assume that the dimension of (X_t) is known to be m (the question of how to assess m from S is discussed in Subsection 9.3). We select m characteristic events $A_1, \ldots, A_m$ of (X_t) (selection criteria are discussed later in this section). Let $\mathcal{A}(A_1, \ldots, A_m)$ be an OOM of (X_t) which is interpretable w.r.t. $A_1, \ldots, A_m$. The learning task, then, is to induce from S an OOM $\tilde{\mathcal{A}}$ which is an estimate of $\mathcal{A}(A_1, \ldots, A_m) = (\mathbb{R}^m, (\tau_a)_{a \in \mathcal{O}}, w_0)$. We require that the estimation be consistent almost surely, i.e. for almost every infinite path $S_\infty = a_0 a_1 \cdots$ of (X_t), the sequence $(\tilde{\mathcal{A}}_n)_{n \geq n_0}$ obtained from estimating OOMs from initial sequences $S_n = a_0 a_1 \cdots a_n$ of S_∞, converges to $\mathcal{A}(A_1, \ldots, A_m)$ (in some matrix norm).

An algorithm meeting these requirements shall now be described.

As a first step we estimate w_0. Prop. 8(*1*) states that $w_0 = (P[A_1], \ldots, P[A_m])$. Therefore, a natural estimate of w_0 is $\tilde{w}_0 = (\tilde{P}[A_1], \ldots, \tilde{P}[A_m])$, where $\tilde{P}[A_i]$ is the estimate for $P[A_i]$ obtained by counting frequencies of occurrence of A_i in S, as follows:

$$\tilde{P}[A_i] = \frac{\text{number of } \bar{a} \in A_i \text{ occurring in } S}{\text{number of } \bar{a} \text{ occurring in } S} = \frac{\text{number of } \bar{a} \in A_i \text{ occurring in } S}{N - k + 1}, \tag{24}$$

where k is the length of events A_i. In the second step, we estimate the operators τ_a. According to prop. 8(*2*), for any sequence $\bar{b}_j$ it holds that

$$\tau_a(\tau_{\bar{b}_j} w_0) \;\; = \;\; (P[\bar{b}_j a A_1], \ldots, P[\bar{b}_j a A_m]). \tag{25}$$

An m-dimensional linear operator is uniquely determined by the values it takes on m linearly independent vectors. This basic fact from linear algebra directly leads us to an estimation of τ_a, using (25). We estimate m linearly independent vectors $v_j := \tau_{\bar{b}_j} w_0$ by putting $\tilde{v}_j = (\tilde{P}[\bar{b}_j A_1], \ldots, \tilde{P}[\bar{b}_j A_m])$ $(j = 1, \ldots, m)$. For the estimation we use a similar counting procedure as in 24:

$$\tilde{P}[\bar{b}_j A_i] = \frac{\text{number of } \bar{b}\bar{a} \text{ (where } \bar{a} \in A_i\text{) occurring in } S}{N - l - k + 1}, \tag{26}$$

where l is the length of $\bar{b}_j$. Furthermore, we estimate the results $v'_j := \tau_a(\tau_{\bar{b}_j} w_0)$ of applying τ_a to v_i by $\tilde{v}'_j = (\tilde{P}[\bar{b}_j a A_1], \ldots, \tilde{P}[\bar{b}_j a A_m])$, where

$$\tilde{P}[\bar{b}_j a A_i] = \frac{\text{number of } \bar{b}a\bar{a} \text{ (where } \bar{a} \in A_i\text{) occurring in } S}{N - l - k}. \tag{27}$$

Thus we obtain estimates $(\tilde{v}_j, \tilde{v}'_j)$ of m argument-value pairs $(v_j, v'_j) = (v_j, \tau_a v_j)$ of applications of τ_a. From these estimated pairs, we can compute an estimate $\tilde{\tau}_a$ of τ_a through an elementary linear algebra construction: first collect the vectors $\tilde{v}_j$ as columns in a matrix $\tilde{V}$, and the vectors $\tilde{v}'_j$ as columns in a matrix $\tilde{W}_a$, then obtain $\tilde{\tau}_a = \tilde{W}_a \tilde{V}^{-1}$.

This basic idea can be augmented in two respects:

1. Instead of simple sequences $\bar{b}_j$, one can just as well take collective events B_j of some common lenght l to construct $\tilde{V} = (\tilde{P}[B_j A_i]), \tilde{W}_a = (\tilde{P}[B_j a A_i])$ (exercise). We will call B_j *indicative events.*

2. Instead of constructing $\tilde{V}, \tilde{W}_a$ as described above, one can also use raw count numbers, which saves the divisions on the rhs in (24),(26),(27). That is, use $V^\# = (\#_{\text{butlast}} B_j A_i), W_a^\# = (\# B_j a A_i)$, where $\#_{\text{butlast}} B_j A_i$ is the raw number of occurrences of $B_j A_i$ in $S_{\text{butlast}} := s_1 \ldots s_{N-1}$, and $\# B_j a A_i$ is the raw number of occurrences of $B_j a A_i$ in S. It is easy to see that this gives the same matrices $\tilde{\tau}_a$ as the original procedure.

Assembled in an orderly fashion, the entire procedure works as follows (assume that model dimension m, indicative events B_j, and characteristic events A_i have already been selected).

Step 1 Compute the $m \times m$ matrix $V^\# = (\#_{\text{butlast}} B_j A_i)$.

Step 2 Compute, for every $a \in \mathcal{O}$, the $m \times m$ matrix $W_a^\# = (\# B_j a A_i)$.

Step 3 Obtain $\tilde{\tau}_a = W_a^\# (V^\#)^{-1}$.

The computational demands of this procedure are modest compared to today's algorithms used in HMM parameter estimation. The counting for $V^{\#}$ and $W_a^{\#}$ can be done by a single sweep of an inspection window (of length $k+l+1$) over S. Multiplying or inverting $m \times m$ matrices essentially has a computational cost of $O(m^3/p)$ (this can be slightly improved, but the effects become noticeable only for very large m), where p is the degree of parallelization. The counting and inverting/multiplying operations together give a time complexity of this core procedure of $O(N + nm^3/p)$, where n is the size of $\mathcal{O}$.

For numerical stability and efficient exploitation of information contained in S, it is important to choose characteristic and indicator events appropriately. I cannot at the current state of the theory offer a method for "optimal" choice. However, some helpful rules of thumb are obvious:

1. The characteristic (and indicative, respectively) events should occur in the data with roughly equal frequencies. This minimizes the average relative error in the sums of entries in the rows (columns, respectively) of $\tilde{V}$ and $\tilde{W}_a$.

2. The matrix $\tilde{V}$ should be "as regular as possible" to make its inversion as insensitive as possible against error in the matrix entries. In terms of numerical linear algebra, this means that the ratio $\sigma_{min}/\sigma_{max}$ of the smallest vs. the greatest singular value σ_{min} of $\tilde{V}$ should be as big as possible (cf. [Golub and van Loan, 1996] for singular value decompositions).

3. The sequences $\bar{a}$ contained in A_i should have a high mutual correlation in the sense that if $\bar{a}_1, \bar{a}_2 \in A_i$, then the random variables $\chi_t^{\bar{a}_x}$ defined by $\chi_t^{\bar{a}_x} = 1$ if $(X_t, \dots, X_{t+k-1}) = \bar{a}_x$, else 0, are highly correlated ($x = 1, 2$). In plain words, this means that when $\bar{a}_1$ is likely to occur next, so is $\bar{a}_2$, and vice versa. Conversely, members of different characteristic events should have low mutual correlation. If the A_i have high inter-variation and low intra-variation in this sense, then the characteristic events are "correlational components" of the process's k-step future distribution. This suggests that PCA or other clustering techniques might be instrumental in finding optimal characteristic events.

4. Similarly, indicative events should have high internal and low external correlation in a related sense. Given characteristic events A_i, then

$\bar{b}_1, \bar{b}_2$ should be members of the same B_j if and only if the vectors $(\tilde{P}_S[A_1 \,|\, \bar{b}_x], \ldots, \tilde{P}_S[A_m \,|\, \bar{b}_x])$ (where $x = 1, 2$) have a small distance in the 2-norm.

5. Indicative events should exhaust $\mathcal{O}^l$ for some l, i.e. $B_1 \cup \ldots \cup B_m = \mathcal{O}^l$. This warrants that when we move the inspection window over S, at every position we get at least one count for $\tilde{V}$; thus we exploit S best.

At the beginning of this subsection we assumed that characteristic events A_i were given. We now see that the task of finding characteristic events in the first place coincides with optimizing the numerical/information-theoretic condition of $\tilde{V}$. It is easy to show (exercise) that if we choose any A_i, B_j such that V is regular, then the A_i are characteristic events. We therefore only have to solve the problem to find an "as regular as possible" counting matrix $\tilde{V}$.

The estimation of $\mathcal{A}(A_1, \ldots, A_m)$ by $\tilde{\mathcal{A}}$ is consistent almost surely in the sense outlined at the beginning of this subsection. This is because (i) the estimates $\tilde{V}$ and $\tilde{W}_a$ converge to the true matrices $V = (P[B_j A_i]), W_a = (P[B_j a A_i])$ almost surely with increasing N (e.g., in the 2-norm of matrices).

We shall now demonstrate the "mechanics" of the algorithm with an artificial toy example. Assume that the following path S of length 20 is given:

$$S = abbbaaaabaabbbabbbbb.$$

We estimate a two-dimensional OOM. We choose the simplest possible characteristic events $A_1 = \{a\}, A_2 = \{b\}$ and indicative events $B_1 = \{a\}, B_2 = \{b\}$.

First we estimate the invariant vector w_0, by putting

$$\tilde{w_0} = (\#a, \#b)/N = (8/20, 12/20).$$

Then we obtain $V^{\#}$ and $W_a^{\#}, W_b^{\#}$ by counting occurrences of subsequences in S:

$$\begin{aligned} V^{\#} &= \begin{pmatrix} \#_{\text{butlast}} aa & \#_{\text{butlast}} ba \\ \#_{\text{butlast}} ab & \#_{\text{butlast}} bb \end{pmatrix} = \begin{pmatrix} 4 & 3 \\ 4 & 7 \end{pmatrix}, \\ W_a^{\#} &= \begin{pmatrix} \#aaa & \#baa \\ \#aab & \#bab \end{pmatrix} = \begin{pmatrix} 2 & 2 \\ 2 & 1 \end{pmatrix}, \end{aligned}$$

$$W_b^\# = \begin{pmatrix} \#aba & \#bba \\ \#abb & \#bbb \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 3 & 5 \end{pmatrix}.$$

From these raw counting matrices we obtain estimates of the observable operators by

$$\begin{aligned} \tilde{\tau}_a &= W_a^\#(V^\#)^{-1} &= \begin{pmatrix} 3/8 & 1/8 \\ 5/8 & -1/8 \end{pmatrix}, \\ \tilde{\tau}_b &= W_b^\#(V^\#)^{-1} &= \begin{pmatrix} -1/16 & 5/16 \\ 1/16 & 11/16 \end{pmatrix}. \end{aligned}$$

That is, we have arrived at an estimate

$$\tilde{\mathcal{A}} = (\mathbb{R}^2, \begin{pmatrix} 3/8 & 1/8 \\ 5/8 & -1/8 \end{pmatrix}, \begin{pmatrix} -1/16 & 5/16 \\ 1/16 & 11/16 \end{pmatrix}, (9/20, 11/20)). \tag{28}$$

This concludes the presentation of the learning algorithm in its core version.

We now illustrate the learning procedure with the "probability clock" $\mathcal{C}$ introduced in sections 6 and 8. $\mathcal{C}$ was run to generate a path S of length $N = 3000$. $\mathcal{C}$ was started in an invariant state (cf. prop. 1), therefore S is stationary. We shall construct[3] from S an estimate $\tilde{\mathcal{C}}$ of $\mathcal{C}$.

Assume that we know that the process dimension is $m = 3$ (cf. subsection 9.3). First we have to find good characteristic and indicative events A_i, B_j, being guided by our "rules of thumb". Since we need three such events each, the smallest length of events compatible with $|\mathcal{O}| = 2$ is $k = l = 2$. That is, for A_1, A_2, A_3 we must partition $\mathcal{O}^2 = \{aa, ab, ba, bb\} =: \{\bar{a}_1, \bar{a}_2, \bar{a}_3, \bar{a}_4\}$ into three subsets. Similarly, we must distribute $\{\bar{a}_1, \bar{a}_2, \bar{a}_3, \bar{a}_4\}$ over B_1, B_2, B_3. We set out by computing the 4×4 *complete raw counting matrix* $V_{\text{complete}}^\# := (\#_S \bar{a}_j \bar{a}_i)$. In order to obtain A_1, A_2, A_3, we must join two of the four sequences aa, ab, ba, bb into a single event. We compute the correlation coefficients between the row vectors of $V_{\text{complete}}^\#$ and find that the third and fourth row have the highest pairwise correlation (of .96). Therefore, following the rule of thumb *3*, we merge ba with bb and put $A_1 = \{aa\}, A_2 = \{ab\}, A_3 =$

[3] The calculations were done using the Mathematica software package. Data and Mathematica programs can be fetched from the author's internet home page at www.gmd.de/People/Herbert.Jaeger/

$\{ba, bb\}$. These are the characteristic events used in the interpretable version $\mathcal{C}(A_1, A_2, A_3)$ in section 8. After merging the third and fourth row in $V^{\#}_{\text{complete}}$, a similar correlation analysis on the columns of the resulting 3×4 matrix reveals that ab and bb should be merged into a single indicative event, since the corresponding column vectors correlate with .99 (rule of thumb *4*). This gives us $B_1 = \{aa\}, B_2 = \{ab, bb\}, B_3 = \{ba\}$.

Using these characteristic and indicative events, $V^{\#}$ is computed (it can be obtained without further counting from $V^{\#}_{\text{complete}}$ by merging columns 2 & 4 and rows 3 & 4). $V^{\#}$ has row sums $486, 899, 1611$ and column sums $486, 1612, 898$, so rule of thumb *1* is not badly stretched. I emphasize that the rules of thumb need not mutually agree; however, rules *3* and *4* intuitively are the most important ones with respect to extracting information from S, so they were given preference here over the others.

The computation steps that remain after choosing A_i and B_j are merely mechanical and omitted here. We will briefly discuss the quality of the model estimate $\tilde{\mathcal{C}}_{3000} = (\mathbb{R}^3, (\tilde{\tau}_a, \tilde{\tau}_b), \tilde{w}_0)$ thus obtained.

The entries in the matrices $\tilde{\tau}_a, \tilde{\tau}_b$ deviate from the entries in the true matrices (23) by an average absolute error of .05 ($\approx 22\%$). The prediction errors made for $P[a \,|\, ba^t]$ by the estimated OOM $\tilde{\mathcal{C}}_{3000}$ are graphically represented in fig. 5(a). While the basic oscillatory character of the process has been captured, its frequency is underestimated, and furthermore, the estimated model exhibits a marked damping of the oscillation.

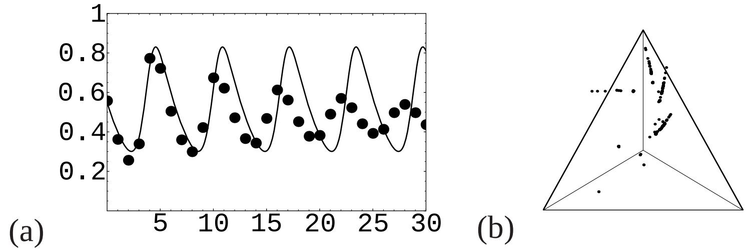

Figure 5: (a) Probabilities $P[a \,|\, a^t]$ according to the "probability clock" $\tilde{\mathcal{C}}_{3000}$ estimated from a 3000 time step sequence. (b) State diagram of a 1000-step run of $\tilde{\mathcal{C}}_{3000}$.

Fig. 5(b) shows a state plot of a 1000-step run of the estimated model. Some points lie outside the triangular area $H^{\geq 0}$, which implies that the model assigns negative "probabilities" to certain future events. Thus, $\tilde{\mathcal{C}}_{3000}$ is not a valid OOM, since the non-negativity condition 3 from definition 1 is violated.

We return to this disturbing fact after a look at another model $\tilde{\mathcal{C}}_{30000}$, which was estimated from a 30000-step sequence, using the same characteristic and indicative events as before.

The average absolute error of matrix entries $\tilde{\mathcal{C}}_{30000}$ was found to be .0038 ($\approx$ 1.7%). Fig. 6 illustrates the performance of $\tilde{\mathcal{C}}_{30000}$. The probabilities $P[a \mid ba^t]$ are captured almost accurately within the plotted time horizon of 30 steps (fig. 6(a)). The states of a 100000-step run of $\tilde{\mathcal{C}}_{30000}$ almost perfectly coincide with the most frequent states of the original OOM (23). The fact that even after 100000 steps the states of $\tilde{\mathcal{C}}_{30000}$ remain in the close vicinity of the true states indicates that $\tilde{\mathcal{C}}_{30000}$ does not violate condition 3 from definition 1, i.e., it is a valid OOM.

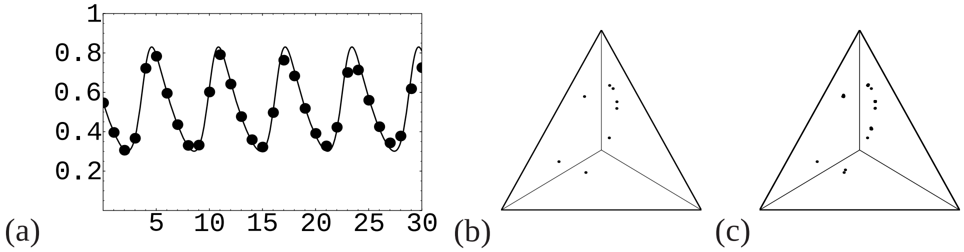

Figure 6: (a) same as fig. 5(a), now for a model estimated from a 30.000-step sequence. (b) The eight most frequent states of the original "probability clock". (c) An overlay of (b) with a 100000-step run of the model estimated from a 30000-step sequence. Every 100-th step of the 100000-step run was additionally plotted into (b) to obtain (c).

The reason that $\tilde{\mathcal{C}}_{3000}$ is not a valid OOM lies in the mathematical nature of the rotation map $1/2\tau_{1.0}$ of the original OOM. In terms of dynamical systems theory, a rotation $\mathbb{R}^3 \rightarrow \mathbb{R}^3$ is *structurally unstable* [Strogatz, 1994]. I.e., any mapping $\tilde{\tau}_{1.0}$ which deviates from $\tau_{1.0}$ by an arbitrarily small amount is not a rotation; it is either a "spiraling out" point repellor or a "spiraling in" point attractor. The error made in estimating $\tilde{\mathcal{C}}_{3000}$ happened to turn the rotation into a repellor (as can be guessed from fig. 5(b)); by pure coincidence, the error made with $\tilde{\mathcal{C}}_{30000}$ ended up with $\tilde{\tau}_a$ being of the attractor type.

