A theoretical basis for emergent pattern discrimination in neural systems through slow feature extraction

Stefan Klampfl, Wolfgang Maass
Institute for Theoretical Computer Science
Graz University of Technology
A-8010 Graz, Austria
{klampfl,maass}@igi.tugraz.at

May 22, 2010

Abstract

Neurons in the brain are able to detect and discriminate salient spatio-temporal patterns in the firing activity of presynaptic neurons. It is open how they can learn to achieve this, especially without the help of a supervisor. We show that a well-known unsupervised learning algorithm for linear neurons, Slow Feature Analysis (SFA), is able to acquire the discrimination capability of one of the best algorithms for supervised linear discrimination learning, the Fisher Linear Discriminant (FLD), given suitable input statistics. We demonstrate the power of this principle by showing that it enables readout neurons from simulated cortical microcircuits to learn without any supervision to discriminate between spoken digits, and to detect repeated firing patterns that are embedded into a stream of noise spike trains with the same firing statistics. Both these computer simulations and our theoretical analysis show that slow feature extraction enables neurons to extract and collect information that is spread out over a trajectory of firing states that lasts several hundred ms. In addition, it enables neurons to learn without supervision to keep track of time (relative to a stimulus onset, or the initiation of a motor response). Hence these results elucidate how the brain could compute with trajectories of firing states, rather than only with fixed point attractors. It also provides a theoretical basis for understanding recent experimental results on the emergence of view- and position-invariant classification of visual objects in inferior temporal cortex.

1 Introduction

The brain is able to extract an astonishing amount of information from its environment without a supervisor or teacher that tells the brain how an external stimulus should be classified. Experimental data show that one method which the brain uses in order to learn the categorization of external objects without a supervisor is the temporal slowness learning principle, which exploits the fact that temporally adjacent sensory stimuli are likely to be caused by the same external object. More precisely, experimental results from the lab of DiCarlo (Cox et al., 2005; Li and DiCarlo, 2008) (see DiCarlo and Cox, 2007, for a review) show that this simple heuristic is sufficient for the formation of position- and view-invariant representations of visual objects in higher cortical areas. This was tested in clever experiments by altering the probability that different objects caused temporally adjacent firing states in primary visual cortex (the external visual stimuli were swapped during the transient blindness while a saccade was performed). Human subjects were reported to merge different visual objects – presented at different retina locations – into single visual percepts as a result of this manipulation of the temporal statistics of visual inputs. Also the firing response of neurons in monkey area IT was reported to change accordingly. As a result of these data it was hypothesized in (Li and DiCarlo, 2008) that "unsupervised temporal slowness learning may reflect the mechanism by which the visual stream builds and maintains tolerant object representations". But a rigorous theoretical basis for the emergent discrimination capability of this unsupervised temporal slowness learning principle proposed

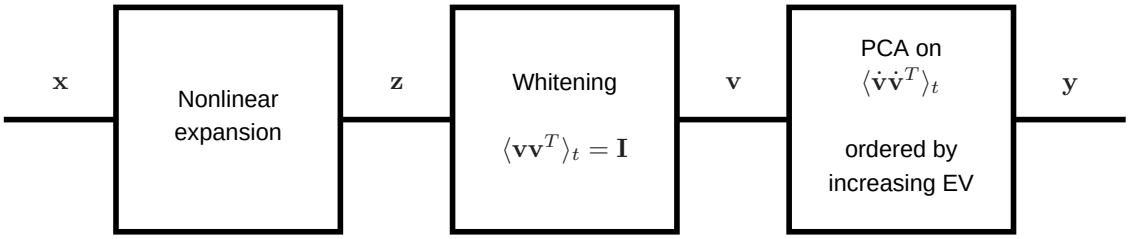

Figure 1: Blockdiagram of SFA. The algorithm is applied to a multi-dimensional time series $\mathbf{x}$. It consists of an optional expansion step which computes a number of fixed static nonlinear combinations $\mathbf{z}$ of the components of $\mathbf{x}$. Later in this article we will show that this step can be performed by a cortical microcircuit of neurons. If this step is left out, $\mathbf{z} = \mathbf{x}$. In the next step, the expanded input $\mathbf{z}$ has to be whitened such that the components of the signal $\mathbf{v}$ have zero mean, unit variance, and are decorrelated. The final step selects from this whitened signal the direction of minimal temporal variation, i.e., the least principal component of the temporal derivative $\dot{\mathbf{v}}$. The projection onto this direction yields the slowest feature y_1. Multiple slow features $\mathbf{y} = (y_1, y_2, \ldots)$ are obtained from orthogonal projection directions which form the eigenvectors of the covariance matrix $\langle \dot{\mathbf{v}}\dot{\mathbf{v}}^T \rangle_t$, ordered by increasing eigenvalue.

by (Li and DiCarlo, 2008) has been missing. Such theoretical foundation, which relates the statistics of the sequence of external stimuli to the emergent discrimination capability of this unsupervised learning method, is provided in this article.

There have been a number of approaches to learn invariant representations in an unsupervised manner from the contingency of temporally adjacent inputs, i.e., by extracting features that vary on a slow time scale (e.g., Földiak, 1991; Mitchison, 1991; Becker and Hinton, 1992; Stone and Bray, 1995). We focus on one particularly transparent computational mechanism for unsupervised temporal slowness learning: Slow Feature Analysis (SFA), introduced by (Wiskott, 1998; Wiskott and Sejnowski, 2002). SFA transforms a (usually high-dimensional) time series $\mathbf{x}$ into an output y, and minimizes the temporal variation of y under the additional constraints of zero mean and unit variance (to avoid the trivial constant solution). The temporal variation of the output y is defined as the average of its squared temporal derivative $\langle \dot{y}^2 \rangle_t$, where $\langle \cdot \rangle_t$ denotes averaging over time. In other words, SFA finds that function[1] g out of a certain predefined function space that produces the slowest possible output $y = g(\mathbf{x})$. This optimization problem is hard to solve in the general case (see Wiskott, 2003), but if the available function space is constrained to linear combinations of a whitened input, the problem has an elegant solution in the form of an eigenvalue problem in the covariance matrix of input time derivatives. More precisely, the slowest output is produced by the eigenvector of this matrix that corresponds to the *smallest* eigenvalue. This results in the standard SFA algorithm as presented in (Wiskott, 1998; Wiskott and Sejnowski, 2002) (see blockdiagram in Figure 1), which contains an optional expansion step that computes a number of fixed nonlinear combinations of the components of $\mathbf{x}$. Such nonlinear expansion boosts the power of any subsequent linear processing (like a kernel for support vector machines (Schölkopf and Smola, 2002)). This nonlinear expansion enables SFA to effectively choose from a much larger set of functions g (containing also nonlinear projections from $\mathbf{x}$), even if the last processing step in the blockdiagram of Figure 1 is constrained to be linear.

The restriction to linear functions in the last step of SFA allows that this processing step could in principle be carried out in a biological neural system by readout or projection neurons that extract information from a cortical microcircuit. A linear function is a reasonable approximation to the expressive capability of a readout neuron. The last step of SFA, the selection of the least principal component of the input time derivatives, could in principle be solved by anti-Hebbian learning on the differential input and output signals (Mitchison, 1991). Furthermore Sprekeler et al. (2007) have shown that this is equivalent to choosing the principal component of a low-pass filtered input, which can in principle be solved by standard Hebbian learning. In addition they have shown that an experimentally supported synaptic plasticity rule, spike-timing-dependent plasticity (STDP), could in principle enable spiking neurons to learn this processing step without supervision, provided that the presynaptic inputs are preprocessed in a suitable manner.

[1]Note that this function is a static input-output mapping $y(t) = g(\mathbf{x}(t))$, which at any time t transforms the input $\mathbf{x}(t)$ into an output value $y(t)$ instantaneously.

This result suggests, that the gap between the abstract SFA learning principle for linear neurons that we examine in this article and neurophysiological data on synaptic plasticity could eventually be closed. However, this analysis leaves open the question how the first two processing stages could be carried out by a biological neural system. We will show that a standard model for a generic cortical microcircuit could carry out the first processing step in the diagram of Figure 1 for the case where the time series $\mathbf{x}$ consists of multiple low-pass filtered spike trains. The question remains how the second processing step of Figure 1, the whitening, could be implemented by a neural circuit. Several learning methods that achieve whitening through a network of neurons have been proposed (Goodall, 1960; Atick and Redlich, 1993) (see chapter 8 of Dayan and Abbott, 2001). There also exist experimental data which suggest that the response of cortical neurons to natural external stimuli tends to be quite decorrelated (see e.g., Vinje and Gallant, 2000).

We establish in section 2 a relationship between the unsupervised SFA learning method and a commonly used learning method for supervised classification learning: the Fisher Linear Discriminant (FLD). More precisely, we show that SFA approximates the discrimination capability of the FLD in the sense that both methods yield the same projection direction, which can be interpreted as a separating hyperplane in the input space. This approximation holds for a simple condition on the temporal statistics of the input time series to SFA: The probability that two successive samples are from different classes has to be low. Through its tendency to produce a slowly varying output, SFA automatically clusters those inputs together that often occur in immediate consecution, and classifies them as different samples from the same category.

SFA is a learning method that does not require explicit supervision in the sense that the input patterns are given together with the target classification (labels). We show instead that it suffices to provide SFA with a very weak or implicit supervisor in the sense that successive input patterns tend to belong to the same class. Li and DiCarlo (2008) have referred to this as "unsupervised temporal slowness learning" and for brevity we use the term *unsupervised learning* in this article.

SFA may also elucidate a puzzle regarding internal codes and computational mechanisms of the brain. A number of experimental data have challenged the classical view of coding and computation in the brain, which was based on the assumption that external stimuli and internal memory items are encoded by firing states of neurons, which assign a certain firing rate to a number of neurons that is maintained for some time interval. This classical view of neural coding has the advantage that one can apply a variety of readily available computational models from computer science and artificial neural networks in order to model computation in the brain. However, numerous recent experimental data suggest that many types of natural sensory stimuli, as well as internally generated traces for episodic memory, are encoded by characteristic trajectories (or sequences) of different firing states of neurons that stretch over several hundred ms. This result has been found both for (seemingly) static external stimuli such as odors (Mazor and Laurent, 2005; Broome et al., 2006) (see Rabinovich et al., 2008, for a review) and tastes (Jones et al., 2007), and for intrinsically time-varying external stimuli such as natural auditory and visual stimuli (see Buonomano and Maass, 2009, for a review). In addition, numerous experimental data on replay of episodic memory from hippocampus to cortex point to sequences of different firing states, rather than single firing states of networks of neurons, as a common form of traces of episodic memory in hippocampus and cortex (see e.g., Euston et al., 2007; Ji and Wilson, 2008). These experimental data give rise to the question, how the brain can compute with such temporally dispersed information in the form of trajectories of firing states. At the latest at the top-level of information processing in the brain, where percepts are formed and decisions are made, the ubiquitous distribution of salient information over a sequence of different firing states (stretching over several hundred ms) has to be inverted, and compressed into a much shorter time interval. The theoretical analysis provided in this article explains why, and under what conditions, this is possible with SFA learning.

In section 3 we test the theoretically predicted emergent discrimination capability of SFA by applying it to the output of a simulated network of spiking neurons, more precisely, a detailed model for a laminar cortical microcircuit (Häusler and Maass, 2007) based on data from (Thomson et al., 2002) and from the lab of Markram (Gupta et al., 2000). We injected spike trains that simulate the response of the cochlea to different spoken digits as inputs to the simulated cortical microcircuit, and examined whether linear readouts that receive as input a whitened version of the continuously varying firing response (in the form of low-pass filtered spike trains) from neurons in this circuit can learn without supervision to discriminate between different spoken digits. This experiment turned out to be successful, and it also revealed a possible functional advantage of this processing scheme: Linear readout neurons learned not only without supervi-

sion to discriminate between different spoken digits, but they provided correct predictions of the currently spoken digit already while the digit was still being spoken. This is what we refer to as "anytime computing": An "anytime computation" is a special form of an online computation, which can be prompted at any time to provide its current best guess of a proper output, by integrating as much information about previously arrived input pieces as possible. In another experiment, SFA was able to both detect and identify spike patterns within a continuous stream of Poisson input. Again this information was available already during the presentation of a pattern. This feature, which is predicted by the theoretical analysis of SFA learning, could enable subsequent processing stages in the brain to begin higher level computational processing already before the trajectory of network states that is characteristic for a particular sensory stimulus has ended. This feature might remove one obstacle for establishing a computational model for hierarchical processing of sensory information in the cortex: if each stage waits with its processing until the trajectory of firing states in the lower area has ended, and then creates a subsequent trajectory as a result of its own computational processing, the resulting total computation time becomes too long. If however readout neurons can transmit "at any time" their current guess regarding the identity of the circuit input, other areas to which these readout neurons project can start their computational processing right away. During the subsequent few hundreds of ms they could in addition collect further evidence which they will receive from the same readout neurons, for or against the initial guess. In this computational paradigm the stream of sensory stimuli could generally be processed in real time, with significant processing delays arising only in the case of ambiguous sensory stimuli.

2 A theoretical basis for the emergent discrimination capability of SFA

In this section, we first give a definition of SFA and FLD. We then present a criterion on the temporal statistics of training examples which clarifies when SFA approximates FLD. Finally, we show how the SFA objective is influenced by applying it to a sequence of whole trajectories of points instead of just to a sequence of individual training examples.

Slow Feature Analysis (SFA) SFA extracts the slowest component y from a multi-dimensional input time series $\mathbf{x}$ by minimizing the temporal variation $\Delta(y)$ of the output signal y (Wiskott and Sejnowski, 2002),

$$\min \quad \Delta(y) := \langle \dot{y}^2 \rangle_t, \tag{1}$$

under the additional constraints of zero mean ($\langle y \rangle_t = 0$) and unit variance ($\langle y^2 \rangle_t = 1$). The notation $\langle \cdot \rangle_t$ is used in this article to denote averaging over time. If multiple slow features are extracted an additional constraint ensures that they are decorrelated ($\langle y_i y_j \rangle_t = 0$) and ordered by decreasing slowness.

If we assume that the time series $\mathbf{x}$ has zero mean ($\langle \mathbf{x} \rangle_t = \mathbf{0}$) and if we only allow linear functions $y = \mathbf{w}^T \mathbf{x}$ the problem simplifies to the following objective

$$\min \quad J_{SFA}(\mathbf{w}) := \frac{\mathbf{w}^T \langle \dot{\mathbf{x}} \dot{\mathbf{x}}^T \rangle_t \mathbf{w}}{\mathbf{w}^T \langle \mathbf{x} \mathbf{x}^T \rangle_t \mathbf{w}}. \tag{2}$$

The matrix $\langle \mathbf{x} \mathbf{x}^T \rangle_t$ is the covariance matrix of the input time series and $\langle \dot{\mathbf{x}} \dot{\mathbf{x}}^T \rangle_t$ denotes the covariance matrix of time derivatives of the input time series (or time differences, for discrete time). The weight vector $\mathbf{w}$ which minimizes the quotient in (2) is the solution to the generalized eigenvalue problem

$$\langle \dot{\mathbf{x}} \dot{\mathbf{x}}^T \rangle_t \mathbf{w} = \lambda \langle \mathbf{x} \mathbf{x}^T \rangle_t \mathbf{w} \tag{3}$$

corresponding to the smallest eigenvalue λ. That is, we consider only the linear part of SFA here and ignore the nonlinear expansion step in Figure 1 for the moment (i.e., $\mathbf{z} = \mathbf{x}$). Note that the whitening step is made implicit here in the formulation of (2), like in (Berkes and Wiskott, 2003).

Fisher's Linear Discriminant (FLD) The FLD is a different data analysis method. It is applied to single data points $\mathbf{x}$, rather than time series. Furthermore it requires *labeled* training examples $\langle \mathbf{x}, c \rangle$, where $c \in \{1, \cdots, C\}$ is the class to which this example belongs (we will first focus on the case $C = 2$). Hence it is a method for *supervised* learning. The goal is to find a weight vector $\mathbf{w}$ so that the class of new (unlabeled) test examples can be predicted from the value of $\mathbf{w}^T\mathbf{x}$ (predicting that $\mathbf{x}$ belongs to class 2 if $\mathbf{w}^T\mathbf{x} \geq \theta$ for some threshold θ, else that $\mathbf{x}$ belongs to class 1).

FLD searches for that projection direction $\mathbf{w}$ which maximizes the separation between classes while at the same time minimizing the variance within classes, thereby minimizing the class overlap of the projected values:

$$\max \quad J_{FLD}(\mathbf{w}) := \frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}}. \tag{4}$$

For two point sets S_1 and S_2 with means $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, $\mathbf{S}_B$ is the between-class covariance matrix given by the separation of the class means

$$\mathbf{S}_B = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T, \tag{5}$$

and $\mathbf{S}_W$ is the within-class covariance matrix given by

$$\mathbf{S}_W = \sum_{\mathbf{x} \in S_1} (\mathbf{x} - \boldsymbol{\mu}_1)(\mathbf{x} - \boldsymbol{\mu}_1)^T + \sum_{\mathbf{x} \in S_2} (\mathbf{x} - \boldsymbol{\mu}_2)(\mathbf{x} - \boldsymbol{\mu}_2)^T. \tag{6}$$

Again, the vector $\mathbf{w}$ optimizing (4) can be viewed as the solution to a generalized eigenvalue problem,

$$\mathbf{S}_B\mathbf{w} = \lambda\mathbf{S}_W\mathbf{w}, \tag{7}$$

corresponding to the *largest* eigenvalue λ. Figure 3A illustrates the idea of FLD. It finds that direction $\mathbf{w}$ that optimizes the separability between the projected values of different classes S_1 and S_2 by additionally taking into account the within-class variances. Choosing the direction $\mathbf{w}'$ that only maximally separates the class means results in an overlap of the projected values. The FLD had been introduced in (Fisher, 1936). Good descriptions can be found in (Duda et al., 2000; Bishop, 2006).

2.1 Application to a classification problem with two classes

SFA and FLD receive different data types as inputs: unlabeled time series for SFA, in contrast to labeled single data points $\langle \mathbf{x}, c \rangle$ for the FLD during training, and unlabeled single data points $\mathbf{x}$ during evaluation of its resulting generalization capability after training.

Therefore, in order to apply the unsupervised SFA learning algorithm to the same classification problem as the supervised FLD, we have to convert the labeled training samples into a time series of unlabeled data points that can serve as an input to the SFA algorithm. In the following we create such a training time series from the classification problem by choosing at each time step a particular point from $S_1 \cup S_2$. We investigate the relationship between the weight vector found by Fisher's Linear Discriminant on the original classification problem and the weight vector found by Slow Feature Analysis applied to the resulting training time series. The idea is that if we create the time series in such a way that most of its transitions, i.e., pairs of consecutive points, consist of point pairs from the same class, SFA should learn to be invariant to points from the same class and to extract the hidden class label of data points as a slowly varying feature of the time series.

First, we consider a classification problem with 2 classes, i.e., assume we are given two point sets $S_1, S_2 \subset \mathbb{R}^n$,

$$S_1 := \{\mathbf{x}_i^1 | i = 1, \ldots, N\}, \tag{8}$$

$$S_2 := \{\mathbf{x}_j^2 | j = 1, \ldots, N\}, \tag{9}$$

where $\mathbf{x}_i^1$ and $\mathbf{x}_j^2$ denote the data points of class 1 and 2, respectively (note that these points are unlabeled, since the superscripts 1 and 2 are not "visible" for the algorithms; it may also occur that $\mathbf{x}_i^1 = \mathbf{x}_j^2$). For simplicity we assume that both sets are of the same size N. We choose the following Markov model (see

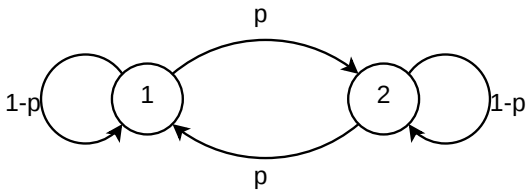

Figure 2: Markov model describing the generation of the input time series to SFA from a two-class FLD problem. The state c corresponds to the class S_c from which the current point in the time series is drawn. After the selection of each point the class of the next point is determined according to the transition probabilities between the states.

