Supplement: Gradient-based Training of Slow Feature Analysis by Differentiable Approximate Whitening

Merlin Schüler MERLIN.SCHUELER@INI.RUB.DE
Hlynur Davíð Hlynsson HLYNUR.HLYNSSON@INI.RUB.DE
Laurenz Wiskott LAURENZ.WISKOTT@INI.RUB.DE
Institute for Neural Computation, Ruhr-Universität Bochum, Germany

Editors: Wee Sun Lee and Taiji Suzuki

S1. Small parameter-study: number of power iterations

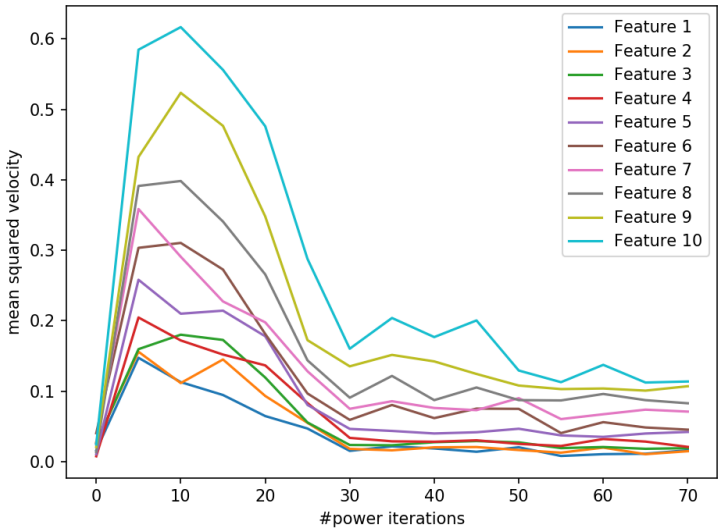

Figure S1: The mean squared velocity (Δ value) per feature for an increasing number of power iterations, averaged over 10 trials. For no power iterations, the velocity is close to 0, but for a too small positive number of power iterations the optimization becomes unstable and results in sub-optimal performance.

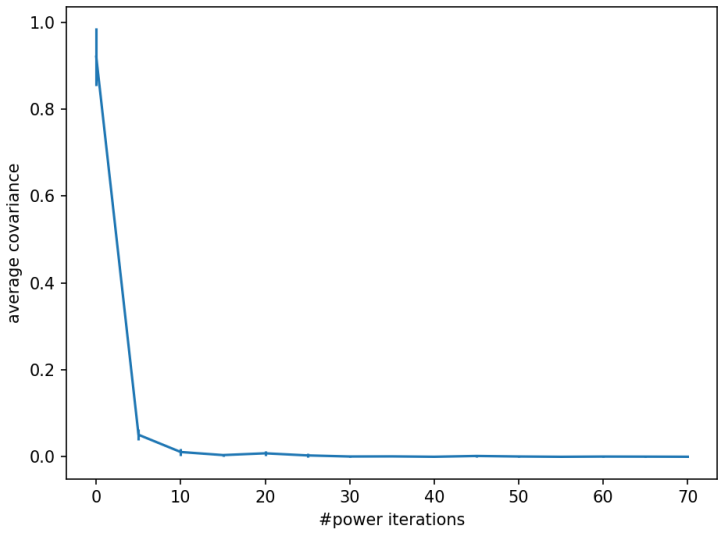

Figure S2: The average covariance (auto-variances not included) for an increasing number of power iterations, averaged over 10 trials. Without whitening, the covariance is close to 1, but the features quickly become decorrelated when the number of iterations is increased.

S2. Comparison architectures (synthetic data)

This supplement presents the architectures used for comparing greedy layer-wise with gradient-based end-to-end training of SFA. Both networks have been trained with both training methods. In the gradient-based setting, the *Keras*'s *Nadam* optimizer has been used with default hyperparameters and a whitening layer has been applied to the network output.

Quadratic expansion network	
Operation	Output dimension
Input Layer	500
Linear Layer	33
Quadratic Expansion	594
Linear Layer	33
Quadratic Expansion	594
Linear Layer	33
Quadratic Expansion	594
Linear Layer	6

Table S1: A network with multiple quadratic expansions, each preceded by linear dimensionality-reduction. These kind of networks are typically used in closed-form SFA as they trade-off model expressivity with memory requirements.

Neural network(tanh)	
Operation	Output dimension
Input Layer	500
Linear Layer	500
Pointwise *tanh*	500
Linear Layer	500
Pointwise *tanh*	500
Linear Layer	500
Pointwise *tanh*	500
Linear Layer	6

Table S2: A simple multi-layer neural network using a *tanh* activation function to induce non-linearity.