Structural instability of operators is not the only reason why the estimation of OOMs from data might yield models which violate the non-negativity condition 3 from definition 1. Namely, if in an interpretable OOM, which is a true model of some process (X_t), states occur which lie very close to or even on the triangular boundary of $H^{\geq 0}$, slight errors made by estimating an OOM from a finite realization of (X_t) might push these states beyond that boundary. This situation cries for a method which would enable one to transform an estimated pseudo-OOM, which violates non-negativity, into a "nearest" valid OOM. Or, for that matter, it would be desirable to have a simple method for checking the non-negativity condition of a candidate OOM in the first place. Unfortunately I cannot offer either of these methods.

In practical applications, however, this does no harm. If the estimated model is used to predict the probability $P[\bar{a}]$ of some sequence $\bar{a}$ and returns a negative value $P[\bar{a}] = \epsilon < 0$, one can simply "correct" this prediction to $P[\bar{a}] = !\ 0$.

9.2 Refined algorithm: unbiased estimation of process parameters

A standard requirement for a statistical estimator of some parameters $\mathbf{p}$ is that it be *unbiased*, i.e., that the expected value of the estimated parameters are the true values: $E(\tilde{\mathbf{p}}) = \mathbf{p}$. In the case at hand, we have a stationary, m-dimensional process (X_t), which is described by an interpretable OOM $\mathcal{A}(A_1, \ldots, A_m) = (\mathbb{R}^m, (\tau_a)_{a \in \mathcal{O}}, w_0)$. This OOM is characterized by the parameters $\mathbf{p}$ in the matrices τ_a (in the stationary case, w_0 can be derived from these matrices, so the parameters in w_0 can be ignored). If we estimate $\mathcal{A}(A_1, \ldots, A_m)$ by $\tilde{\mathcal{A}}$, as described in the previous subsection, the estimated parameters $\tilde{\mathbf{p}}$ of the matrices $\tilde{\tau}_a$ give, as far as I can see, no unbiased estimates of the true parameters $\mathbf{p}$. The reason for this pessimism is that while $\tilde{V}$ and $\tilde{W}_a$ clearly are unbiased estimates of the matrices V and W_a, the inverting of $\tilde{V}$ in step 3 of the construction is a nonlinear operation, and unbiasedness is not generally preserved under nonlinear operations.

However, unbiasedness of the matrix parameter estimation is actually not what one would most desire. The major use of OOMs is to calculate probabilities $P[\bar{a}]$ via (6). It would be extremely nice if we knew that the probabilities $P_{\tilde{\mathcal{A}}}[\bar{a}]$, computed with estimated OOMs $\tilde{\mathcal{A}}$ using eq. (6), are unbiased estimates of the true probabilities $P[\bar{a}]$ in (X_t). I do not know whether this generally holds, but first steps toward such a theorem shall be done in this subsection. More specifically, we will see that if characteristic and indicative events are chosen properly in the estimation procedure, for certain *defining events* D it holds that $P[D] = E(P_{\tilde{\mathcal{A}}}[D])$. The defining events have the special property that their probabilities uniquely specify (X_t). Thus, the collection of probabilities $P[D]$ can be considered an parametrization $\mathbf{q}$ of (X_t) which is an alternative to the parametrization $\mathbf{p}$ afforded by the matrix parameters of the τ_a. For these parameters $\mathbf{q}$, the OOM construction procedure yields unbiased estimates.

We start by observing that if one constructs $\tilde{\mathcal{A}}$ using characteristic events $A_1, \ldots, A_m$ as described in the previous subsection, $\tilde{\mathcal{A}}$ is not generally an interpretable OOM itself. However, by a suitable choice of characteristic events $A_1, \ldots, A_m$, one can guarantee that $\tilde{\mathcal{A}} = \tilde{\mathcal{A}}(A_1, \ldots, A_m)$. For a convenient formulation of this fact, we use the shorthand $\bar{a}^{\to k} := \{\bar{a}\bar{b} \,|\, \bar{a}\bar{b} \in \mathcal{O}^k\}$ to denote complex events of length k which are characterized by a beginning sequence $\bar{a}$.

Proposition 9 *If for some $1 \leq n \leq m$, the characteristic events $A_1, \ldots, A_m$ can be grouped into n disjoint groups $A_1^1, \ldots, A_{l_1}^1; A_1^2, \ldots, A_{l_2}^2; \ldots; A_1^{n-1}, \ldots, A_{l_{n-1}}^{n-1}$; $A_1^n = A_m$ (where the last group consists only of A_m), such that the following conditions are satisfied:*

1. *each characteristic event from the first group is a union of complex events characterized by beginning singletons, i.e., $A_x^1 = \bigcup\{a_1^{\to k}, \ldots, a_r^{\to k}\}$ for some $a_1, \ldots, a_r \in \mathcal{O}$,*

2. *each characteristic event from the ν-th group $(1 < \nu < n)$ is of the form $A_x^\nu = \bigcup\{(b_1\bar{c})^{\to k}, \ldots, (b_s\bar{c})^{\to k} \,|\, \bar{c}^{\to k} \subseteq A_y^{\nu-1}\}$ for some $b_1, \ldots, b_s \in \mathcal{O}$ and for some $1 \leq y \leq l_{\nu-1}$,*

then an OOM $\tilde{\mathcal{A}}$ estimated from data, using these characteristic events, is interpretable w.r.t. $A_1, \ldots, A_m$, i.e., $\tilde{\mathcal{A}} = \tilde{\mathcal{A}}(A_1, \ldots, A_m)$.

The proof is given in the appendix. We proceed by considering the indicative events $B_1, \ldots, B_m$ used in constructing the matrices $\tilde{V}$ and $\tilde{W}_a$. Again, it does not generally hold that the probabilities $P_{\tilde{\mathcal{A}}}[B_j]$, obtained in the process characterized by $\tilde{\mathcal{A}}$, are equal to the estimates $\tilde{P}_S[B_j]$ of the same events, obtained from counting in the training data. The following proposition shows that by a suitable choice of indicative events, one can guarantee this and more:

Proposition 10 *Let $A_1, \ldots, A_m$ be characteristic events with the properties from proposition 9. Let $B_1, \ldots, B_m$ be indicative events which satisfy the following conditions:*

1. *$B_1 = \varepsilon$, the empty sequence.*
2. *For $1 < \nu \leq m$, B_ν has the form $B_\nu = C_1^\nu \ldots C_{x_\nu}^\nu := \{c_1 \ldots c_{x_\nu} \mid c_i \in C_i^\nu \subseteq \mathcal{O}$ for $i = 1, \ldots, x_\nu\}$. Furthermore, some $\nu' < \nu$ exists such that $B_{\nu'} = C_1^{\nu'} \ldots C_{x_{\nu'}}^{\nu'} = C_1^\nu \ldots C_{x_\nu - 1}^\nu$.*

Then for all complex events D of the form $B_j A_i, B_j a A_i$, where $a \in \mathcal{O}$, it holds that $P_{\tilde{\mathcal{A}}}[D] = \tilde{P}_S[D]$. In other words, the model $\tilde{\mathcal{A}}$ replicates in the model-computed probabilities of the events used in constructing the matrices $\tilde{V}, \tilde{W}_a$, the empirical relative frequencies of them.

Corollary: Let $B'_1, \ldots, B'_m$ be indicative events satisfying the above conditions. Construct from them new indicative events $B_1, \ldots, B_m$ by the operations of (i) joining disjoint sets, (ii) subtracting subsets (i.e., if X, Y are given and $X \subseteq Y$, obtain $Z = Y \setminus X$). Then the statement of the proposition is true also for these new $B_1, \ldots, B_m$.

The proof is given in the appendix.

The characteristic and indicative events used for estimating the "probability clock" $\mathcal{C}(A_1, A_2, A_3)$ in the previous subsection satisfy the conditions in propositions 9 and the corollary of 10 (put $A_1^1 = \{ba, bb\}, A_1^2 = \{ab\}, A_1^3 = \{aa\}, B'_1 = \varepsilon \equiv \{aa, ab, ba, bb\}(!), B'_2 = \{b\} \equiv \{ab, bb\}, B'_3 = ba, B_1 = \{aa\} = (B'_1 \setminus B'_2) \setminus B'_3, B_2 = B'_2, B_3 = B'_3$). Therefore, the probabilities of events $B_j A_i, B_j a A_i$ obtained in the estimated models $\tilde{\mathcal{C}}_{3000}, \tilde{\mathcal{C}}_{30000}$ are identical with the values obtained from counting in the 3000-step (30000-step, respectively) sequences.

Propositions 9 and 10 have many implications. One concerns the notion of degrees of freedom $\delta_{m,M}$ of an m-dimensional, M-symbol OOM. How many parameters are actually needed to characterize an OOM? The values $\tilde{P}_S[B_j A_i], \tilde{P}_S[B_j a A_i]$ (or equivalently, the values $\#_S B_j A_i, \#_S B_j a A_i$, or the values $P_{\tilde{\mathcal{C}}}[B_j A_i], P_{\tilde{\mathcal{C}}}[B_j a A_i]$, or the values $P[B_j A_i], P[B_j a A_i]$), are redundant. Some of them are implied by others. One can show (exercise) that the matrix $\tilde{V}$ can be constructed from the matrices $\tilde{W}_a$, if the characteristic and indicative events are chosen according to the previous two propositions. But even among the remaining Mm^2 parameters in the latter matrices, many are redundant. For instance, in the probability clock matrices we find $\tilde{W}_a(3,2) =$

$\tilde{P}_S[bab] = \tilde{P}_S[babaa] + \tilde{P}_S[babab] + \tilde{P}_S[babb] = \tilde{W}_b(1,3) + \tilde{W}_b(2,3) + \tilde{W}_b(3,3)$. I conjecture that an m-dimensional OOM (and thus, an m-dimensional process (X_t)) can be characterized by $\delta_{m,M} = (M-1)m^2$ parameters.

Another unanswered question is whether every finite-dimensional LDP possesses characteristic and indicative events which satisfy the conditions from the previous two propositions. I conjecture the answer is yes but have been unable to prove this.

The bottom line of all of this is that among the events B_jA_i, B_jaA_i we can often (maybe always), for a given process (X_t), select certain *defining* events $D_1, \ldots, D_{\delta_{m,M}}$ such that

1. (X_t) is uniquely determined by $P[D_1], \ldots, P[D_{\delta_{m,M}}]$.
2. If from realizations S of the process, OOMs $\tilde{\mathcal{A}}_S$ are estimated using the events D_i as characteristic/indicative events, these OOMs reproduce the empirical frequencies $\tilde{P}_S[D_i]$ of the events $D_1, \ldots, D_{\delta_{m,M}}$ in the respective training sequences, i.e. $P_{\tilde{\mathcal{A}}}[D_i] = \tilde{P}_S[D_i]$.
3. $$E(P_{\tilde{\mathcal{A}}}[D_i]) = P[D_i]. \tag{29}$$

In this sense, the OOM induction procedure affords an unbiased estimation of model-defining parameters, if characteristic and indicative events are chosen properly.

Note that (29) is valid even if the true dimension m' of the process is greater than the model dimension m. That is, OOM's constructed according to propositions 9 and 10 give unbiased estimates of at least $\delta_{m,M}$ of the $\delta_{m',M}$ parameters defining the true process.

9.3 Determination of model dimension

In the preceding subsections, it was assumed that the true process dimension m was known, and the task was to learn an m-dimensional model. These assumptions were the basis for an elementary statistical characterization of learning algorithm's properties.

But in real life the task usually is not to learn models with the true process dimension. Empirical processes that are generated by complex physical systems quite likely are very high-dimensional (even infinite-dimensional).

However, one can hope that only some few of the true dimensions are responsible for most of the stochastic phenomena that appear in the data, and that higher dimensions contribute with increasing residuality. Given finite data, then, the question is to determine m such that learning an m-dimensional model reveals m "significant" process dimensions, while any contributions of higher process dimensions are insignificant in the face of the estimation error due to finite data. Put bluntly, the task is to fix m such that data are neither overfitted nor underexploited.

We shall now describe a practical method for dealing with the notion of "significance". Assume first that characteristic and indicative events for an $m \times m$ raw counting matrix $V^\#$ are given. What, precisely, does it mean to say that $V^\#$ captures m "significant" process dimensions?

A canonical approach is to consider the singular value decomposition (SVD) $\sigma_{max} = \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m = \sigma_{min}$ of $V^\#$. I assume that the reader is familiar with the concept of SVD and its role in numerical linear algebra (cf. [Golub and van Loan, 1996]). If the smallest singular value σ_{min} is "significantly" larger than 0, then $V^\#$ has "numerical rank" m, and we are justified in claiming that an OOM estimated using $\tilde{V}$ captures m significant process dimensions.

Any rigorous attempt to decide whether $\sigma_{min} > 0$ by a "significant" amount must take into account the distribution of σ_{min} under the null hypothesis that $\sigma_{min} = 0$. This distribution depends on the distribution of the matrix entries, i.e. of the random variables $V^\#(ij) = \#_{\text{butlast}}[B_j A_i]$ defined by counting the occurrences of $B_j A_i$ in paths S_{butlast} of length $N - 1$. As a first approximation, one can assume that $\#_{\text{butlast}}[B_j A_i]$ is binomially distributed, with expected value $(N-1)P[B_j A_i]$ and variance $(N - 1)P[B_j A_i](1 - P[B_j A_i])$.

Unfortunately, it is an unsolved (and difficult to analyze) problem in numerical linear algebra to determine the distributions of singular values, given the distribution of matrix entries [Sabine Van Huffel, personal communication]. Thus, what would be the canonical way to answer our question is barred.

A practical escape from this deadlock is offered by heuristic methods used in numerical linear algebra to decide $\sigma_{min} > 0$ in the face of numerical (small) perturbations. Following [Golub and van Loan, 1996] (section 5.5.8) we accept the hypothesis $\sigma_{min} > 0$, if $\sigma_{min} > \epsilon \|V^\#\|_\infty$, where the ∞-norm of an $m \times m$ matrix (α_{ij}) is $\max_{1 \leq i \leq m} \sum_{1 \leq j \leq m} \mid \alpha_{ij} \mid$. The constant ϵ is fixed by convention: the greater ϵ is taken, the more likely it becomes to reject

$\sigma_{min} > 0$. Golub and van Loan advise that ϵ should approximately amount to the average relative error of the matrix entries.

While better alternatives are lacking, I suggest to adopt this strategy. The value for ϵ that I use is the average relative error of the entries of $V^{\#}$, which can be estimated from estimates of the variances by

$$\tilde{\epsilon} = 1/m^2 \sum_{i,j} \alpha_{i,j}, \tag{30}$$

where

$$\alpha_{i,j} = \begin{cases} \sqrt{\#[B_jA_i](1-\#[B_jA_i]/N)}/\#[B_jA_i], & \text{if } \#[B_jA_i] > 0 \\ 0 & \text{if } \#[B_jA_i] = 0 \end{cases} \tag{31}$$

If we find that the m-dimensional counting matrix $V^{\#}$ is regular, using this criterion, then the information in S warrants the construction of a model of dimension at least m. This is the core idea of determining model dimension.

Theoretically, then, one could find the appropriate model dimension by checking all possible counting matrices (i.e., all possible selections of characteristic and indicative events) of dimensions $m = 1, 2, 3, \ldots$ and stop when for some m_0 all m_0-dimensional counting matrices are found to be numerically singular. Then take $m = m_0 - 1$ as the appropriate dimension. Although this procedure terminates in finite time (due to the finiteness, there are only finitely many ways to construct non-empty characteristic and indicative events for each m), this is clearly impractical.

One has to find some way to keep in check the number of counting matrices considered for each m. A simple method is to consider not the counting matrices $V^{\#}$ but instead the complete counting matrices introduced in subsection 9.1, namely, $V^{\#}_{\text{complete}} = (\#\bar{a}_j\bar{a}_i)_{1\leq i\leq |\mathcal{O}|^k, 1\leq j\leq |\mathcal{O}|^l} =: V^{\#}(k,l)$. Many different counting matrices $V^{\#}$ can be made from such $V^{\#}(k,l)$ by merging columns and rows, but only the latter has to be tested for its numerical rank. This is because for the corresponding true $k \times l$ matrices $V(k,l) = (P[\bar{a}_j\bar{a}_i])$ it holds that

$$\dim(X_t) = m \Leftrightarrow \exists l_0, k_0 : \text{rank}(V(l_0,k_0)) = \text{rank}(V(l_0+1,k_0+1)). \tag{32}$$

This statement is proven in [Jaeger, 1997b]; it can also easily be obtained as a consequence of a closely related theorem from [Gilbert, 1959] (Lemma 2).

(32) allows us to estimate the appropriate process dimension in a way which circumvents the computation of matrices $V^{\#}$. Let $\sigma_1, \ldots, \sigma_p$ be the singular values of $V^{\#}(k,k)$, ordered by decreasing size. Compute ϵ for $V^{\#}(k,k)$ according to (30). Take as the numerical rank m of $V^{\#}(k,k)$ the minimal index m such that $\sigma_m > \epsilon \|V^{\#}(k,k)\|$. Repeat same procedure for $V^{\#}(k+1,k+1)$ to obtain numerical rank m'. If $m' = m$, the appropriate dimension has been found. If $m' > m$, continue by increasing k.

This procedure is still potentially impractical because the size of $V^{\#}(k,k)$ explodes as k increases. Further simplifications suggest themselves. The most important is that the procedure can be terminated without further increasing k once $m <| \mathcal{O} |^k$, since higher k will in most cases not lead to any further increase of m due to sharp increase in variance of matrix entries (relative to absolute size of them), which drives ϵ up relative to singular values.

A side remark: at the end of section 4 I remarked that the learning procedure can be used to construct a minimal-dimensional OOM from a given OOM $\mathcal{A}$. This is how: first, use (32) to determine the process dimension. Second, replay the learning algorithm with precise matrices V, W_a, which can be computed with $\mathcal{A}$.

As a demonstration we reconsider the estimation of the probability clock. The 4×4 matrix $V^{\#}(2,2)$, was obtained from sequences of different length generated by the probability clock. Table 1 shows the findings. In this table, the singular values corresponding to each sequence were re-scaled such that the greatest one was set to 100; the cutoff value $\epsilon \|V^{\#}(k,k)\|$ was proportionally rescaled. The outcome justifies a 3-dimensional model for the 30000-step sequence. But it turns out that for the 3000-step sequence, in subsection 9.1 we would have been better advised to construct only a 2-dimensional model, since the effect of the third process dimension is too weak to become significantly discernible in that short sequence.

