ESANN'1993 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 7-8-9 April 1993, D-Facto public., ISBN 2-9600049-0-6, pp. 189-194

Locally implementable learning with isospectral matrix flows. [1]

Jeroen Dehaene[2] and Joos Vandewalle,
Deptartment of electrical engineering, ESAT-SISTA,
K.U.Leuven, Kardinaal Mercierlaan 94, B-3001 Heverlee, Belgium
e-mail: dehaenej@esat.kuleuven.ac.be

Abstract

We describe a time continuous uniform learning algorithm, that can be implemented on a network structure, in a way that is related to Hebbian learning. On a local scale, the weight $h_{i,j}$ of each connection between nodes i and j obeys the same law, using information, available in nodes i and j. This information is either supplied externally, or obtained by propagation of information over the network in neural network style. On a global scale, the weight matrix follows an isospectral matrix flow (preserving the eigenvalues), directed by an external teaching signal, supplied to the nodes. The weight matrix converges to an instantiation, useful for Hopfield type and related neural networks, stabilizing all (and only those) binary patterns in a subspace described by the teaching signal.

1. introduction

An important general idea in neural network research is the idea of computing with a dynamical system with on a *local* scale (neurons, weights) simple and often uniform behavior, all elements evolving by the same law using information available through the network, and resulting in some interesting global behavior such as memory on a *global* scale.

In our case the *global* behavior is an isospectral matrix flow, a dynamical system with a matrix state, $H \in R^{n\times n}$ (n is the number of nodes or neurons), which preserves its eigenvalues, during evolution. An external time varying signal $p(t) : R \rightarrow R^n$ is supplied to the system. If $p(t)$ evolves in a subspace $\mathcal{P}$ of R^n (and doesn't settle in a smaller subspace), the principal eigenspace of H converges to $\mathcal{P}$, under initial conditions specified below. The flow can be applied to neural network learning, and can be useful for other applications where directional information is to be extracted from a signal $p(t)$.

On a *local* scale, the system can be implemented on a network structure in the following sense. The matrix entries are stored in the connections of the network, by storing entry $h_{i,j}$ in the connection between node i and node j, like the weights of a fully interconnected Hopfield network. The entries of a time varying vector p are supplied to the nodes, entry p_i to node i. Other vectors can be obtained in the nodes by propagating a vector over the network. That is, if a vector x is available in the nodes, the vector Hx can be obtained as follows.

[1]This research work was carried out at the ESAT laboratory of the Katholieke Universiteit Leuven, in the framework of a Concerted Action Project of the Flemish Community, entitled *Applicable Neural Networks*. The scientific responsibility is assumed by its authors.
[2]Research assistant of the Belgian National Fund for Scientific Research.

Each node i sends $x_i h_{i,j}$ over its connection to j. By summing all incoming information, a node j can calculate the j-th component of Hx. To realize the matrix flow, each matrix entry $h_{i,j}$ evolves according to the same fixed law, of the form $\dot{h}_{i,j} = \sum_k \alpha_k x_i^{(k)} y_j^{(k)}$, where $x^{(k)}$ and $y^{(k)}$ are vectors available at the nodes and α_k are coefficients. In matrix notation we obtain

$$\dot{H} = \sum_k \alpha_k x^{(k)} y^{(k)T} \tag{1}$$

We concentrate on time continuous formulations, because of the elegant theoretical framework and because of the attractivity of analog implementations. However, we don't claim the presented work is ready for analog implementation. We also discuss discrete implementation.

2. Theory

2.1. a special case of the double bracket flow

Consider the matrix flow

$$\dot{H} = H^2 pp^T - 2Hpp^T H + pp^T H^2 = rp^T - 2qq^T + pr^T \tag{2}$$

where $q = Hp$ and $r = Hq = H^2 p$. This flow is clearly of the form (1).

It is a special case of the double bracket flow [1], with time varying parameter $N = pp^T$,

$$\dot{H} = [H, [H, N]] \tag{3}$$

where $N \in R^{n \times n}$ is symmetric, and $[X, Y] = XY - YX$ denotes the matrix bracket. The global behavior of the double bracket flow is determined by three main properties.

1) The flow is isospectral. This means that during evolution of H its eigenvalues (spectrum) remain constant. The matrix $H(t)$ can be decomposed as $H(t) = \Theta(t)\Lambda\Theta(t)^T$, where $\Lambda \in R^{n\times n}$ is a constant diagonal matrix, with the eigenvalues as its entries, and $\Theta(t) \in R^{n \times n}$ is orthogonal and has the time varying eigenvectors as its columns. Θ obeys

$$\dot{\Theta} = [N, H]\Theta \tag{4}$$

It is easy to verify that this implies (3) for the evolution of H, and that Θ remains orthogonal because $[N, H]$ is skew symmetric.