Figure 2) to create a time series $\mathbf{x}_t$ out of these two point sets S_1 and S_2: First, we choose one of the two classes with equal probability. Then we select a random point from the corresponding set (S_1 or S_2). This is then the first point in the input time series, $\mathbf{x}_1$. Next, we switch the class with a certain probability p (or leave it unchanged with probability $1-p$) and choose a point from the resulting class as the next input point, $\mathbf{x}_2$. This is repeated until the time series has a certain predefined length T. The states in the underlying Markov model correspond to the class from which the data point is currently drawn. After each drawing, the class is either switched with probability p, or left unchanged with probability $1-p$. The stationary distribution of this Markov model is

$$\boldsymbol{\pi} = \left(\frac{1}{2}, \frac{1}{2}\right). \tag{10}$$

Because we have chosen the initial distribution $\mathbf{p}_0 = \boldsymbol{\pi}$ we can say that at any time the current point is drawn from class 1 or class 2 with probability $1/2$.

In this case we can express the matrices $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ and $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ of the SFA objective (2) in terms of the within-class and between-class scatter matrices of the FLD (4), $\mathbf{S}_W$ and $\mathbf{S}_B$ (for a derivation see appendix A.1):

$$\langle \mathbf{x}\mathbf{x}^T \rangle_t = \frac{1}{2N}\mathbf{S}_W + \frac{1}{4}\mathbf{S}_B, \tag{11}$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t = \frac{1}{N}\mathbf{S}_W + p \cdot \mathbf{S}_B. \tag{12}$$

Note that only $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ depends on p, whereas $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ does not.

For small p we can neglect the effect of $\mathbf{S}_B$ on $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ in (12). In this case the time series consists mainly of transitions within a class, whereas switching between the two classes is relatively rare. Therefore the covariance of time derivatives is mostly determined by the within-class scatter of the two point sets, and both matrices become approximately proportional: $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \approx 1/N \cdot \mathbf{S}_W$. Moreover, if we assume that $\mathbf{S}_W$ (and therefore $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$) has only nonzero eigenvalues, we can rewrite the SFA objective as

$$\begin{aligned}
\min \quad J_{SFA}(\mathbf{w}) &\Leftrightarrow \max \quad \frac{1}{J_{SFA}(\mathbf{w})} \\
&\Leftrightarrow \max \quad \frac{\mathbf{w}^T \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{w}}{\mathbf{w}^T \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \mathbf{w}} \\
&\overset{(11),(12)}{\Leftrightarrow} \max \quad \frac{1}{2} + \frac{N}{4} \cdot \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} \\
&\Leftrightarrow \max \quad J_{FLD}(\mathbf{w}).
\end{aligned} \tag{13}$$

In the third line we inserted the expressions for $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ (11) and the approximation for $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ (12) for small p. That is, in this case where switching between different classes is rare compared to transitions within a class, the weight vector that yields the slowest output function is approximately equal to the weight vector that is optimal in separating the two classes in the sense of FLD.

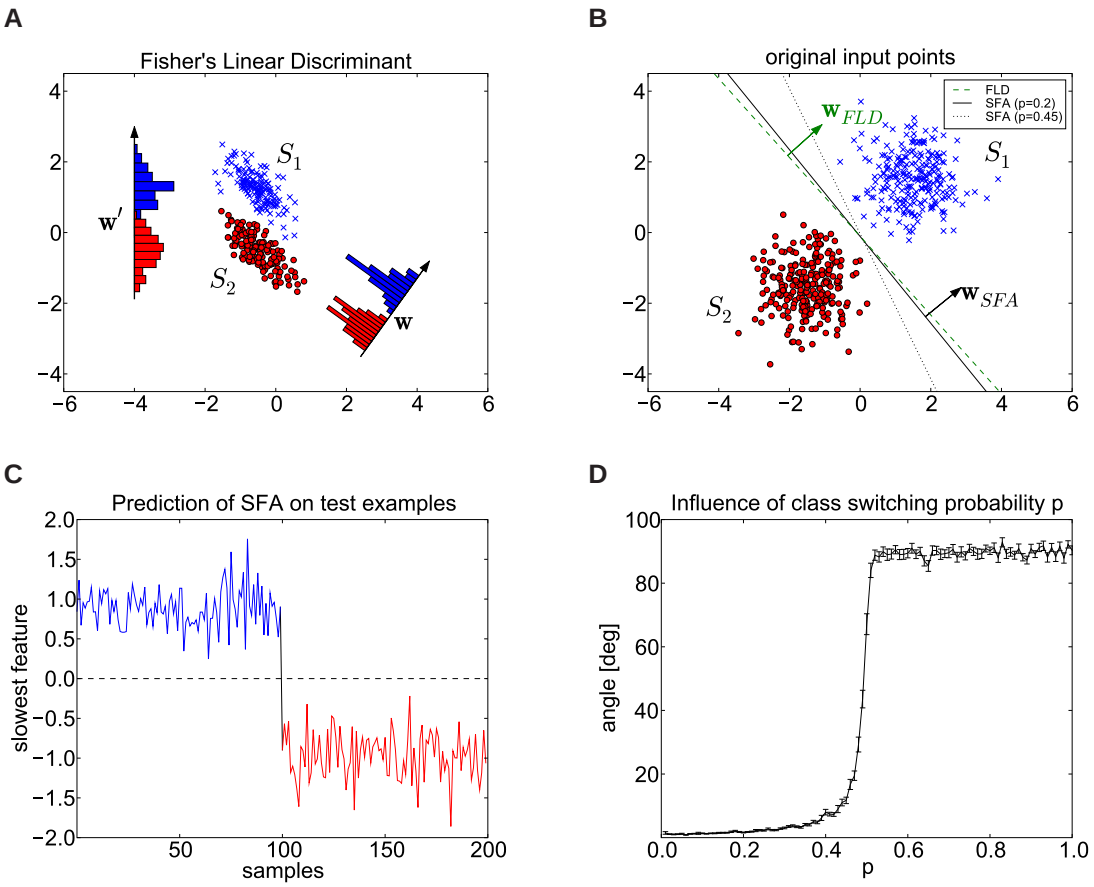

Figure 3: Relationship between unsupervised SFA and supervised FLD for a two-class problem in 2D. (**A**) Illustration of the concept of FLD. Shown are two point sets and histograms of values obtained by projecting points onto two different directions, $\mathbf{w}$, the direction resulting from FLD, and $\mathbf{w}'$, the direction of maximal separation of the class means. (**B**) Sample point sets with 250 points for each class, drawn from two different Gaussian distributions. The green arrow and the green dashed line indicates the weight vector ($\mathbf{w}_{FLD}$) and a corresponding hyperplane, respectively, resulting from the application of FLD to the two-class problem. The black arrow and the black solid line shows the weight vector ($\mathbf{w}_{SFA}$) and a hyperplane resulting from SFA applied to the time series generated from these training points as described in the text ($T = 5000$, $p = 0.2$). The black dotted line displays an additional SFA hyperplane resulting from a time series generated with $p = 0.45$. All hyperplanes are placed onto the mean value of all training points. (**C**) Output of the SFA algorithm (slowest feature) applied to a test time series consisting of 100 points from class 1 (blue) and 100 points from class 2 (red; colors as in **B**). These test points were drawn from the same Gaussian distributions as in **B**, but were not used for training. Each value of the trace corresponds to a projection of a point onto the weight vector of the slowest feature ($\mathbf{w}_{SFA}$ in **B**). The dashed line at 0 corresponds to points on the solid SFA-hyperplane shown in **B**. (**D**) Dependence of the error between the weight vectors found by FLD and SFA on the switching probability p. This error is defined as the average angle between the weight vectors obtained on 100 randomly chosen classification problems. Error bars denote the standard error of the mean. Good approximations can still be achieved with rather high values of p (up to 0.5).

Figure 3 demonstrates this relationship on a sample two-class problem in two dimensions. We interpret the weight vectors found by both methods as normal vectors of hyperplanes in the input space. Since an additional bias value is required to determine a unique hyperplane for each weight vector, we place the

hyperplanes in Figure 3B simply onto the mean value[2] $\boldsymbol{\mu}$ of all training data points (i.e., the hyperplanes are defined as $\mathbf{w}^T\mathbf{x} = \theta$ with $\theta = \mathbf{w}^T\boldsymbol{\mu}$). One sees that the weight vector found by the application of SFA to the training time series $\mathbf{x}_t$ generated with $p = 0.2$ is approximately equal to the weight vector resulting from FLD on the initial sets of training points. The deviation comes from the fact that the covariance matrix of time differences, $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$, is not solely determined by the within-class scatter (in eq. (12)), because the time series switches several times between the classes.

We interpret the slowest feature found by the SFA algorithm as the hypothesis of a linear classifier ($h(\mathbf{x}) = \text{sign}(\mathbf{w}_{SFA}^T(\mathbf{x} - \boldsymbol{\mu}))$). Figure 3C shows the prediction of this hypothesis for unseen test points from each class, drawn from the same distribution as the training point sets S_1 and S_2. It can be seen that the output of the slowest feature of this test time series (which corresponds just to the projection of its points onto the weight vector $\mathbf{w}_{SFA}$) takes on distinct values for different classes. This demonstrates that SFA has extracted the class of the points as the slowest varying feature by finding a direction that separates both classes, and that this ability generalizes to test points not used for training.

Figure 3D quantifies the deviation of the weight vector resulting from the application of SFA to the time series from the one found by FLD on the original points. We use the angle between both weight vectors as an error measure. For each value of p we generate 100 random classification problems such as the one shown in Figure 3B and calculate the average angle between the vectors obtained by both methods on these problems (see appendix B.1 for details). Since the sign of the vectors is arbitrary, we always took the smaller of the two possible angles. Thus, an angle of 0° means perfect equivalence, and the maximal achievable angle (i.e., error) is 90°. It can be seen that if p is low, i.e., transitions between classes are rare compared to transitions within a class, the angle between the vectors is small and SFA approximates FLD very well. The angle increases moderately with increasing p; even with higher values of p (up to 0.45) the approximation is reasonable and a good classification by the slowest feature can be achieved. As soon as p reaches a value of about 0.5, the error grows almost immediately to the maximal value of 90°. It can be seen from equations (11) and (12) that for $p = 0.5$ the covariance of time derivatives, $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$, becomes proportional to the covariance of the input, $\langle \mathbf{x}\mathbf{x}^T \rangle_t$, which means that every possible vector $\mathbf{w}$ is a solution to the generalized eigenvalue problem (3), resulting in an average angle of about 45°. With $p = 0.5$ points are drawn randomly from the union of the two point sets, independently of the class previously chosen, i.e., the class information is neglected altogether. For values of $p > 0.5$ switching between classes becomes so frequent that SFA cannot extract the class information anymore resulting in vectors orthogonal to the FLD vector.

2.2 Application to classification problems with more than two classes

The results in the previous section can also be extended to the case of C classes ($C > 2$), showing the equivalence between the space spanned by the $C - 1$ slow features extracted by SFA and the $C - 1$-dimensional subspace resulting from the application of a generalized version of Fisher's Linear Discriminant (Duda et al., 2000).

Again, we start from a classification problem with C disjoint point sets $S_c \subset \mathbb{R}^n, c = 1, \ldots, C$,

$$S_c := \{\mathbf{x}_i^c | i = 1, \ldots, N_c\}, \tag{14}$$

where $\mathbf{x}_i^c$ denote the data points of class c. In contrast to the previous section we consider here the more general case that the number of points in each class is different. Let N_c denotes the number of data points in class c, and let $N_T = \sum_{c=1}^{C} N_c$ be the total number of points. From these point sets we generate a time series $\mathbf{x}_t$ analogously as in the previous section, using a generalization of the Markov model in Figure 2 with C states $S = \{1, 2, \ldots, C\}$. We define the transition probability from state $i \in S$ to state $j \in S$ as

$$P_{ij} = \begin{cases} a \cdot \frac{N_j}{N_T} & \text{if } i \neq j, \\ 1 - \sum_{k \neq j} P_{ik} & \text{if } i = j, \end{cases} \tag{15}$$

[2]Note that for this particular choice of time series generation the expected mean of the training time series is equal to the total mean of the training data points. Since SFA subtracts the mean of the training time series beforehand, this value is mapped to 0 in the SFA output.

with some appropriate constant[3] $a > 0$. It is easy to show (see appendix A.2) that

$$\boldsymbol{\pi} = \left(\frac{N_1}{N_T}, \frac{N_2}{N_T}, \ldots, \frac{N_C}{N_T} \right) \tag{16}$$

is a stationary distribution of this Markov model. This means that the probability that any point in the time series is chosen from a particular class is proportional to the size of the corresponding point set compared to the number of total points.

For this particular way of generating a time series from the input points we can calculate the following expressions for the covariance matrices of the input and time derivatives in terms of the within-class and between-class covariances (see appendix A.2):

$$\langle \mathbf{x}\mathbf{x}^T \rangle_t = \frac{1}{N_T}\mathbf{S}_W + \frac{1}{N_T}\mathbf{S}_B, \tag{17}$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t = \frac{2}{N_T}\mathbf{S}_W + \frac{2a}{N_T}\mathbf{S}_B. \tag{18}$$

Note that equations (17) and (18) are similar to (11) and (12). Again, $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ depends on a, whereas $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ does not. Note that the commonly used definition for the between-class scatter matrix $\mathbf{S}_B$ (see e.g., Duda et al., 2000) for the multi-class case is slightly different from the two class case (5). For small a, i.e., when transitions between classes are rare compared to transitions within a class, we can approximate $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \approx 2/N_T \cdot \mathbf{S}_W$.

We recall the definition of SFA as a generalized eigenvalue problem (3) and insert (17) and (18) for negligible a:

$$\begin{aligned}
&\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \mathbf{W} = \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{W}\boldsymbol{\Lambda} \\
\overset{(17),(18)}{\Leftrightarrow}\ & \frac{2}{N_T}\mathbf{S}_W\mathbf{W} = \frac{1}{N_T}\mathbf{S}_W\mathbf{W}\boldsymbol{\Lambda} + \frac{1}{N_T}\mathbf{S}_B\mathbf{W}\boldsymbol{\Lambda} \\
\Leftrightarrow\ & 2\mathbf{S}_W\mathbf{W}\boldsymbol{\Lambda}^{-1} = \mathbf{S}_W\mathbf{W} + \mathbf{S}_B\mathbf{W} \\
\Leftrightarrow\ & \mathbf{S}_B\mathbf{W} = \mathbf{S}_W\mathbf{W}\left[2\boldsymbol{\Lambda}^{-1} - \mathbf{E}\right],
\end{aligned} \tag{19}$$

where $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_n)$ is the matrix of generalized eigenvectors and $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ is the diagonal matrix of generalized eigenvalues. We used the assumption that $\mathbf{S}_W$ (and therefore $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ and $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$) are positive definite, i.e., all eigenvalues λ_i are strictly positive ($\boldsymbol{\Lambda}^{-1}$ exists). The last line of (19) is just the formulation of FLD as a generalized eigenvalue problem (see (7)). More precisely, the eigenvectors of the SFA problem are also eigenvectors of the FLD problem, i.e., the $C-1$ slowest features extracted by SFA applied to the time series $\mathbf{x}_t$ span the subspace that optimizes separability in terms of Fisher's Linear Discriminant. Note that the eigenvalues correspond by

$$\lambda_i^{FLD} = \frac{2}{\lambda_i^{SFA}} - 1, \tag{20}$$

which means the order of eigenvalues is reversed, since all eigenvalues λ_i^{SFA} are positive. The slowest feature (corresponding to the smallest eigenvalue in the first line of (19)) is the weight vector which achieves maximal separation (largest eigenvalue in the last line of (19)).

This similarity of the subspace found by FLD on the initial point sets and by SFA on the time series is demonstrated in Figure 4. Panel B shows the projection of the data points shown in panel A onto the 2-dimensional subspace resulting from FLD, while Panel C plots the trajectory of the two slowest features found by SFA applied to a test time series generated from points drawn from the same distributions as the original points in panel A. One sees that both projections are almost identical, which means that the subspace that maximizes separability in terms of Fisher is equal to the subspace spanned by the slowest features of our particular time series. Note that there is more than one particular pair of directions which span the same 2-dimensional subspace. Therefore, while both methods extract the same subspace, the exact projections might look different (e.g., the signs of individual eigenvectors may be flipped, or the projections could be rotated against each other, if the eigenvalues are close to degenerate).

[3] For $C = 2$ and $N_1 = N_2 = N_T/2$ the class is switched at each time with probability $p = a/2$ and left unchanged with probability $1 - p$.

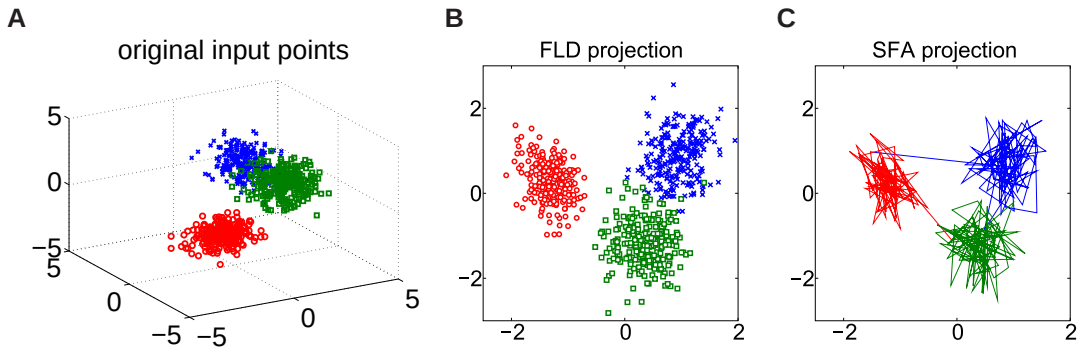

Figure 4: Relationship between SFA and FLD for a three-class problem in 3D. (**A**) Sample point sets with 250 points for each class, drawn from three different Gaussian distributions. (**B**) Point sets projected onto the 2-dimensional subspace found by FLD (colors and markers as in **A**). The FLD maximizes the between-class scatter while minimizing the within-class scatter. (**C**) Phase plot of the two slowest features found by SFA applied to a test time series consisting of 100 points from each class, which were drawn from the same Gaussian distributions as in **A**, but not used for training. The training sequence for SFA was generated from the input points in **A** as described in the text ($T = 5000$, $a = 0.5$). This corresponds to a projection of these test points onto the subspace spanned by the two slowest features. The color encodes the class of the respective point in the test sequence (colors as in **A**,**B**). Note the similarity between panels **B** and **C**.

2.3 Application to trajectories of training examples

In sections 2.1 and 2.2 we have shown that SFA approximates the classification capability of FLD if the probability is low that two successive points in the input time series to SFA are from different classes. In order to generate a time series from the classification problems we chose at each time step the class of the points with a certain probability according to a Markov model, but apart from that class information, however, each point was chosen independently from the preceding point in the time series. The optimal response to such a time series is to produce a constant response during periods where only points from a single class are presented (see also Berkes, 2006). This approximately piecewise constant function will become more smooth as the size of the function space increases, but it will remain a step function. This classification capability of SFA relies on the fact that SFA sees each possible transition between two points from the same class approximately equally often, and therefore produces a similar output for each point from that class.

What happens if these time series consist of whole *trajectories* of single points, e.g., repeated occurrences of characteristic sequences of firing states in neural circuits? In this section we investigate how the SFA objective changes when the input time series consists of trajectories of points instead of individual points only.

2.3.1 Repetitions of a fixed trajectory

First, we consider a time series $\mathbf{x}_t$ consisting of multiple repetitions of a fixed predefined trajectory $\tilde{\mathbf{t}} := (\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_{\tilde{T}})$ of $\tilde{T}$ n-dimensional points $\tilde{\mathbf{x}}_k$, which are embedded into noise input. Initially the trajectory points $\tilde{\mathbf{x}}_k$ are drawn from a certain distribution. Between any two repetitions of this trajectory noise input is presented, which consists of a random number of points drawn from the same distribution, but independently at each time step.

