Artificial Neural Networks – Proc. Int. Conf. ICANN, pp. 157–164, Springer 1996

Associative Completion and Investment Learning using PSOMs

Jörg Walter and **Helge Ritter**

Department of Information Science
University of Bielefeld, D-33615 Bielefeld, FRG
Email: walter@techfak.uni-bielefeld.de
http://www.techfak.uni-bielefeld.de/~walter/

Abstract

We describe a hierarchical scheme for rapid adaptation of context dependent "skills". The underlying idea is to first invest some learning effort to specialize the learning system to become a rapid learner for a restricted range of contexts. This is achieved by constructing a "Meta-mapping" that replaces an slow and iterative context adaptation by a "one-shot adaptation", which is a context-dependent skill-reparameterization.
The notion of "skill" is very general and includes a task specific, hand-crafted function mapping with context dependent parameterization, a complex control system, as well as a general learning system.
A representation of a skill that is particularly convenient for the investment learning approach is by a *Parameterized Self-Organizing Map* (PSOM). Its direct constructability from even small data sets significantly simplifies the investment learning stage; its ability to operate as a continuous associative memory allows to represent skills in the form of "multi-way" mappings (relations) and provides an automatic mechanism for sensor data fusion.
We demonstrate the concept in the context of a (synthetic) vision task that involves the associative completion of a set of feature locations and the task of one-shot adaptation of the transformation between world and object coordinates to a changed camera view of the object.

1 Introduction

The data-driven construction of non-linear mappings for interpolation and classification has been among the most widely used features of artificial neural networks (ANN). This has led to a strong emphasis of feed-forward networks and to much fruitful work at elucidating their capabilities from the perspective of interpolation, statistics, and approximation theory [3, 11].

However, there is clearly more to ANNs than the approximation of a nonlinear function[1]. As applications of neural networks have become more demanding, the issue of how to build neural *systems* that are composed of multiple mapping modules has received increasing attention [4], raising the question of how such systems can be trained without requiring inordinately large training sets. This is particularly urgent in areas where training samples are not abundant or are costly, such as, e.g., in robotics. Related

[1] the reader will be aware of the fact that *any* computation can be viewed as a sufficiently complex vector valued function; however, this theoretical possibility does not mean that this is always the most productive viewpoint.

with this is the important question of how to achieve *"one-shot"-learning*, i.e., the ability of a system to adapt to a particular context within a single or at least a very small number of trials. Finally, we should weaken the emphasis of "one-way" mappings. A much more natural and flexible formulation of many tasks, such as, e.g., flexible sensor fusion, is in terms of *continuous relations* and the corresponding required ability is that of a *continuous associative memory* instead of the much more rigid one-way mappings of feedforward networks.

In the present paper we attempt to address some of these issues. First, we show how the recently introduced Parameterized Self-Organizing Maps ("PSOMs",[6, 8]) can be used as a flexible continuous associative memory that offers a very natural approach to the problem of *sensor fusion* [1].

As an illustration we consider the task to associate for a continuous set of camera views of a rigid 3D-object the spatial positions of occluded point features from varying subsets of visible point features. Sensor fusion occurs automatically when the PSOM is offered more than a minimally necessary set to determine the orientation of the object. In this case, the redundant information is exploited for an increased accuracy of the inferred point locations.

We then describe a learning architecture in which a PSOM can be combined with other mapping modules (some or all of which may be PSOMs again) in such a way that the resulting system can be prestructured for the ability of "one-shot-learning" or "one-shot-adaptation" by means of a prior "investment learning" phase. Here, the underlying idea is that one-shot adaptation can be achieved with a "Meta-mapping" for the dynamic reparameterization of one or several mapping module(s) that represent(s) the "skill" that is to be adapted; this "Meta-mapping" is constructed during the investment learning phase and its input is a sufficient (small) number of sensor measurements that characterise the current context.

To make the paper self-contained, we give a brief summary of the main characteristics of the PSOM approach in the next section, and append a condensed description in the appendix. A more detailed account, together with result of applications in the domain of vision and robotics, see [9, 10, 7].