9.4 Adaptive system identification

Assume that an ever-ongoing sequence $S_\infty = a_1, a_2, a_3, \ldots$ is generated by a source which drifts on a slow timescale. A common task is to maintain online a current model of the shifting process. This can be achieved with the OOM learning algorithm, as follows:

1. In preliminary investigations of similar time series (or beginnings of S_∞) choose appropriate characteristic and indicative events. Build $V^{\#}, W_a^{\#}$

N	SV	*cutoff*	m
300	(100 20 10 0.42)	29	1
1000	(100 12 10 1.5)	13	1
3000	(100 19 6.8 0.57)	8.6	2
10000	(100 15 7.2 1.5)	4.5	3
30000	(100 15 6.8 0.2)	2.5	3

Table 1: Determining appropriate model dimensions. N: length of sequence, SV: rescaled singular values $(\sigma_1, \ldots, \sigma_4)$, *cutoff*: $\epsilon \| V^{\#}(2,2) \|$, m: appropriate model dimension.

from a beginning of S_∞, and compute an initial OOM.

2. Continually update the counting matrices $V^{\#}, W_a^{\#}$ as new data come in, attributing more weight to more recent data.

3. If $V^{\#}, W_a^{\#}$ drift by more than a fixed tolerance threshold, recompute the model.

In step 1 one has to determine an appropriate model dimension m. This task is more intricate here than in the case of a finite, stationary sequence, since an appropriate choice of m depends on the tolerance threshold, and on the issue of whether the shift in the process concerns the "dominant" dimensions and/or the "less significant" ones. A rule of thumb that suggests itself is to choose m such that it is appropriate for subsequences of S_∞ obtained between the model updates.

Due to the pleasant computational properties of the recomputation 3, this scheme will in many cases allow to recompute models on a faster than the process drift timescale, thereby yielding adaptive system identification.

10 Input-output OOMs

In many applications one wishes to model stochastic systems which generate stochastic output while they are influenced by stochastic inputs. Input-output systems are most important in control theory, where it is investigated how a piece of machinery (*plant*) can be made to behave optimally in some sense by administering to it a suitable sequence of inputs (*controls*). Input-output systems appear in many other areas besides control engineering, too.

For instance, in some approaches in robotics that a robot models its environment as a stochastic input-output-system. The input in this case is constituted by the robot's actions, and the output is the sensory feedback the robot receives. Other instances of stochastic input-output systems are found in signal processing (stochastic transducers) or as modules in speech processing systems, to name but a few.

In this section it is shown how the OOM approach can be generalized to cover input-output systems. We treat the case of finite-valued input which is administered at the same temporal rate as output is generated. The terminology used in this section is taken largely from [Gihman and Skorohod, 1979], who first provided a comprehensive, general mathematical treatment of controlled stochastic processes. Since the perspective in that book is systems control, the terminology used here will inherit that flavor.

10.1 Formal description of controlled stochastic processes

Let $\mathcal{U} = \{r_1, \ldots, r_k\}$ be the set of possible inputs (or *controls*), and as before let $\mathcal{O} = \{a_1, \ldots, a_{k'}\}$ be the possible outputs.

In terms of stochastic processes, an input-output system (or *controlled object* in the terminology of Gihman & Skorohod) can be most conveniently characterized by a family of probability measures $(\mu_{\mathbf{r}})_{\mathbf{r} \in \mathcal{U}^\infty}$, which is indexed by the infinite control sequences. For every such control sequence $\mathbf{r}$, $\mu_{\mathbf{r}}$ is a probability measure on $(\mathrm{P}(\mathcal{O}))^{\mathbb{N}}$, where $\mathrm{P}(\mathcal{O})$ is the power set σ-algebra on $\mathcal{O}$. Each measure $\mu_{\mathbf{r}}$ is almost certainly defined by its projections $\mu_{\mathbf{r}}^n$ $(n \geq 0)$ on finite initial cylinders $\mathrm{P}(\mathcal{O})_0 \times \cdots \times \mathrm{P}(\mathcal{O})_n$. $\mu_{\mathbf{r}}^n[(a_0, \ldots, a_n)]$ gives the probability that an output sequence $a_0 \ldots a_n$ is produced by the controlled object if it is given the control sequence $r_0 \ldots r_n$, where $r_0 \ldots r_n$ is an initial sequence of $\mathbf{r}$. If $\mathbf{r}$ and $\mathbf{r}'$ coincide over the first $n+1$ time steps, it holds that $\mu_{\mathbf{r}}^n = \mu_{\mathbf{r}'}^n$. This implies that a controlled object can also be characterized by the family of finite-time-step initial probability distributions $(\mu_{\bar{r}})_{\bar{r} \in \mathcal{U}^+}$, where $\mu_{r_0 \ldots r_n}[(a_0, \ldots, a_n)]$ is again the probability that an output sequence $a_0 \ldots a_n$ is produced if the controlled object is given the control sequence $r_0 \ldots r_n$. We shall use this latter characterization[4].

[4] Gihman & Skorohod condition $n+1$ outputs $a_0, \ldots, a_n$ on n controls $r_0, \ldots, r_{n-1}$ up to the last but one time step, which suggests a perspective of feedback control. Conditioning output at time n on control input at time n, by contrast, suggests a direct control strategy. This is a matter of taste. All definitions and results carry over to either of the alternative

An input-output OOM (IO-OOM) of a controlled object must allow to compute the measures $\mu_{\bar{r}}$. The definition of an m-dimensional IO-OOM is straightforward and analogous to the definition of input-output hidden Markov models given in [Bengio, 1999]. The basic idea is that for every input r one defines a separate OOM with operators $(\tau_a^r)_{a \in \mathcal{O}}$ operating on $\mathbb{R}^m$. Then, if at time t the input r_t is given to the system, the operators from the collection $(\tau_a^{r_t})_{a \in \mathcal{O}}$ are used to generate the next output, and update the state vector w_{t-1} to w_t. This yields the following definition:

Definition 3 *An m-dimensional IO-OOM is a triple $(\mathbb{R}^m, ((\tau_a^r)_{a \in \mathcal{O}})_{r \in \mathcal{U}}, w_0)$, where $w_0 \in \mathbb{R}^m$ and all $\tau_a^r : \mathbb{R}^m \mapsto \mathbb{R}^m$ are linear operators, satisfying*

1. $\mathbf{1} w_0 = 1$,
2. *for every $r \in \mathcal{U}$, the matrix $\sum_{a \in \mathcal{O}} \tau_a^r$ has column sums equal to 1,*
3. *for all sequences $(r_0, a_0) \ldots (r_k, a_k)$ it holds that $\mathbf{1} \tau_{a_k}^{r_k} \cdots \tau_{a_0}^{r_0} w_0 \geq 0$.*

*For every $r \in \mathcal{U}$, the ordinary OOM $(\mathbb{R}^m, (\tau_a^r)_{a \in \mathcal{O}}, w_0)$ is called the r-*constituent *of the IO-OOM.*

An IO-OOM defines measures $\mu_{r_0 \ldots r_n}$ via

$$\mu_{r_0 \ldots r_n}[a_0 \ldots a_n] = \mathbf{1} \tau_{a_n}^{r_n} \cdots \tau_{a_0}^{r_0} w_0, \tag{33}$$

and thereby specifies a controlled object.

Control input is usually administered to a controlled object according to some *control strategy.* Following again Gihman & Skorohod, and basically repeating the considerations made above for the measures μ, we model a control strategy by a family $(\nu_{\bar{a}})_{\bar{a} \in \mathcal{O}^*}$, where $\nu_{a_0 \ldots a_{n-1}}$ is a probability measure on $\mathrm{P}(\mathcal{U})_0 \times \ldots \times \mathrm{P}(\mathcal{U})_n$ for nonempty sequences $a_0 \ldots a_{n-1}$, and ν_ε is a measure on $\mathrm{P}(\mathcal{U})$. $\nu_\varepsilon[r]$ gives the probability that r is selected as first control input; $\nu_{a_0 \ldots a_{n-1}}[r_0 \ldots r_n]$ is the probability that a control sequence $r_0 \ldots r_n$ is given if output $a_0 \ldots a_{n-1}$ occurs. Note that in this formalization, control input at time n depends on all prior controls and the output history up to the previous time step $n - 1$[5].

formalizations.

[5] Gihman & Skorohod have control at time n depending on output at time n.

It is natural to describe a controlled process incrementally, considering (i) the conditioned probabilities $P[a_n \,|\, r_0, \ldots, r_n, a_0, \ldots, a_{n-1}]$ of observing output a_n after inputs $r_0, \ldots, r_n$ and prior output history $a_0, \ldots, a_{n-1}$, and (ii) the conditioned probabilities $P[r_n \,|\, r_0, \ldots, r_{n-1}, a_0, \ldots, a_{n-1}]$ of giving input r_n after an input-output history $r_0, \ldots, r_{n-1}, a_0, \ldots, a_{n-1}$. These conditioned probabilities can be computed from the $\mu_{\bar{r}}$'s and $\nu_{\bar{a}}$'s by observing

$$\begin{aligned}
&P[a_n \,|\, r_0, \ldots, r_n, a_0, \ldots, a_{n-1}] = && (34)\\
&\quad= \frac{P[a_0 \ldots a_n \,|\, r_0 \ldots r_n]}{P[a_0 \ldots a_{n-1} \,|\, r_0 \ldots r_n]}\\
&\quad= \frac{P[a_0 \ldots a_n \,|\, r_0 \ldots r_n]}{P[a_0 \ldots a_{n-1} \,|\, r_0 \ldots r_{n-1}]}\\
&\quad= \mu_{r_0 \ldots r_n}[a_0 \ldots a_n]/\mu_{r_0 \ldots r_{n-1}}[a_0 \ldots a_{n-1}],\\
&P[r_n \,|\, r_0, \ldots, r_{n-1}, a_0, \ldots, a_{n-1}] = && (35)\\
&\quad= \frac{P[r_0 \ldots r_n \,|\, a_0 \ldots a_{n-1}]}{P[r_0 \ldots r_{n-1} \,|\, a_0 \ldots a_{n-1}]}\\
&\quad= \frac{P[r_0 \ldots r_n \,|\, a_0 \ldots a_{n-1}]}{P[r_0 \ldots r_{n-1} \,|\, a_0 \ldots a_{n-2}]}\\
&\quad= \nu_{a_0 \ldots a_{n-1}}[r_0 \ldots r_n]/\nu_{a_0 \ldots a_{n-2}}[r_0 \ldots r_{n-1}],
\end{aligned}$$

which are defined if $\mu_{r_0 \ldots r_{n-1}}[a_0 \ldots a_{n-1}] > 0$ and $\nu_{a_0 \ldots a_{n-2}}[r_0 \ldots r_{n-1}] > 0$. Note that $P[a_0 \mid r_0] = \mu_{r_0}[a_0]$, $P[r_0] = \nu_\varepsilon[r_0]$, and $P[r_1 \mid r_0, a_0] = \nu_{a_0}[r_0 r_1]/\nu_\varepsilon[r_0]$. Furthermore note that the $\mu_{\bar{r}}$'s and $\nu_{\bar{a}}$'s can be fully recovered from (34) and (35), which means that (34) is an alternative characterization of a controlled object and (35) of a control strategy.

A control strategy together with a controlled object yield a stochastic process $(V_t)_{t \geq 0}$ with values in $\mathcal{U} \times \mathcal{O}$. The finite distributions of this process are defined inductively via the incremental probabilities derived in (34) and (35):

$$\begin{aligned}
&P[V_0 = (r_0, a_0)] \;=\; \nu_\varepsilon[r_0]\mu_{r_0}[a_0]\\
& && (36)\\
&P[V_n = (r_n, a_n) \,|\, V_0 = (r_0, a_0), \ldots, V_{n-1} = (r_{n-1}, a_{n-1})] =\\
&\quad= P[V_0 = (r_0, a_0), \ldots, V_{n-1} = (r_{n-1}, a_{n-1})]\\
&\qquad \cdot P[r_n \,|\, r_0, \ldots, r_{n-1}, a_0, \ldots, a_{n-1}]
\end{aligned}$$

$$\cdot P[a_n \,|\, r_0, \ldots, r_n, a_0, \ldots, a_{n-1}].$$

The process (V_t) describes the combined input-output sequence, as it results from the application of some particular control strategy on a certain controlled object. This process is called a *controlled stochastic process* (CSP) by Gihman and Skorohod. Note that a controlled object and a control strategy yield a unique CSP. Conversely, a CSP specifies some or all of the incremental probabilities in (34) and (35). Specifically, it holds that

$$\begin{aligned} & P[a_n \,|\, r_0, \ldots, r_n, a_0, \ldots, a_{n-1}] = & (37) \\ & = \frac{P[V_0 = (r_0, a_0), \ldots, V_n = (r_n, a_n)]}{\sum_{b \in \mathcal{O}} P[V_0 = (r_0, a_0), \ldots, V_{n-1} = (r_{n-1}, a_{n-1}), V_n = (r_n, b)]}, \\ & P[r_n \,|\, r_0, \ldots, r_{n-1}, a_0, \ldots, a_{n-1}] = & (38) \\ & = \frac{\sum_{b \in \mathcal{O}} P[V_0 = (r_0, a_0), \ldots, V_{n-1} = (r_{n-1}, a_{n-1}), V_n = (r_n, b)]}{P[V_0 = (r_0, a_0), \ldots, V_{n-1} = (r_{n-1}, a_{n-1})]}. \end{aligned}$$

These incremental probabilities are only defined for nonzero denominators on the r.h.s. of (37) and (38). In particular, this implies that if the control strategy omits certain input sequences, the CSP does not contain information about the controlled object's response to that input. If one wishes to infer a complete model of the controlled object from such "defective" CSPs, further assumptions must be made. Specifically, we shall see that if one assumes that the controlled object is an IO-OOM, complete models can be inferred from "defective" CSPs.

A treatment of stationarity involves some subtleties which are not necessary in the case of ordinary stochastic processes and OOMs. First observe that one cannot speak of a stationary controlled object, or a stationary IO-OOM, since there is no stochastic process defined by a controlled object or an IO-OOM alone. One must treat stationarity as a property of CSP's. A straightforward approach would be to investigate CSP's which are stationary. However, given a controlled object, typically no stationary CSP exists even for time-invariant, finite-past-depending control strategies. Therefore, stationary CSP's are of little use.

This important fact shall be illustrated with a little example. The insights afforded by this example will later guide us in establishing a correct learning procedure.

Specify a controlled object by a 2-input, 2-output, 2-dimensional IO-OOM $\mathcal{A} = (\mathbb{R}^2, ((\tau_a^r)_{a \in \{1,2\}})_{r \in \{1,2\}}, (w_0^1, w_0^2))$. As a control strategy, select control 1 vs. 2 with equal probability at time 0, and flip control ever afterwards, i.e. select control 1 at time $t+1$ iff at time t control 2 was selected. This gives rise to a CSP which can be modeled by a 4-dimensional, 4-symbol OOM $\mathcal{B}_{\mathrm{CSP}} = (\mathbb{R}^4, (\tau_{(r,a)})_{(r,a)\in\{1,2\}\times\{1,2\}}, v_0)$, where the 4×4 matrices corresponding to the observable operators are given by

$$\begin{aligned}
\tau_{(1,a)} &= \begin{pmatrix} \mathbf{0} & \tau_a^1 \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \quad (a=1,2),\\
\tau_{(2,a)} &= \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \tau_a^2 & \mathbf{0} \end{pmatrix} \quad (a=1,2),
\end{aligned}$$

where $\mathbf{0}$ indicates the 2×2 null matrix.

Obviously, the starting vector of $\mathcal{B}_{\mathrm{CSP}}$ is $v_0 = 1/2(w_0^1, w_0^2, w_0^1, w_0^2)$. Typically, $\mathcal{B}_{\mathrm{CSP}}$ will not be stationary, i.e., v_0 will not be an invariant vector of $\sum_{(r,a)\in\{1,2\}\times\{1,2\}} \tau_{(r,a)}$. Conversely, consider the invariant state vector $u_0 = (u^1, u^2, u^3, u^4)$ of $\sum_{(r,a)\in\{1,2\}\times\{1,2\}} \tau_{(r,a)}$, which renders $\mathcal{B}_{\mathrm{stationary}} = (\mathbb{R}^4, (\tau_{(r,a)})_{(r,a)\in\{1,2\}\times\{1,2\}}, u_0)$ stationary. Typically, u_0 will not be of the form $1/2(x,y,x,y)$. This implies that typically no starting vector (x,y) exists which can be used with $\mathcal{A}$ instead of w_0, such that the resulting CSP would be stationary.

Nevertheless, the control strategy in this example has a distinctive "stationary" flavor, and the CSP modeled by $\mathcal{B}_{\mathrm{CSP}}$ quickly becomes stationary after an initial transient. Control strategies which are intuitively stationary, and ultimately stationary CSP's occur very commonly. This motivates the following definition:

Definition 4 *1. A stochastic process (X_t) with values in a measure space $(B, \mathfrak{B})$ is* asymptotically stationary *if for every finite sequence $A_0, \ldots, A_n$ of events from $\mathfrak{B}$ the limit*

$$\lim_{t\to\infty} P[X_t \in A_0, \ldots, X_{t+n} \in A_n]$$

exists.

2. A control strategy is called asymptotically stationary *if for all $r \in \mathcal{U}$, for all infinite sequences $r_0, r_{-1}, r_{-2}\ldots$ and $a_0, a_{-1}, a_{-2}\ldots$ of inputs and outputs the limit*

$$\lim_{t\to\infty} P[r \,|\, r_{-t}, r_{-t+1}, \ldots, r_0, a_{-t}, a_{-t+1}, \ldots, a_0]$$

exists, where $P[\cdot\,|\,\cdot]$ represents the control strategy according to (35).

3. First some notation: for a control sequence $\bar{r} = r_0 \dots r_n$ and an output sequence $\bar{a} = a_0 \dots a_n$ let $v_{\bar{a}}^{\bar{r}} := \tau_{a_n}^{r_n} \cdots \tau_{a_0}^{r_0} w_0 / \mathbf{1}\tau_{a_n}^{r_n} \cdots \tau_{a_0}^{r_0} w_0$ be the state vector that occurs in an IO-OOM after a history of inputs $\bar{r}$ and outputs $\bar{a}$.