It is easy to show (see appendix A.3) that for such a time series the SFA objective (2) reduces to

$$\min \quad J_{SFA}(\mathbf{w}) \Leftrightarrow \max \quad \frac{\mathbf{w}^T \tilde{\mathbf{\Sigma}}_t \mathbf{w}}{\mathbf{w}^T \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{w}}, \tag{21}$$

where

$$\tilde{\mathbf{\Sigma}}_t := \frac{1}{\tilde{T}-1}\sum_{k=2}^{\tilde{T}}\left(\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_{k-1}^T + \tilde{\mathbf{x}}_{k-1}\tilde{\mathbf{x}}_k^T\right) \tag{22}$$

is the covariance matrix of the trajectory $\tilde{\mathbf{t}}$ with $\tilde{\mathbf{t}}$ delayed by one time step, i.e., it measures the temporal covariances (hence the index t) of $\tilde{\mathbf{t}}$ with time lag 1. Such time-delayed correlation matrices have also been introduced in (Blaschke et al., 2006, 2007) to show the relationship between SFA and second-order ICA. Note that in the standard classification problems described previously the time series $\mathbf{x}_t$ had no temporal correlations apart from the class information at all, i.e., consecutive points were uncorrelated given their class labels.

That is, choosing the weight vector $\mathbf{w}$ that produces the slowest output is equivalent to choosing the vector that maximizes the temporal correlations of the output during instances of the trajectory $\tilde{\mathbf{t}}$. In other words, $\mathbf{w}$ is the (generalized) eigenvector of $\tilde{\mathbf{\Sigma}}_t$ which corresponds to the largest eigenvalue of this matrix. Since the transitions between two successive points of the trajectory $\tilde{\mathbf{t}}$ occur much more often in the time series $\mathbf{x}_t$ than transitions between any other possible pair of points, SFA has to respond as smoothly as possible during $\tilde{\mathbf{t}}$ in order to produce the slowest possible output, whereas the average response to noise samples should ideally be zero. This means that SFA is able to detect these repetitions of $\tilde{\mathbf{t}}$ by responding during such instances with a distinctive shape.

Figure 5A shows the response of SFA, which was trained on a sequence of 100 repetitions of a fixed trajectory $\tilde{\mathbf{t}}$, interleaved with random intervals of noise input from the same distribution. It can be seen that during each instance of $\tilde{\mathbf{t}}$ SFA responds with the same smooth curve. Due to the intermittent noise input, this curve has to be cyclic and have zero mean. Typically this response is similar to a section of a sine wave, which is theoretically the slowest possible response for the general SFA optimization problem (1) (Wiskott, 2003). The smoothness of this sine wave critically depends on the number of trajectory repetitions (the proportion of time trajectories are presented compared to noise), the dimensionality of the state space, and the complexity of the function space (which is constrained to be linear here). For display purposes we have chosen an overfitting regime in Figure 5A, since the dimensionality of the state space is larger than the length of the trajectory. In this example, SFA also responds with an increased amplitude during trajectory presentations. This can be explained by the fact that the slowest signal with a constrained variance is one which distributes this variance to times when it varies more slowly (i.e., during trajectory repetitions).

2.3.2 Several classes of trajectories

Next, we consider a classification problem given by two sets of trajectories, $\mathcal{T}_1, \mathcal{T}_2 \subset (\mathbb{R}^n)^{\tilde{T}}$, i.e., the elements of each set $\mathcal{T}_c$ are sequences of $\tilde{T}$ n-dimensional points[4]. We assume that all those points are distinct and that $\mathcal{T}_1$ and $\mathcal{T}_2$ are of the same size N. Moreover, we emphasize that we draw the trajectories from distributions with different means, $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, as we did in the point discrimination examples. We generate a time series according to the same Markov model as in Figure 2. However, we do not choose individual points at each time step; rather we generate a sequence of trajectories: Initially, we choose a class from which the first trajectory is drawn, $\mathcal{T}_1$ or $\mathcal{T}_2$, and draw a random trajectory from this set. After each trajectory, we select a new trajectory of points after the previous one has ended. The class of this new trajectory is determined according to the transition probabilities in Figure 2.

For this time series consisting of such a trajectory sequence we can now express the matrices $\langle \mathbf{x}\mathbf{x}^T\rangle_t$ and $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T\rangle_t$ of the SFA objective (2) as (see appendix A.3)

$$\langle \mathbf{x}\mathbf{x}^T\rangle_t = \frac{1}{2N\tilde{T}}\mathbf{S}_W + \frac{1}{4}\mathbf{S}_B, \tag{23}$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T\rangle_t = \frac{1}{N\tilde{T}}\mathbf{S}_W + \frac{p}{\tilde{T}}\cdot\mathbf{S}_B - \frac{\tilde{T}-1}{\tilde{T}}\cdot\tilde{\mathbf{\Sigma}}_t. \tag{24}$$

The matrices $\mathbf{S}_W$ and $\mathbf{S}_B$ describe here the within-class and between-class scatter of the FLD objective (4) applied to point sets S_1 and S_2, which are composed of the individual points of the trajectories in $\mathcal{T}_1$

[4]The generalization to C classes is analogous to section 2.2.

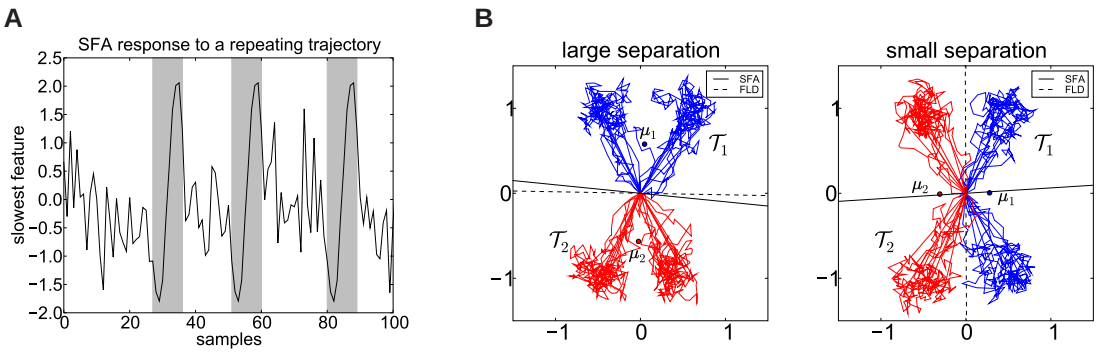

Figure 5: Relationship between SFA and FLD for time series consisting of trajectories. (**A**) SFA response to a time series consisting of a single repeating trajectory of training examples. A trajectory $\tilde{\mathbf{t}}$ is generated by randomly selecting $\tilde{T} = 10$ points from the uniform distribution of binary vectors, $\{0,1\}^n$ ($n = 50$). Repetitions of this trajectory $\tilde{\mathbf{t}}$ (shaded areas) are interleaved with a random number (drawn uniformly between 10 and 30) of individual single points drawn from the same distribution. The input time series that was used for training SFA consisted of 100 such repetitions of $\tilde{\mathbf{t}}$; a sample SFA response with 3 repetitions is shown. (**B**) Classification problem with two classes of artificial trajectories in 2D (blue, $\mathcal{T}_1$, and red, $\mathcal{T}_2$), each consisting of 20 trajectories of 100 points. The class means are denoted by $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$. In both panels the trajectories were drawn from the same distribution, but in the left panel the class labels were chosen in order to yield a large separation between the class means, whereas in the right panel this separation is small. The dashed line indicates a hyperplane corresponding to the weight vector obtained by application of FLD to the individual points of the trajectories. The solid line is the hyperplane found by SFA on a random sequence of 1000 trajectories. Both hyperplanes are placed onto the mean value of the trajectories. Note that the result of SFA is independent of p (here, $p = 0.5$).

and $\mathcal{T}_2$, respectively. Note that the covariance matrix $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ in (23) is equal to the case where the time series was composed of individual points instead of trajectories (see equation (11)). However, the temporal correlations induced by the use of trajectories has an effect on the covariance of temporal differences $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ in (24) compared to (12). First, it additionally depends again on the temporal covariance matrix $\tilde{\boldsymbol{\Sigma}}_t$, which is in this case the average temporal covariance with time lag 1 of all available trajectories in $\mathcal{T}_1$ and $\mathcal{T}_2$. Second, the switching probability p enters with a factor $1/\tilde{T}$, which becomes apparent when noting that whenever a trajectory is selected, $\tilde{T}$ points from the same class are presented in succession. Thus the effective switching probability is $p/\tilde{T}$. Note that for $\tilde{T} = 1$ and $\tilde{\boldsymbol{\Sigma}}_t = \mathbf{0}$ equations (11) and (12) follow as a special case.

Equations (23) and (24) suggest that even for a small value of p the objective of SFA cannot be solely reduced to the FLD objective, but rather that there is a trade-off between the tendency to separate trajectories of different classes (as explained by the relation between $\mathbf{S}_B$ and $\mathbf{S}_W$) and the tendency to produce smooth responses during individual trajectories (determined by the temporal covariance matrix $\tilde{\boldsymbol{\Sigma}}_t$):

$$\min \quad J_{SFA}(\mathbf{w}) = \frac{\mathbf{w}^T \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \mathbf{w}}{\mathbf{w}^T \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{w}} \approx \frac{\mathbf{w}^T \left[\frac{1}{N\tilde{T}} \mathbf{S}_W \right] \mathbf{w}}{\mathbf{w}^T \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{w}} - \frac{\tilde{T}-1}{\tilde{T}} \cdot \frac{\mathbf{w}^T \tilde{\boldsymbol{\Sigma}}_t \mathbf{w}}{\mathbf{w}^T \langle \mathbf{x}\mathbf{x}^T \rangle_t \mathbf{w}}, \tag{25}$$

where the approximation is valid if $p/\tilde{T}$ is small[5]. That is, the SFA objective can be written as the difference between two terms. The weight vector $\mathbf{w}$ which minimizes the first term is equal to the weight vector found by the application of FLD to the classification problem of the individual trajectory points (note that $\mathbf{S}_B$ enters (25) through $\langle \mathbf{x}\mathbf{x}^T \rangle_t$, cf. eq. (13)). The weight vector which maximizes the second term is the one which produces the slowest possible response during individual trajectories. The factor $(\tilde{T}-1)/\tilde{T}$ is the proportion of transitions between successive points in the time series that belong to the same trajectory. If the separation between the trajectory classes is large compared to the temporal correlations (i.e., the first

[5]Note that the values of both numerators are in the same range because $\tilde{\boldsymbol{\Sigma}}_t$ is already a normalized covariance matrix (22) whereas $\mathbf{S}_W$ (6) needs to be normalized by a factor $1/N\tilde{T}$.

term in (25) dominates for the resulting $\mathbf{w}$) the slowest feature will be similar to the weight vector found by FLD on the corresponding classification problem. On the other hand, as the temporal correlations of the trajectories increase, i.e., the trajectories themselves become smoother, the slowest feature will tend to favor exploiting this temporal structure of the trajectories over the separation of different classes (in this case, eq. (25) is dominated by the second term for the resulting $\mathbf{w}$).

In the point discrimination example SFA derives its classification capability from seeing each possible transition between two points approximately equally often. This is not the case anymore when a sequence of trajectories is presented: now there are pairs of points from the same class which have too few transitions between them because most transitions are not between randomly chosen points, but within pre-defined trajectories. Furthermore, since the effective switching probability of the classes of two consecutive trajectories is reduced to $p/\tilde{T}$, the SFA objective (25) becomes essentially independent of the switching probability p, if the trajectories are sufficiently long. This means, that the SFA output does not depend on the temporal order of the trajectories any more, rather, the result is completely determined by the set of trajectories used for training. That is, by using a time series consisting of trajectories instead of individual points one loses the possibility to control the classification problem to be learned by changing the temporal statistics of the input. All possible class labellings of a given set of trajectories lead to the same direction learned by SFA. The class labelling which is in this case approximated by SFA according to (25) is the one which has the maximal separability in terms of the FLD, i.e., the one which corresponds to a scatter $\mathbf{S}_W$ which minimizes the first term in (25). This is demonstrated in Figure 5B, which shows two classification problems with artificial trajectories chosen from the same distribution of points, but with different assignment of class labels: one with a large and one with a small separation between the means of the trajectory classes. It can be seen that while the FLD always finds a separating hyperplane, SFA always approximates that classification problem with the larger separation. However, even if the slowest feature is not able to separate the classes, later SFA components, which find orthogonal directions to the previous ones, might be useful. For example, in the right panel of Figure 5B the second slowest feature would find a separating hyperplane.

In theory, the optimal response of SFA in this trajectory example would again be a piecewise constant function. However, if we introduce zero or noise input between two trajectories the optimal response would be half sine waves during presentations of individual trajectories, which are the typical SFA responses shown in (Wiskott and Sejnowski, 2002; Wiskott, 2003). If the means of the trajectory classes (e.g., $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ in Figure 5B) are equal, there would be no effect to discriminate classes in terms of Fisher's Linear Discriminant, because the first term in (25) vanishes. However, the theoretical analysis in (Wiskott, 2003) predicts that even in that case of equal class means SFA still provides a certain discrimination capability through the decorrelation constraint of multiple slow features: a feature that responds with half sine waves of different amplitudes for different patterns also varies slowly and can still be decorrelated to other responses. Thus, with an infinite function space SFA always produces a feature that responds with a different amplitude for each individual pattern. That is, in general SFA will try to distinguish all trajectories, but if the available function space is limited it might respond with the same amplitude to all trajectories which are similar, i.e., belong to the same class.

2.4 When does linear separation of trajectories of network states suffice?

Linear SFA can at best achieve a linear separation of trajectories of points. Although linear separation of complex trajectories of points is difficult in low dimensions, mathematical arguments imply that linear separation of such trajectories becomes much easier in higher dimensions. Consider artificial trajectories which are simply defined as a sequence of random points drawn uniformly from the d-dimensional hypercube $[0,1]^d$. Each point in this space corresponds to the vector of firing activities of the d presynaptic neurons of a readout at a particular time t. Each linear readout neuron defines a hyperplane in this state space by the particular setting of its weights. It assigns values 1 for points on one side of this hyperplane and values 0 to points on the other side of the hyperplane. Two trajectories are called linearly separable if they lie on different sides of some hyperplane. Figure 6A shows an example of such a pair of linearly separable trajectories in 3 dimensions. However, such a perfect separation of randomly drawn trajectories is very unlikely in this low-dimensional space.

Figure 6B shows that the situation changes drastically if one moves to higher-dimensional spaces. The

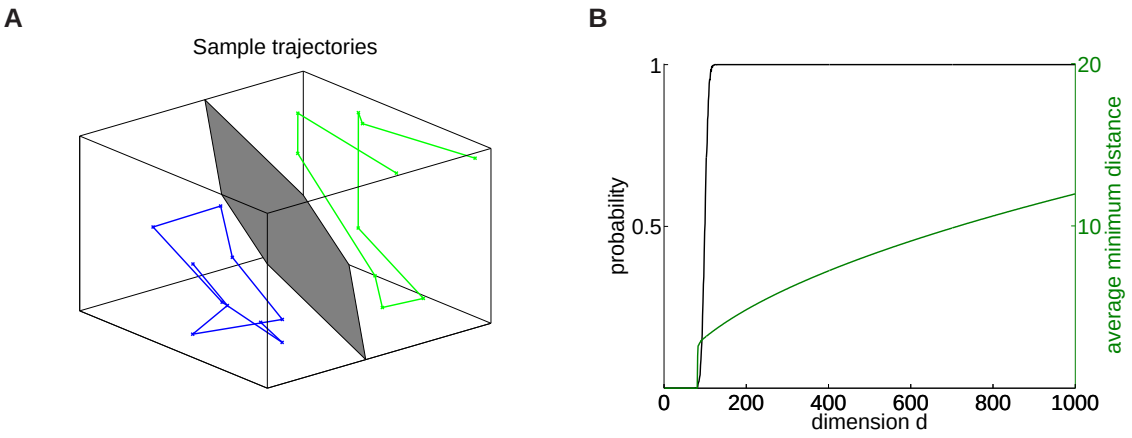

Figure 6: Probability of linear separability increases with higher dimensionality of the state space. (**A**) Two sample trajectories (green and blue curve) defined by connecting randomly drawn points from a 3-dimensional cube. These trajectories can be separated by a hyperplane (gray surface) defined by the synaptic weights of a linear discriminator. (**B**) Probability of linear separability of two randomly drawn trajectories of length 100 (black curve, left scale), and average minimum Euclidean distance between any two points of these two trajectories (green curve, right scale), as a function of the dimension d. Trajectories are defined as a sequence of random points drawn uniformly from the d-dimensional unit cube.

black curve indicates the probability that any two randomly drawn trajectories of length 100 (i.e., each trajectory is defined by connecting 100 random points drawn uniformly from the d-dimensional unit cube) are linearly separable in d dimensions, for different values of d (see appendix B.2). One sees that as soon as the dimension grows beyond 100, any two such trajectories become linearly separable with almost 100% probability. This holds for any length l of trajectories: for $d = l$, the probability of separation is 0.5 (see also Cover, 1965), if $d > l$ the probability converges very fast to 1. In other words, a linear readout neuron with d presynaptic inputs can separate almost any pair of trajectories that are each defined by connecting less than d randomly drawn points.

The green curve in Figure 6B shows the average of the minimal distance between such a pair of trajectories, which is defined as the minimal Euclidean distance between any point of trajectory 1 and any point of trajectory 2. This distance also grows with increasing dimension[6]. Thus, at higher dimensions d, it is not only more likely that any two trajectories of the length $l < d$ can be separated by a linear readout, but they can also be separated with an increasing "safety-margin" from the hyperplane. This implies that noisy variations of the same trajectories can be correctly classified by the same linear readout, which hints to a better generalization capability of linear readout neurons for higher dimensions.

3 Application to unsupervised training of linear readouts from a cortical microcircuit model

In the previous section we have shown that Slow Feature Analysis can directly be used for unsupervised linear discrimination of different point sets, if a time series is generated from these point sets in a way that the class is a slowly varying feature. Furthermore, we have shown how this property is affected if this time series consists of a sequence of trajectories instead of individual points. Now we turn our attention to SFA as a possible mechanism for training readouts of a biological microcircuit. The sequence of states that such a recurrent network undergoes in response to a specific stimulus forms a trajectory in state space. When presented with a sequence of such trajectories SFA should again be able to extract information about the

[6]Note that the length of the main diagonal of a d-dimensional hypercube, i.e., the largest possible distance between any two points from the hypercube, is $\sqrt{d}$.

stimulus in a similar way.

We have argued in the section 2.3 that the application of SFA to such a sequence of trajectories of network states differs from the application to individual points of a classification problem. In the latter case, a different input pattern has been presented at every single time step, whereas in the former case a single trajectory forms a sequence of input patterns from the same class. Due to the temporal correlations of these trajectories we do not expect that the slowest feature always perfectly extracts the class of the trajectories, as it did for the example with individual points in Figure 3. Rather, we predict that the class information will be distributed over multiple slow features. If multiple slow features are extracted, the feature y_i is the slowest feature under the additional constraint to be decorrelated to all slower features $y_1, \ldots, y_{i-1}$. This means that the slowest features are ordered by decreasing slowness, i.e., y_1 is the slowest feature, y_2 is the second slowest feature, and so on. In the following, features y_i with a higher index i are also called "higher order" features.