2 Properties of PSOMs

The PSOM is characterized by the following important features:

- The PSOM is the continuous analog of the standard discrete self-organizing map ([5]). It shows *excellent generalization* capabilities based on *attractor manifolds* instead of just attractor points.
- For training data sets with the *known topology* of a multi-dimensional cartesian grid the map manifold can be constructed *directly*. The resulting PSOM is immediately usable – without any need for time consuming adaptation sequences. However, further on-line *fine-tuning* is possible, e.g., in the case of coarsely sampled data or when the original training data were corrupted by noise.
- The price for rapid learning is the cost when using. The PSOM needs an iterative search for the best-matching parameter location $\mathbf{s}^*$ (Eq. 1 in the appendix).
- The *non-linear associative completion* ability of the PSOM can be used to represent continuous relations instead of just functions ("dynamic multi-way mapping"). This is illustrated in Fig. 1. The choice of an input variable set is made by specifying a suitable diagonal projection matrix $\mathbf{P}$ whose non-zero elements

select the input subspace(s) together with a possible relative weighting of the input components for the non-linear least-square best-match (see Eq. 1 in the appendix for the best-match vector $\mathbf{w}(\mathbf{s}^*)$.)

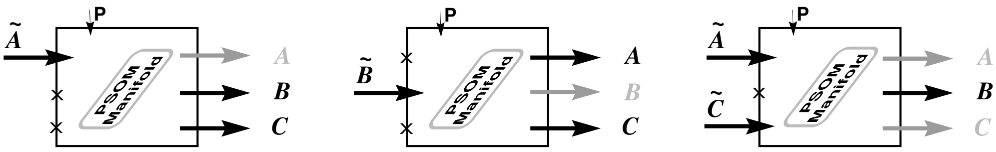

Figure 1: "Continuous associative memory" supports multiple mapping directions. The specified $\mathbf{P}$ matrices select different subspaces (here symbolized by $\tilde{A}$, $\tilde{B}$ and $\tilde{C}$) of the embedding space as inputs. Values of variables in the selected input subspaces are considered as "clamped" (indicated by a tilde) and determine the values found by the iterative least square minimization (Eq. 1). for the "best-match" vector $\mathbf{w}(\mathbf{s}^*)$. This provides an associative memory for the flexible representation of continuous relations.

3 Image Completion and Object Orientation Identification

To illustrate the flexibility of the PSOM-approach, we consider a task from the vision domain. Here, we are often confronted with the following *sensor-fusion* problem: given the identity of a geometric object together with a number of measurements of continuous valued features (such as the 2D-image locations of a subset of salient point features), use this information to infer a set of "missing" variable values, such as the location of occluded object parts or the 3D-orientation and/or location of the object.

Although the approach is not restricted to the use of camera sensors, we consider the case when the sensor measurements is a set of 2D locations of salient object features (of known identity). Such information would have to be obtained by some suitable preprocessing stage; in the present context we consider a synthetic vision task in which the set of 2D-locations has been computed from a perspective projection of a geometric object model. We also assume that the object has already been centered (by some suitable tracking process), leaving us with four remaining degrees of freedom: orientation and depth, which we describe by roll ϕ, pitch θ, yaw ψ and z [2].

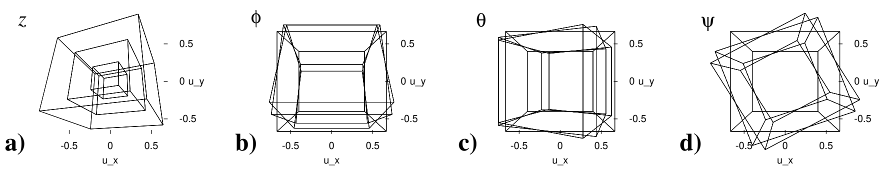

Figure 2: *(a)* The cubical test object seen by the camera when rotated and shifted in several depths z (ϕ=10°, θ=20°, ψ=30°, z=2...6L, cube size L.) *(b–d)* 0°, 20°, and 30° rotations in the roll ϕ, pitch θ, and yaw ψ system.

Fig. 2 shows the camera view of the test object (a unit cube, but note that the method works in the same way with any (fixed) shape) for different orientations and depths. This provides a different characterization of object pose by the coordinate pairs $\vec{u}_{P_i}$ of a set of salient feature points P_i, $i = 1, 2, \ldots n$ on the object (for the cube, we choose its eight corners, i.e., $n = 8$). Therefore, a convenient embedding space X is spanned by the $4 + 2n$ variables $\mathbf{x} = (\phi, \theta, \psi, z,\ \vec{u}_{P1}, \vec{u}_{P2}, \cdots, \vec{u}_{P_n})$. For the construction of the PSOM, X was sampled at the 81 points of a regular $3{\times}3{\times}3{\times}3$ grid (in ϕ, θ, ψ, z-space) covering a range of 150° for each orientation and twice the edge length L of the cube for the depth dimension.