An IO-OOM $(\mathbb{R}^m, ((\tau_a^r)_{a\in\mathcal{O}})_{r\in\mathcal{U}}, w_0)$ is called asymptotically detectable *w.r.t. a control strategy $(\nu_{\bar{a}})_{\bar{a}\in\mathcal{O}^*}$, if in the resulting CSP $(\Omega, \mathfrak{C}, P, (V_t)_{t\geq 0})$ there exists a set $C \in \mathfrak{C}$ of paths of measure $P[C] = 1$, such that for every path $\omega \in C$, where $V_t(\omega) =: (r_t^\omega, a_t^\omega)$, it holds that*

$$\begin{aligned}
\lim_{n\to\infty} \sup\{\|v_{\bar{a}}^{\bar{r}} - v_{\bar{a}'}^{\bar{r}'}\| \mid & \ \bar{r} = r_0^\omega \dots r_n^\omega, \ \ \bar{a} = a_0^\omega \dots a_n^\omega, \\
& \bar{r}' = r_0^{\omega'} \dots r_l^{\omega'}, \ \ \bar{a}' = a_0^{\omega'} \dots a_l^{\omega'}, \\
& \omega' \in C, l \geq n, \\
& r_{l-n}^{\omega'} r_{l-n+1}^{\omega'} \dots r_l^{\omega'} = \bar{r}, \\
& a_{l-n}^{\omega'} a_{l-n+1}^{\omega'} \dots a_l^{\omega'} = \bar{a}\} \\
= \ & 0.
\end{aligned}$$

In simple words, an IO-OOM is asymptotically detectable if finite-past knowledge about control and output history fixes the current state vector with arbitrary precision when increasingly long pasts are considered.

Asymptotic detectability is a key concept for a deeper understanding of IO-OOMs (and of OOMs, too). Preliminary investigations suggest that only certain "pathological" IO-OOMs are not asymptotically detectable[6]. The IO-OOM learning algorithm rests on asymptotic detectability. The three concepts introduced in the definition above are connected. It is not difficult to see that if an IO-OOM is asymptotically detectable and a control strategy is asymptotically stationary, the corresponding CSP is asymptotically stationary. Conversely, if a CSP is asymptotically stationary, the control strategy is asymptotically stationary.

10.2 Interpretable IO-OOMs

In this subsection we introduce interpretable IO-OOMs. They are largely analog to interpretable ordinary OOMs, and will be needed in the learning algorithm for IO-OOMs.

[6] These issues are currently being investigated by Arend Streit at GMD

Two IO-OOMs $\mathcal{A} = (\mathbb{R}^m, ((\tau_a^r)_{a\in\mathcal{O}})_{r\in\mathcal{U}}, w_0)$, $\mathcal{A}' = (\mathbb{R}^{m'}, ((\tau^{r\prime}_a)_{a\in\mathcal{O}})_{r\in\mathcal{U}}, w_0')$ are said to be *equivalent* if they specify the same controlled object. A full mathematical treatment of equivalence, which would correspond to the results obtained for ordinary OOMs, remains to be worked out. For the time being, we shall make do with two conditions which together are obviously sufficient for equivalence of $\mathcal{A}$ and $\mathcal{A}'$:

1. $m = m'$, and
2. there exists a linear isomorphism $\varrho : \mathbb{R}^m \to \mathbb{R}^m$, which preserves component sums of vectors, and for which it holds that

$$\begin{aligned} \varrho w_0 &= w_0' \\ \tau^{r\prime}_a &= \varrho \tau_a^r \varrho^{-1} \text{ for all } r, a. \end{aligned} \tag{39}$$

We will now investigate transformations ϱ which map $\mathcal{A}$ on an equivalent version which is interpretable in a certain sense. Let $\mathcal{O}^k = A_1 \dot\cup \cdots \dot\cup A_m$ be a collection of (output) events of length k, $\bar{r} = r_0 \ldots r_{k-1}$ an input sequence of length k, and $\tau_{A_i}^{\bar{r}} := \sum_{a_0\ldots a_{k-1}\in A_i} \tau_{a_{k-1}}^{r_{k-1}} \circ \cdots \circ \tau_{a_0}^{r_0}$. (The symbol $\dot\cup$ denotes disjoint union). Consider the mapping

$$\begin{aligned} \varrho_{\mathcal{A};\bar{r};A_1,\ldots,A_m} : \mathbb{R}^m &\rightarrow \mathbb{R}^m \\ v &\mapsto (\mathbf{1}\tau_{A_1}^{\bar{r}} v, \ldots, \mathbf{1}\tau_{A_m}^{\bar{r}} v). \end{aligned} \tag{40}$$

It is easy to see that $\varrho_{\mathcal{A};\bar{r};A_1,\ldots,A_m}$ is linear and preserves component sums of vectors. We assume that the output events $A_1, \ldots, A_m$ are selected such that $\varrho_{\mathcal{A};\bar{r};A_1,\ldots,A_m}$ is regular, in which case we call the events $A_1, \ldots, A_m$ *characteristic* w.r.t. $\bar{r}$. Then $\varrho_{\mathcal{A};\bar{r};A_1,\ldots,A_m}$ maps $\mathcal{A}$ on an equivalent IO-OOM $\mathcal{A}'$ according to (39).

States v' of $\mathcal{A}'$ are *interpretable* in the sense that the j-th component of v' is the probability that the output event A_j is observed during time steps $t, t+1, \ldots, t+k-1$, if $\mathcal{A}'$ is in state v' at time t and is given the input $\bar{r}$ during those time steps. We shall write $\mathcal{A}(\bar{r}; A_1, \ldots, A_m)$ to denote IO-OOMs which are interpretable in this sense, and call $(\bar{r}; A_1, \ldots, A_m)$ the *characterization frame* of $\mathcal{A}(\bar{r}; A_1, \ldots, A_m)$:

10.3 Example of an IO-OOM, a control strategy, and a CSP

In this subsection we describe a simple IO-OOM, specify a control strategy, and describe the resulting CSP as an ordinary OOM. In the next subsection we will see how the IO-OOM can be recovered (learnt) from the CSP[7].

Let $\mathcal{O} = \{1, 2\}$ and $\mathcal{U} = \{1, 2, 3\}$ be the outputs and controls for a controllable object, which is specified by the following 2-dimensional IO-OOM:

$$\mathcal{A} = (\mathbb{R}^2, (\tau_a^r)_{a \in \{1,2\}})_{r \in \{1,2,3\}}, (.5, .5)), \tag{41}$$

where

$$\begin{aligned} \tau_1^1 &= \begin{pmatrix} 0 & .5 \\ .5 & .5 \end{pmatrix} & \tau_2^1 &= \begin{pmatrix} 0 & 0 \\ .5 & 0 \end{pmatrix} \\ \tau_1^2 &= \begin{pmatrix} .1 & .5 \\ .1 & 0 \end{pmatrix} & \tau_2^2 &= \begin{pmatrix} .4 & .5 \\ .4 & 0 \end{pmatrix} \\ \tau_1^3 &= \begin{pmatrix} .25 & .2 \\ .25 & 0 \end{pmatrix} & \tau_2^3 &= \begin{pmatrix} .25 & .8 \\ .25 & 0 \end{pmatrix}. \end{aligned}$$

As a control strategy, we select control 1 with probability 2/3 and control 2 with probability 1/3 at $t = 0$. For $t > 0$, we either leave the control unchanged, i.e. $r_t = r_{t-1}$, or we increase it by 1 (mod 3), i.e. $r_t = r_{t-1}(\text{mod}3)+1$. We make the probability of increasing the control depend on the previous output, by putting $r_t = r_{t-1}$ with probability 1/3 if $a_{t-1} = 1$, and with probability 2/3 if $a_{t-1} = 2$. In simple words, the control cycles through $1 \dots 12 \dots 23 \dots 31 \dots$ where the probability of increasing the control is coupled to the prior output at a fixed rate. This is a simple control strategy with stochastic feedback.

The control strategy depends only on finite (depth 1) pasts. This implies that it is asymptotically stationary. Furthermore, it is easy to show that $\mathcal{A}$ is asymptotically detectable (exploit that after application of τ_2^1 the state vector is uniquely determined regardless of prior history, and that τ_2^1 occurs in almost all paths of the CSP). As a consequence, the CSP is asymptotically stationary.

[7] Mathematica files and notebooks containing all data, definitions and calculations pertaining to this example can be obtained from the author.

The CSP in this example actually is a 6-dimensional LDP. I will now describe a 6-dimensional OOM $\mathcal{B} = (\mathbb{R}^6, (\tau_{(r,a)})_{(r,a)\in\{1,2\}\times\{1,2,3\}}, v_0)$ which models the CSP. The idea to construct $\mathcal{B}$ is to partition the 6-dimensional state vectors into 3 segments of two components x, y each. The segments are indexed by the possible inputs. Thus, we write state vectors as $v = (x_1, y_1, x_2, y_2, x_3, y_3)$. Now we arrange matters such that if v_{t+1} is a state vector obtained after a history $(r_0, a_0), \ldots, (r_t, a_t)$, i.e.

$$v_{t+1} = \tau_{(r_t,a_t)} \cdots \tau_{(r_0,a_0)} v_0 / \mathbf{1}\tau_{(r_t,a_t)} \cdots \tau_{(r_0,a_0)} v_0,$$

then the following conditions are met:

1. The sum of components in the i-th segment indicates the probability that the next input is i $(i = 1, 2, 3)$. More precisely, we wish to achieve that $\mathbf{1}(\tau_{(i,1)} + \tau_{(i,2)})(x_1, y_1, x_2, y_2, x_3, y_3) = x_i + y_i$.
2. If $v_{t+1} = (x_1, y_1, x_2, y_2, x_3, y_3)$, and for some $i \in \{1, 2, 3\}$ it holds that $(x_i, y_i) \neq (0, 0)$, then $(x_i, y_i)/\mathbf{1}(x_i, y_i)$ is the state vector of the IO-OOM after the input-output history $r_0, \ldots, r_t;\ a_0, \ldots, a_t$.

Thus, state vectors v_t transparently code both the information necessary for determining the probabilities of the next control input, and the information required for determining the probabilities of the next output of the controlled object. The following 6×6 matrices satisfy our requirements:

$$\tau_{11} = \begin{pmatrix} 2/3\tau_1^1 & \mathbf{0} & \mathbf{0} \\ 1/3\tau_1^1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}, \qquad \tau_{12} = \begin{pmatrix} 1/3\tau_2^1 & \mathbf{0} & \mathbf{0} \\ 2/3\tau_2^1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}$$
$$\tau_{21} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 2/3\tau_1^2 & \mathbf{0} \\ \mathbf{0} & 1/3\tau_1^2 & \mathbf{0} \end{pmatrix}, \qquad \tau_{22} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 1/3\tau_2^2 & \mathbf{0} \\ \mathbf{0} & 2/3\tau_2^2 & \mathbf{0} \end{pmatrix},$$
$$\tau_{31} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & 1/3\tau_1^3 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 2/3\tau_1^3 \end{pmatrix}, \qquad \tau_{32} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & 2/3\tau_2^3 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 1/3\tau_2^3 \end{pmatrix}.$$

where again $\mathbf{0}$ denotes the 2×2 null matrix.

The reader may convince himself or herself that together with a starting vector $v_0 = (1/3, 1/3, 1/6, 1/6, 0, 0)$ this OOM correctly models the CSP which results from the controlled object $\mathcal{A}$ and the control strategy described above.

Now consider the invariant vector v_{stat} of component sum 1 of the sum map $\tau_{(1,1)} + \cdots + \tau_{(3,2)}$. Using this vector as a starting vector instead of v_0 yields a stationary version of $\mathcal{B}$, namely, $\mathcal{B}_{\text{stat}} = (\mathbb{R}^6, (\tau_{(r,a)})_{(r,a)\in\{1,2\}\times\{1,2,3\}}, v_{\text{stat}})$. Computing v_{stat} numerically yields

$$v_{\text{stat}} = (.18, .21, .14, .15, .21, .095).$$

It is easy to see that if the IO-OOM $\mathcal{A}$ is started with a starting vector $w_0 = (x, y)$, then the resulting CSP can be modeled by $\mathcal{B}$, with v_0 replaced by $(2/3x, 2/3y, 1/3x, 1/3y, 0, 0)$. Clearly v_{stat} is not of this form. This implies that there exists no starting vector w_0^* for $\mathcal{A}$ such that the resulting CSP becomes stationary.

To sum up, we have described a controlled object with a particular starting state and an asymptotically stationary control strategy, which give rise to an asymptotically stationary CSP. Furthermore we have a model of the stationary CSP. But, there exists no starting state of the controlled object which would yield a stationary CSP. As we shall see in the next subsection, this somewhat counterintuitive fact must be suitably acknowledged by the learning algorithm for IO-OOMs.

10.4 Learning algorithm for IO-OOMs

This section describes a learning algorithm for IO-OOMs in mathematical language.

We address the following learning task. Assume that an unknown IO-OOM $\mathcal{A}$ is controlled according to a fixed control strategy, and a (long) empirical trace $S = (r_0, a_0), \ldots, (r_N, a_N)$ from the resulting CSP is available. Then, estimate an IO-OOM $\tilde{\mathcal{A}}$ of the controlled object from S.

The core idea of the learning algorithm is simple. It proceeds in two major steps. First, an ordinary OOM $\tilde{\mathcal{B}}$ of the CSP is estimated. Second, $\tilde{\mathcal{B}}$ is used to estimate probabilities of the kind $P[A_i \,|\, \bar{r}; r_0, \ldots, r_n; a_0, \ldots, a_n]$, i.e. the probability that an output event A_i is generated after a prior history $r_0, \ldots, r_n; a_0, \ldots, a_n$ and a current control $\bar{r}$. These probabilities correspond to the components $\mathbf{1}\tau_{A_1}^{\bar{r}} v$ of state vectors v of an interpretable IO-OOM $\mathcal{A}(\bar{r}; A_1, \ldots, A_m)$ according to (40). $\tilde{\mathcal{A}}(\bar{r}; A_1, \ldots, A_m)$ can be constructed straightforwardly from the estimates of these probabilities.

The two steps are now described in detail. We assume that the dimension m of the original IO-OOM is known (the topic of determining m from S is deferred to the next subsection). Let $\bar{r} = r_0 \dots r_{k-1}$ be the control sequence of the characterization frame $(\bar{r}; A_1, \dots, A_m)$ to be used. To obtain the matrices $\tilde{\tau}^r_a(\bar{r}; A_1, \dots, A_m)$, proceed as follows.

1. Construct from S a stationary OOM $\tilde{\mathcal{B}} = (\mathbb{R}^l, (\tilde{\tau}_{(r,a)})_{(r,a) \in \mathcal{U} \times \mathcal{O}}, v_0)$ of the CSP, using the techniques described in previous sections.

2. From $\tilde{\mathcal{B}}$ construct the operators $\tilde{\tau}^r_a$ of $\tilde{\mathcal{A}}(\bar{r}; A_1, \dots, A_m)$ in the following substeps:

   (a) Obtain m state vectors $v_1, \dots, v_m$ of $\tilde{\mathcal{B}}$ by computing m initial input-output histories $\bar{r}_i = r^i_0 \dots r^i_{k_i}, \bar{a}_i = a^i_0 \dots a^i_{k_i}$, i.e. obtain $v_i := v^{\bar{r}_i}_{\bar{a}_i}$ ($i = 1, \dots, m$, where $v^{\bar{r}_i}_{\bar{a}_i}$ is defined as in definition 4(3)).

   (b) For $i = 1, .., m$ compute m-dimensional vectors

   $$w_i = \frac{1}{\mathbf{1}\tilde{\tau}_{(\bar{r},\mathcal{O}^k)} v_i} (\mathbf{1}\tilde{\tau}_{(\bar{r},A_1)} v_i, \dots, \mathbf{1}\tilde{\tau}_{(\bar{r},A_m)} v_i), \tag{42}$$

   where $\tilde{\tau}_{(\bar{r},A_j)} := \sum_{a_0 \dots a_{k-1} \in A_j} \tilde{\tau}_{(r_{k-1},a_{k-1})} \circ \dots \circ \tilde{\tau}_{(r_0,a_0)}$ and $\tilde{\tau}_{(\bar{r},\mathcal{O}^k)} := \sum_{a_0 \dots a_{k-1} \in \mathcal{O}^k} \tilde{\tau}_{(r_{k-1},a_{k-1})} \circ \dots \circ \tilde{\tau}_{(r_0,a_0)}$. w_i is the state vector obtained in $\tilde{\mathcal{A}}(\bar{r}; A_1, \dots, A_m)$ after an input-output history $\bar{r}_i, \bar{a}_i$, provided $\tilde{\mathcal{A}}(\bar{r}; A_1, \dots, A_m)$ is asymptotically detectable and $\bar{r}_i, \bar{a}_i$ is a sufficiently long history.

   (c) For $i = 1, .., m$, compute the probability

   $$P[a \,|\, r, \bar{r}_i, \bar{a}_i] = \mathbf{1}\tilde{\tau}_{(r,a)} v_i / \mathbf{1}\tilde{\tau}_{(r,\mathcal{O})} v_i, \tag{43}$$

   where $\tilde{\tau}_{(r,\mathcal{O})} := \sum_{a \in \mathcal{O}} \tilde{\tau}_{r,a}$. $P[a \mid r, \bar{r}_i, \bar{a}_i]$ is the probability that output a is produced, in the process described by $\tilde{\mathcal{B}}$, after a prior history of $\bar{r}_i, \bar{a}_i$ and a current control input r.

   (d) For $i = 1, .., m$, compute $v'_i = \tilde{\tau}_{(r,a)} v_i / \mathbf{1}\tilde{\tau}_{(r,a)} v_i$. From these, compute m-dimensional state vectors w'_i as in step 2b. w'_i is the state vector obtained in $\tilde{\mathcal{A}}(\bar{r}; A_1, \dots, A_m)$ after an input-output history $\bar{r}_i r, \bar{a}_i a$.

(e) Put $w''_i := P[a \mid r, \bar{r}_i, \bar{a}_i] w'_i$. It holds that $\tilde{\tau}^r_a w_i = w''_i$. Collect $w_1, \ldots, w_m$ as columns in a $m \times m$ matrix V, and $w''_1, \ldots, w''_m$ as columns in a $m \times m$ matrix W^r_a. Now obtain $\tilde{\tau}^r_a = W^r_a V^{-1}$.

This core version of the learning algorithm works only if the various denominators occurring in steps 2b,2c,2d are nonzero. A closer inspection reveals that this is warranted if and only if the control strategy assigns a nonzero probability to controls $\bar{r}$ and $r\bar{r}$ after prior histories $\bar{r}_i, \bar{a}_i$. Furthermore it is necessary that the vectors w''_i obtained in step 2e be linearly independent.