When computing with state trajectories in order to be able to extract reliable information about the stimulus, we want readouts of the circuit to produce an informative output not only at the end of the trajectory, but already while the trajectory is still being presented to the readout. Furthermore, this output should be as temporally stable as possible throughout the duration of a trajectory, hence providing an "anytime classification" of the stimulus. This requirement of temporal stability renders SFA a promising candidate for training readouts in an unsupervised fashion to discriminate "at any time" between trajectories in response to different stimulus classes. In the following we will discuss several computer simulations of a cortical microcircuit of spiking neurons where we trained a number of linear SFA readouts[7] on a sequence of network state trajectories, each of which is defined by the low-pass filtered spike trains of those neurons in the circuit that provide synaptic input to the readout neuron. Such recurrent circuits typically provide a temporal integration of the input stream and project it nonlinearly into a high-dimensional space (Maass et al., 2002), thereby boosting the expressive power of the subsequent linear SFA readouts. In the setup of Figure 1 the circuit therefore provides the mapping from the inputs $\mathbf{x}$ to the expanded signals $\mathbf{z}$, i.e., the trajectories of network states. The readouts then compute the slowest features $\mathbf{y}$ from these trajectories. Note, however, that the whitening step is performed implicitly in the SFA optimization (2). As a model for a cortical microcircuit model we use the laminar circuit from (Häusler and Maass, 2007) consisting of 560 spiking neurons organized into layers 2/3, 4, and 5, with layer-specific connection probabilities obtained from experimental data (Gupta et al., 2000; Thomson et al., 2002).

3.1 Detecting Embedded Spike Patterns

In the first experiment we investigated the ability of SFA to detect a repeating firing pattern within noise input of the same firing statistics. We recorded circuit trajectories in response to a sequence of 200 repetitions of a fixed spike pattern which are embedded into a continuous Poisson input stream. The input to the circuit consisted of 10 input spike trains. The pattern itself is defined as fixed Poisson spike trains of length 250ms and of rate 20Hz, the same rate as the background Poisson input (in the following also called noise input). We then trained linear SFA readouts on the 560-dimensional circuit trajectories, defined as the low-pass filtered spike trains of the spike response of all 560 neurons of the circuit (we used an exponential filter with $\tau = 30$ms and a sample time of 1ms). The period of Poisson input in between two such patterns was also randomly chosen; it was drawn uniformly between 100ms and 500ms.

Figure 7A shows a sample test stimulus consisting of a sequence of four pattern instances interleaved by random intervals of noise input, as well as the circuit response to this test stimulus and the 5 slowest features, y_1 to y_5, in response to the trajectory obtained by low-pass filtering this circuit response. At first glance, no clear difference can be seen between the raw SFA responses during periods of pattern presentations and during phases of noise input. The slow features are of course nonzero during noise input since the circuit response is quite similar to the response during patterns. However, we found that if we take the mean over the responses of multiple different noise phases, the average SFA output cancels away whereas a characteristic response remains during pattern presentations (see Figure 7C). This effect is predicted by the theoretical arguments in section 2.3 and can to some extent be seen in phase plots of traces that are obtained by a leaky integration of the slowest features in response to a test sequence of 50

[7] We interpret the linear combination defined by each slow feature as the weight vector of a hypothetical linear readout.

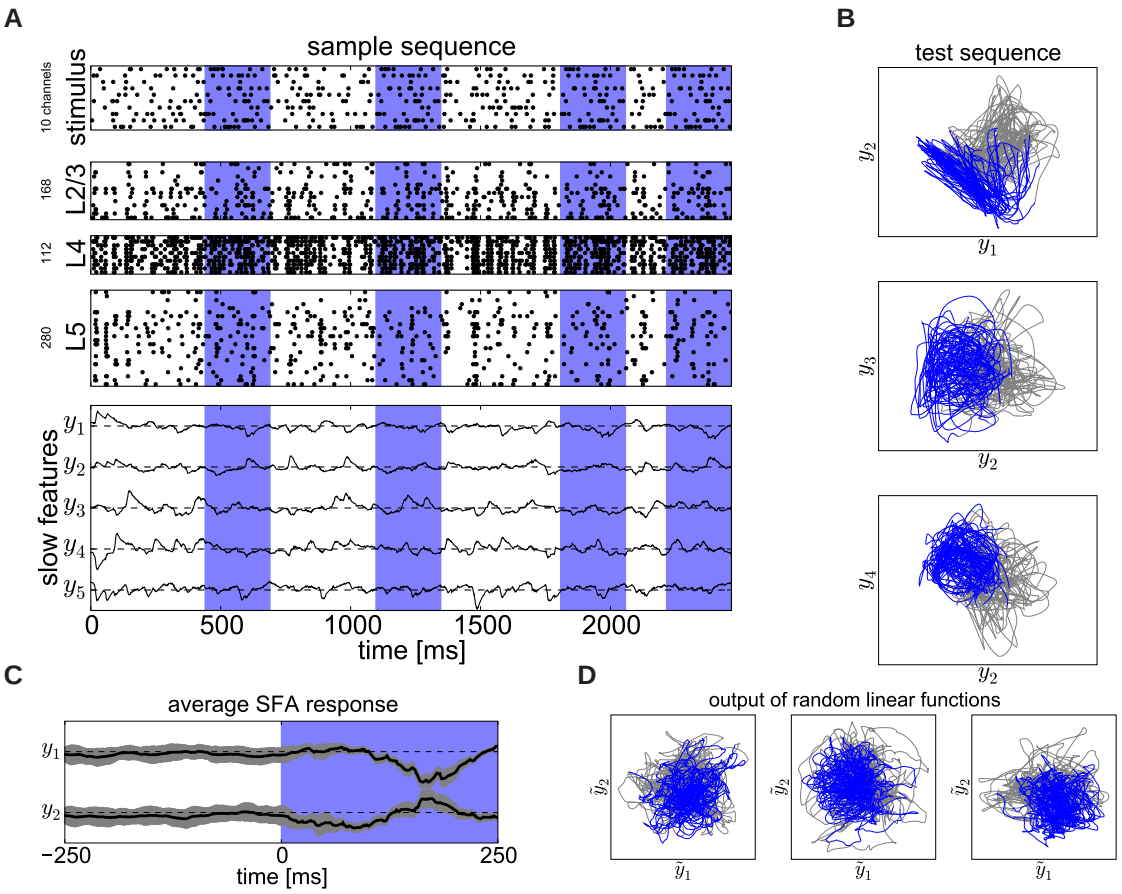

Figure 7: Unsupervised learning of the detection of spike patterns. (**A**) From top to bottom: sample stimulus sequence, response spike trains of the network, and slowest features. The stimulus consists of 10 channels and is defined by repetitions of a fixed spike pattern (blue shaded regions) which are embedded into random Poisson input of the same rate. The pattern has a length of 250ms and is made up by Poisson spike trains of rate 20Hz. The period between two patterns is drawn uniformly between 100ms and 500ms. The response spike trains of the laminar circuit of (Häusler and Maass, 2007) are shown separated into layers 2/3, 4, and 5. The numbers of neurons in the layers are indicated on the left, but only the response of every 12th neuron is plotted. Shown are the 5 slowest features, y_1 to y_5, for the network response shown above. The dashed lines indicate values of 0. (**B**) Phase plots of low-pass filtered versions (leaky integration, $\tau = 100$ms) of individual slow features in response to a test sequence of 50 embedded patterns plotted against each other (blue: traces during the pattern, gray: during random Poisson input). Note that equal increments in x- and y-direction have the same length, i.e., a circle is circular. (**C**) Average response of the two slowest features, y_1 and y_2, during the 250ms spike pattern (blue) and a preceding 250ms noise period (white). Note that the spike pattern is fixed, but the noise is drawn anew each time. The average was taken over 50 pattern repetitions not used for training, as those in **B**. The dashed line denotes the value 0; the shaded area indicates the standard deviation across these 50 repetitions. (**D**) Phase plots of two features $\tilde{y}_1$ and $\tilde{y}_2$ obtained from three randomly chosen orthogonal projections (compare with the top panel in **B**).

embedded patterns (see Figure 7B). The slowest features span a subspace where the response during pattern presentations can be nicely separated from the response during noise input. Concerning this separability, SFA yields a significant improvement over randomly chosen linear functions, as shown in Figure 7D. That is, by simple threshold operations on the low-pass filtered versions of the slowest features one can in principle detect the presence of patterns within the continuous input stream. Furthermore, this extracted information is not only available after a pattern has been presented, but already during the presentation of the pattern, which supports the idea of "anytime computing".

One interesting property of this setup is that if we apply SFA directly on the stimulus trajectories, we basically achieve the same result. In fact, the application to the circuit trajectories is the harder task because of the variability of the response to repeated presentations of the same pattern and because of temporal integration: The circuit integrates input over time making the response during a pattern dependent on the noise input immediately before the start of the pattern. Figure 7C shows these two effects. The standard deviation during the noise input is due to different stimulus spike trains which are drawn anew each time. On the other hand, the variability during the pattern presentations results from the inherent noise of the network, i.e., from different responses to the same stimulus. Figure 7C shows that the standard deviation during patterns is smaller than during noise. However, at the start of the pattern it does not decrease immediately, but gradually, due to temporal integration. That means that even though the average SFA response becomes different from zero just after the pattern onset, the output still depends on the previous noise input. Figure 7C suggests that this forgetting time of the circuit, the time after which the output of the laminar circuit does not depend on the noise any more, is at least 50ms.

3.2 Recognizing Isolated Spoken Digits

In the second experiment we tested whether SFA is able to discriminate two classes of trajectories as described in section 2.3. We performed a speech recognition task using the dataset considered originally in (Hopfield and Brody, 2000, 2001) and later in the context of biological circuits in (Maass et al., 2002, 2004) as well as in (Verstraeten et al., 2005) and in (Legenstein et al., 2008). This isolated spoken digits dataset consists of the audio signals recorded from 5 speakers pronouncing the digits "zero", "one", ..., "nine" in ten different utterances (trials) each. We preprocessed the raw audio files with a model of the cochlea (Lyon, 1982) and converted the resulting analog cochleagrams into spike trains that serve as input to our microcircuit model (see appendix B.3.2 for details). This biologically realistic preprocessing is computationally more expensive than the original encoding used in (Hopfield and Brody, 2000), but it has been shown that it can drastically improve the performance of a circuit for a specific speech recognition task (Verstraeten et al., 2005). Figure 8A shows sample cochleagrams, stimulus spike trains, and response spike trains for two utterances of digits "one" and "two" by the same speaker.

First, we tried to discriminate between trajectories in response to inputs corresponding to utterances of digits "one" and "two", of a single speaker (speaker 2, as shown in Figure 8). We split the 20 available samples (2 digits $\times$ 10 utterances) into 14 training and 6 test samples (i.e., three utterances of each digit is kept for testing). To produce an input to SFA, we generated from these 14 training samples a random sequence of 100 input patterns, recorded for each pattern the response of the circuit, and concatenated the resulting trajectories in time. Note that the same pattern is presented many times. Here we did not switch the classes of two successive trajectories with a certain probability because, as explained in the previous section, for long trajectories the SFA response is independent of this switching probability. Rather, we trained linear SFA readouts on a completely random trajectory sequence.

We then trained linear SFA readouts on the 560-dimensional circuit trajectories, defined as the low-pass filtered spike trains of the spike response of all 560 neurons of the circuit. All responses were recorded for the same amount of time such that all trajectories had the same length; after the circuit activity had stopped, the trajectories descended back to zero. Once there is zero (or noise) input between trajectories the result of SFA becomes independent of the temporal order of the trajectories because only adjacent time steps play a role. However, according to section 2.3 this is anyway the case for sufficiently long trajectories. Note that the network responses for repeated presentations of the same stimulus were different due to the inherent noise in the network that was used to model the background synaptic activity *in vivo* (see appendix B.3.2).

Figure 8B shows the 5 slowest features, y_1 to y_5, ordered by decreasing slowness in response to the trajectories corresponding to the three remaining test utterances for each class, digit "one" and digit "two". As a measure of slowness we used the index η of a signal $y(t)$ defined in (Wiskott and Sejnowski, 2002),

$$\eta(y) := \frac{T}{2\pi}\sqrt{\Delta(y)}. \tag{26}$$

This is a slightly different measure than (1), and denotes the number of oscillations of a sine wave with the same Δ-value. We found that the two slowest features, y_1 and y_2, responded with shapes similar to half sine waves during the presence of a trajectory (each 500ms a trajectory starts and lasts for several 100ms),

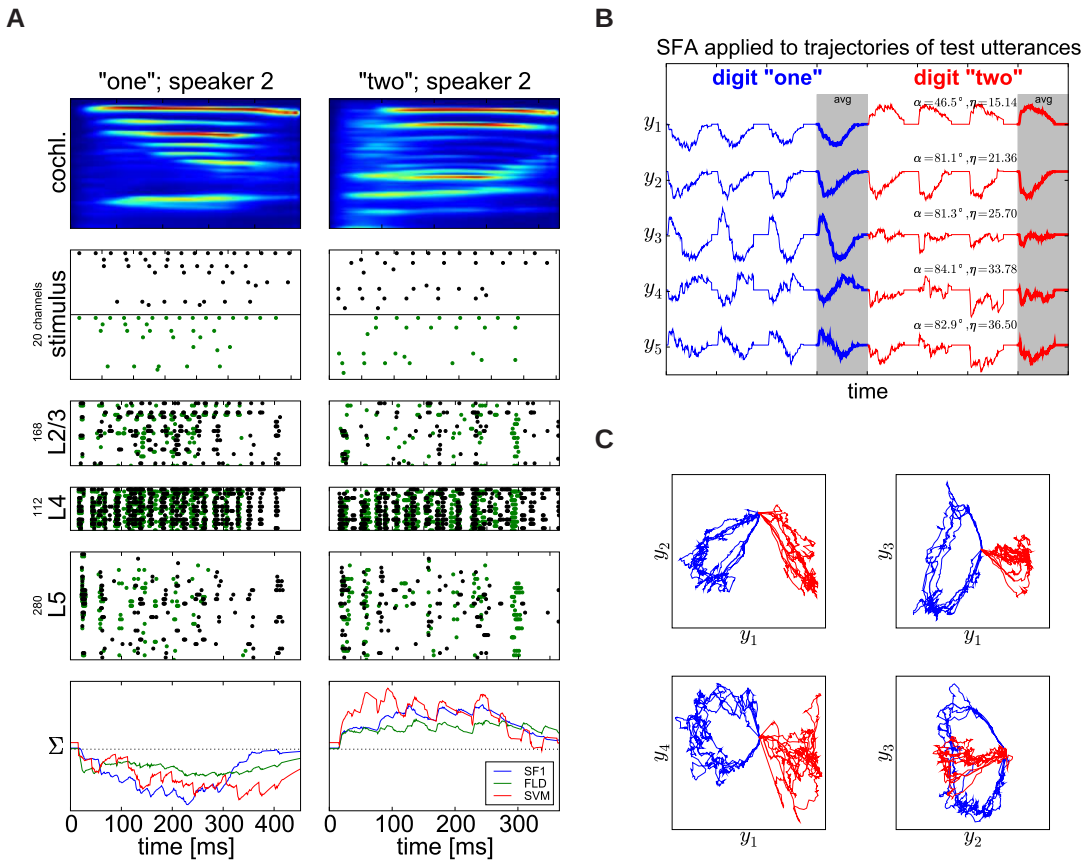

Figure 8: SFA applied to unsupervised digit recognition for a single speaker. (**A**) From top to bottom: sample cochleagrams, input spike trains, response spike trains of the network, and traces of different linear readouts. Each cochleagram has 86 channels with analog values between 0 and 1 (red, near 1; blue, near 0). Stimulus spike trains are shown for two different utterances of the given digit (black and green; the black spike times correspond to the cochleagram shown above). The response spike trains of the laminar circuit from (Häusler and Maass, 2007) are shown separated into layers 2/3, 4, and 5. The numbers of neurons in the layers are 168, 112, and 280, respectively, but only subsets of these neurons are plotted (14, 10, 24). The responses to the two stimulus spike trains in the panel above are shown superimposed with the corresponding color. Each readout trace corresponds to a weighted sum (Σ) of network states of the black responses in the panel above. The trace of the slowest feature ("SF1", blue line; see **B**) is compared to traces of readouts trained by FLD (green line) and SVM with linear kernel (red line) to discriminate at any time between the network states of the two classes. All weight vectors are normalized to length 1. The dashed line denotes the threshold of the respective linear classifier. (**B**) Response of the 5 slowest features y_1 to y_5 of the previously learned SFA in response to trajectories of the three test utterances of each class not used for training (blue, class 1; red, class 2). The slowness index η (26) is calculated from these output signals. The angle α denotes the deviation of the projection direction of the respective feature from the direction found by FLD. The thick curves in the shaded area display the mean SFA responses over all three test trajectories for each class. (**C**) Phase plots of individual slow features plotted against each other (thin lines: individual responses, thick lines: mean response over all test trajectories). Note that equal increments in x- and y-direction have the same length, i.e., a circle is circular.

which is in fact the slowest possible response under the unit variance constraint. Higher order features partly consisted of full sine wave responses, which are the slowest possible responses under the additional constraint to be decorrelated to previous slow features.

In this example already the slowest feature y_1 extracts the class of the input patterns almost perfectly: it responds with positive values for trajectories in response to utterances of digit "two" and with negative values for utterances of digit "one", and generalizes this behavior to unseen test examples. As a measure

for the discriminative capability of a specific SFA response, i.e., its quality as a possible classifier, we measured the angle between the projection direction corresponding to this slow feature and the direction of the FLD. Since each slow feature as well as the weight vector that specifies the projection direction of the FLD is only determined up to the sign, we only report the smaller value. These angular values therefore vary between 0° and 90°. It can be seen in Figure 8B that the slowest feature y_1 is closest to the FLD. Hence, according to (25), this constitutes an example where the separation between classes dominates, but is already significantly influenced by the temporal correlations of the circuit trajectories.

We call this property of the extracted features, to respond differently for different stimulus classes, the *What*-information (Wiskott and Sejnowski, 2002). The second slowest feature y_2, on the other hand, responds with half sine waves whose sign is independent of the pattern identity. One can say that, in principle, y_2 encodes simply the presence of a circuit response. This is a typical example of a representation of *Where*-information (Wiskott and Sejnowski, 2002), i.e., the "pattern location" regardless of the identity of the pattern. Full sine wave responses would further encode the position within the trajectory. The other slow features y_3 to y_5 do not extract either *What*- or *Where*-information explicitly, but rather a mixed version of both. For repeated runs of the same experiment with different training utterances the explicit *What*- and *Where*-information of y_1 and y_2 are reliably extracted, but the exact shape of the higher order features might differ depending on the particular training utterances.

Figure 8C shows phase plots of these slow features shown in Figure 8B plotted against each other. In theory, in the phase plot of two features encoding *What*-information the responses should form straight lines from the origin in a pattern-specific angle. In the three plots involving feature y_1 it can be seen that these response directions are distinct for different pattern classes. On the other hand, phase plots of two features encoding *Where*-information ideally form loops in the phase space, independent of the identity of the pattern, where each point on this loop corresponds to a position in the trajectory. This can only be seen to some extent in the plot y_2 vs y_3, but not explicitly because in this example no two features encode *Where*-information alone. Similar responses have been theoretically predicted in (Wiskott, 2003) and found in simulations of a hierarchical (nonlinear) SFA network trained with a sequence of one-dimensional trajectories (Wiskott and Sejnowski, 2002). Furthermore, we found that the response vector $\mathbf{r}(t) := (y_1(t), \ldots, y_5(t))$, which is composed of the values of all 5 slowest features at a particular point in time, clusters at different directions for different classes. The average angle between two response vectors from different classes is around 90 degrees throughout the duration of a trajectory. This effect arises from the decorrelation constraint and is also a theoretical result of (Wiskott, 2003).

Note that the information extracted by SFA about the identity of the stimulus is provided not only at the end of a specific trajectory, but is made available right from the start. After sufficient training, the slowest feature y_1 in Figure 8B responds with positive or negative values indicating the stimulus class during the whole duration of the network trajectory[8]. This supports the aforementioned idea of "anytime computing". Moreover, as a measure for the performance of SFA we can train a linear classifier on the extracted features, i.e., at each point in time the response vector $\mathbf{r}(t)$, composed of the values of the 5 slowest features at that time, and labelled with the class of the corresponding trajectory, serves as one data point for the classification. The performance that a particular classifier is able to achieve can be viewed as a lower bound for the information that the extracted slow features convey about the trajectory class. Applied to the features of Figure 8B, sampled every 1ms, an SVM with linear kernel achieves a classification performance of 98% (evaluated by 10-fold cross validation). Note again that this is an "anytime" classification, since samples during the whole duration of the trajectories are taken into account.