Fig. 3 depicts some examples, in which orientation and depth of the cube were inferred using the constructed PSOM with the image positions of four of its corners (indicated by asterisks) chosen as input (dotted lines indicate true, solid lines indicate reconstructed object pose).

The achieved root mean square (RMS) errors for recovering the object pose were 2.7°, 3.2°, 2.8°, and $0.12L$ (for ϕ, θ, ψ, z). The remaining four corner point locations could be predicted within an accuracy of 1.3 % of the mean edge image size. A more detailed analysis of the impact of parameter variations, such as the range of the involved variables, the number of training vectors, the sensor noise and the number of available input points will be presented elsewhere [7].

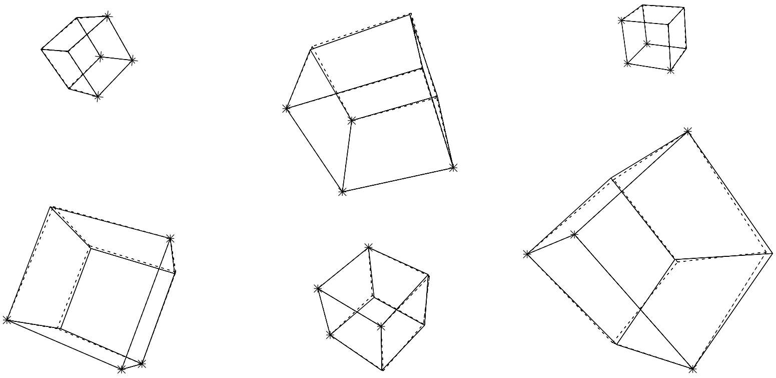

Figure 3: Six test poses. *Dotted* lines indicate the cubical test objects seen by the camera. *Asterisks* mark the positions of four corner points used for the reconstruction of the object pose, indicated with *full* lines.

4 Investment Learning in Prototypical Contexts

In this section we consider the problem of efficiently learning "skills" and there adaptation to changing context. To be concrete, we consider a "skill", in form of a multivariate function mapping $T : \vec{x}_1 \to \vec{x}_2$, transforming between two task variable sets $\vec{x}_1$ and $\vec{x}_2$. We assume: *(i)* that the "skill" can be acquainted by a "**t**ransformation box" ("T-Box"), which is a suitable building block with learning capabilities; *(ii)* the mapping "skill" T is internally modeled and determined by a set of parameters ω (which can be accessed from outside the "black box", which makes the T-Box rather an open, "white box"); *(iii)* the correct parameterization ω changes smoothly with the *context* of the system; *(v)* the situational context can be observed and is associated with a set of suitable sensors values $\vec{c}$ (some of them are possibly expensive and temporarily unavailable); *(vi)* the context changes only from time to time, or on a much larger time scale, than the task mapping T is employed.

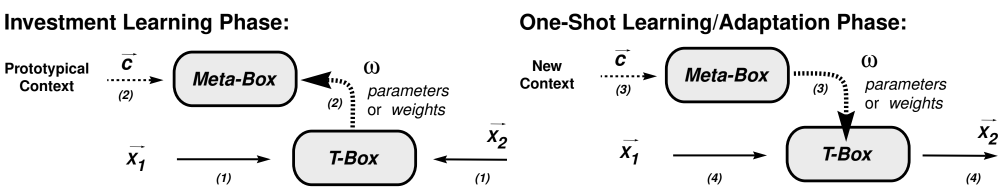

Figure 4: The transformation T-Box maps between task input $\vec{x}_1$ and output $\vec{x}_2$ spaces and gets encoded in the parameter set ω.

The conventional, integrated approach is to consider the problem of learning the mapping from all relevant input values, $\vec{x}_1, \vec{c}$ to the desired output $\vec{x}_2$. This leads to a larger, specialized network. The disadvantages are *(i)* the possible catastrophic interference (after-learning in a situated context may effect other contexts in an uncontrolled way); *(ii)* low modularity and re-usability.