Often there exist no prior histories $\bar{r}_i, \bar{a}_i$ such that these requirements of nonzero probabilities and linear independence are satisfied for all (r, a). A more sophisticated version of the learning algorithm must then be used. The idea is not to utilize a single characterization frame $(\bar{r}; A_1, \ldots, A_m)$ uniformly for the construction of all $\tilde{\tau}^r_a$, but to construct each $\tilde{\tau}^r_a$ with respect to its own appropriate characterization frame $(\bar{r}(r,a); A_1(r,a), \ldots, A_m(r,a))$, in the same way as indicated in the core version of the learning procedure. This basically means that the $\tilde{\tau}^r_a$ are learnt for different but equivalent interpretable IO-OOMs $\tilde{\mathcal{A}}(\bar{r}(r,a); A_1(r,a), \ldots, A_m(r,a))$. In a novel, third step, transformations $\tilde{\varrho}$ between these equivalent IO-OOMs must be constructed, which allow to translate the differently interpretable $\tilde{\tau}^r_a$ into a common characterization frame, thereby arriving at a single IO-OOM.

Thus, assume that steps 1 and 2 have been carried out, using p different characterization frames $F^i = (\bar{r}^i; A^i_1, \ldots, A^i_m)$ $(i = 1, \ldots, p)$ for obtaining all $\tilde{\tau}^r_a =: \tilde{\tau}^r_a(\bar{r}(r,a); A_1(r,a), \ldots, A_m(r,a))$, where $(\bar{r}(r,a); A_1(r,a), \ldots, A_m(r,a)) = F^i$ for some i. I will now describe how in a third step, appropriate transformations between these characterization frames can be constructed from $\tilde{\mathcal{B}}$.

3. Order the characterization frames in a tree. Write $F^i \to F^j$ to denote that F^i is a direct predecessor of F^j. Let F^* be the terminal node in this tree, i.e. from every F^i there leads a path to F^*. Construct, for every pair $F^i \to F^j$, a transformation between the characterization frames F^i and F^j, i.e., a transformation $\tilde{\varrho}_{ij} : \tilde{\mathcal{A}}(\bar{r}^i; A^i_1, \ldots, A^i_m) \to \tilde{\mathcal{A}}(\bar{r}^j; A^j_1, \ldots, A^j_m)$, according to (39), as follows:

   (a) Select m state vectors $v_1, \ldots, v_m$ of $\tilde{\mathcal{B}}$, which occur after reasonably long initial histories like in step 2a, such that the following state vectors of $\tilde{\mathcal{A}}(F^i)$ and $\tilde{\mathcal{A}}(F^j)$, respectively,

$$\begin{aligned} w^i_s &= \frac{1}{\mathbf{1}\tilde{\tau}_{(\bar{r}^i,\mathcal{O}^k)}v_s}(\mathbf{1}\tilde{\tau}_{(\bar{r}^i,A^i_1)}v_s,\ldots,\mathbf{1}\tilde{\tau}_{(\bar{r}^i,A^i_m)}v_s), \\ w^j_s &= \frac{1}{\mathbf{1}\tilde{\tau}_{(\bar{r}^j,\mathcal{O}^k)}v_s}(\mathbf{1}\tilde{\tau}_{(\bar{r}^j,A^j_1)}v_s,\ldots,\mathbf{1}\tilde{\tau}_{(\bar{r}^j,A^j_m)}v_s), \quad (s = 1,\ldots,m) \end{aligned} \tag{44}$$

are well-defined (nonzero denominators are obtained), and such that $w^i_1,\ldots,w^i_m$ are linearly independent.

(b) Collect the vectors w^i_s as columns into a matrix U^i and the vectors w^j_s as columns into a matrix U^j and obtain $\tilde{\varrho}_{ij} = U^j(U^i)^{-1}$.

The maps $\tilde{\varrho}_{ij}$ can be used to transform all $\tilde{\tau}^r_a(\bar{r}(r,a); A_1(r,a),\ldots,A_m(r,a))$ obtained in steps 1 and 2, into operators of an IO-OOM with characterization frame F^*. More precisely, if $\tilde{\tau}^r_a(\bar{r}(r,a); A_1(r,a),\ldots,A_m(r,a)) = \tilde{\tau}^r_a(F^{i_0})$ and $F^{i_0} \to F^{i_1} \to \cdots \to F^{i_x} = F^*$, one obtains

$$\tilde{\tau}^r_a(F^*) = \tilde{\varrho}_{i_{x-1}i_x}\cdots\tilde{\varrho}_{i_0i_1}\tilde{\tau}^r_a(F^{i_0})\tilde{\varrho}^{-1}_{i_0i_1}\cdots\tilde{\varrho}^{-1}_{i_{x-1}i_x}. \tag{45}$$

The vectors v required in step 3a can often be the same as the vectors v used in step 2a, which streamlines the whole procedure. We shall adopt this strategy in the next subsection.

The tree in step 3 should be as flat as possible, in order to avoid long concatenations of transformations in (45), which accumulate error.

With some control strategies it is impossible to find any characterization frames F^i for which this algorithm works. In such cases there exist non-equivalent IO-OOMs which are consistent with the observed CSP. A formal characterizations of these cases remains to be found.

The learning algorithm generalizes from the particular control strategy that was used in generating S. The learnt IO-OOM can be used to predict system responses to control input which has zero probability in the original control strategy. This will be demonstrated in the next subsection.

If the original sequence S was indeed generated by an m-dimensional, asymptotically detectable IO-OOM $\mathcal{A}$, and if the resulting CSP is an l-dimensional LDP, then the learning algorithm is asymptotically consistent almost certainly in the following sense. For almost every infinite path S_∞ of the CSP, it holds that if an m-dimensional interpretable IO-OOM $\tilde{\mathcal{A}}_N(F)$ is learnt from initial sequences S_N of length N of S_∞, using K initial steps for generating the state vectors v_i in steps 2a and 3a, then $\lim_{N,K\to\infty}\tilde{\mathcal{A}}_N(F) =$

$\varrho(\mathcal{A}, F)\mathcal{A}$ in the sense of convergence of the observable operator matrices entries.

Another learning task of interest would occur when many short initial paths of the CSP are available, and we wish to identify an IO-OOM $\mathcal{A}$ including an initial state. The techniques described above can trivially be accomodated to this case. In the first step, learn a non-stationary OOM $\tilde{\mathcal{B}}$ with a starting vector v_0. The second and third step essentially remain unchanged. The vectors v required in steps 2a and 3a can be obtained from arbitrarily short initial input-output sequences, since the complications induced by asymptotic stationarity are no longer relevant. The starting vector v_0 directly corresponds to the starting vector w_0 of the IO-OOM. The latter can be obtained from the former by a calculation similar to the ones in steps 2a and 3a.

10.5 Determination of appropriate model dimension

For determining an appropriate dimension of the IO-OOM, I propose to use a method which is largely similar to the one proposed for ordinary OOMs in section 9.3. However, the situation is different now from what it was then. Formerly the determination of model dimension relied on raw counting matrices. Counting matrices are no longer available; they have essentially been replaced by the matrix V described in step 2e of the learning procedure. We will use the trick to transform a suitable version of V into a “pseudo” counting matrix first, and then apply the familiar procedure for determining an appropriate model dimension m.

Assume that step 1 of the learning procedure has been carried out, i.e. an appropriate l-dimensional model $\tilde{\mathcal{B}}$ of the CSP is available. It is easy to see that m cannot be greater than l. Therefore, it suffices to exploit pseudo counting matrices of dimension l. Proceed as follows:

1. Select l output events $\mathcal{O}^k = A_1 \dot{\cup} \cdots \dot{\cup} A_l$ and a control sequence $\bar{r} \in \mathcal{U}^k$, i.e., select a characterization frame F. Select l vectors $v_1, \ldots, v_l$ which (i) occur after a reasonably long initial history, and (ii) satisfy the condition $\mathbf{1}\tilde{\tau}_{(\bar{r}, \mathcal{O}^k)} v_i > 0$. Furthermore, the ratio of the biggest vs. the smallest singular value of the $l \times l$ matrix made from $v_1, \ldots, v_l$ should not be too big, i.e. the $v_1, \ldots, v_l$ should be as “numerically l.i.” as possible. For $i = 1, \ldots, l$ compute the vectors

$$w_i = \frac{1}{\mathbf{1}\tilde{\tau}_{(\bar{r},\mathcal{O}^k)} v_i} (\mathbf{1}\tilde{\tau}_{(\bar{r},A_1)} v_i, \ldots, \mathbf{1}\tilde{\tau}_{(\bar{r},A_l)} v_i), \tag{46}$$

as in step 2b of the learning procedure. Collect them as columns in a $l \times l$ matrix V. Normalize the sum of all entries of V to a value of 1 by putting $V' := 1/l\ V$.

2. Multiply V' with a scalar to arrive at a pseudo counting matrix by putting

$$V^{\#} := N(\mathbf{1}\tilde{\tau}_{(\bar{r},\mathcal{O}^k)} v_0) V', \tag{47}$$

where N is the length of the original training sequence S and v_0 is the starting vector of the stationary OOM $\tilde{\mathcal{B}}$. Thus, $\mathbf{1}\tilde{\tau}_{(\bar{r},\mathcal{O}^k)} v_0$ gives the relative frequency of the control sequence $\bar{r}$ within S. The rationale behind (47) is that the vectors in V contain information gained from only that fraction of S where control $\bar{r}$ occurs.

3. Use $V^{\#}$ to determine an approximate model dimension $m(F)$, as described in section 9.3.

4. Repeat steps 1–3 with different input sequences $\bar{r} \in \mathcal{U}^k$ and different events $A_1, \ldots, A_l$, i.e. different characterization frames F' (which however share the length k of characteristic events and control). Do this exhaustively (all possible frames) or by random probing. Take as appropriate model dimension the maximal number $m(F)$ found in this way.

This method admittedly has a heuristic flavor, but seems to work well in made-up examples where the correct model dimension is known. A more mathematically justified procedure would be welcome.

10.6 An example of utilizing the learning algorithm

We shall now illustrate the use of the learning algorithm by applying it to the example introduced in subsection 10.3.

The IO-OOM (41) was run with the control strategy described in subsection 10.3 to yield an input-output sequence S of length 10000. From S a 6-dimensional stationary OOM $\tilde{\mathcal{B}}$ was learnt, using singletons as characteristic

N	SV	*cutoff*	m
10000	(100 55 35 8.2 4.3 1.6)	4.9	4

Table 2: Determining appropriate model dimension for $\tilde{\mathcal{B}}$. Table is similar to table 1. For detail compare text.

and indicative events $(\{(1,1)\},\{(1,2)\},\{(2,1)\},\{(2,2)\},\{(3,1)\},\{(3,2)\})$. This selection of characteristic events ensured that $\tilde{\mathcal{B}}$ is itself interpretable, i.e. $\tilde{\mathcal{B}} = \tilde{\mathcal{B}}(\{(1,1)\},\{(1,2)\},\{(2,1)\},\{(2,2)\},\{(3,1)\},\{(3,2)\})$, by virtue of proposition 9.

The raw counting matrix $V^{\#}$ was subjected to a test for appropriate model dimension. The result is given in table 2.

It turned out that a 4-dimensional, maybe a 5-dimensional model would be supported by S, but not a 6-dimensional one. However, in this case the 6 singleton characteristic events were kept (and thus a 6-dimensional model obtained) since these singletons are extremely convenient in the further steps of the learning procedure, as will become apparent presently. Trading appropriateness for convenience is justified here since we are not actually interested in $\tilde{\mathcal{B}}$ itself. For the construction of the IO-OOM from $\tilde{\mathcal{B}}$ it is irrelevant whether $\tilde{\mathcal{B}}$ overfits S.

Next, the appropriate model dimension m was computed. To this end, 10 state vectors $v_1, \ldots, v_{10} \in \mathbb{R}^6$ were obtained by performing 10 runs of $\tilde{\mathcal{B}}$, starting from $\tilde{\mathcal{B}}$'s invariant starting vector, with v_i being the 50-th state vector obtained in the i-th run. An initial history of length 50 is long enough to ensure that the state vectors v_i of correspond to state vectors of the unknown IO-OOM (cf. the discussion on asymptotic stationarity and detectability in section 10.1).

Now ten trials for estimating m were performed, each consisting of the following steps:

1. A control sequence $r_1 r_2 r_3$ was randomly chosen which occurs in the CSP with nonzero probability. Six output events $A_1 \dot{\cup} \cdots \dot{\cup} A_6 = \{1,2\}^3$ were arbitrarily chosen. From $v_1, \ldots, v_{10}$, six vectors $v^1, \ldots, v^6$ were chosen such that $\mathbf{1}\tilde{\tau}_{(r_1 r_2 r_3, A_i)} v^j > 0$ for at least one $j \in \{1, \ldots, 6\}$.

2. For each v^j, a vector w^j was computed according to (46). These w^j were collected in a matrix V, which was transformed into a pseudo counting matrix $V^{\#}$ as described in the previous subsection.

#	SV				*cutoff*	m
1	(100	45	2.5	1.1)	19	2
2	(100	29	.52	.02)	6.2	2
3	(100	52	1.5	.08)	11	2
4	(100	33	7.4	.22)	26	2
5	(100	12	5.9	.44)	20	1
6	(100	9.5	5.4	1.2)	20	1
7	(100	28	2.2	.14)	23	2
8	(100	25	23	.80)	24	2
9	(100	19	7.1	2.4)	23	1
10	(100	19	5.7	1.9)	12	2

Table 3: Results from ten trials for determining the appropriate dimension for the IO-OOM. Only nonzero singular values are shown. The table is organized similar to table 9.3, with the exception of the first column, which gives the trial Nr here.

3. $V^{\#}$ was evaluated for an appropriate model dimension as described in section 9.3.

Table 10.6 lists the results obtained from the 10 trials. The majority of trials yielded $m = 2$ and a few $m = 1$. Note that the latter are no evidence that the appropriate model dimension should be 1, since low outcomes can be attributed to poorly distinguishing characteristic events, low probability of the control sequence used, or ill-chosen vectors v^i. Only the maximal values obtained are indicative. It thus turns out that a model dimension of $m = 2$ should be adopted.

The next task was to estimate the maps $\tilde{\tau}^r_a$. The simple version of the learning algorithm, where all $\tilde{\tau}^r_a$ are estimated w.r.t. a common characterization frame $F = (r_1 \ldots r_k; A_1, A_2)$, is inapplicable. This is because the control strategy excludes, after control i at time t, a control $i + 2$ (mod 3) at time $t + 1$. For input r with $r_1 = r + 2$ (mod 3), this renders the expression $\mathbf{1}\tilde{\tau}_{(\bar{r},\mathcal{O}^k)}\tilde{\tau}_{(r,a)}v'_i$ zero for $a = 1, 2$, which means a zero denominator in step 2d. Therefore, the more sophisticated version of the learning algorithm has to be used.

The maps $\tilde{\tau}^r_a$ were thus estimated w.r.t. different characterization frames. Concretely, for each map $\tilde{\tau}^r_a$ the frame $(r; \{1\}, \{2\})$ was used $(a = 1, 2)$. We describe in detail the procedure for estimating $\tilde{\tau}^2_1(2; \{1\}, \{2\})$.

Step 2a: Two state vectors v_1, v_2 of $\tilde{\mathcal{B}}$ were selected, which occurred after a 50-step initial sequence of $\tilde{\mathcal{B}}$, and after which an input of 2 had nonzero probability. Exploiting the fact that $\tilde{\mathcal{B}}$ is interpretable, the latter simply means that the third and fourth component of v_1, v_2 must not both be zero. Some further heuristic considerations for selecting "good" v_1, v_2 were that the probabilities of obtaining control 2 and output 1 at v_i, i.e. $\mathbf{1}\tilde{\tau}_{(2,1)} v_i$, should be high. Furthermore v_1 should be as different from v_2 as possible. For a concrete measure of "difference", the 1-norm of $\mathbb{R}^6$ was taken. The first of these heuristic requirements aims at finding such v_i where the future distribution after input 2 and output 1 is modeled relatively precisely by $\tilde{\mathcal{B}}$. The second requirement aims at taking probes from as different as possible states of the CSP, which again increases precision. From the ten vectors used in the model dimension determination described before, the two were selected which best satisfied these requirements. Concretely,

$$\begin{aligned} v_1 &= (0.345, 0, 0.306, 0.348, 0, 0), \\ v_2 &= (0, 0, 0.140, 0.240, 0.187, 0.433) \end{aligned}$$

were selected.

Step 2b: $w_1 = (\mathbf{1}\tilde{\tau}_{(2,1)} v_1, \mathbf{1}\tilde{\tau}_{(2,2)} v_1) / \mathbf{1}\tilde{\tau}_{(2,\{1,2\})} v_1$ was computed. Note that due to the interpretability of v_1, w_1 is simply the 2-vector obtained from the middle two entries of v_1, and dividing them by the sum of these two entries, i.e., $w_1 = (v_1^3, v_1^4)/(v_1^3 + v_1^4)$. w_2 is computed accordingly. One obtains

$$\begin{aligned} w_1 &= (0.467, 0.533), \\ w_2 &= (0.369, 0.6319). \end{aligned}$$

Step 2c: For v_1, the probability $P[1 \mid \ldots] = \mathbf{1}\tilde{\tau}_{(2,1)} v_1 / \mathbf{1}\tilde{\tau}_{(2,\{1,2\})} v_1$ (cf. (43)) was computed. Again, the fact that v_1 is interpretable can be exploited to obtain this probability as $v_1^3/(v_1^3 + v_1^4)$. $P[2 \mid \ldots]$ was treated analogously, obtaining

$$\begin{aligned} P[1 \mid \ldots] &= 0.467, \\ P[2 \mid \ldots] &= 0.369. \end{aligned}$$

Step 2d: w_1' and w_2' were computed from the vectors $\tilde{\tau}_{(2,1)} v_1 / \mathbf{1}\tilde{\tau}_{(2,1)} v_1$ and $\tilde{\tau}_{(2,1)} v_2 / \mathbf{1}\tilde{\tau}_{(2,1)} v_2$ like w_1 and w_2 from v_1 and v_2.

Step 2e: w_1'' and w_2'' were then obtained as

$$\begin{aligned} w_1'' = P[1 \mid \ldots]w_1' &= (0.0907, 0.376), \\ w_2'' = P[2 \mid \ldots]w_2' &= (0.0878, 0.281). \end{aligned}$$

w_1, w_2 were put as columns in a matrix V, and w_1'', w_2'' as columns in a matrix W. Then the desired estimate was obtained by

$$\tilde{\tau}_1^2(2; \{1\}, \{2\}) = WV^{-1} = \begin{pmatrix} 0.106 & 0.0774 \\ 0.895 & -0.0774 \end{pmatrix}.$$

The observable operator $\tau_1^2(2; \{1\}, \{2\})$ of the original IO-OOM $\mathcal{A}$, interpreted w.r.t. the frame $(2; \{1\}, \{2\})$, is

$$\tau_1^2(2; \{1\}, \{2\}) = WV^{-1} = \begin{pmatrix} 0.15 & 0.05 \\ 0.85 & -0.05 \end{pmatrix}.$$

In a similar way, all $\tilde{\tau}_j^i(i; \{1\}, \{2\})$ $(i = 1, 2, 3, j = 1, 2)$ were constructed.