The bottom panel of Figure 8A shows readout traces of three different linear discriminators applied to specific test trajectories, one from each class. Each point on a trace represents a weighted sum of the network states at a particular time, just before the threshold operation of the corresponding linear classifier. That is, a value above (below) zero means that the state at that time is classified to belong to class 2 (class 1) by this particular linear discriminator. Here, we interpret the slowest feature extracted, y_1 from Figure 8B, as a linear discriminator with this particular weight vector and the average over the training time series as the discrimination threshold. We compare the trace of this "SFA classifier" to traces of linear readouts trained as Fisher's discriminant and Support Vector Machine (SVM) (Schölkopf and Smola,

[8] Since the optimal SFA response is not a piecewise constant curve, but a sequence of half sine waves, an even better discriminator would be the direction of the response vector $\mathbf{r}(t)$ which theoretically stays constant throughout a trajectory (Wiskott, 2003).

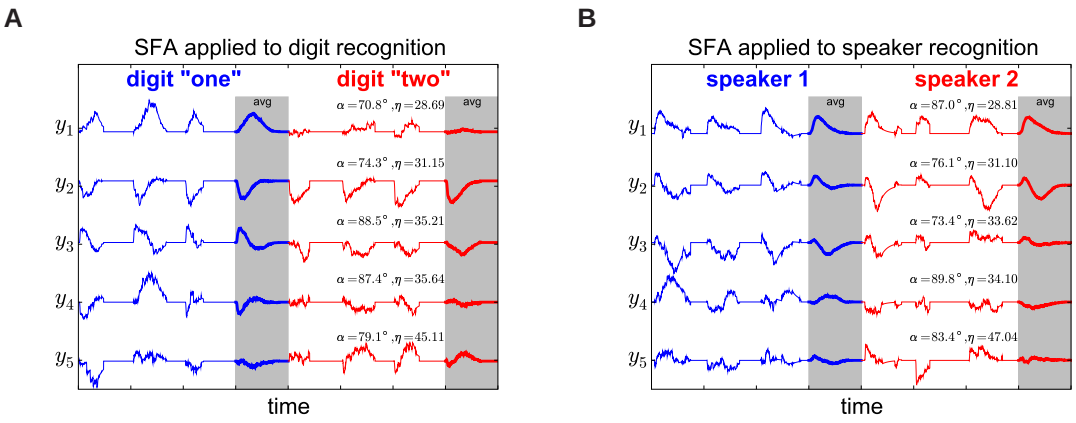

Figure 9: SFA applied to unsupervised speaker-independent digit recognition and digit-independent speaker recognition. Both panels show the response of the 5 slowest features y_1 to y_5 of the previously learned SFA in response to trajectories of three test utterances of each class not used for training. Trajectories are padded with zeros such that each trajectory has the same length. The slowness index η (26) is calculated from these output signals. The angle α denotes the deviation of the projection direction of the respective feature from the direction found by FLD. The thick curves in the shaded area display the average SFA responses over all available test trajectories for each class. (**A**) SFA applied to speaker-independent digit recognition. Shown are the responses for three random test trajectories of digit "one" (blue) and digit "two" (red) from three different speakers as well as the average SFA response over all 30 available test trajectories. (**B**) SFA applied to digit-independent speaker recognition. Shown are the responses for three random test trajectories of speaker 1 (blue) and speaker 2 (red) from three different digits as well as the average SFA response over all 60 available test trajectories.

2002) to discriminate between the network states of trajectories of different classes[9]. Both FLD and SVM are trained on the same input as SFA, which consists of the network states sampled with $\Delta t = 1$ms of 100 trajectories chosen randomly as described above (but, of course without the information about the temporal sequence of states). The discrimination threshold for both SFA and FLD was chosen as the average over all training points. It can be seen that in this case the slowest feature, which has been learned in an unsupervised manner, is able to achieve a perfect separation, comparable to those of the supervised methods of FLD and SVM. That is, if we interpret the weight vector of this slowest feature as the weight vector of a linear discriminator, this classifier achieves a performance of almost 100% on deciding which class of input stimuli has caused these unseen network state trajectories, even in an "anytime" manner, i.e., during the whole duration of the trajectories.

Figure 9A shows the responses of SFA trained on a sequence of 500 trajectories corresponding to utterances of digits "one" and "two" of all 5 speakers. From the 100 available samples (2 digits $\times$ 5 speakers $\times$ 10 utterances) we have used 70 for training and kept the remaining 30 for testing. The response of the learned SFA to trajectories in response to three of these testing utterances for each of the two classes, as well as the mean SFA response over all 30 test utterances of each class, is shown in Figure 9A. It can be seen that, qualitatively, the performance decreases compared to the case where only a single speaker is used (see Figure 8B). No single feature extracts the class information alone, but significant *What*-information is still represented: First, the slowest feature y_1 responds more strongly to trajectories corresponding to samples with digit "one". Second, feature y_3 responds with negative values only for trajectories in response to digit "two", whereas for those of digit "one" it consistently has an initial positive response. Again feature y_1 has the smallest angular distance to the FLD direction, even if it is larger than in Figure 8B.

Similarly, we can apply SFA to a sequence of trajectories in response to utterances of speakers 1 and 2 (but now with all 10 digits) try to extract information about the speaker feature, independent of the spoken

[9]Note that the absolute scale of different readout traces relative to each other is arbitrary since only the direction of the weight vectors are relevant. In this presentation all three weight vectors are normalized to length 1, in order to be comparable to each other.

nonlinear expansion	# dimensions	classifier performance
none (stimulus)	20 (10)	75%
quadratic	65	81%
cubic	285	83%
laminar circuit	560 (100)	88%

Table 1: Performance values of a linear classifier trained on the slow features in response to different nonlinear expansions of the input, for the speaker recognition experiment in Figure 9B. The nonlinearity implicitly provided the laminar circuit is compared to a quadratic and cubic expansion of the stimulus, as well as to the naked stimulus. The second column gives the dimensionality of the state space provided by the respective nonlinear projection. The numbers in brackets denote the effective dimensions used to train SFA, after PCA is applied (see appendix B.3.3). The quadratic (cubic) kernel contains all monomials up to degree 2 (3) of the 10 effective stimulus dimensions (Wiskott and Sejnowski, 2002). Performance values are evaluated by 10-fold cross validation.

digit. Now there are 200 available samples (10 digits $\times$ 2 speakers $\times$ 10 utterances), where we have used 140 for training and kept the remaining 60 for testing. Figure 9B shows the responses of the learned SFA to 3 trajectories of these test utterances, as well as the average SFA response over all 60 test trajectories. Due to the increased number of different samples for each class (for each speaker there are now 10 different digits) this task is more difficult than the speaker-independent digit recognition. No single slow feature extracts *What*-information alone; the closest feature to the FLD is feature y_3. To some extent also y_4 extracts discriminative information about the stimulus.

In these experiments, the separation between the classes (expressed by the first term in (25)) obviously decreases compared to the single-speaker case. In such a situation where the distance between the class means is very small, the tendency to extract the trajectory class itself as a slow feature becomes negligible. In that case the theory predicts that SFA tries to distinguish each individual trajectory due to the decorrelation constraint, and clusters similar trajectories because of the finite (linear) function space. It can be seen in Figure 8 that higher-order features start to discriminate between different samples of the same class. This demonstrates that multiple SFA responses are important and collectively convey discriminative information about the class of the trajectory currently being presented, and that in these examples one should view SFA as a powerful preprocessing stage for a subsequent classification, rather than a classifier itself.

It is important to note that the different classification results in Figures 9A and 9B are not obtained due to a different temporal order of the trajectories within the training input (i.e., whether the speaker is varying more slowly than the digit, or vice versa), but due to the use of a different training set of trajectories. The result of SFA does not depend on the temporal order of the trajectories within the training input because of the intermittent zero phases, and is therefore completely determined by the training set of trajectories.

The performance of a linear classifier trained on the 5 slowest features in response to all available test trajectories to predict the class label of the stimulus is 90% for the speaker-independent digit recognition (Figure 9A) and slightly lower (88%) for the digit-independent speaker recognition (Figure 9B). If linear SFA is applied directly to the 20-dimensional trajectories obtained by low-pass filtering the stimulus spike trains directly, the same classifier achieves a performance of about 75%. This indicates that the circuit provides a useful nonlinear combination of input components. Table 1 compares these performance values to different nonlinear expansions of the stimulus for this experiment. It can be seen that the laminar circuit yields a better performance than a cubic kernel, even though the number of dimensions already have the same order of magnitude. Other than the quadratic and cubic expansion, which are static, the circuit additionally provides a temporal integration of the stimulus which might provide a significant performance improvement in this case.

Note again that these are performance values for an unsupervised "anytime" speech recognition task. A comparable performance has been achieved in (Maass et al., 2004) on a different task (digit "one" against all other digits) on the encoding by (Hopfield and Brody, 2000) by training the readout weights with a linear SVM. The performance values reported in (Verstraeten et al., 2005) are not for "anytime" speech recognition in the sense that snapshots across different time points of network trajectories are used for training the readout, but a majority vote across different classifiers trained at different time points is used to predict the currently spoken digit. If the decision about which stimulus class has been presented should

not be made "anytime", but only at the end of each stimulus/trajectory, almost perfect performance can be achieved by integrating the slow features during the duration of a trajectory, i.e., by accumulating evidence for or against a given speaker or digit.

Finally, we note that the qualitative performance of SFA (i.e., how "good" the features look, or in which order the features are extracted) depends on the smoothness of the trajectories that are used for training. The circuit model of (Häusler and Maass, 2007) typically shows a bursting behavior, which is mostly due to the short-term dynamics of synapses. Thus the performance of SFA can even be improved by using a circuit model that generates smoother trajectories of network states. Also, we obtain similar results if we apply SFA directly on a sequence of the high-dimensional analog cochleagrams shown in Figure 8A.

4 Discussion

4.1 SFA as a principle for neural computation

We have shown in this article that Slow Feature Analysis (SFA) can in principle be used for learning *unsupervised* (or implicitly supervised) linear discrimination. SFA (Wiskott and Sejnowski, 2002) belongs to a family of algorithms for learning invariant representations from *temporal* input sequences, which maximize the "slowness" of their output signals (e.g., Földiak, 1991; Mitchison, 1991; Becker and Hinton, 1992; Stone and Bray, 1995). This objective is based on the assumption that signals that encode invariant representations, such as the location or identity of an object in the visual field, vary on a much slower time scale than raw sensory signals, such as the intensity of the visual input at a single fixed point on the retina, for example. Therefore, the extraction of slow features of the quickly varying input signal is likely to yield invariant properties of this input signal. The unique aspect about SFA is its appealing formulation as an eigenvalue problem in the covariance matrices of the (possibly nonlinearly expanded) multi-dimensional input signal.

This formulation has allowed us to establish a relationship between this unsupervised learning rule and a powerful supervised method for classification, Fisher's Linear Discriminant (FLD), which can be expressed as a similar eigenvalue problem. In particular, we have demonstrated that by converting the input to a classification problem (two labeled point sets) into an unlabeled time series in a special way, SFA is able to closely resemble the result of FLD on this classification problem. More precisely, if two consecutive points in the time series are likely to be chosen from the same class (i.e., the switching probability p between the classes is low), both methods yield similar projection directions, which can be interpreted as hypotheses of linear discriminators (i.e., separating hyperplanes). Due to this tendency that temporally contiguous points are from the same class SFA is able to learn to become invariant to different points within a class, but to respond differently for points from different classes, i.e., to extract the class as a slowly varying feature.

In this paper we have basically considered three cases of application of SFA for pattern recognition: (i) point discrimination, (ii) trajectory discrimination with different means, and (iii) trajectory discrimination with identical means. In case (i), the point discrimination, the class membership is implicitly encoded in the temporal sequence of samples that serves as input to SFA. The optimal response is a piecewise constant function during periods where points from the same class are presented, and for a linear function, converges to the result of FLD on the original classification problem. Regarding case (ii), the trajectory discrimination with different means, we have analyzed how the SFA objective changes if it is applied to a time series that consists of a sequence of such trajectories of training examples instead of individual points that are independently chosen at each time step. More precisely, we have considered a trajectory classification problem, which consists of sets of point sequences rather than sets of individual points. We generated a time series from this classification problem by randomly choosing trajectories from these two sets and by concatenating them into a single sequence. We found that for such a sequence of sufficiently long trajectories the result of SFA becomes independent of the class switching probability between two successive trajectories, thus of the temporal order of the trajectories within the time series. Applied to such a time series, the optimization problem of SFA can be viewed as a composition of two effects: the tendency to extract the trajectory class as a slow feature and the tendency to produce a smooth response during individual trajectories. The first effect can be described by the scatter matrices of the FLD, whereas the second effect depends on the temporal correlations (with time lag 1) of the trajectories.

Case (iii) occurs when the class means are so close together that they are almost identical. In this case the effect of the FLD vanishes. If the trajectories are interleaved with zero (or noise) input the optimal solution to SFA would be to respond with a half sine wave to each trajectory. For that case, (Wiskott and Sejnowski, 2002; Wiskott, 2003) explain the emergence of discriminative information in the SFA responses by the decorrelation constraint: a feature that responds with different amplitudes for different patterns also varies slowly and can still be decorrelated to other features that exhibit the same response for each pattern. Thus, with an infinite function space SFA always produces a feature that responds with a different amplitude for each individual pattern. If the available function space is limited (e.g., linear, as in our case) SFA might cluster similar trajectories, e.g., those that belong to the same class, by responding to them with similar amplitude.

In the context of biologically realistic neural circuits this ability of an unsupervised learning mechanism is of particular interest, because it could enable readout neurons, which typically receive inputs from a large number of presynaptic neurons of the circuit, to extract from the trajectory of network states information about the stimulus that has caused this particular sequence of states – without any "teacher" or reward. In previous simulation studies of neural circuit models, so far training of readouts of biological microcircuits has mostly been performed in a supervised manner (Maass et al., 2002, 2004; Legenstein et al., 2005) or in a reward-based trial-and-error setting (Legenstein et al., 2008).

We have tested the potential biological relevance of this learning principle in computer simulations of a quite realistic model of a cortical microcircuits (Häusler and Maass, 2007). More precisely, we have tested whether SFA would enable projection or "readout" neurons to learn without supervision to detect and discriminate salient input streams to the microcircuit. These readouts were modelled as linear neurons, i.e., we have neither used a particular nonlinear expansion (Wiskott and Sejnowski, 2002), which would have likely suffered from the curse of dimensionality when applied to these high-dimensional trajectories, nor an explicit kernel (Bray and Martinez, 2003). Rather, we have taken advantage of the kernel property of the circuit itself, which provides intrinsic nonlinear combinations of input components by its recurrent connections, and thereby boosts the expressive power of a subsequent linear readout.

In particular, we have shown that SFA is able to detect a repeating spike pattern within a continuous stream of Poisson input with the same firing statistics in an unsupervised manner. Furthermore, we demonstrated that the recognition of isolated spoken digits is possible using a biologically realistic preprocessing for audio samples. SFA was able to almost perfectly discriminate between two digits of a single speaker, and to a lesser extent also to extract information about the spoken digit independent of the speaker as well as the speaker independent of the spoken digit.

The laminar circuit transforms the input spike trains in three different ways. First, it provides a nonlinear expansion of the input by projecting it into a higher dimensional space through its recurrent connections. We have shown in one of the speech discrimination tasks that the circuit significantly improves the performance of a subsequent linear SFA readout compared to the case where this readout is directly applied to the stimulus spike trains. Moreover, the circuit performs better than a static quadratic or cubic expansion of the stimulus (see Table 1). The second effect of the circuit is to provide temporal integration. While this may be benefitial in the spoken digits tasks, it certainly decreases the performance in the pattern detection task because it makes the response of the circuit at the beginning of a pattern depend on the noise input immediately before (see Figure 7C). The third effect is the inherent noise of the network, i.e., its property to respond differently each time the same stimulus is presented. This noise models the background synaptic input *in vivo* (Destexhe et al., 2001). SFA should perform better if this intrinsic noise is low.

We find that the response of the learned SFA readouts to a sequence of test trajectories contains both *What*- and *Where*-information, i.e., they encode the class of the trajectory currently presented (pattern identity) as well as the current position within a trajectory (location within a pattern). This is in agreement with the objective of SFA, because both the location and identity vary on a slower time scale than the raw sequence of network states. The extracted features tend to be sections of sine waves, which are the slowest possible responses under the constraints of unit variance and decorrelation. Features encoding *Where*-information usually detect the presence of the trajectories (and encode the current position within the trajectory) independent of their identity and respond with similar shapes to each trajectory. Such information is very useful for neural systems, since it allows them to keep track of time relative to a stimulus onset or the initiation of a motor response (Buonomano and Mauk, 1994; Buonomano and Maass, 2009). The fact that such timing information becomes automatically available through unsupervised SFA could

in fact point to a general advantage of coding and computing with trajectories of firing states, rather than with single firing states (as many classical theories of neural computation propose). Obviously the ability to keep track of time on the time scale of a few hundred ms is essential for biological organisms, e.g., for motor control. In contrast, features encoding *What*-information discriminate between different types of trajectories and respond differently for different classes of trajectories. The response vector defined by the slowest features at a particular point in time takes on specific directions for each trajectory class; we have found that the average response vectors of different classes are around 90 degrees apart. These properties of SFA have been theoretically predicted in a thorough analysis of this algorithm (Wiskott, 2003). In (Wiskott and Sejnowski, 2002), they have been also found in computer simulations, where a hierarchical SFA network has been trained with a sequence of short one-dimensional trajectories. There, the particular organization of sequential quadratic SFA stages provides the nonlinear function space, from which the function is chosen that generates the slowest possible output from the input signal. In our case this function space is implicitly given by the nonlinearity of the circuit.

In contrast to the results for classification problems on point sets, due to the temporal structure of trajectories a single SFA readout of a cortical microcircuit might not extract the class of network trajectories explicitly, but usually a mixture of both *What*- and *Where*-information. This is what we had expected from our theoretical analysis, which suggested that there is a trade-off between the tendency to separate different classes and the tendency to respond as smoothly as possible during individual trajectories. Moreover, as the distance between the class means decreases, the separation tendency becomes negligible, and SFA tries to distinguish all individual trajectories. However, the slowest features span a subspace where the trajectories are nicely separated, thereby rendering SFA a powerful preprocessing stage by improving the computational performance of a subsequent classification. Furthermore, the results show that SFA readouts are able to distinguish between different stimulus classes in an "anytime" manner, i.e., they provide the correct classification already before the trajectory has ended. This makes the information about the stimulus available to later processing or decision making stages not only after a trajectory has settled into an attractor, but already while the stimulus is still being presented.

In these circuit simulations, SFA responds with amplitudes of different sign to patterns of different classes, and even generalizes this behavior to unseen test examples. We argue that the function space that is implicitly provided by the cortical microcircuit together with the linear SFA readouts might just have the property that different trajectories yield the same responses if they are similar enough. More precisely, it might correspond to an imperfect kernel that maps similar input patterns (patterns that are likely to be from the same class) into similar trajectories, and sufficiently distinct input patterns to trajectories that are significantly separated. Previous studies (e.g., Legenstein and Maass, 2007) suggest that if such circuits operate in a regime called *edge of chaos*, they might have this desired property.

Furthermore, our theory predicts and our experiments show that the ability of SFA to discriminate between different classes of trajectories is strongly influenced by the temporal correlations of the trajectories, as explained by the temporal covariance matrix with time lag 1. It would be interesting to investigate the effect of different magnitudes of these correlations, e.g., by comparing the effect of different sampling frequencies (we use a quite short sampling time in our examples).