Here, we approach a solution in a modular way and suggest to *split learning*: *(i)* (structurally) among two modules, the Meta-Box and the T-Box, and *(ii)* (temporally) in two phases. The first, the *investment learning* stage may be slow and has the task of learning prototypical context situations. It does not yet produce the final solution, but instead pre-structures the system such that in the subsequent, the *one-shot adaptation phase*, the specialization to a particular solution (within the chosen domain) can be achieved extremely rapidly.

As illustrated in Fig. 4, the Meta-Box is responsible for providing the mapping from sensory context observations $\vec{c}$ to the parameter set ω. Therefore, each T-Box parameter/weight set ω together with its associated context information $\vec{c}$ can serve as a high-dimensional training data vector for constructing the Meta-Box mapping during the investment learning stage.

To obtain these training data vectors, we must choose a suitable set of *prototypical contexts* $j = 1, 2, \ldots$ and determine for each of them the parameter set ω_j of an adapted T-Box *(1)* and the context sensor observation $\vec{c}_{,j}$ *(2)*. After the Meta-Box has been trained, the task of adapting the "skill" to a new system context is tremendously accelerated. Instead of any time-consuming re-learning of the (T) mapping this adjustment now takes the form of an *immediate* Meta-Box $\rightarrow$ T-Box mapping (*one-shot adaptation*): The Meta-Box maps a new (unknown) context observation $\vec{c}_{,new}$ *(3)* into the parameter / weight set ω_{new} for the T-Box. Equipped with ω_{new}, the T-Box provides the desired mapping T_{new} *(4)*.

In contrast to a *mixture-of-experts* architecture [4], suggesting a linear combination of multiple, parallel working "expert" networks, the described scheme could be viewed as *non-linear "interpolation-of-expertise"*. It is efficient w.r.t. network requirements in memory and in computation: the "expertise" (ω) is gained and implemented in a single "expert" network (T-Box). Furthermore, the Meta-Box needs to be re-engaged only when the context is changed, which is indicated by a deviating sensor value $\vec{c}$.

However, this scheme requires from the learning implementation of the T-Box, that the parameter/weight set ω is represented in a -with $\vec{c}$ smoothly varying- "non-degenerated" manner. E.g. a regular multilayer perceptron allows many weight permutations ω. Employing a MLP in the T-Box would additionally require a suitable stabilizer to avoid grossly inadequate interpolation between prototypical "expertises" ω_j, denoted in different kinds of permutations.

5 Object Coordinate Transformation by Investment Learning

To illustrate this approach let us revisit the vision example of Sec.3. For a robot, an interesting skill in that setting could be to transform coordinates from a camera centered (world or tool) frame (yielding coordinate values $\vec{x}_1$) to the object centered frame (yielding coordinate values $\vec{x}_2$). This mapping would have to be represented by the T-Box and its "environmental context" would be the current orientation of the object relative to the camera. Fig. 5 shows three ways how the investment learning scheme can be implemented in that situation. All three share the same PSOM network type as the META-BOX building block. The "Meta-PSOM" bears the advantage, that the architecture can cope easily with situations, where various (redundant) sensory values are un-available.

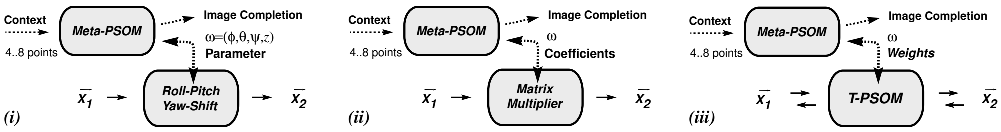

Figure 5: Three different ways to solve the context dependent, or investment learning task.

The first solution (i) uses the Meta-PSOM for the reconstruction of object pose in roll-pitch-yaw-depth values from Sec. 3. The T-Box is given by the four succesive homogeneous transformation on the basis of the ϕ, θ, ψ, z values obtained from the Meta-PSOM.

The middle solution (ii) represents the coordinate transformation as the product of the four successive transformations. Thus, in this case the Meta-PSOM controls the coefficients of a matrix multiplication. As in (i), the required ω are gained by a suitable calibration, or system identification procedure.

When no explicit ansatz for the T-Box is already available, we can use method (iii). Here, for each prototypical context, the required T-mapping is learned by a network and becomes encoded in its weight set ω. For this, one can use any trainable network that meets the requirement stated at the end of the previous section; however, PSOMs are a particularly convenient choice, since they can be directly constructed from a small data set and since they offer the advantage of multi-way mappings.