Thus, three characterization frames were used, namely, $F^i = (i; \{1\}, \{2\})$ $(i = 1, 2, 3)$. They were ordered by putting $F^2 \to F^1, F^3 \to F^1$. Now two transformations $\tilde{\varrho}_{21}, \tilde{\varrho}_{31}$ had to be constructed for unifying the characterization frames into a common frame F^1, according to steps 3a and 3b.

Step 3a: $\tilde{\varrho}_{21}$ was constructed from two of the ten state vectors v_i originally used for determining the model dimension. These two vectors v had to be selected such that the probabilities $\mathbf{1}\tilde{\tau}_{(1,\{1,2\})}v$ and $\mathbf{1}\tilde{\tau}_{(2,\{1,2\})}v$ were nonzero. Considering again the interpretability of $\tilde{\mathcal{B}}$, this simply means that among the first two and among the second two entries of v there must be nonzeros. The following vectors were chosen:

$$\begin{aligned} v_1 &= (0.50, 0.16, 0.13, 0.22, 0, 0), \\ v_2 &= (0.35, 0, 0.31, 0.39, 0, 0). \end{aligned}$$

The vectors w_s^1, w_s^2 $(s = 1, 2)$ could have been constructed by direct computation according to (44). However, interpretability of $\tilde{\mathcal{B}}$ can again be exploited, obtaining $w_1^1 = (v_1^1, v_1^2)/(v_1^1 + v_1^2), w_2^1 = (v_2^1, v_2^2)/(v_2^1 + v_2^2), w_1^2 = (v_1^3, v_1^4)/(v_1^3 + v_1^4), w_2^2 = (v_2^3, v_2^4)/(v_2^3 + v_2^4)$.

Step 3b: Putting w_1^1, w_2^1 as columns in a matrix U^1 and w_1^2, w_2^2 in U^2, the desired transformation was estimated as

$$\tilde{\varrho}_{21} = U^1(U^2)^{-1} = \begin{pmatrix} 2.281 & -0.123 \\ -1.28 & 1.12 \end{pmatrix}.$$

For comparison: the correct transformation mapping would have been

$$\varrho_{21} = \begin{pmatrix} 1.833 & 0.166 \\ -0.833 & 0.833 \end{pmatrix}.$$

Finally, $\tilde{\varrho}_{21}$ was applied to $\tilde{\tau}_1^2(2;\{1\},\{2\})$ to give $\tilde{\tau}_1^2(1;\{1\},\{2\})$.

Similar transformations yielded uniform F^1 versions of all operator estimates. They compare to the F^1 versions of the original operators as follows (let $\delta_j^i := \tau_j^i(F^1) - \tilde{\tau}_j^i(F^1)$):

$$\delta_1^1 = \begin{pmatrix} -0.007 & 0.03 \\ 0.007 & -0.03 \end{pmatrix}, \quad \delta_2^1 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix},$$
$$\delta_1^2 = \begin{pmatrix} 0.09 & -0.1 \\ -0.06 & -0.02 \end{pmatrix}, \quad \delta_2^2 = \begin{pmatrix} 0.02 & 0.1 \\ -0.06 & 0.02 \end{pmatrix},$$
$$\delta_1^3 = \begin{pmatrix} 0.08 & -0.1 \\ -0.1 & 0.1 \end{pmatrix}, \quad \delta_2^3 = \begin{pmatrix} 0.03 & -0.09 \\ 0.01 & 0.08 \end{pmatrix}.$$

This amounts to an average absolute error in matrix entry estimates of about 0.052.

$\tilde{\mathcal{B}}$ was interpretable w.r.t. $(\{(1,1)\},\{(1,2)\},\{(2,1)\},\{(2,2)\},\{(3,1)\},\{(3,2)\})$ in this example. This fact could be nicely exploited in order to avoid costly matrix multiplications. There are many ways to make the characteristic events of $\tilde{\mathcal{B}}$ agree with the characterization frames used in constructing $\tilde{\mathcal{A}}$, such that matrix multiplications can be avoided in a similar way. A systematic treatment of this speed-up remains to be done.

Note that the induced model $\tilde{\mathcal{A}}$ allows to compute the response probabilities of the controlled object for any possible control sequence. By contrast, the original control strategy did not allow input 3 after 1, 2 after 3, or 1 after 2. In this sense, the learning algorithm generalizes from training data.

10.7 Discussion of learning IO-OOM procedure

The reader will have noticed that the learning algorithm described in this section involves some heuristics. Specifically, this concerns the finding of "good" state vectors v_i of the OOM $\tilde{\mathcal{B}}$ of the CSP. I could give only some hints when I described the example. A mathematically founded strategy for achieving optimal estimates $\tilde{\tau}_a^r$ is lacking.

Little is known about the length K of initial sequences of $\tilde{\mathcal{B}}$ which have to be run to generate state vectors v_i which correspond with sufficient precision to state vectors w_i of the IO-OOM. In the example, $K = 50$ was used. However, when the correct OOM $\mathcal{B}$ of the CSP, as described in subsection 10.3, was used instead of the estimated model $\tilde{\mathcal{B}}$ as a basis for steps 2 and 3 of the learning procedure, it was found that $K = 1$ already gave perfectly correct "estimates"! In other words, in this example the starting vector v_0 of the stationary CSP model $\mathcal{B}$, which does *not* correspond to states of the IO-OOM in the sense of (42), turns into states which *do* correspond to IO-OOM states immediately after only one time step. Whether this is related to the fact that the control strategy in this example also depends on only one prior time step remains to be investigated.

Throughout this section, the notation $\tilde{\mathcal{A}}(F)$ was used to denote an estimated IO-OOM which was constructed using the characterization frame F. Actually, using F during the construction does not imply that the estimated IO-OOM finally obtained is, in fact, interpretable w.r.t. F. A correlate of proposition 9, which would provide us the means to ensure interpretability of the estimated IO-OOM, remains to be found.

The computational cost of the learning algorithm comes from (i) estimating $\tilde{\mathcal{B}}$ (step 1) and (ii) constructing $\tilde{\mathcal{A}}$ from $\tilde{\mathcal{B}}$. Putting $M :=|\mathcal{U}|\,|\mathcal{O}|$, the contribution of (i) is, as we already know, $O(N+Ml^3)$. A closer inspection of the learning algorithm reveals that the computational load of (ii) is bounded by $O(KMml^2)$, where K is the number of steps taken for initial histories to generate state vectors v_i of $\tilde{\mathcal{B}}$. The example suggests that it might be possible to keep K bounded under some constant. Since $m \leq l$, this would lead to a combined computational complexity ((i) and (ii)) of $O(N+Ml^3)$. Of course, this measure omits costs of heuristics used for finding "good" vectors v_i.

In many realistic cases, one has complete knowledge of the control strategy. In such cases, the learning method described here is certainly inappropriate, since such knowledge is not exploited. Furthermore, in some applications one can even change the control strategy in order to gain specific information about the controlled object. The methods described here do not allow to take advantage of this possibility.

All in all, the results reported in this section mark only a beginning in the art of estimating IO-OOMs, which appears to be a much wider field than estimating ordinary OOMs.

10.8 IO-OOMs and PSRs

This subsection relates IO-OOMs to the predictive state representations (PSRs) of controlled stochastic systems that were recently introduced by Littman, Sutton, Singh et al [Littman *et al.*, 2001] [Singh *et al.*, 2003] . The subsection is written in July 2003 and inserted into the original techreport text from 1998, and addresses readers who already have read about PSRs. PSRs have been partially inspired by the OOM article in Neural Computation [Jaeger, 2000], but the authors were not aware of the treatment of IO-OOMs in this techreport and thus independently developed basically similar mathematical objects introducing, however, a different terminology and working out some interestingly different details. Here I provide "translation aids" to mutually relate the underlying concepts.

A note on history. I abandoned the subject of IO-OOMs after 1998, concentrating on ramifications of standard OOMs (continuous-valued OOMs [Jaeger, 1998a], general OOM theory of stochastic processes [Jaeger, 1999b], OOMs for optimal action selection [Jaeger, 1999a]) and, most importantly, on the subtelties of learning OOMs from small data sets under the auspices of statistical learning theory. Thanks mostly to the work of Klaus Kretzschmar and Tobias Oberstein, the elementary learning algorithm is by now refined to a degree that an automated learning procedure could be implemented which extracts the information contained in (possibly poor) datasets in a statistically and computationally very efficient way. The main ingredients are (i) preprocessing the training sequence into a novel form of suffix tree representation (called context trees), (ii) translating the statistical learning problem into a problem of numerical linear algebra, namely, optimizing the condition of a matrix related to the V matrix of the early learing procedure, (iii) invoking a nonlinear optimizer to solve this optimization task. The refined learning procedure outperforms by far the basic OOM learning method reported in earlier sections of this techreport, and even more starkly surpasses EM-based learning methods for HMMs. These results are summed up in [Kretzschmar, 2003] [ingredients (ii) and (iii)] and [Oberstein, 2002] [for (i)]. The further refinement of the implementation, which has reached a size of about 10,000 lines of code, is now pursued as an open source development project at *omk.sourceforge.net*. The project of developing an optimal learning method for standard OOMs is essentially completed.

PSRr and *interpretable* IO-OOMs are very close relatives. The connection is established through the PSR-notion of *tests* and the IO-OOM notion of *characterization frames*. Both concepts describe the current state of a controlled object in terms of the probabilities that certain, pre-defined future events will happen. There is a finite number of such events. In the following I use the PSR terminology introduced in [Littman *et al.*, 2001] when I speak about PSRs.

Tests and predictive states. The *tests* of a PSR are q input-output sequences

$$t_i = a^i_1, \ldots, a^i_{l_i},\, o^i_1, \ldots, o^i_{l_i}, \tag{48}$$

where $i = 1, \ldots, q$, the o^i_j are output observations from a finite set $\mathcal{O}$, and the the a^i_j are control actions (inputs to the controlled object) from a finite set $\mathcal{A}$. A *prediction vector* is a $1 \times q$-dimensional vector $p(h) = (P(t_1 \mid h), \ldots, P(t_q \mid h))$ containing the conditional probabilities $P(t_1 \mid h) = P(o^i_1, \ldots, o^i_{l_i} \mid h, a^i_1, \ldots, a^i_{l_i})$ that after an initial input-output history $h = o_1, \ldots, o_t, a_1, \ldots, a_t$ the system produces outputs $o^i_1, \ldots, o^i_{l_i}$ when input $a^i_1, \ldots, a^i_{l_i}$ is given. The probabilities collected in $p(h)$ must form a sufficient statistics for the prediction of the controlled object, that is, for every finite input-output sequence $t = a'_1, \ldots, a'_n,\, o'_1, \ldots, o'_n$, the probability $P(t \mid h) = P(o'_1, \ldots, o'_n \mid h, a'_1, \ldots, a'_n)$ must be computable by a *projection function* f_t which does not depend on h, that is,

$$P(t \mid h) = f_t(p(h)). \tag{49}$$

A prediction vector $p(h)$ with this sufficient statistics property is called a *predictive state representation* (PSR), and the tests $t_1, \ldots, t_q$ that are used in making it up are the *core tests* of a PSR. .

In a *linear* PSR, applying the projection function f_t of a test t to a prediction vector $p(h)$ becomes multiplication with a *projection vector* m_t: $f_t(p(h)) = p(h)^\mathsf{T} m_t$. The PSR approach is in principle open to both linear and nonlinear PSTs. The first papers are all concerned with linear models.

Characterization frames. As introduced earlier in this section, a characterization frame $(\bar{r}; A_1, \ldots, A_m)$ for in interpretable IO-OOM is a sequence $\bar{r} = r_0, \ldots, r_{k-1}$ of inputs together with m characteristic events A_i, which form a partition of $\mathcal{O}^k$ subject to a certain algebraic condition of linear independence, namely, that the transformation matrix $\varrho_{\mathcal{A};\bar{r};A_1,\ldots,A_m}$ (cf. Subsection 10.2) is regular, i.e., invertible. It is easy to see that this regularity condition holds iff m state vector v_i of the IO-OOM exists such that the $1 \times m$ vectors $(P(A_1 \mid v_i, \bar{r}), \ldots, P(A_m \mid v_i, \bar{r}))$ are linearly independent.

Comparison of basic concepts.

1. First, a note on notation. In the PSR articles [Littman *et al.*, 2001] [Singh *et al.*, 2003], the letter a refers to inputs and o refers to outputs, whereas I use r for inputs and a for outputs. In order to avoid confusion, I will henceforth use letters $u \in \mathcal{U}$ for controls and $y \in \mathcal{Y}$ for system outputs, both for PSRs and IO-OOMs, in accordance with common notation in the control engineering literature.

2. A PSR test $t_i = u_1^i, \ldots, u_{l_i}^i, y_1^i, \ldots, y_{l_i}^i$ tests whether a particular single observation sequence $y_1^i, \ldots, y_{l_i}^i$ is observed under inputs $u_1^i, \ldots, u_{l_i}^i$. Such a test has the following counterpart in IO-OOMs: given a characterization frame $(\bar{u}; Y_1, \ldots, Y_m)$ select one of the characteristic events Y_i and "test" whether *any* of the observation sequences $y_0, \ldots, y_{k-1} \in Y_i$ is obtained under input $\bar{u}$. A pair $t = (\bar{u}, Y)$, where $\bar{u}$ is an input sequence of some length l and $Y \subseteq \mathcal{O}^l$, shall be called an IO-OOM test.

3. The correlate in the IO-OOM framework of a PSR projection function f_t for a PSR test t is the linear function $\mathbf{1}\tau_Y^{\bar{u}}$ for an IO-OOM test $t = (\bar{u}, Y)$.

4. A major difference between PSRs and IO-OOMs is that the PSRs in principle admit arbitrary projection functions, whereas IO-OOMs consider exclusively *linear* projection functions $\mathbf{1}\tau_Y^{\bar{u}}$. This is a very important difference, whose implications remain to be understood. Here are some relevant observations:

   - Clearly, admitting nonlinear projection functions may lead to more compact models (i.e., a smaller number of tests may be sufficient to determine the probabilities of all possible tests when nonlinear projection functions are allowed).
   - An interesting question related to the sufficient statistics requirement for prediction vectors concerns their minimal dimension: What is the minimal number of core tests needed for a PSR of a

given stochastic system? Clearly, if one admits nonlinear projection functions, this number might be smaller than when only linear projection functions are allowed. It might turn out that when arbitrary projection functions are admitted, even a one-dimensional predictive state representation is generically possible, that is, a single test t^* generically exists such that the probability $P(t^* \mid h)$ is sufficient to compute the probabilities $P(t \mid h)$ of all other tests t. In order to illustrate this eventuality, consider for simplicity a degenerate controlled system which has only a single input value, that is, $\mathcal{U} = \{u\}$. Such a system can be considered a purely generative system whith no input at all, like a Markov chain, a HMM or a standard OOM. A previous history h boils down to a sequence $\bar{y}$ of observations; likewise, tests are then of the form $y_1, \ldots, y_l$. Now consider the probability clock example introduced in Section 6. This is certainly not a trivial stochastic system: it cannot be modelled by any finite-dimensional HMM, and the minimal OOM dimension is 3. Consider the singleton test $t^* = a$, i.e., the observation of an a, and the corresponding one-dimensional prediction vector $P(t^* \mid h) = P(a \mid h)$. Consider again Figure 3. The y-values of the dots mark the possible values that $P(a \mid h)$ can take. By virtue of the transcendent rotation angle of $\varphi = 1/(2\pi)$, no two such dots at different x-values (i.e. after a different prior history) will have identical y-values. In other words, knowing the precise value of $P(a \mid h)$ amounts to knowing how many a's have been generated before the current time and after the last "reset" event b. This amounts to full knowledge of the OOM's current state, and thus allows one to compute the probabilities of any other tests.

Note that this involves a highly nonlinear, infinite-precision computation within the execution of the projection functions, namely, going from the *exact* value of $P(a \mid h)$ to the number of a's generated previously in a row. The previous history is coded in the digit sequence of the probability value $P(a \mid h)$ in a complicated way; using this value as a "sufficient statistics" amounts to decoding this sequence of digits.

It seems plausible (or at least possible) that history coding in infinite-precision single-test probabilities is a generic phenomenon. If this turns out to be true, then the PSR idea of defining predictive state representations through a sufficient statistics property might need some further qualifications, e.g. by restricting the class of projection functions.

- Linear projection functions are not only computationally convenient, they are also only natural choice from the perspective of abstract OOM theory (as introduced briefly in Section 4 in this tutorial and more thoroughly in [Jaeger, 1999b]). The abstract view on future event probabilities in terms of predictor functions, and the resulting operator calculus for updating future distribution states (cf. Section 4), naturally leads to linear operators, the fundamental equivalence theorem from Section 5, and from there to the learning algorithm. All the advanced results obtained in the last years by Klaus Kretzschmar depend on the fact that one can cast a general theory of stochastic systems as a subtheory of linear algebra ([Jaeger, 1999b]). The advantage of nonlinear PSRs, namely, that they may be more compact than linear PSRs, opposes the disadvantage that currently no general theory or learning algorithm is available for nonlinear PSRs.

5. The similarity between PSRs and the OOM approach is further highlighted in the following observation. Given a prediction vector $p(h)$ after some history h and some arbitrary test $t = u_1, \ldots, u_l, y_1, \ldots, y_l$, what is the probability of success for this test, that is, what is $P(y_1, \ldots, y_l \mid h, u_1, \ldots, u_l)$? For linear PSRs derived from POMDPs, an answer is given in Theorem 1 of [Littman *et al.*, 2001]. In that theorem it is shown how from the POMDP one can compute certain matrices M_{u_i,y_i} and a projection vector m_{u_l,y_l}, such that $P(y_1, \ldots, y_l \mid h, u_1, \ldots, u_l) = p(h)M_{u_1,y_1} \cdots M_{u_{l-1},y_{l-1}} m_{u_l,y_l}$, which is essentially an IO-OOM representation (up to reverse ordering and using an outcome-specific projection vector m_{u_l,y_l} instead of the universal projection vector $\mathbf{1}$ found in the OOM approach). The IO-OOM representation is more general in that it is not restricted to systems equivalant to POMDPs.