4.2 Relation to preceding work

Slow Feature Analysis has already been applied for unsupervised pattern recognition in (Berkes, 2005b, 2006), where SFA has been used to discriminate between handwritten digits. There the SFA objective is reformulated to optimize slowness for time series consisting of just two patterns, averaged over all possible pairs of patterns. The idea is to search for functions that respond similarly to patterns of the same class, and therefore ignore the transformation between the individual patterns. The optimization (1) in (Berkes, 2006) is performed over the set of time derivatives of all possible pairs of samples of a class,

$$\min \quad \Delta(y_j) = a \cdot \sum_{c=1}^{C} \sum_{\substack{k,l=1 \\ k<l}}^{N_c} \left(g_j(\mathbf{x}_k^c) - g_j(\mathbf{x}_l^c)\right)^2, \tag{27}$$

under the constraints of zero mean, unit variance, and decorrelation, where C is the number of classes, N_c is the number of samples of class c, $\mathbf{x}_k^c$ is the k-th sample of class c, and a is a normalization constant

dividing by the number of all possible pairs. Obviously, the functions g_j that minimize (27) are ones which are constant for all patterns belonging to the same class, in which case the objective function is zero. As a consequence, patterns from the same class will cluster in the feature space formed by the output signals of the $(C-1)$ slowest functions g_j, where classification can be performed using simple techniques (Berkes, 2005b, 2006).

One problem with this approach is that it is often computationally intractable to consider all pairs of patterns, since the number of pairs grows very fast with the number of patterns. Furthermore, it might be implausible to have access to such an artificial time series, e.g., from the perspective of a readout of a cortical microcircuit which receives input on-the-fly. We take a different approach and apply the standard SFA algorithm to a time series consisting of randomly selected patterns of the classification problem, where we switch the class of the current pattern at each time step with a certain probability. We have found that if this switching probability p is low SFA extracts features which separate the classes and finds approximately the same subspace as Fisher's Linear Discriminant. In particular, we have demonstrated the dependence of the deviation on p: as p goes to zero, the weight vector of SFA converges to the weight vector of FLD. Note that with this approach perfect equivalence between SFA and FLD cannot be reached because the time series would have to consist only of transitions within a class, but at the same time contain patterns from all classes, which is not possible. In this hypothetical case the SFA problem would become equivalent to the reformulated objective in (Berkes, 2005b, 2006). In (Berkes, 2005a) the author applied a nonlinear version of Fisher's Discriminant to the same handwritten digits dataset as in (Berkes, 2005b, 2006) (using a fixed polynomial expansion of the input as a kernel) and achieved a similar result, but, however, no relationship between the two methods was shown.

In (Franzius et al., 2008) the authors show that a hierarchical network of quadratic SFA modules can extract the identity of objects from an image sequence presenting these objects at continuously changing positions, sizes, and viewing angles. They create the input sequence in a similar way as we do: after each time step, the object identity is switched with a low probability. The resulting features extracted by SFA contain information about the identity of the object currently shown, as well as the current position, size and rotation angles of the object. However, this information is usually not made explicit in the sense that a single slow feature codes for exactly one "configuration variable" (such as object identity or position), rather, each such variable is distributed over multiple slow features. The original variables, however, can be recovered from the slowest features using linear regression or simple classifiers with high accuracy. The tendency for this "linear mixing" of information increases as the input sequence gets more and more complex (i.e., contains more transformations of the same object). We also find this effect in our experiments: in the experiment where we discriminated between spoken digits of a single speaker (Figure 8) the slowest feature extracted the class information explicitly, whereas in the experiment where more speakers were used (Figure 9A) this information was distributed over multiple features. In principle, one can view the nonlinear expansion of the image sequence that belongs to a single object presentation (i.e., between two object switching events) as a particular trajectory in response to this object. Different trajectories for the same object vary in the specific sequence of poses of that object during a particular presentation phase. In this sense, SFA is trained on a sequence of trajectories, each resulting from a specific presentation of a particular object. According to (Franzius et al., 2008) the classifier performance for extracting the object identity is maximized if all other variables are made very fast. This is in agreement with our theory because faster configuration variables produce weaker temporal correlations of these image trajectories. This means that SFA more closely approximates the result of FLD on these images.

This work in (Franzius et al., 2008) offers one explanation how the visual system learns invariant object recognition from the temporal statistics of the input stimuli: images that occur in immediate succession tend to belong to the same object. In fact, a considerable amount of work has been done that investigates temporal slowness as a computational principle in the visual system. In (Berkes and Wiskott, 2003) quadratic SFA (i.e., linear SFA in the expanded input of all polynomials of degree 2 of the original input dimensions) has been applied to natural image sequences and the learned quadratic forms have been interpreted as receptive fields (Berkes and Wiskott, 2006). These resulting receptive fields resemble many properties of complex cells, such as their Gabor-like shape, shift invariance, or direction selectivity. Furthermore, when presented with a visual input sequence that is generated by the movement of a simulated rat in a virtual environment, SFA has been shown to reproduce the spatial firing patterns of place cells, head-direction cells, spatial-view cells, and grid cells (Franzius et al., 2007a). Depending on the movement statistics of this simulated rat

different types of invariances are learned (e.g., the head direction independent of the current position in the environment). To obtain the final response characteristics of these cell types, however, an additional sparse coding stage has been incorporated (Franzius et al., 2007b), which extracts the representations of single cells from the more distributed representations resulting from SFA. Using a slightly different slowness objective, similar invariance properties of the visual system have been found in (Einhäuser et al., 2002; Wyss et al., 2006).

In this work we have trained the readouts of our cortical microcircuit with the standard batch algorithm of SFA. This has the advantage that there are no parameters which have to be tuned for a specific problem. Since SFA is based on an eigenvalue problem it finds the solution in a single iteration and has no convergence problems (e.g., to be trapped in local minima). In biological systems, however, processing has to be performed on-the-fly, and therefore learning rules that optimize temporal stability in an online manner are of particular interest. Several computational models exist that are based on this slowness principle and that show how invariances in the visual system can be learned through a variety of Hebbian-like learning rules (Földiak, 1991; Wallis and Rolls, 1997; Wyss et al., 2006; Masquelier and Thorpe, 2007). In this article we do not propose a biologically realistic learning rule, rather, we investigate the properties of one well-known algorithm, Slow Feature Analysis, out of this family of optimization methods based on the slowness principle and analyze its unsupervised discrimination capabilities. A recent paper demonstrates that this learning rule can in principle be implemented by a spiking neuron with a form of STDP (Sprekeler et al., 2007). Although this result is purely analytical and has yet to be verified in computer simulations, it supports the hypothesis that the objective of slowness is an important ingredient in the unsupervised learning mechanisms of biological systems. In fact, STDP has been successfully applied to robust online unsupervised detection of repeating spatiotemporal spike patterns hidden within spike trains of the same firing statistics (Masquelier et al., 2009). Moreover, it has been shown that spiking neurons equipped with a special form of STDP which receives a global reward signal (Izhikevich, 2007) can learn to discriminate between different trajectories of firing states using reinforcement learning (Legenstein et al., 2008).

SFA is not only inspired by the slowness principle for learning invariances, but might also be motivated by information-theoretic principles, such as the Information Bottleneck (IB) method (Tishby et al., 1999), or Independent Component Analysis (ICA) (Hyvärinen et al., 2001). In (Creutzig and Sprekeler, 2008) a relationship is shown between SFA and the IB method for predictive coding, which optimizes the objective to compress the information of the past into the current state of a system, such that as much information as possible about the future is preserved. In other words, it minimizes $I(\text{past}; \text{state}) - \beta I(\text{state}; \text{future})$ with some trade-off parameter β. It turns out that for the case of one-time-step prediction and of a linear system with Gaussian noise this problem becomes equivalent to linear SFA. On the other hand, ICA tries to uncover statistically independent signals from an observed linear mixture of these signals. Blaschke et al. (2006, 2007) show that for a particular measure of independence, which involves the temporal correlations with a time delay of one time step, ICA becomes formally equivalent to linear SFA. Finally, Turner and Sahani (2007) provides a probabilistic interpretation for SFA, where it is assumed that the observed time series $\mathbf{x}$ is generated by a linear mixture of latent variables (the slow features y_i). The mixing matrix $\mathbf{W}$ is recovered by maximizing the likelihood function. This attractive formulation has the advantages that constraints and extensions can be included in the model in a very natural way, and that noise and missing data in the input are handled elegantly by this probabilistic setup. These results establish an interesting connection between the slowness objective and both probability and information theory and further demonstrate the power of the elegant algorithm of linear SFA.

4.3 Conclusion

Summarizing, we have established a theoretical basis that explains when Slow Feature Analysis can be expected to have emergent pattern discrimination capabilities. Both our theoretical results and our computer simulations suggest that Slow Feature Analysis – and more generally the concept of slowness or temporal stability – could be a powerful mechanism for extracting temporally stable information from trajectories of network states of biological circuits without supervision, and hence an important ingredient for spatiotemporal processing in cortical networks (Buonomano and Maass, 2009). In particular, it provides a basis for explaining how brains can arrive at stable percepts in spite of continuously changing network states in a completely unsupervised way.

A Derivation of the relationship between the SFA and FLD objective

A.1 Derivation for the case of two classes

In this section we derive the expressions for the temporal covariance matrices $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ and $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ of the SFA objective (2) for the two-class case in terms of the within-class and between-class scatter matrices of the FLD objective (4), $\mathbf{S}_W$ and $\mathbf{S}_B$, for the particular method of time series generation described in the main text.

Assume we are given two disjoint point sets $S_1, S_2 \subset \mathbb{R}^n$,

$$S_1 := \{\mathbf{x}_i^1 | i = 1, \ldots, N\}, \tag{28}$$

$$S_2 := \{\mathbf{x}_j^2 | j = 1, \ldots, N\}, \tag{29}$$

where $\mathbf{x}_i^1$ and $\mathbf{x}_j^2$ denote the data points of class 1 and 2, respectively, and N denotes the number of data points for each of the two classes. Both point sets can be characterized by their mean vectors ($\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$) and covariance matrices ($\boldsymbol{\Sigma}_1$, $\boldsymbol{\Sigma}_2$), given by

$$\boldsymbol{\mu}_1 = \langle \mathbf{x}_i^1 \rangle = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i^1, \tag{30}$$

$$\boldsymbol{\mu}_2 = \langle \mathbf{x}_j^2 \rangle = \frac{1}{N} \sum_{j=1}^{N} \mathbf{x}_j^2, \tag{31}$$

and

$$\boldsymbol{\Sigma}_1 = \frac{1}{N} \left\langle \left(\mathbf{x}_i^1 - \boldsymbol{\mu}_1\right) \left(\mathbf{x}_i^1 - \boldsymbol{\mu}_1\right)^T \right\rangle = \sum_{i=1}^{N} \mathbf{x}_i^1 {\mathbf{x}_i^1}^T - \boldsymbol{\mu}_1 \boldsymbol{\mu}_1^T, \tag{32}$$

$$\boldsymbol{\Sigma}_2 = \frac{1}{N} \left\langle \left(\mathbf{x}_j^2 - \boldsymbol{\mu}_2\right) \left(\mathbf{x}_j^2 - \boldsymbol{\mu}_2\right)^T \right\rangle = \sum_{i=1}^{N} \mathbf{x}_j^2 {\mathbf{x}_j^2}^T - \boldsymbol{\mu}_2 \boldsymbol{\mu}_2^T. \tag{33}$$

The within-class and between-class scatter matrices of Fisher's Linear Discriminant are then given by (see (6) and (5))

$$\begin{aligned} \mathbf{S}_W &= \sum_{i=1}^{N} (\mathbf{x}_i^1 - \boldsymbol{\mu}_1)(\mathbf{x}_i^1 - \boldsymbol{\mu}_1)^T + \sum_{j=1}^{N} (\mathbf{x}_j^2 - \boldsymbol{\mu}_2)(\mathbf{x}_j^2 - \boldsymbol{\mu}_2)^T \\ &= N \left(\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2\right) \end{aligned} \tag{34}$$

and

$$\mathbf{S}_B = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T. \tag{35}$$

We now generate a time series $\mathbf{x}_t$ from these two input point sets S_1 and S_2 as described in the main text, using the Markov model in Figure 2. We can now express the mean and covariance of this time series $\mathbf{x}_t$ in terms of $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$, $\boldsymbol{\Sigma}_1$, and $\boldsymbol{\Sigma}_2$. For the mean we get

$$\boldsymbol{\mu} := \langle \langle \mathbf{x}_t \rangle_t \rangle = \langle \langle \mathbf{x}_t \rangle \rangle_t = \frac{1}{T} \sum_{t=1}^{T} \langle \mathbf{x}_t \rangle = \langle \mathbf{x}_t \rangle = \frac{1}{2}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2, \tag{36}$$

because the stationary distribution of the Markov model in Figure 2 is $\boldsymbol{\pi} = \left(\frac{1}{2}, \frac{1}{2}\right)$ (10). More generally, the mean of the time series is given by the weighted mean between the two class means, weighted by the probability that a point is drawn from the corresponding class. Note the different expectation operators: $\langle \cdot \rangle_t$ denotes the temporal average over the time series $\mathbf{x}_t$, whereas the average over all possible time series $\mathbf{x}_t$ generated from S_1 and S_2 is given by $\langle \cdot \rangle$. That is, $\langle \langle \mathbf{x}_t \rangle_t \rangle$ refers to the temporal average of a specific time series $\mathbf{x}_t$, averaged over all possible realizations of $\mathbf{x}_t$, whereas $\langle \langle \mathbf{x}_t \rangle \rangle_t$ refers to the temporal average

of the expected value of $\mathbf{x}_t$ at a specific time step t. Since this Markov model yields a stationary random process, we can exchange the expectation operators. Similarly, the expected covariance matrix is given by

$$\begin{aligned}\boldsymbol{\Sigma} := \left\langle \langle (\mathbf{x}_t - \boldsymbol{\mu})(\mathbf{x}_t - \boldsymbol{\mu})^T \rangle_t \right\rangle &= \frac{1}{T}\sum_{t=1}^{T} \langle \mathbf{x}_t \mathbf{x}_t^T \rangle - \boldsymbol{\mu}\boldsymbol{\mu}^T = \langle \mathbf{x}_t \mathbf{x}_t^T \rangle - \boldsymbol{\mu}\boldsymbol{\mu}^T \\ &= \frac{1}{2}(\boldsymbol{\Sigma}_1 + \boldsymbol{\mu}_1\boldsymbol{\mu}_1^T) + \frac{1}{2}(\boldsymbol{\Sigma}_2 + \boldsymbol{\mu}_2\boldsymbol{\mu}_2^T) - \boldsymbol{\mu}\boldsymbol{\mu}^T \\ &= \frac{1}{2}\boldsymbol{\Sigma}_1 + \frac{1}{2}\boldsymbol{\Sigma}_2 + \frac{1}{4}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T, \end{aligned} \tag{37}$$

where in the last step we used (36). Note that the covariance matrix of the time series is not only determined by the covariance matrices of the two classes, but also by their spatial separation as expressed by (37). We assume without loss of generality that $\boldsymbol{\mu} = \mathbf{0}$, i.e., $\boldsymbol{\Sigma} = \langle \mathbf{x}_t \mathbf{x}_t^T \rangle$.

Next we consider the covariance matrix of time derivatives. For the expected covariance matrix we write

$$\begin{aligned}\left\langle \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \right\rangle &= \frac{1}{T-1}\sum_{t=2}^{T} \left\langle (\mathbf{x}_t - \mathbf{x}_{t-1})(\mathbf{x}_t - \mathbf{x}_{t-1})^T \right\rangle \\ &= \left(\langle \mathbf{x}_t \mathbf{x}_t^T \rangle + \langle \mathbf{x}_{t-1} \mathbf{x}_{t-1}^T \rangle \right) - \left(\langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle + \langle \mathbf{x}_t \mathbf{x}_{t-1}^T \rangle \right). \end{aligned} \tag{38}$$

The two terms in the first part of (38) consist of covariances between input samples of the same time index and can be rewritten as (using (37))

$$\langle \mathbf{x}_t \mathbf{x}_t^T \rangle \approx \frac{1}{2}\left(\boldsymbol{\Sigma}_1 + \boldsymbol{\mu}_1\boldsymbol{\mu}_1^T\right) + \frac{1}{2}\left(\boldsymbol{\Sigma}_2 + \boldsymbol{\mu}_2\boldsymbol{\mu}_2^T\right) \tag{39}$$

$$\langle \mathbf{x}_{t-1} \mathbf{x}_{t-1}^T \rangle = \langle \mathbf{x}_t \mathbf{x}_t^T \rangle. \tag{40}$$

Because of the stationarity of $\mathbf{x}_t$ the covariance matrix is independent of a time shift. The approximation in (39) holds for large T, since the summation in (37) contains T terms and the summation in (38) contains $T-1$ terms. Similarly, the two terms in the second part of (38) consist of the cross-covariances between adjacent time steps. If the classes of $\mathbf{x}_t$ and $\mathbf{x}_{t-1}$ are fixed, then $\mathbf{x}_t$ is chosen independently of $\mathbf{x}_{t-1}$ and we can split up the expectation operator $\langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle = \langle \mathbf{x}_{t-1} \rangle \langle \mathbf{x}_t^T \rangle$ into the product of the two class means. Considering the 4 possible class transitions we write

$$\langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle = \frac{1}{2}(1-p)\boldsymbol{\mu}_1\boldsymbol{\mu}_1^T + \frac{1}{2}(1-p)\boldsymbol{\mu}_2\boldsymbol{\mu}_2^T + \frac{1}{2}p\boldsymbol{\mu}_1\boldsymbol{\mu}_2^T + \frac{1}{2}p\boldsymbol{\mu}_2\boldsymbol{\mu}_1^T \tag{41}$$

$$\langle \mathbf{x}_t \mathbf{x}_{t-1}^T \rangle = \langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle^T = \langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle. \tag{42}$$

Plugging (39) to (42) back into (38) yields

$$\left\langle \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t \right\rangle = \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2 + p(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T. \tag{43}$$

In equations (37) and (43) we have expressed the covariance matrix of the time series $\mathbf{x}_t$, $\langle \mathbf{x}\mathbf{x}^T \rangle_t$, and the covariance matrix of its time derivatives, $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$, in terms of the means and covariances of the two point sets of the FLD problem. We repeat these here for clarity (we drop the expectation $\langle \cdot \rangle$ for convenience):

$$\langle \mathbf{x}\mathbf{x}^T \rangle_t = \frac{1}{2}\boldsymbol{\Sigma}_1 + \frac{1}{2}\boldsymbol{\Sigma}_2 + \frac{1}{4}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T, \tag{44}$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t = \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2 + p(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T. \tag{45}$$

Remember that p is the transition probability between the classes, according to Figure 2. Recalling the definition of $\mathbf{S}_W$ (34) and $\mathbf{S}_B$ (35), we finally obtain the result

$$\langle \mathbf{x}\mathbf{x}^T \rangle_t = \frac{1}{2N}\mathbf{S}_W + \frac{1}{4}\mathbf{S}_B, \tag{46}$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t = \frac{1}{N}\mathbf{S}_W + p \cdot \mathbf{S}_B. \tag{47}$$

A.2 Derivation for the case of more than two classes

In this section we derive the expressions for the temporal covariance matrices $\langle \mathbf{x}\mathbf{x}^T \rangle_t$ and $\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t$ of the SFA objective (2) for the general case of more than two classes in terms of the within-class and between-class scatter matrices of the FLD objective (4), $\mathbf{S}_W$ and $\mathbf{S}_B$, for the particular method of time series generation described in the main text. We proceed analogously to the previous section for the two-class case.