To illustrate this for the present example, we chose for the T-Box a 2×2×2 "T-PSOM" that implements the coordinate transform for *both directions simultaneously.* Its training required eight training vectors arranged at the corners of a cubical grid.

In order to compare approaches $(i)-(iii)$, we computed the transformation T-Box accuracy averaged over a set of 50 contexts (given by 50 randomly chosen object poses), each with 100 object volume points $\vec{x}_2$ to be transformed into camera coordinates $\vec{x}_1$. For the computed RMS value for the camera x, y, and z direction method (i) gave the result $[0.025, 0.023, 0.14]L$. Method (ii) led to the slightly better values of $[0.016, 0.015, 0.14]L$. Approach (iii), although not requiring any geometric analysis, led to practically the same accuracy, namely $[0.015, 0.014, 0.12]L$.

6 Discussion and Conclusion

To be practical, learning algorithms must provide solutions that can compete with solutions hand-crafted by a human who has analyzed the system. The criteria for success can vary, but usually the costs of gathering data and of teaching the system are a major factor on the side of the learning system, while the effort to analyze the problem and to design an algorithmn is on the side of the hand crafted solution.

Here we suggest the PSOM as a versatile module for the rapid learning of high-dimensional, non-linear, smooth relations. Based on a strong bias, introduced by structuring the training data in a topological order, the PSOM can generalize from very few examples - if this model is a good approximation to the system. The multi-way mapping ability PSOM is generated by the *associative completion* of partial inputs. We showed an example in which this ability was used to infer the positions of occluded feature points. In addition, this property leads to an interesting approach for the task of *sensor fusion*, since redundant data are exploited to achieve a higher accuracy for the inferred variable values.

As a general framework to achieve one-shot or very rapid learning we introduced the idea of *investment learning*. While PSOMs are extremely well suited for this approach, the underlying idea to "compile" the effect of a longer learning phase into a one-step META-BOX is more general and is independent from the PSOMs. The META-BOX controls the parameterization of a set of context specific "skills" which are implemented in the form of one or several transforms or T-BOX. Thus, learning at the level of the skills is replaced *by their context-sensitive dynamic re-parameterization* through the META-BOX-mapping. This emphasises an important point for the construction of more flexible and more powerful learning systems: in addition to focusing on output *values* we should increasingly embrace mappings that *produce as their result other mappings* – a point of view that has in recent years also become increasingly emphasized in the realm of functional programming languages.

We illustrated three versions of this approach when the output mapping was a coordinate transform between a camera centered and an object centered frame of reference. They differed in their choice of the T-BOX that was used. Comparing the three choices we found that the neural network based T-PSOM can fully compete with dedicated one-way function mapping boxes while at the same time offering the additional advantage of providing forward and backward transform simultaneously.

Appendix: The PSOM Algorithm

A PSOM is a parameterized, m-dimensional hyper-surface $M = \{\mathbf{w}(\mathbf{s}) \in X \subseteq \mathbb{R}^d | \mathbf{s} \in S \subseteq \mathbb{R}^m\}$ that is embedded in some higher-dimensional vector space X. M is used in a very similar way as the standard discrete self-organizing map: given a distance measure $dist(\mathbf{x}, \mathbf{x}')$ and an input vector $\mathbf{x}$, a best-match location $\mathbf{s}^*(\mathbf{x})$ is determined by minimizing

$$\mathbf{s}^* := \underset{\mathbf{s} \in S}{\operatorname{argmin}} \; dist(\mathbf{x}, \mathbf{w}(s)) \tag{1}$$

The associated "best-match vector" $\mathbf{w}(\mathbf{s}^*)$ provides the best approximation of input $\mathbf{x}$ in the manifold M. If we require $dist(\cdot)$ to vary only in a input sub-space X^{in} of X (i.e., $dist(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T \mathbf{P}(\mathbf{x} - \mathbf{x}') = \sum_{k=1}^{d} p_k (x_k - x'_k)^2$, where the diagonal matrix $\mathbf{P} = \text{diag}(p_1, \cdots, p_d)$ projects into X^{in}), $\mathbf{s}^*(\mathbf{x})$ actually will only depend on $\mathbf{Px}$. The components of $\mathbf{w}(\mathbf{s}^*(x))$ with zero diagonal elements p_k can be viewed as the output. The best-match vector $\mathbf{w}(\mathbf{s}^*)$ is the (non-linear) *associative completion of a fragmentary input* $\mathbf{x}$ of which only the *part* $\mathbf{Px}$ *is reliable*. It is this associative mapping that we will exploit in applications of the PSOM, see Fig. 1.