6. In the available PSR papers, concrete linear PSRs are always derived from POMDPs (or weaker models). It is shown that the dimension of a PSR (i.e., the number of core tests) is bounded from above by the dimension of the original POMDP (i.e., number of hidden states). It is an interesting question of whether linear PSRs (or IO-OOMs) exist which

are equivalent to a POMDP, but whose dimension is properly smaller than the POMDP dimension. OOM theory provides an affirmative answer: POMDPs of arbitrarily large minimal dimension exist that have a three-dimensional IO-OOM (or PSR) representation. Consider again for simplicity the degenerate case of a single input $\mathcal{U} = \{u\}$, that is, POMDPs which are in fact just HMMs (or IO-OOMs which are in fact standard OOMs). The probability clock example introduced in Section 6 is an example of an OOM of dimension $m = 3$ which has no equivalent OOM at all. By a slight modification, it can be turned into an OOM which has equivalent HMMs, but where the minimal HMM dimension m_{HMM} is larger than 3 by an arbitrary factor, say, $m_{\mathrm{HMM}} \geq 3k$ for a given k. Just put the rotation angle to a value of $\varphi = 2\pi/3k$. Then the resulting analog of Figure 3 would be periodic after $3k$ steps. It is clear that this process can be modelled by a HMM which however would have to invest a minimum of $3k$ states to visit the steps of the rotation (a formal argument would use the fact that the convex cone K associated with this process would be $3k$-polyhedral, see the remark at the end at Section 6). Going back to proper IO-OOMs: If OOMs of this type were used as r-constituents of a proper IO-OOM (as described in Definition 3), the IO-OOM would be 3-dimensional but the minimal dimension of an equivalent POMDP would be at least $3k$.

7. (this point added in April 2004). In a recent manuscript[8] it has been pointed out that PSRs of controlled processes are more general than *interpretable* IO-OOMs. The authors provide an example of a 4-dimensional IO-OOM/PSR that has no interpretable version in the sense of the definition given in Subsection 10.2. The reason is that characterization frames rely on a single given sequence of inputs $\bar{r}$; in cases when all observation operators belonging to a given input project the state vector on a proper subspace (in the sense that the images of all τ_a^r, where r is fixed, fall in a proper subspace of the shared model space $\mathbb{R}^m$, and this holds for all r), the construction of an interpretable IO-OOM fails.

Learning. In [Singh *et al.*, 2003] an online learning algorithm for linear PSRs is presented. It is essentially an approximate ("myopic" in the words of the authors) stochastic gradient descent algorithm and works as follows.

[8] Predictive State Representations: A New Theory for Modeling Dynamical Systems. Satinder Singh, Michael R. James and Matthew R. Rudary. Submitted to UAI, 2004.

First, the set of core tests is extended to the set of *extension tests* by adding a test of the form $uyt = u, u_1, \ldots, u_{l_i}, y, y_1, \ldots, y_l$ for every core test $t = u_1, \ldots, u_{l_i}, y_1, \ldots, y_l$, and adding single-step tests of the form uy. The associated projection vectors m_x (where x indexes the extension tests) are initialized randomly to values $\hat{m}_x(0)$. Then, as a training input-output sequence is read in, the estimates $\hat{m}_x(n)$ are updated at time n according to

$$\hat{m}_x(n+1) \quad = \quad \hat{m}_x(n) - \alpha \, \frac{1}{w_x^\pi(n)} \, [\chi_x(n) - \hat{p}^\mathsf{T}(n) \, \hat{m}_x(n)] \, \hat{p}(n), \qquad (50)$$

where α is a learning rate (may depend on test and time), $x = u_1, \ldots, u_l, \, y_1, \ldots, y_l$ is an extension test that is applicable at time n (that is, the sequence $y(n+1), \ldots, y(n+l)$ of the next l inputs is the same as the input sequence of test t), π refers to a known policy of generating inputs, $w_x^\pi(n) = \prod_{i=1}^l P(y_i \mid h_{n+i}, \pi)$ is the probability of the next input sequence under policy π, $\chi_x(n) = 1$ if the sequence of next l outputs observed after time n is equal to the output sequence of the test (else $\chi_x(n) = 0$), and $\hat{p}(n)$ is the estimated predictive state vector at time n (this estimation uses the currently available estimates $\hat{m}_x(n)$ for core test projection vectors). Projection vectors of extension tests x which are not applicable at time n are not updated. Some details are omitted here, the reader is referred to [Singh *et al.*, 2003] for a complete treatment.

Connections with the IO-OOM learning algorithm. Although the PSR learing algorithm is an online algorithm and the IO-OOM is a batch algorithm, they share some important details that again emphasize the close relationship between the two approaches.

1. The introduction of extension tests is mirrored in the W_y^u matrices used in the IO-OOM learning algorithm (step 2(e)) and the $W_{(u,y)}$ matrices used in step 1 (they were not explicitly introduced in Subsection 10.4). This all reflects the need to obtain "statistics of change", that is, statistics about what happens to predictive states when at some time $n+1$ an input/output pair (u, y) occurs.

2. In 2000, I investigated several versions of an online learning algorithm for standard OOMs which is very closely related to the PSR learning algorithm. In a simple special case, this OOM learning algorithm works

as follows. Assume that a process is generated by an m-dimensional OOM, and that the number $\mid \mathcal{Y} \mid$ of observations is also m, that is, $\mathcal{Y} = \{y_1, \ldots, y_m\}$. A training sequence $S = y(1), y(2), \ldots$ is generated by this OOM. To learn from S an OOM that is interpretable with respect to singleton characteristic events $A_i = \{y_i\}$, start with initial random estimates observable operators $\hat{\tau}_{y_i}(0)$ and a random state vector $\hat{v}(0)$. The initialization $\hat{\tau}_{y_i}(0)$ of each operator must observe the constraint that all column sums are zero except for the i-th column, which must sum to 1 (this is a necessary condition for an interpretable OOM of this sort). At time n, estimates $\hat{\tau}_{y_i}(n)$ and $\hat{v}(n)$ are available $(i = 1, \ldots, m)$. Update the observable operator $\hat{\tau}_{y(n)}(n)$ and the state vector $\hat{v}(n)$, using $y(n)$ and $y(n+1)$, as follows:

(a) Compute a preliminary estimate of the next state vector through $v^*(n+1) = \hat{\tau}_{y(n)}(n)\hat{v}(n)/\mathbf{1}\hat{\tau}_{y(n)}(n)\hat{v}(n)$.

(b) Let i be the index of $y(n+1)$, that is $y_i = y(n+1)$. Put $w = -v^*(n+1)$ except at position i, where $[w]_i = 1 - [v^*(n+1)]_i$ ($[\cdot]_i$ denotes i-th vector component). Note: the components of w sum to zero because $v^*(n+1)$ sums to 1.

(c) Let j be the index of $y(n)$. Update $\hat{\tau}_{y(n)}(n+1) = \hat{\tau}_{y(n)}(n) + \alpha\, w(\hat{v}(n))^{\mathsf{T}}$ and $\hat{v}(n+1) = \hat{\tau}_{y(n)}(n+1)\hat{v}(n)/\mathbf{1}\hat{\tau}_{y(n)}(n+1)\hat{v}(n)$. Notes: (i) This leaves the column sums unaffected. (ii) α is a learning rate.

(d) If any values of $\hat{v}(n+1)$ fall outside the range $[0, 1]$, clip them back into this range and renormalize the resulting clipped state vector to unit sum. Note: This occurs frequently in the initial learning phase when the model is still bad. Observe that in an interpretable OOM, all state components are probabilities and must fall into $[0, 1]$.

This simple algorithm is "myopic" like the PSR learning algorithm. Its basic idea is that the update changes $\hat{\tau}_{y_i}(n)$ in a way that increases the component of the next state vector that corresponds to the probability of the actually observed event $y(n+1)$. Technical (but manageable) complications arise when $\mid \mathcal{Y} \mid$ is not equal to the OOM dimension.

The model estimates obtained during a learning run are all interpretable w.r.t. $A_i = \{y_i\}$. This means that the estimated observable operator matrices can be directly compared with the known correct matrices; the original interpretable OOM is the unique optimal solution for the learning task.

The learning algorithm was found through heuristic considerations. It looks like a stochastic gradient descent algorithm but was not formally analyzed to be one.

This algorithm worked basically as expected. Define a prediction error by $\varepsilon(n) = [(\hat{v}(n) - v(n))^{\mathsf{T}}(\hat{v}(n) - v(n))]/m$, where $v(n)$ is the state vector obtained in the correct OOM at time n. Because $v(n)$ contains the true probabilities of the next observations at time $n + 1$, this error measures the accuracy of the model predictions of probabilities of next observations. It typically goes down stochastically to values in the order of 0.0001 after 100,000 update steps in experiments where three-dimensional OOMs had to be learned.

However, I encountered several difficulties with this algorithm which in the end let me abandon this line of investigation. The most salient problem is the presence of many modes of convergence. This is an intrinsic problem of stochastic gradient descent techniques. In practice this means that once a fairly low error is reached, further progress can extremely slow down. A typical observation is that at error levels of $\varepsilon(n) = 0.0001$, the estimated matrices still deviate from the correct ones by 5–10 per cent in some parameters. This reflects the fact that some parameters have much smaller effects on the probabilities computed by an OOM than others. Further convergence would require enormously long training, because the error level (which drives the gradient descent) is already very small, but the learning rate cannot not be increased for stability reasons. This prevents one from effectively testing experimentally whether a learning run would "in the end" converge to the correct solution. Therfore, it is hard if not impossible to practically rule out the possibility that the learning gets stuck in a local minimum. All that I can say is that in all experiments, the algorithm yielded models with small prediction errors after 100,000 steps.

Another difficulty, again characteristic for stochastic gradient descent algorithms, is instability. In order to speed up learning, one wishes to increase the learning rate. However, too large learning rates trigger instability. If one has practical applications in mind, then a reasonably low error must be reachable much faster than after the 100,000 or so steps I had to wait. However, this seemed not easy to achieve in the presence of luring instability. By comparison: the standard batch algorithm requires a few hundred to thousand data points only to reach similarly accurate models.

A third difficulty results from the circumstance that not only the model parameters are updated online, but that also the state vector is only available in the form of estimates which depend on the current model estimate. Conversely, the model update depends on the (error-bearing) state estimates. This mutual dependence renders a mathematical analysis of the convergence properties of the algorithm difficult.

I should mention here that in 1999, Vladislav Tadic (then a guest researcher at my former institute) developed another kind of online learning algorithm for standard OOMs based on a combination of the LSM algorithm and stochastic approximation. The algorithm maintains estimates of the matrices V and W_a, as defined in the batch algorithm for standard OOMs, and does not need matrix inversions to compute the observable operators from V and W_a. The algorithm almost surely converges to the correct solution if it exists. Unfortunately, Vladislav did not continue this work because he entered a research position in a different field. His results are however documented in an unpublished paper, and work could be resumed.

It is not immediately clear how my online learning algorithm for standard OOMs can be adapted to the IO-OOM case. One obvious way would be to adopt the strategy of the PSR learning algorithm: fix one characterization frame (say, with length one input, for instance $(u_1; y_1, \ldots, y_m)$ and then update the model only at times n when $u(n+1) = u_1$. However, this would grossly underexploit the training data. An alternative might be to train different IO-OOM models, one for each characterization frame $(u_i; y_1, \ldots, y_m)$ (where $i = 1, \ldots, m$) and merge the resulting models online, by individually transforming them all into a common characterization frame by algeabric means, computing the mean model, and re-transforming the mean model back into the models with the different characterization frames used in the learning run. However, this would increase the computational cost.

Comments on PSR and IO-OOM learning.

1. The outcome of the PSR learning algorithm is estimates for the projection vectors of the extension tests (and among them, the core tests). It is not clear to me how from these projection vectors one can construct projection vectors for other tests not among the extension tests. The construction of such other projection vectors in Theorem 1 in [Littman *et al.*, 2001] exploits the knowledge of an underlying POMDP. By contrast, the batch IO-OOM learning algorithm and Tadic's IO-OOM online learning algorithm yield estimates of the observable operators, which can be used to calculate the probabilities of arbitrary IO-OOM tests.

2. The PSR algorithm underexploits the training data because updates occur only at times where some extension test is applicable. By contrast, the IO-OOM batch learning algorithm fully exploits the training data (in step 1), and so does Tadic's IO-OOM online learning algorithm.

3. In the light of my experiences with slow residual convergence of my versions of stochastic gradient descent algorithms for standard OOMs, I suspect that the PSR algorithm likewise might suffer from this problem. A more detailed analysis (both for PSR and online OOM learning) is needed.

4. The IO-OOM batch learning algorithm is equipped with an add-on mechanism for determining an appropriate model size that neither overfits nor underexploits training data (see Subsection 10.5). This is a precious commodity in the light of statistical learning theory. Klaus Kretzschmar has further refined this mechanism for standard OOMs [Kretzschmar, 2003].

5. It is an open question whether the PSR learning algorithm (or my online OOM learing algorithm) are guaranteed to converge to a correct model even if it exists and the right model dimension is used in learning. By contrast, the batch IO-OOM algorithm should inherit the asymptotic correctness property from its standard OOM counterpart: step 1 certainly leads to asymptotic correct estimates of the CSP, because this is just a standard OOM estimation, and the remaining steps are just algebraic transformations. However, I have not formally proven

that. The Newton method based online learning algorithm for standard OOMs developed by Vladislav Tadic is proven to converge almost surely.

6. Despite their current weaknesses, the PSR learning algorithm and possible IO-OOM adaptations of my former OOM online learning algorithm should not be discarded too quickly. First, they are the only known computationally cheap online algorithms. Second, the slow convergence problem may become mitigated by introducing techniques known for speeding up LSM algorithms in the field of adaptive signal processing. Third, in many applications one might be satisfied with the initial fast convergence if the most salient properties of the system are captured relatively quickly.

To conclude, I would like to point to a difficulty which has not yet surfaced in the PSR literature, but which is of central importance from a statistical learning angle, and where OOM theory might contribute some insight. Namely, how can one determine whether some set of predictive states is, in fact, linearly independent? With linear PSRs, this question is intimately connected to detecting minimal-dimensional PSRs, because the minimal dimension m of a linear PSR (or an IO-OOM, for that matter) is defined by the circumstance that m linearly independent (but not $m+1$ linearly independent) state vectors can arise during the evolution of the process. In the presence of finite training data, all models carry some error, and all computed state vectors must be considered noisy. But, any set of noisy vectors is generically linearly independent in the strict mathematical sense. This implies that one cannot simply check for linear independence by Gaussian eliminiation or other purely algebraic methods for determining matrix rank. Instead, one must resort to methods developed in numerical linear algebra for determining the *numerical* rank of noisy matrices, that is, reject the hypothesis that a set of vectors is linearly dependent under assumptions about the size and distribution of their noise components. OOM theory offers a working solution, see Subsections 9.3 and 10.5 for an introduction and [Kretzschmar, 2003] for a more refined approach.

11 Conclusion

The theoretical and algorithmic results reported in this tutorial are all variations on a single insight: The *change* of predictive knowledge that we have about a stochastic system, is a *linear* phenomenon. This leads to the concept of observable operators, which has been detailed out here for discrete-time, finite-valued processes, but is valid for every stochastic process [Jaeger, 1998b]. The linear nature of observable operators allows to solve the system identification task purely by means of numerical linear algebra, which arguably is the best understood of all areas of applied mathematics. The techniques reported in this tutorial still involve a good deal of heuristics and need to be refined in many places. Nevertheless, considering the expressiveness of OOMs (stronger than HMMs) and the efficiency of model estimation (essentially O(model-dimension^3) for output-only systems), I hope that OOMs will soon be taken up and further developed in many fields. I would be very happy indeed if this tutorial would enable practicians to use OOMs, and if OOMs would turn out to be of real help in dealing with stochastic systems.

A Proof of theorem 1

A numerical function P on the set of finite sequences of observable events can be uniquely extended to the probability distribution of a stochastic process, if the following two conditions are met:

1. P is a probability measure on the set of initial sequences of length $k+1$ of observable events for all $k \geq 0$. That is, (i) $P[a_{i_0} \ldots a_{i_k}] \geq 0$, and (ii) $\sum_{a_{i_0} \ldots a_{i_k} \in \mathcal{O}^{k+1}} P[a_{i_0} \ldots a_{i_k}] = 1$.

2. The values of P on the initial sequences of length $k+1$ agree with continuation of the process in the sense that $P[a_{i_0} \ldots a_{i_k}] = \sum_{b \in \mathcal{O}} P[a_{i_0} \ldots a_{i_k} b]$.

The process is stationary, if additionally the following condition holds:

3. $P[a_{i_0} \ldots a_{i_k}] = \sum_{b_0 \ldots b_s \in \mathcal{O}^{s+1}} P[b_0 \ldots b_s a_{i_0} \ldots a_{i_k}]$ for all $a_{i_0} \ldots a_{i_k}\ in \mathcal{O}^{k+1}$ and $s \geq 0$.