Assume we are given C disjoint point sets $S_c \subset \mathbb{R}^n, c = 1, \ldots, C$,

$$S_c := \{\mathbf{x}_i^c | i = 1, \ldots, N_c\}, \tag{48}$$

where $\mathbf{x}_i^c$ denote the data points of class c, and N_c denotes the number of data points in each class. Let $N_T = \sum_{c=1}^{C} N_c$ be the total number of points. Each of these point sets can be characterized by its mean vector and covariance matrix, given by

$$\boldsymbol{\mu}_c = \frac{1}{N_c} \sum_{i=1}^{N_c} \mathbf{x}_i^c, \tag{49}$$

$$\boldsymbol{\Sigma}_c = \frac{1}{N_c} \sum_{i=1}^{N_c} \mathbf{x}_i^c \mathbf{x}_i^{cT} - \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T. \tag{50}$$

The within-class and between-class covariance matrices of the Fisher Linear Discriminant in the multi-class case are defined by

$$\begin{aligned} \mathbf{S}_W &= \sum_{c=1}^{C} \sum_{i=1}^{N_c} (\mathbf{x}_i^c - \boldsymbol{\mu}_c)(\mathbf{x}_i^c - \boldsymbol{\mu}_c)^T \\ &= \sum_{c=1}^{C} N_c \boldsymbol{\Sigma}_c, \end{aligned} \tag{51}$$

and

$$\begin{aligned} \mathbf{S}_B &= \sum_{c=1}^{C} N_c (\boldsymbol{\mu}_c - \boldsymbol{\mu})(\boldsymbol{\mu}_c - \boldsymbol{\mu})^T \\ &= \sum_{c=1}^{C} N_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - N_T \boldsymbol{\mu} \boldsymbol{\mu}^T, \end{aligned} \tag{52}$$

where $\boldsymbol{\mu} = 1/N_T \sum_{c=1}^{C} N_c \boldsymbol{\mu}_c$ is the total mean of the input points.

We generate a time series $\mathbf{x}_t$ from these point sets as described in the main text, using the Markov model with states $S = \{1, 2, \ldots, C\}$ and transition probabilities

$$P_{ij} = \begin{cases} a \cdot \frac{N_j}{N_T} & \text{if } i \neq j, \\ 1 - \sum_{k \neq j} P_{ik} & \text{if } i = j, \end{cases} \tag{53}$$

for $i, j \in S$. First, we show that

$$\boldsymbol{\pi} = \left(\frac{N_1}{N_T}, \frac{N_2}{N_T}, \ldots, \frac{N_C}{N_T} \right) \tag{54}$$

is a stationary distribution of (53). This can be easily seen by verifying that for all $j \in S$,

$$\begin{aligned}\pi_j &= \sum_{i \in S} \pi_i P_{ij} \\ &= \sum_{i \neq j} \frac{N_i}{N_T} \cdot a \cdot \frac{N_j}{N_T} + \frac{N_j}{N_T} \left(1 - \sum_{k \neq j} a \cdot \frac{N_k}{N_T}\right) \\ &= a \cdot \frac{N_j}{N_T} \sum_{i \neq j} \frac{N_i}{N_T} + \frac{N_j}{N_T} - a \cdot \frac{N_j}{N_T} \sum_{k \neq j} \frac{N_k}{N_T} \\ &= \frac{N_j}{N_T}. \quad \square \end{aligned} \tag{55}$$

For the mean of the time series $\mathbf{x}_t$ we get, analogously to (36),

$$\langle \mathbf{x} \rangle_t = \sum_{c=1}^{C} \pi_c \boldsymbol{\mu}_c = \frac{1}{N_T} \sum_{c=1}^{C} N_c \boldsymbol{\mu}_c. \tag{56}$$

Note that for the particular choice of (53) the mean of the time series becomes equal to the total mean of the input points. Similarly, we obtain for the covariance matrix

$$\begin{aligned}\langle \mathbf{x}\mathbf{x}^T \rangle_t &= \sum_{c=1}^{C} \pi_c \left(\boldsymbol{\Sigma}_c + \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T\right) - \boldsymbol{\mu}\boldsymbol{\mu}^T \\ &= \frac{1}{N_T} \sum_{c=1}^{C} N_c \boldsymbol{\Sigma}_c + \frac{1}{N_T} \sum_{c=1}^{C} N_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - \boldsymbol{\mu}\boldsymbol{\mu}^T \\ &= \frac{1}{N_T} \mathbf{S}_W + \frac{1}{N_T} \mathbf{S}_B. \end{aligned} \tag{57}$$

For the covariance of time derivatives we proceed analogously to the previous section and write

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t = \left(\langle \mathbf{x}_t \mathbf{x}_t^T \rangle + \langle \mathbf{x}_{t-1} \mathbf{x}_{t-1}^T \rangle\right) - \left(\langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle + \langle \mathbf{x}_t \mathbf{x}_{t-1}^T \rangle\right), \tag{58}$$

with

$$\langle \mathbf{x}_t \mathbf{x}_t^T \rangle = \sum_{c=1}^{C} \pi_c \left(\boldsymbol{\Sigma}_c + \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T\right) \tag{59}$$

$$\langle \mathbf{x}_{t-1} \mathbf{x}_{t-1}^T \rangle = \langle \mathbf{x}_t \mathbf{x}_t^T \rangle \tag{60}$$

$$\langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle = \sum_{i,j \in S} \pi_i P_{ij} \boldsymbol{\mu}_i \boldsymbol{\mu}_j^T \tag{61}$$

$$\langle \mathbf{x}_t \mathbf{x}_{t-1}^T \rangle = \langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle^T = \langle \mathbf{x}_{t-1} \mathbf{x}_t^T \rangle. \tag{62}$$

The last equation holds because $\pi_i P_{ij} = \pi_j P_{ji}$. Plugging (59) to (62) back into (58) yields

$$\begin{aligned}
\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t &= 2\sum_{c=1}^{C} \pi_c \boldsymbol{\Sigma}_c + 2\sum_{c=1}^{C} \pi_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - 2\sum_{i=1}^{C}\sum_{j=1}^{C} \pi_i P_{ij} \boldsymbol{\mu}_i \boldsymbol{\mu}_j^T \\
&= \frac{2}{N_T}\sum_{c=1}^{C} N_c \boldsymbol{\Sigma}_c + \frac{2}{N_T}\sum_{c=1}^{C} N_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - \frac{2}{N_T}\sum_{c=1}^{C} N_c P_{cc} \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - \frac{2}{N_T}\sum_{c=1}^{C} N_c \boldsymbol{\mu}_c \left(\sum_{k \neq c} P_{ck} \boldsymbol{\mu}_k^T \right) \\
&= \frac{2}{N_T}\sum_{c=1}^{C} N_c \boldsymbol{\Sigma}_c + \frac{2}{N_T}\sum_{c=1}^{C} N_c \left(\sum_{k \neq c} a \cdot \frac{N_k}{N_T} \right) \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - \frac{2}{N_T}\sum_{c=1}^{C} N_c \boldsymbol{\mu}_c \left(\sum_{k \neq c} a \cdot \frac{N_k}{N_T} \boldsymbol{\mu}_k^T \right) \\
&= \frac{2}{N_T}\sum_{c=1}^{C} N_c \boldsymbol{\Sigma}_c + \frac{2a}{N_T^2}\sum_{c=1}^{C} N_c (N_T - N_c) \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - \frac{2a}{N_T^2}\sum_{c=1}^{C} N_c \boldsymbol{\mu}_c (N_T \boldsymbol{\mu} - N_c \boldsymbol{\mu}_c)^T \\
&= \frac{2}{N_T}\sum_{c=1}^{C} N_c \boldsymbol{\Sigma}_c + \frac{2a}{N_T} \left[\sum_{c=1}^{C} N_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - N_T \boldsymbol{\mu} \boldsymbol{\mu}^T \right] \\
&= \frac{2}{N_T} \mathbf{S}_W + \frac{2a}{N_T} \mathbf{S}_B.
\end{aligned} \tag{63}$$

Finally, we repeat the results (57) and (63),

$$\langle \mathbf{x}\mathbf{x}^T \rangle_t = \frac{1}{N_T} \mathbf{S}_W + \frac{1}{N_T} \mathbf{S}_B, \tag{64}$$

$$\langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle_t = \frac{2}{N_T} \mathbf{S}_W + \frac{2a}{N_T} \mathbf{S}_B, \tag{65}$$

and note the similarity to the results for the case of two classes in the previous section, (46) and (47).

A.3 Derivation for time series consisting of trajectories

In this section we derive the expressions given in equations (21) to (24), which reformulate the SFA objective for the case of time series consisting of trajectories of training examples rather than a sequence of individual points that are independently chosen.

First we consider the case where the time series $\mathbf{x}_t$ consists of multiple repetitions of a fixed trajectory $\tilde{\mathbf{t}} := (\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_{\tilde{T}})$ of length $\tilde{T}$, and random intervals of independently drawn "noise" samples drawn from the same distribution (characterized by mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$) as the $\tilde{\mathbf{x}}_k$. We assume without loss of generality that $\boldsymbol{\mu} = \langle \mathbf{x}_t \rangle_t = \mathbf{0}$. Furthermore, let T be the total length of $\mathbf{x}_t$, and let $\tilde{p}$ be the fraction of these T time steps of $\mathbf{x}_t$ that are occupied by the trajectory $\tilde{\mathbf{t}}$.

For the expected covariance matrix of $\mathbf{x}_t$ we get

$$\begin{aligned}
\left\langle \langle \mathbf{x}\mathbf{x}^T \rangle_t \right\rangle &= \frac{1}{T}\sum_{t=1}^{T} \langle \mathbf{x}\mathbf{x}^T \rangle \\
&= \frac{1}{T}\sum_{t \in \tilde{\mathbf{t}}} \langle \mathbf{x}\mathbf{x}^T \rangle + \frac{1}{T}\sum_{t \notin \tilde{\mathbf{t}}} \langle \mathbf{x}\mathbf{x}^T \rangle \\
&= \frac{\tilde{p}}{\tilde{T}}\sum_{k=1}^{\tilde{T}} \mathbf{x}_k \mathbf{x}_k^T + (1 - \tilde{p}) \boldsymbol{\Sigma} \\
&= \tilde{p} \tilde{\boldsymbol{\Sigma}} + (1 - \tilde{p}) \boldsymbol{\Sigma}.
\end{aligned} \tag{66}$$

We use the notation $t \in \tilde{\mathbf{t}}$ to denote that a time step t within the time series $\mathbf{x}_t$ belongs to an instance of $\tilde{\mathbf{t}}$. The matrix

$$\tilde{\boldsymbol{\Sigma}} := \frac{1}{\tilde{T}}\sum_{k=1}^{\tilde{T}} \tilde{\mathbf{x}}_k \tilde{\mathbf{x}}_k^T \tag{67}$$

is the covariance matrix of $\tilde{\mathbf{t}}$ with itself. Note that the average $\langle\cdot\rangle$ in (66) is over all realizations of $\mathbf{x}_t$ with a fixed trajectory $\tilde{\mathbf{t}}$. If we also average over different realizations of $\tilde{\mathbf{t}}$, the covariance becomes $\mathbf{\Sigma}$.

The covariance matrix of time derivatives can be written as

$$\begin{aligned}\left\langle\langle\dot{\mathbf{x}}\dot{\mathbf{x}}^T\rangle_t\right\rangle &= 2\left\langle\langle\mathbf{x}\mathbf{x}^T\rangle_t\right\rangle - \frac{1}{T-1}\sum_{t=2}^{T}\left(\langle\mathbf{x}_t\mathbf{x}_{t-1}^T\rangle + \langle\mathbf{x}_{t-1}\mathbf{x}_t^T\rangle\right)\\ &\approx 2\cdot\tilde{p}\tilde{\mathbf{\Sigma}} + 2\cdot(1-\tilde{p})\mathbf{\Sigma} - \tilde{p}\cdot\frac{\tilde{T}-1}{\tilde{T}}\cdot\tilde{\mathbf{\Sigma}}_t,\end{aligned} \tag{68}$$

where

$$\tilde{\mathbf{\Sigma}}_t := \frac{1}{\tilde{T}-1}\sum_{k=2}^{\tilde{T}}\left(\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_{k-1}^T + \tilde{\mathbf{x}}_{k-1}\tilde{\mathbf{x}}_k^T\right) \tag{69}$$

is the covariance of $\tilde{\mathbf{t}}$ with $\tilde{\mathbf{t}}$ delayed by one time step, i.e., it captures the temporal correlations of time lag 1. This matrix enters equation (68) with a coefficient $\tilde{p}(\tilde{T}-1)/\tilde{T}$, because each of the $\tilde{p}T/\tilde{T}$ trajectories of the time series contributes $\tilde{T}-1$ times the expected value (69) to the sum in the first line of (68). Note that all other temporal correlations of $\mathbf{x}_t$ apart from those caused by $\tilde{\mathbf{t}}$ are zero. The approximation in (68) is valid for large T (i.e., when $T/(T-1)\approx 1$).

Inserting (66) and (68) into the SFA objective (2) yields

$$J_{SFA}(\mathbf{w}) = \frac{\mathbf{w}^T\langle\dot{\mathbf{x}}\dot{\mathbf{x}}^T\rangle_t\mathbf{w}}{\mathbf{w}^T\langle\mathbf{x}\mathbf{x}^T\rangle_t\mathbf{w}} = 2 - \tilde{p}\cdot\frac{\tilde{T}-1}{\tilde{T}}\cdot\frac{\mathbf{w}^T\tilde{\mathbf{\Sigma}}_t\mathbf{w}}{\mathbf{w}^T\langle\mathbf{x}\mathbf{x}^T\rangle_t\mathbf{w}}, \tag{70}$$

and therefore

$$\min\quad J_{SFA}(\mathbf{w}) \Leftrightarrow \max\quad \frac{\mathbf{w}^T\tilde{\mathbf{\Sigma}}_t\mathbf{w}}{\mathbf{w}^T\langle\mathbf{x}\mathbf{x}^T\rangle_t\mathbf{w}}. \tag{71}$$

Next, we consider the two-class problem, where the time series $\mathbf{x}_t$ consists of a sequence of trajectories chosen from two classes $\mathcal{T}_1$ and $\mathcal{T}_2$. After each trajectory, the class of the next trajectory is switched with probability p, or left unchanged with probability $1-p$, according to the Markov model in Figure 2. These two trajectory sets can be characterized by their means, $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, and their covariances, $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$. Each of these quantities equals equations (30) to (33) evaluated for point sets S_1 and S_2 composed of the individual points of the trajectories in $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively. Furthermore, let $\tilde{\mathbf{\Sigma}}_t^{(1)}$ and $\tilde{\mathbf{\Sigma}}_t^{(2)}$ be the average temporal covariance matrices with time lag 1 for trajectories in $\mathcal{T}_1$ and $\mathcal{T}_2$, i.e.,

$$\tilde{\mathbf{\Sigma}}_t^{(c)} := \frac{1}{N(\tilde{T}-1)}\sum_{i=1}^{N}\sum_{k=2}^{\tilde{T}}\left[(\tilde{\mathbf{x}}_{i,k}^c-\boldsymbol{\mu}_c)(\tilde{\mathbf{x}}_{i,k-1}^c-\boldsymbol{\mu}_c)^T + (\tilde{\mathbf{x}}_{i,k-1}^c-\boldsymbol{\mu}_c)(\tilde{\mathbf{x}}_{i,k}^c-\boldsymbol{\mu}_c)^T\right] \tag{72}$$

where N is the number of trajectories in each of the sets $\mathcal{T}_1$ and $\mathcal{T}_2$ and $\tilde{T}$ is the length of a trajectory (we assume for simplicity that all trajectories have the same length). $\tilde{\mathbf{x}}_{i,k}^c$ is the k-th point in the i-th trajectory of class c. Note that in contrast to (69) the mean $\boldsymbol{\mu}_c$ is class-specific and different from zero.

The expected covariance matrix of the time series $\mathbf{x}_t$ is not affected by temporal correlations and is therefore equal to the case where individual points are chosen instead of trajectories (see equation (37)):

$$\left\langle\langle\mathbf{x}\mathbf{x}^T\rangle_t\right\rangle = \sum_{c=1}^{C}\pi_c\left(\mathbf{\Sigma}_c + \boldsymbol{\mu}_c\boldsymbol{\mu}_c^T\right), \tag{73}$$

where $C=2$ is the number of classes and $\pi_c = 1/2$ is the probability of being in state c for the stationary distribution of the Markov model. For the expected covariance matrix of time derivatives we write

$$\left\langle\langle\dot{\mathbf{x}}\dot{\mathbf{x}}^T\rangle_t\right\rangle = 2\left\langle\langle\mathbf{x}\mathbf{x}^T\rangle_t\right\rangle - \frac{1}{T-1}\sum_{t=2}^{T}\left(\langle\mathbf{x}_t\mathbf{x}_{t-1}^T\rangle + \langle\mathbf{x}_{t-1}\mathbf{x}_t^T\rangle\right). \tag{74}$$

The time series has length T and consists of $T/\tilde{T}$ trajectories. Therefore we can split up the sum in the second term on the right hand side of the last equation into $T - T/\tilde{T}$ contributions from transitions

$(\mathbf{x}_{t-1}, \mathbf{x}_t)$ within a trajectory and $T/\tilde{T} - 1$ contributions of switches between two temporally adjacent trajectories (i.e., at time points t when a new trajectory starts). Concerning the first part of the sum, each of the $T/\tilde{T}$ trajectories contributes $\tilde{T} - 1$ times the expected value $\sum_c \pi_c \left(\tilde{\mathbf{\Sigma}}_t^{(c)} + 2\boldsymbol{\mu}_c \boldsymbol{\mu}_c^T \right)$. The second part is determined according to the transition probabilities between the classes, similar to (41) and (61).

$$\left\langle \langle \dot{\mathbf{x}} \dot{\mathbf{x}}^T \rangle_t \right\rangle \approx 2 \sum_{c=1}^{C} \pi_c \mathbf{\Sigma}_c + 2 \sum_{c=1}^{C} \pi_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - \frac{\tilde{T} - 1}{\tilde{T}} \sum_{c=1}^{C} \pi_c \tilde{\mathbf{\Sigma}}_t^{(c)} - 2 \left(1 - \frac{1}{\tilde{T}} \right) \sum_{c=1}^{C} \pi_c \boldsymbol{\mu}_c \boldsymbol{\mu}_c^T - \frac{2}{\tilde{T}} \sum_{c_1, c_2} \pi_{c_1} P_{c_1 c_2} \boldsymbol{\mu}_{c_1} \boldsymbol{\mu}_{c_2}^T. \quad (75)$$

Again we approximated $T/(T-1) \approx 1$. For the Markov model in Figure 2 and $\tilde{\mathbf{\Sigma}}_t = \sum_{c=1}^{C} \pi_c \tilde{\mathbf{\Sigma}}_t^{(c)}$ we can write

$$\left\langle \langle \dot{\mathbf{x}} \dot{\mathbf{x}}^T \rangle_t \right\rangle = \mathbf{\Sigma}_1 + \mathbf{\Sigma}_2 - \frac{\tilde{T} - 1}{\tilde{T}} \tilde{\mathbf{\Sigma}}_t + \frac{p}{\tilde{T}} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T. \quad (76)$$

Using the definitions of the within-class and between-class scatter matrices of the FLD (equations (34) and (35)) we can rewrite equations (73) and (76):

$$\langle \mathbf{x} \mathbf{x}^T \rangle_t = \frac{1}{2N\tilde{T}} \mathbf{S}_W + \frac{1}{4} \mathbf{S}_B, \quad (77)$$

$$\langle \dot{\mathbf{x}} \dot{\mathbf{x}}^T \rangle_t = \frac{1}{N\tilde{T}} \mathbf{S}_W + \frac{p}{\tilde{T}} \cdot \mathbf{S}_B - \frac{\tilde{T} - 1}{\tilde{T}} \cdot \tilde{\mathbf{\Sigma}}_t. \quad (78)$$

B Simulation Details

B.1 Estimating the error between SFA and FLD

We estimated the deviation between the result of FLD applied to a two-dimensional two-class classification problem and the result of SFA applied to a time series generated from this classification problem using the Markov model in Figure 2 as the angle α between the weight vectors yielded by both methods,

$$\alpha = \arccos \frac{\mathbf{w}_{SFA} \cdot \mathbf{w}_{FLD}}{||\mathbf{w}_{SFA}|| \cdot ||\mathbf{w}_{FLD}||}. \quad (79)$$

We evaluated this angular error as a function of p, the switching probability in Figure 2, i.e., the probability that two consecutive points in the time series are from different classes.