S3. Place-Fields

This supplement contains the more detailed behavior of slow and sparse features during rotation of the agent in Figures S3 and S4.

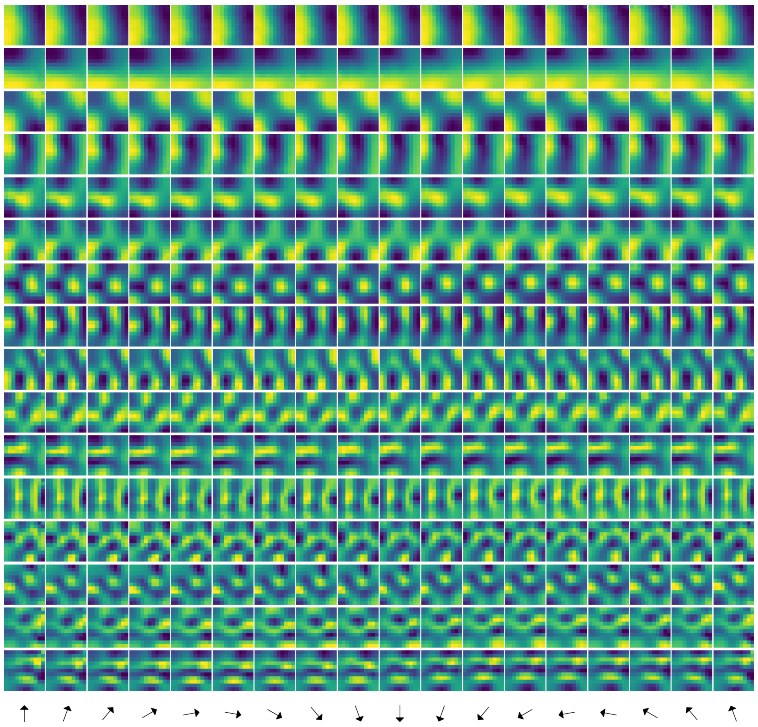

Figure S3: Activation maps for all 16 (ordered) slow output features during a full rotation in 20° degree steps. The features are largely invariant to rotation of the agent, despite the narrow field of view. They encode distinct positions, though this is not directly visible without sparse coding.

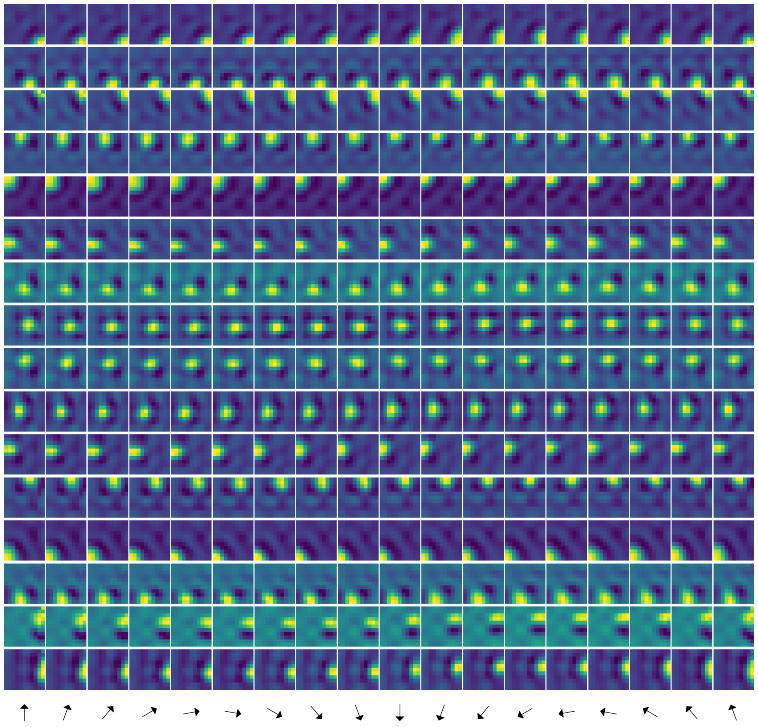

Figure S4: Activation maps for all 16 sparse output features during a full rotation in 20° degree steps. The features clearly encode position and are largely invariant to rotation of the agent, despite the narrow field of view.

S4. Code

The code is available as *Python 3 + Keras/Tensorflow* implementation at: https://s3.amazonaws.com/acml2019/powersfa_code.zip