Point 1(i) is warranted by virtue of condition 3 from definition 1. 1(ii) is a consequence of conditions 1 and 2 from the definition (exploit that condition

2 implies that left-multiplying a vector by μ does not change the sum of components of the vector):

$$\begin{aligned}\sum_{\bar{a}\in\mathcal{O}^{k+1}} P[\bar{a}] = \sum_{\bar{a}\in\mathcal{O}^{k}} \mathbf{1}\tau_{\bar{a}} w_0 = \\ = \ \mathbf{1}(\sum_{a\in\mathcal{O}}\tau_a)\cdots(\sum_{a\in\mathcal{O}}\tau_a)w_0 \quad (k+1 \text{ terms } (\sum_{a\in\mathcal{O}}\tau_a)) \\ = \ \mathbf{1}\mu\cdots\mu w_0 \quad = \quad \mathbf{1}w_0 \quad = \quad 1.\end{aligned}$$

For proving point 2, again exploit condition 2:

$$\sum_{b\in\mathcal{O}} P[\bar{a}b] = \sum_{b\in\mathcal{O}} \mathbf{1}\tau_b\tau_{\bar{a}}w_0 = \mathbf{1}\mu\tau_{\bar{a}}w_0 = \mathbf{1}\tau_{\bar{a}}w_0 = P[\bar{a}].$$

Finally, the stationarity criterium 3 is obtained by exploiting $\mu w_0 = w_0$:

$$\begin{aligned}\sum_{\bar{b}\in\mathcal{O}^{s+1}} P[\bar{b}\bar{a}] = \sum_{\bar{b}\in\mathcal{O}^{s+1}} \mathbf{1}\tau_{\bar{a}}\tau_{\bar{b}}w_0 \\ = \ \mathbf{1}\tau_{\bar{a}}\mu\ldots\mu w_0 \quad (s+1 \text{ terms } \mu) \\ = \ \mathbf{1}\tau_{\bar{a}}w_0 \quad = \quad P[\bar{a}].\end{aligned}$$

B Proof of proposition 2

Let $\bar{b} \in \mathcal{O}^*$, and $\mathfrak{g}_{\bar{b}} = \sum_{i=1}^{n} \alpha_i \mathfrak{g}_{\bar{c}_i}$ be the linear combination of $\mathfrak{g}_{\bar{b}}$ from basis elements of $\mathfrak{G}$. Let $\bar{d} \in \mathcal{O}^+$. Then, we obtain the statement of the proposition through the following calculation:

$$\begin{aligned}(\mathfrak{t}_a(\mathfrak{g}_{\bar{b}}))(\bar{d}) \quad = \\ = \ (\mathfrak{t}_a(\sum_{i=1}^{n}\alpha_i\mathfrak{g}_{\bar{c}_i}))(\bar{d}) \quad = \quad (\sum\alpha_i\mathfrak{t}_a(\mathfrak{g}_{\bar{c}_i}))(\bar{d}) \\ = \ (\sum\alpha_i P[a\,|\,\bar{c}_i]\ \mathfrak{g}_{\bar{c}_i a})(\bar{d}) \quad = \quad \sum\alpha_i P[a\,|\,\bar{c}_i]P[\bar{d}\,|\,\bar{c}_i a] \\ = \ \sum\alpha_i P[a\,|\,\bar{c}_i]\frac{P[\bar{c}_i a\bar{d}]}{P[a\,|\,\bar{c}_i]P[\bar{c}_i]} \quad = \quad \sum\alpha_i\frac{P[\bar{c}_i]P[a\bar{d}\,|\,\bar{c}_i]}{P[\bar{c}_i]} \\ = \ \mathfrak{g}_{\bar{b}}(a\bar{d}) \quad = \quad P[a\bar{d}\,|\,\bar{b}] \quad = \quad P[a\,|\,\bar{b}]\ P[\bar{d}\,|\,\bar{b}a] \\ = \ P[a\,|\,\bar{b}]\ \mathfrak{g}_{\bar{b}a}(\bar{d}).\end{aligned}$$

C Proof of proposition 3

From an iterated application of (12) one obtains $\mathfrak{t}_{a_{i_k}} \dots \mathfrak{t}_{a_{i_0}} \mathfrak{g}_\varepsilon = P[a_{i_0} \dots a_{i_k}] \mathfrak{g}_{a_{i_0} \dots a_{i_k}}$. Therefore, it holds that

$$\mathfrak{g}_{a_{i_0}\dots a_{i_k}} = \sum_{i=1,\dots,n} \frac{\alpha_i}{P[a_{i_0}\dots a_{i_k}]} \mathfrak{g}_{\bar{b}_i}$$

Interpreting the vectors $\mathfrak{g}_{\bar{b}_i}$ and $\mathfrak{g}_{a_{i_0}\dots a_{i_k}}$ as probability distributions (cf. (11)), it is easy to see that $\sum_{i=1,\dots,n} \frac{\alpha_i}{P[a_{i_0}\dots a_{i_k}]} = 1$, from which the statement immediately follows.

D Proof of proposition 4

To see *1*, let $(\mathfrak{G}, (\mathfrak{t}_a)_{a \in \mathcal{O}}, \mathfrak{g}_\varepsilon)$ be the predictor-space OOM of (X_t). Choose $\{\bar{b}_1, \dots, \bar{b}_m\} \subset \mathcal{O}^*$ such that the set $\{\mathfrak{g}_{\bar{b}_i} \,|\, i = 1, \dots, m\}$ is a basis of $\mathfrak{G}$. Then, it is an easy exercise to show that an OOM $\mathcal{A} = (\mathbb{R}^m, (\tau_a)_{a \in \mathcal{O}}, w_0)$ and an isomorphism $\pi : \mathfrak{G} \rightarrow \mathbb{R}^m$ exist such that (i) $\pi(\mathfrak{g}_\varepsilon) = w_0$, (ii) $\pi(\mathfrak{g}_{\bar{b}_i}) = e_i$, where e_i is the i-th unit vector, (iii) $\pi(\mathfrak{t}_a \mathfrak{d}) = \tau_a \pi(\mathfrak{d})$ for all $a \in \mathcal{O}, \mathfrak{d} \in \mathfrak{G}$. These properties imply that $\mathcal{A}$ is an OOM of (X_t).

In order to prove *2*, let again $(\mathfrak{G}, (\mathfrak{t}_a)_{a \in \mathcal{O}}, \mathfrak{g}_\varepsilon)$ be the predictor-space OOM of (X_t). Let Γ be the linear subspace of $\mathbb{R}^k$ spanned by the vectors $\{w_0\} \cup \{\tau_{\bar{a}} w_0 \mid \bar{a} \in \mathcal{O}^+\}$. Let $\{\tau_{\bar{b}_1} w_0, \dots, \tau_{\bar{b}_l} w_0\}$ be a basis of Γ. Define a linear mapping σ from Γ to $\mathfrak{G}$ by putting $\sigma(\tau_{\bar{b}_i} w_0) := P[\bar{b}_i]\ \mathfrak{g}_{\bar{b}_i}$, where $i = 1, \dots, l$. σ is called the *canonical projection* of $\mathcal{A}$ on the predictor-space OOM. By a straightforward calculation, it can be shown that $\sigma(w_0) = \mathfrak{g}_\varepsilon$ and that for all $\bar{c} \in \mathcal{O}^+$ it holds that $\sigma(\tau_{\bar{c}} w_0) = P[\bar{c}] \mathfrak{g}_{\bar{c}}$ (cf. [Jaeger, 1997a] for these and other properties of σ). This implies that σ is surjective, which in turn yields $m \leq k$.

E Proof of proposition 7

From the definition of process dimension (def. 2) it follows that m sequences $\bar{a}_1, \dots, \bar{a}_m$ and m sequences $\bar{b}_1, \dots, \bar{b}_m$ exist such that the $m \times m$ matrix $(P[\bar{a}_j \,|\, \bar{b}_i])$ is regular. Let k be the maximal length occurring in the sequences $\bar{a}_1, \dots, \bar{a}_m$. Define complex events C_j of length k by using the sequences a_j as initial sequences, i.e. put $C_j := \{\bar{a}_j \bar{c} \mid \bar{c} \in \mathcal{O}^{k - |\bar{a}_j|}\}$, where $|\ \bar{a}_j\ |$ denotes the length of $\bar{a}_j$. It holds that $(P[C_j \mid \bar{b}_i]) = (P[\bar{a}_j \mid \bar{b}_i])$. We transform the complex events C_j in two steps in order to obtain characteristic events.

In the first step, we make them disjoint. Observe that due to their construction, two complex events C_{j_1}, C_{j_2} are either disjoint, or one is properly included in the other. We define new, non-empty, pairwise disjoint, complex events $C'_j := C_j \setminus \bigcup_{C_x \subset C_j} C_x$ by taking away from C_j all complex events properly included in it. It is easily seen that the matrix $(P[C'_j \mid \bar{b}_i])$ can be obtained from $(P[C_j \mid \bar{b}_i])$ by subtracting certain rows from others. Therefore, this matrix is regular, too.

In the second step, we enlarge the C'_j (while preserving disjointness) in order to arrive at complex events A_j which exhaust $\mathcal{O}^k$. Put $C'_0 = \mathcal{O}^k \setminus (C'_1 \cup \ldots \cup C'_m)$. If $C'_0 = \emptyset$, $A_j := C'_j$ $(j = 1, \ldots, m)$ are characteristic events. If $C'_0 \neq \emptyset$, consider the $m \times (m+1)$ matrix $(P[C'_j \mid \bar{b}_i])_{i=1,\ldots,m, j=0,\ldots,m}$. It has rank m, and column vectors $v_j = (P[C'_j \mid \bar{b}_1], \ldots, P[C'_j \mid \bar{b}_m])$. If v_0 is the null vector, put $A_1 := C'_0 \cup C'_1, A_2 := C'_2, \ldots, A_m := C'_m$ to obtain characteristic events. If v_0 is not the null vector, let $v_0 = \sum_{\nu=1,\ldots,m} \alpha_\nu v_\nu$ be its linear combination from the other column vectors. Since all v_ν are non-null, non-negative vectors, some α_{ν_0} must be properly greater than 0. A basic linear algebra argument (exercise) shows that the $m \times m$ matrix made from column vectors $v_1, \ldots, v_{\nu_0} + v_0, \ldots, v_m$ has rank m. Put $A_1 := C'_1, \ldots, A_{\nu_0} := C'_{\nu_0} \cup C'_0, \ldots, A_m := C'_m$ to obtain characteristic events.

F Proof of proposition 9

It suffices to show that for every column vector $v = (\#BA_1, \ldots, \#BA_m)$ of the counting matrix V it holds that $\mathbf{1}\tilde{\tau}_{A_i} v = v^i$, where v^i is the i-th component of v. We show this first for $A_i = A^1_x = \bigcup\{a_1^{\to k}, \ldots, a_r^{\to k}\}$ from the first group:

$$\begin{aligned}
&\mathbf{1}\tilde{\tau}_{A^1_x} v = \\
&\quad = \mathbf{1}\tilde{\tau}_{a_1} v + \cdots + \mathbf{1}\tilde{\tau}_{a_r} v \\
&\quad = \mathbf{1}(\#(Ba_1A_1), \ldots, \#(Ba_1A_m)) + \cdots + \mathbf{1}(\#(Ba_rA_1), \ldots, \#(Ba_rA_m)) \\
&\quad = \#(Ba_1) + \cdots + \#(Ba_r) \quad = \quad \#(BA_i) \quad = \quad v^i.
\end{aligned}$$

In the cases $A_i = A^\nu_x = \bigcup\{(b_1\bar{c})^{\to k}, \ldots, (b_s\bar{c})^{\to k} \mid \bar{c}^{\to k} \subseteq A^{\nu-1}_y\}$, where $A^{\nu-1}_y = A_j$, we use induction on ν to conclude

$$\begin{aligned}
&\mathbf{1}\tilde{\tau}_{A_x^\nu} v = \\
&\quad = \mathbf{1}\tilde{\tau}_{A_y^{\nu-1}}(\tilde{\tau}_{b_1}v + \cdots + \tilde{\tau}_{b_s}v) \\
&\quad = (\tilde{\tau}_{b_1}v + \cdots + \tilde{\tau}_{b_s}v)^j \\
&\quad = \#(Bb_1A_j) + \cdots + \#(Bb_sA_j) \quad = \quad v^i.
\end{aligned}$$

Finally, for A_m the statement follows from

$$\mathbf{1}\tilde{\tau}_{A_m}v = \mathbf{1}\tilde{\mu}v - \sum_{i=1}^{m-1}\mathbf{1}\tilde{\tau}_{A_i}v = \mathbf{1}v - \sum_{i=1}^{m-1}v^i = v^m.$$

G Proof of proposition 10

In this proof, column vectors of $\tilde{V}$ and $\tilde{W}_a$ are denoted by $v_i, w_i^a (i = 1, \ldots, m)$. Note that $v_i = (\tilde{P}[B_iA_1], \ldots, \tilde{P}[B_iA_m])$ and $w_i^a = (\tilde{P}[B_iaA_1], \ldots, \tilde{P}[B_iaA_m])$.

Case B_1: (i) $P_{\tilde{\mathcal{A}}}[A_i] = \mathbf{1}\tilde{\tau}_{A_i}\tilde{w}_0 = \mathbf{1}\tilde{\tau}_{A_i}v_1 =$ (by interpretability) $(v_1)^i = \tilde{P}[A_i]$.

(ii) $P_{\tilde{\mathcal{A}}}[aA_i] = \mathbf{1}\tilde{\tau}_{A_i}\tilde{\tau}_a\tilde{w}_0 = \mathbf{1}\tilde{\tau}_{A_i}w_1^a = (w_1^a)^i = \tilde{P}[aA_i]$.

Case B_ν: (i) $P_{\tilde{\mathcal{A}}}[C_1^\nu \ldots C_{x_\nu}^\nu A_i] = \sum_{c \in C_{x_\nu}^\nu} P_{\tilde{\mathcal{A}}}[C_1^\nu \ldots C_{x_\nu - 1}^\nu cA_i] =$ (by induction on ν) $\sum_{c \in C_{x_\nu}^\nu} \tilde{P}[C_1^\nu \ldots C_{x_\nu - 1}^\nu cA_i] = \tilde{P}[C_1^\nu \ldots C_{x_\nu}^\nu A_i]$. Note that $\tilde{\tau}_{B_\nu}\tilde{w}_0 = v_\nu$ is a direct consequence of this equality.

(ii) $P_{\tilde{\mathcal{A}}}[C_1^\nu \ldots C_{x_\nu}^\nu aA_i] = \mathbf{1}\tilde{\tau}_{A_i}\tilde{\tau}_a\tilde{\tau}_{B_\nu}\tilde{w}_0 = \mathbf{1}\tilde{\tau}_{A_i}\tilde{\tau}_a v_\nu = \mathbf{1}\tilde{\tau}_{A_i}w_\nu^a =$ (by interpretability) $(w_\nu^a)^i = \tilde{P}[C_1^\nu \ldots C_{x_\nu}^\nu aA_i]$.

The corollary follows directly if one observes that the set operations in question correspond to adding and subtracting of corresponding columns in the matrices $\tilde{V}$ and $\tilde{W}_a$, which does not alter the operators $\tilde{\tau}_a$ obtained.

Acknowledgments I feel deeply grateful toward Thomas Christaller for confidence and great support. This tutorial has benefitted enormously from advice given by Hisashi Ito, Shun-Ichi Amari, Rafael Nuñez, Volker Mehrmann, and two extremely constructive anonymous reviewers from Neural Computation, whom I heartily thank. The research described in this tutorial was carried out under a postdoctoral grant from the German National Research Center for Information Technology (GMD).

References

[Bengio, 1999] Y. Bengio. Markovian models for sequential data. *Neural Computing Surveys*, 2:129–162, 1999.

[Berman and Plemmons, 1979] A. Berman and R.J. Plemmons. *Nonnegative Matrices in the Mathematical Sciences.* Academic Press, 1979.

[Doob, 1953] J.L. Doob. *Stochastic Processes.* John Wiley & Sons, 1953.

[Elliott *et al.*, 1995] R.J. Elliott, L. Aggoun, and J.B. Moore. *Hidden Markov Models: Estimation and Control*, volume 29 of *Applications of Mathematics.* Springer Verlag, New York, 1995.

[Gihman and Skorohod, 1979] I.I. Gihman and A.V. Skorohod. *Controlled Stochastic Processes.* Springer Verlag, 1979.

[Gilbert, 1959] E.J. Gilbert. On the identifiability problem for functions of finite Markov chains. *Annals of Mathematical Statistics*, 30:688–697, 1959.

[Golub and van Loan, 1996] G.H. Golub and Ch.F. van Loan. *Matrix Computations, Third Edition.* The Johns Hopkins University Press, 1996.

[Heller, 1965] A. Heller. On stochastic processes derived from Markov chains. *Annals of Mathematical Statistics*, 36:1286–1291, 1965.

[Iosifescu and Theodorescu, 1969] M. Iosifescu and R. Theodorescu. *Random Processes and Learning*, volume 150 of *Die Grundlagen der mathematischen Wissenschaften in Einzeldarstellungen.* Springer Verlag, 1969.

[Ito *et al.*, 1992] H. Ito, S.-I. Amari, and K. Kobayashi. Identifiability of hidden Markov information sources and their minimum degrees of freedom. *IEEE transactions on information theory*, 38(2):324–333, 1992.

[Ito, 1992] H. Ito. *An algebraic study of discrete stochastic systems.* Phd thesis, Dpt. of Math. Engineering and Information Physics, Faculty of Engineering, The University of Tokyo, Bunkyo-ku, Tokyo, 1992. ftp'able from http://kuro.is.sci.toho-u.ac.jp:8080/english/D/.

[Jaeger, 1997a] H. Jaeger. Observable operator models and conditioned continuation representations. Arbeitspapiere der GMD 1043, GMD, Sankt Augustin, 1997. http://www.gmd.de/People/ Herbert.Jaeger/Publications.html.

[Jaeger, 1997b] H. Jaeger. Observable operator models II: Interpretable models and model induction. Arbeitspapiere der GMD 1083, GMD, Sankt Augustin, 1997. http://www.gmd.de/People/ Herbert.Jaeger/Publications.html.

[Jaeger, 1998a] H. Jaeger. Modeling and learning continuous-valued stochastic processes with OOMs. GMD Report 42, GMD, Sankt Augustin, 1998. http://www.gmd.de/People/Herbert.Jaeger/Publications.html.

[Jaeger, 1998b] H. Jaeger. A short introduction to observable operator models of discrete stochastic processes. In R. Trappl, editor, *Proceedings of the Cybernetics and Systems 98 Conference, Vol.1*, pages 38–43. Austrian Society for Cybernetic Studies, 1998. http://www.gmd.de/People/ Herbert.Jaeger/Publications.html.

[Jaeger, 1999a] H. Jaeger. Action selection for delayed, stochastic reward. In I. Wachsmuth and B. Jung, editors, *Proc. 4th Annual Conf. of the German Cognitive Science Society (KogWis99)*, pages 213–219. Infix Verlag, 1999.

[Jaeger, 1999b] H. Jaeger. Characterizing distributions of stochastic processes by linear operators. GMD Report 62, German National Research Center for Information Technology, 1999. http://www.gmd.de/publications/report/0062/.

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

[Kretzschmar, 2003] K. Kretzschmar. Learning symbol sequences with Observable Operator Models. GMD Report 161, Fraunhofer Institute AIS, 2003. ftp://borneo.ais.fraunhofer.de/pub/indy/publications_klaus/OomLearn.pdf.

[Littman *et al.*, 2001] M. L. Littman, R. S. Sutton, and S. Singh. Predictive representation of state. In *Advances in Neural Information Processing Systems 14 (Proc. NIPS 01)*, pages 1555–1561, 2001. http://www.eecs.umich.edu/ baveja/Papers/psr.pdf.

[Narendra, 1995] K.S. Narendra. Identification and control. In M.A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 477–480. MIT Press/Bradford Books, 1995.

[Oberstein, 2002] T. Oberstein. *Efficient Training of Observable Operator Models.* Master thesis, Köln University, 2002. http://www.ais.fraunhofer.de/INDY/tobias/eloom.pdf.

[Rabiner, 1990] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In A. Waibel and K.-F. Lee, editors, *Readings in Speech Recognition*, pages 267–296. Morgan Kaufmann, San Mateo, 1990. Reprinted from Proceedings of the IEEE 77 (2), 257-286 (1989).

[Singh *et al.*, 2003] S. Singh, M. Littman, N. Jong, D. Pardoe, and P. Stone. Learning predictive state representations. In *Proc.ICML 2003, to appear*, 2003. http://www.eecs.umich.edu/ baveja/Papers/ICMLfinal.ps.gz.

[Smallwood and Sondik, 1973] R.D. Smallwood and E.J. Sondik. The optimal control of partially observable markov processes over a finite horizon. *Operations Research*, 21:1071–1088, 1973.

[Strogatz, 1994] S.H. Strogatz. *Nonlinear Dynamics and Chaos.* Addison Wesley, 1994.