For each probability p (we varied p from 0.01 to 1.0 linearly in intervals of 0.01) we generated 100 different random classification problems in the following way. For each of the two classes a two-dimensional mean vector and 2-by-2 covariance matrix was chosen. The coordinates of the mean were drawn independently and uniformly from the interval $[-4, 4]$. The covariance matrix was determined by its two eigenvalues (drawn uniformly from $[0, 1]$) and a rotation angle (drawn uniformly from $[0, 2\pi]$). For each class, 250 points were drawn from a Gaussian distribution with the selected mean and covariance. The time series for SFA is generated using the Markov model in Figure 2 with the given switching probability p. The length T of this time series is chosen to be 10000 samples.

We computed the average angle between the weight vectors found by SFA and FLD on those 100 classification problems, yielding values between $0°$ and $180°$. We replaced angles $\alpha > 135°$ with angles $180° - \alpha$, since projection directions with different signs are equivalent. Angles between $45°$ and $135°$ were only obtained for $p > 0.5$ where they averaged to about $90°$.

B.2 Calculating the probability of linear separability

To calculate the probability of linear separability in Figure 6B we proceeded in the following way: We generated pairs of point sets (i.e., trajectories) each consisting of 100 points drawn uniformly from the d-dimensional hypercube $[0, 1]^d$. We tested whether these two random point sets are linearly separable using

an efficient method proposed in (Yogananda et al., 2007). We evaluated the probability of linear separability for each dimension d as the percentage of 1000 such randomly generated classification problems that resulted in linearly separable point sets. For each classification problem we also searched for the minimal distance between any two points from different sets. We calculated the average minimum distance over all 1000 classification problems for each dimension d.

We found that the curve for the probability of linear separability closely resembles the analytical result of (Cover, 1965), which considered the fraction of all possible dichotomies of N given data points in general position in d dimensions which are linearly separable.

B.3 Detailed description of the network simulations

B.3.1 Generation of input spike trains

In our circuit simulations we use two different types of input: spike trains generated from isolated spoken digits preprocessed with a model of the cochlea and spike patterns embedded in a continuous stream of Poisson input.

In the speech recognition tasks we use the isolated spoken digits dataset in (Hopfield and Brody, 2000, 2001). This dataset consists of of the audio signals recorded from 5 speakers pronouncing the digits "zero", "one", ..., "nine" in ten different utterances (trials) each, i.e., overall there are 500 speech samples. The duration of an utterance is several 100ms.

To generate a biologically realistic network input, the raw audio signals are converted into the output of a cochlea ("cochleagram") using Lyon's Passive Ear model (Lyon, 1982). This computational model consists of a linear filterbank and a nonlinear gain control network and captures the filtering properties of the cochlea and hair cells of the inner ear. The resulting analog cochleagram is a 86-dimensional time series with values between 0 and 1. An implementation of this cochlea model can be found in the Auditory Toolbox for Matlab (Slaney, 1998).

This analog waveform is then transformed into spike trains using the BSA algorithm (Schrauwen and Campenhout, 2003). This method is able to reconstruct a spike train from an analog trace with a given reconstruction filter. Filtering the spike train with this reconstruction filter should yield a trace with a minimal deviation from the original waveform. We used the implementation from the Reservoir Computing Toolbox (Verstraeten et al., 2007). We chose a reconstruction filter with an exponential form ($\tau = 30$ms) and selected the threshold parameter of the algorithm to be 0.97 (standard value). In order to obtain lower firing rates of the spiking stimuli, we scaled the amplitude of the reconstruction filter such that it has an integral of 40. Furthermore we selected 20 from these 86 spike trains in equidistant steps. The same spike patterns have also been used for the speech recognition task in (Legenstein et al., 2008).

The embedded spike patterns, on the other hand, consist of 10 Poisson spike train segments of length $T_{seg} = 250$ms and with rate $r = 20$Hz. Poisson spike trains are generated by positioning spikes in time according to inter-spike intervals drawn from an exponential distribution with rate r until the segment length T_{seg} is reached. Additionally a refractory period of 3ms after a spike is considered, during which no further spike can occur. Similar spike patterns have been considered for example in (Häusler and Maass, 2007). For each pattern class one such pattern is generated. To model the continuous Poisson input, we preceded each pattern instance with a random Poisson input with a duration uniformly drawn between 100ms and 500ms.

B.3.2 Our model of a cortical microcircuit

As a cortical microcircuit we use the laminar circuit model from (Häusler and Maass, 2007) consisting of 560 spiking neurons (Izhikevich neuron model) with dynamic conductance-based synapses. The short-term dynamics of these synapses has been modelled according to the phenomenological model proposed in (Markram et al., 1998). To reproduce the background synaptic input that cortical neurons typically receive *in vivo*, additional synaptic noise is incorporated as an Ornstein-Uhlenbeck (OU) process as conductance input (Destexhe et al., 2001). All parameters of this model, including short-term synaptic dynamics and background synaptic activity, are chosen as in (Häusler and Maass, 2007).

The neurons are organized in six pools; an excitatory and inhibitory pool for each of the layers 2/3, 4, and 5. The numbers of neurons in each layer are 168, 112, and 280, respectively. The connection strengths and probabilities within a pool and between the pools are obtained from data found in (Thomson et al., 2002) and (Gupta et al., 2000). All of the stimulus spike trains (5 for the spike pattern task; 20 for the speech recognition task) are fed into the circuit via the input stream that connects mainly to Layer 4 ("input stream 1" in Figure 1 of (Häusler and Maass, 2007)). The second input stream into Layer 2/3 is switched off.

B.3.3 Training the readouts of the circuit

We instantiated a single circuit and simulated the same network for each stimulus in all the experiments described in this article. In the speech recognition tasks the network is simulated for the same amount of time for all stimuli (500ms for Figure 8 and 750ms for Figure 9). We low-pass filtered the response spike trains with an exponential filter in order to model the contribution of these spikes to the membrane potential of a hypothetical readout neuron. The time constant of this exponential filter is chosen to be 30ms. We refer to this low-pass filtered high-dimensional analog trace as the trajectory of network states in response to a particular stimulus.

To generate a training input for SFA we sampled these trajectories with a sampling time of 1ms and concatenated a random sequence of such trajectories in time (100 trajectories for Figure 8; 1000 trajectories for Figure 9; 200 trajectories for Figure 7). For the embedded spike pattern task one trajectory is defined by the response during one noise/pattern pair. Note that the same stimulus yields different trajectories due to the intrinsic OU-noise of the network that is used to model the background synaptic activity. We proceeded in a similar way as we generated the time series from a classification problem: After each drawing of a trajectory we switched the class from which the next trajectory is drawn according to a Markov model such as that in Figure 2. The probability p for switching the class is chosen to be 0.2 for all experiments, except for the experiment in Figure 7 we had to choose a lower value of $p = 0.01$. We ensured that in the resulting training sequence the number of trajectories was balanced across different classes by requiring that the standard deviation of the numbers of trajectories for each class was at most $T/20$. Before applying SFA or FLD, we projected the trajectories onto the first 100 principal components in order to prevent the covariance matrices from becoming singular, which would lead to numerical issues in the corresponding eigenvalue problems. For the SVM classification of the network states in Figure 8A we used a linear kernel with $C = 10$. The training set for both FLD and SVM consisted of the network states sampled every 1ms of all trajectories considered, but only states during stimulus presentation are taken into account. The same applies to the SVM classification of the slow features for the evaluation of the SFA performance. This performance is evaluated using 10-fold stratified cross validation, where the folds are sampled according to the class size.

B.4 Software

We performed all simulations using Python and NumPy. We used the implementations of SFA and FLD contained in the MDP toolkit (Zito et al., 2008). The Modular toolkit for Data Processing (MDP) is a data processing framework written in Python. The circuit simulations were carried out with the PCSIM software package (http://www.lsm.tugraz.at/pcsim). PCSIM is a parallel simulator for biologically realistic neural networks with a fast C++ simulation core and a Python interface. For Support Vector Machines (SVM) we used the libSVM toolbox contained in the PyML package (http://pyml.sourceforge.net/). Figures were created using Python/Matplotlib and Matlab.

Acknowledgments

We are very grateful to Laurenz Wiskott who provided particularly helpful comments on this paper. Furthermore, we would like to thank Henning Sprekeler and Lars Büsing for stimulating comments and discussions. This paper was written under partial support by the Austrian Science Fund FWF project # S9102-

N13 and project # FP6-015879 (FACETS), project # FP7-216593 (SECO) and project # FP7-231267 (ORGANIC) of the European Union.

References

Atick, J. J. and Redlich, A. N. (1993). Convergent algorithm for sensory receptive field development. *Neural Computation*, 5:45–60.

Becker, S. and Hinton, G. E. (1992). A self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*, 355(6356):161–163.

Berkes, P. (2005a). Handwritten digit recognition with nonlinear fisher discriminant analysis. In *Proc. of ICANN Vol. 2*, volume 3696 of *Lecture Notes in Computer Science*, pages 285–287. Springer.

Berkes, P. (2005b). Pattern recognition with slow feature analysis. Cognitive Sciences EPrint Archive (CogPrint) 4104. `http://cogprints.org/4104/`.

Berkes, P. (2006). *Temporal slowness as an unsupervised learning principle*. PhD thesis, Humboldt-Universität zu Berlin.

Berkes, P. and Wiskott, L. (2003). Slow feature analysis yields a rich repertoire of complex cell properties. *Journal of Vision*, 5(6):579–602.

Berkes, P. and Wiskott, L. (2006). On the analysis and interpretation of inhomogeneous quadratic forms as receptive fields. *Neural Computation*, 18(8):1868–1895.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Blaschke, T., Berkes, P., and Wiskott, L. (2006). What is the relation between slow feature analysis and independent component analysis? *Neural Computation*, 18(10):2495–2508.

Blaschke, T., Zito, T., and Wiskott, L. (2007). Independent slow feature analysis and nonlinear blind source separation. *Neural Computation*, 19(4):994–1021.

Bray, A. and Martinez, D. (2003). Kernel-based extraction of slow features: Complex cells learn disparity and translation invariance from natural images. In *Proc. of NIPS 2002, Advances in Neural Information Processing Systems*. MIT Press.

Broome, B. M., Jayaraman, V., and Laurent, G. (2006). Encoding and decoding of overlapping odor sequences. *Neuron*, 51:467–482.

Buonomano, D. and Maass, W. (2009). State-dependent computations: Spatiotemporal processing in cortical networks. *Nature Reviews in Neuroscience*, 10(2):113–125.

Buonomano, D. V. and Mauk, M. D. (1994). Neural network model of the cerebellum: temporal discrimination and the timing of motor responses. *Neural Computation*, 6:38–55.

Cover, T. M. (1965). Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers*, 14(3):326–334.

Cox, D. D., Meier, P., Oertelt, N., and DiCarlo, J. J. (2005). "Breaking" position-invariant object recognition. *Nature Neuroscience*, 8(9):1145–1147.

Creutzig, F. and Sprekeler, H. (2008). Predictive coding and the slowness principle: an information-theoretic approach. *Neural Computation*, 20(4):1026–1041.

Dayan, P. and Abbott, L. F. (2001). *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. MIT-Press.

Destexhe, A., Rudolph, M., Fellous, J. M., and Sejnowski, T. J. (2001). Fluctuating synaptic conductances recreate in vivo-like activity in neocortical neurons. *Neuroscience*, 107(1):13–24.

DiCarlo, J. J. and Cox, D. D. (2007). Untangling invariant object recognition. *Trends in Cognitive Sciences*, 11(8):333–341.

Duda, R. O., Hart, P. E., and Stork, D. G. (2000). *Pattern Classification*. Wiley.

Einhäuser, W., Kayser, C., König, P., and Körding, K. P. (2002). Learning the invariance properties of complex cells from their responses to natural stimuli. *European Journal of Neuroscience*, 15(3):475–486.

Euston, D. R., Tatsuno, M., and McNaughton, B. L. (2007). Fast-forward playback of recent memory sequences in prefrontal cortex during sleep. *Science*, 318(5853):1147–1150.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annuals of Eugenics*, 7:179–188.

Földiak, P. (1991). Learning invariance from tranformation sequences. *Neural Computation*, 3:194–200.

Franzius, M., Sprekeler, H., and Wiskott, L. (2007a). Slowness and sparseness lead to place, head-direction, and spatial-view cells. *PLoS Computational Biology*, 3(8):e166.

Franzius, M., Vollgraf, R., and Wiskott, L. (2007b). From grids to places. *Journal of Computational Neuroscience*, 22(3):297–299.

Franzius, M., Wilbert, N., and Wiskott, L. (2008). Invariant object recognition with slow feature analysis. In Kurkova, V., Neruda, R., and Koutnik, J., editors, *Proc. 18th Intl. Conf. on Artificial Neural Networks, ICANN'08, Prague*, volume 5163 of *Lecture Notes in Computer Science*, pages 961–970. Springer.

Goodall, M. C. (1960). Statistics: Performance of a stochastic net. *Nature*, 185(4712):557–558.

Gupta, A., Wang, Y., and Markram, H. (2000). Organizing principles for a diversity of GABAergic interneurons and synapses in the neocortex. *Science*, 287:273–278.

Häusler, S. and Maass, W. (2007). A statistical analysis of information processing properties of lamina-specific cortical microcircuit models. *Cerebral Cortex*, 17(1):149–162.

Hopfield, J. J. and Brody, C. D. (2000). What is a moment? "Cortical" sensory integration over a brief interval. *Proc. Nat. Acad. Sci. USA*, 97(25):13919–13924.

Hopfield, J. J. and Brody, C. D. (2001). What is a moment? Transient synchrony as a collective mechanism for spatio-temporal integration. *Proc. Nat. Acad. Sci. USA*, 98(3):1282–1287.

Hyvärinen, A., Karhunen, J., and Oja, E. (2001). *Independent Component Analysis*. Wiley, New York.

Izhikevich, E. M. (2007). Solving the distal reward problem through linkage of STDP and dopamine signaling. *Cerebral Cortex*, 17:2443–2452.

Ji, D. and Wilson, M. A. (2008). Firing rate dynamics in the hippocampus induced by trajectory learning. *J Neurosci*, 28(18):4679–4689.

Jones, L. M., Fontanini, A., Sadacca, B. F., Miller, P., and Katz, D. B. (2007). Natural stimuli evoke dynamic sequences of states in sensory cortical ensembles. *Proc. Nat. Acad. Sci. USA*, 104(47):18772–18777.

Legenstein, R. and Maass, W. (2007). Edge of chaos and prediction of computational performance for neural microcircuit models. *Neural Networks*, 20(3):323–334.

Legenstein, R., Naeger, C., and Maass, W. (2005). What can a neuron learn with spike-timing-dependent plasticity? *Neural Computation*, 17:2337–2382.

Legenstein, R., Pecevski, D., and Maass, W. (2008). A learning theory for reward-modulated spike-timing-dependent plasticity with application to biofeedback. *PLoS Computational Biology*, 4(10):1–27.

Li, N. and DiCarlo, J. J. (2008). Unsupervised natural experience rapidly alters invariant object representation in visual cortex. *Science*, 321:1502–1507.

Lyon, R. F. (1982). A computational model of filtering, detection, and compression in the cochlea. In *Proc. IEEE Int. Conf. Acoustics Speech and Signal Processing*, pages 1282–1285.

Maass, W., Natschläger, T., and Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560.

Maass, W., Natschläger, T., and Markram, H. (2004). Computational models for generic cortical microcircuits. In Feng, J., editor, *Computational Neuroscience: A Comprehensive Approach*, chapter 18, pages 575–605. Chapman & Hall/CRC, Boca Raton.

Markram, H., Wang, Y., and Tsodyks, M. (1998). Differential signaling via the same axon of neocortical pyramidal neurons. *Proc. Nat. Acad. Sci. USA*, 95:5323–5328.

Masquelier, T., Guyonneau, R., and Thorpe, S. J. (2009). Competitive STDP-based spike pattern learning. *Neural Computation*, 21(5):1259–1276.

Masquelier, T. and Thorpe, S. J. (2007). Unsupervised learning of visual features through spike timing dependent plasticity. *PLoS Computational Biology*, 3(2):e31.

Mazor, O. and Laurent, G. (2005). Transient dynamics versus fixed points in odor representations by locust antennal lobe projection neurons. *Neuron*, 48:661–673.

Mitchison, G. (1991). Removing time variation with the anti-hebbian differential synapse. *Neural Computation*, 3(3):312–320.

Rabinovich, M., Huerta, R., and Laurent, G. (2008). Transient dynamics for neural processing. *Science*, 321:48–50.

Schölkopf, B. and Smola, A. J. (2002). *Learning with Kernels*. MIT Press, Cambridge, MA.

Schrauwen, B. and Campenhout, J. V. (2003). BSA, a fast and accurate spike train encoding scheme. In *Proceedings of the International Joint Conference on Neural Networks*.

Slaney, M. (1998). Auditory Toolbox: A MATLAB toolbox for auditary modeling work. Technical report, Apple Computer, Inc. Apple Computer Technical Report #45.

Sprekeler, H., Michaelis, C., and Wiskott, L. (2007). Slowness: An objective for spike-timing-plasticity? *PLoS Computational Biology*, 3(6):1136–1148.

Stone, J. V. and Bray, A. J. (1995). A learning rule for extracting spatio-temporal invariances. *Network: Computation in Neural Systems*, 6(3):429–436.

Thomson, A. M., West, D. C., Wang, Y., and Bannister, A. P. (2002). Synaptic connections and small circuits involving excitatory and inhibitory neurons in layers 2–5 of adult rat and cat neocortex: triple intracellular recordings and biocytin labelling in vitro. *Cerebral Cortex*, 12(9):936–953.

Tishby, N., Pereira, F. C., and Bialek, W. (1999). The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377.

Turner, R. and Sahani, M. (2007). A maximum-likelihood interpretation for slow feature analysis. *Neural Computation*, 19(4):1022–1038.

Verstraeten, D., Schrauwen, B., D'Haene, M., and Stroobandt, D. (2007). An experimental unification of reservoir computing methods. *Neural Networks*, 20(3):391 – 403. Echo State Networks and Liquid State Machines.

Verstraeten, D., Schrauwen, B., Stroobandt, D., and Campenhout, J. V. (2005). Isolated word recognition with the liquid state machine: a case study. *Inf. Process. Lett.*, 95(6):521–528.

Vinje, W. E. and Gallant, J. L. (2000). Sparse coding and decorrelation in primary visual cortex during natural vision. *Science*, pages 1273–1275.

Wallis, G. and Rolls, E. T. (1997). A model of invariant object recognition in the visual system. *Prog. Neurobiol*, 51:167–194.

Wiskott, L. (1998). Learning invariance manifolds. In *Proc. of the 5th Joint Symp. on Neural Computation, May 16, San Diego, CA*, volume 8, pages 196–203, San Diego, CA. Univ. of California.

Wiskott, L. (2003). Slow feature analysis: A theoretical analysis of optimal free responses. *Neural Computation*, 15(9):2147–2177.

Wiskott, L. and Sejnowski, T. J. (2002). Slow feature analysis: unsupervised learning of invariances. *Neural Computation*, 14(4):715–770.

Wyss, R., König, P., and Verschure, P. F. M. J. (2006). A model of the ventral visual system based on temporal stability and local memory. *PLoS Computational Biology*, 4(5):0001–0008.

Yogananda, A. P., Murthy, M. N., and Gopal, L. (2007). A fast linear separability test by projection of positive points on subspaces. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 713–720, New York, NY, USA. ACM.

Zito, T., Wilbert, N., Wiskott, L., and Berkes, P. (2008). Modular toolkit for Data Processing (MDP): a Python data processing framework. *Frontiers in Neuroinformatics*, 2.