M is constructed as a manifold that passes through a given set D of data examples. To this end, we assign to each data sample a point $\mathbf{a} \in S$ and denote the associated data sample by $\mathbf{w}_{\mathbf{a}}$. The set $\mathbf{A}$ of the assigned parameter values $\mathbf{a}$ should provide a good discrete "model" of the topology of our data set. The assignment between data vectors and points $\mathbf{a}$ must be made in a topology preserving fashion to ensure good interpolation by the manifold M that is obtained by the following steps.

For each point $\mathbf{a} \in \mathbf{A}$, we construct a "basis function" $H(\cdot, \mathbf{a}; \mathbf{A})$ or simplified[2] $H(\cdot, \mathbf{a}) : S \mapsto \mathbb{R}$ that obeys *(i)* $H(\mathbf{a}_i, \mathbf{a}_j) = 1$ for $i = j$ and vanishes at all other points of $\mathbf{A}$ $i \neq j$ (orthonormality condition,) and *(ii)* $\sum_{\mathbf{a} \in \mathbf{A}} H(\mathbf{s}, \mathbf{a}) = 1$ for $\forall \mathbf{s}$ ("partition of unity" condition.) We will mainly be concerned with the case of $\mathbf{A}$ being a m-dimensional rectangular hyper-lattice; in this case, the functions $H(\cdot, \mathbf{a})$ can be constructed as products of Lagrange interpolation polynomials, see [10]. Then,

$$\mathbf{w}(\mathbf{s}) = \sum_{\mathbf{a} \in \mathbf{A}} H(\mathbf{s}, \mathbf{a})\ \mathbf{w}_{\mathbf{a}}. \tag{2}$$

defines a manifold M that passes through all data examples. Minimizing $dist(\cdot)$ in Eq. 1 can be done by some iterative procedure, such as gradient descent or – preferably – the Levenberg-Marquardt algorithm [10]. This makes M into the attractor manifold of a (discrete time) dynamical system. Since M contains the data set D, any at least m-dimensional "fragment" of a data example $\mathbf{x} = \mathbf{w} \in D$ will be attracted to the correct completion $\mathbf{w}$. Inputs $\mathbf{x} \notin D$ will be attracted to the interpolating manifold point.

References

[1] B. Brunner, K. Arbter, and G. Hirzinger. Task directed programming of sensor based robots. In *Intelligent Robots and Systems (IROS-94)*, pages 1081–1087, September 1994.

[2] K. Fu, R. Gonzalez, and C. Lee. *Robotics : Control, Sensing, Vision, and Intelligence*. McGraw-Hill, 1987.

[3] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

[4] M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6(2):181–214, 1994.

[5] T. Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, Berlin, Heidelberg, 1995.

[6] H. Ritter. Parametrized self-organizing maps. In S. Gielen and B. Kappen, editors, *Proc. Int. Conf. on Artificial Neural Networks (ICANN-93), Amsterdam*, pages 568–575. Springer Verlag, Berlin, 1993.

[7] J. Walter. *Rapid Learning in Robotics*. Cuvillier Verlag Göttingen, 1996. also http://www.techfak.uni-bielefeld.de/~walter/pub/.

[8] J. Walter and H. Ritter. Local PSOMs and Chebyshev PSOMs – improving the parametrised self-organizing maps. In *Proc. Int. Conf. on Artificial Neural Networks (ICANN-95), Paris*, volume 1, pages 95–102, 1995.

[9] J. Walter and H. Ritter. Investment learning with hierarchical PSOM. In D. Touretzky, M. Mozer, and M. Hasselmo, editors, *Advances in Neural Information Processing Systems 8 (NIPS*95)*, pages 570–576. Bradford MIT Press, 1996.

[10] J. Walter and H. Ritter. Rapid learning with parametrized self-organizing maps. *Neurocomputing*, 12:131–153, 1996.

[11] H. White. Learning in artificial neural networks: a statistical perspective. *Neural Computation*, 1:425–464, 1989.

[2] In contrast to kernel methods, the basis functions may depend on the relative position to all other knots. However, we drop in our notation the dependency $H(\mathbf{s}, \mathbf{a}) = H(\mathbf{s}, \mathbf{a}; \mathbf{A})$ on the latter.