2) When N is constant, the matrix H converges to a matrix $H(\infty)$, which is commutable with N, i.e. $[H(\infty), N] = 0$. This means that the eigenvectors of H and N are the same (or can be chosen so in the case of repeated eigenvalues).

3) When the eigenvectors of $H(\infty)$ are ordered by decreasing eigenvalues, the corresponding eigenvalues of N are ordered in the same way.

When $N = pp^T$ and time varying, the flow is still isospectral and the principal eigenvector of H instantaneously turns toward p, the principal eigenvector of N.

Although p varies in time, we can still get convergence, if the principal eigenspace is of the same dimension as a subspace in which $p(t)$ evolves. We can prove the following theorem [2].

Theorem 1 *Consider the system (2). If the piecewise continuous vector signal $p(t)$ evolves in a k-dimensional subspace $\mathcal{P} \subset R^N$ and $span(\{p(t) \mid t > t_0\}) = \mathcal{P}$, for all $t_0 > 0$, and if the principal eigenvalue λ of H_0 has multiplicity k, then $H(t)$ converges to a matrix $H(\infty)$ with the same eigenvalues as H_0 and for which $\mathcal{P}$ is the eigenspace with eigenvalue λ, if the initial state H_0 avoids some thin manifolds of unstable trajectories.*

We can get a picture of what happens, by interpreting equation (4) for the case $N = pp^T$, giving $\dot{\Theta} = [pp^T, H]\Theta = (pq^T - qp^T)\Theta$. Instantaneously, each column θ_i changes with $\dot{\theta}_i = (pq^T - qp^T)\theta_i$. That is, θ_i is rotated parallel to the plane spanned by p and q in the direction from q to p. Consider for example the case where H has only two distinct eigenvalues, 1 and 0. Then H is the matrix of an orthogonal projection. We can choose Θ (without affecting H) such that one column θ_1 is parallel to q and the others are orthogonal to q. Then, only θ_1 changes, towards p, and the whole subspace $R(H)$ onto which H projects, rotates towards p. (Fig. 1.).

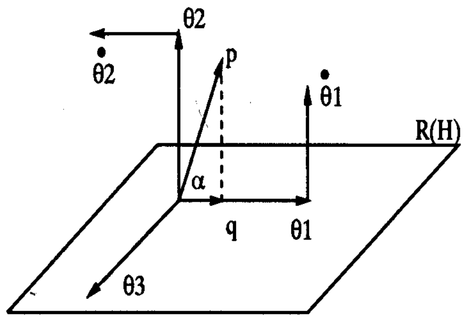

Fig. 1. Rotation of the principal eigenspace

From these observations, it is clear that if the principal eigenvalue λ of H_0 has multiplicity smaller than k, H won't converge but will keep adapting to the current $p(t)$. If the eigenvalue has a multiplicity, greater than k, H will converge, but the principal eigenspace of $H(\infty)$ will depend on the initial state H_0.

2.2 Stabilization of the eigenvalues

For an isospectral flow, deviations from the spectrum (for instance due to an inaccurate realization) are not restored but result in the (exact) system preserving the new spectrum. This is not only a problem for inaccurate analog realizations, but also for discrete realizations, like $H(k+1) = H(k) + \epsilon[H(k), [H(k), p(k)p(k)^T]]$ where ϵ is some small positive real number. We introduce some extra terms, which ensure the stability of given eigenvalues. Doing so, the local implementability is lost at first, but will be regained by some further modifications.

Consider the flow

$$\dot{H} = v'(H) \qquad (5)$$

where $v(x)$ and $v'(x) = dv(x)/dx$ are polynomials. The matrix $v'(H)$ has the same eigenvectors as H but with eigenvalues $v'(\lambda_i)$ instead of λ_i. During evolution according to (5) the eigenvectors remain constant, and the eigenvalues follow

$$\dot{\lambda} = v'(\lambda) \qquad \lambda = \lambda_1, \ldots, \lambda_n \qquad (6)$$

which is the (one-dimensional) gradient ascent flow of $v(\lambda)$. It can also be proved that (5) is the gradient ascent flow of trace($v(H)$), as can be easily verified when H is diagonal.

From (6), it follows immediately that if $v(\lambda)$ is upper-bounded, all λ_i converge (when initial states in an unstable equilibria are excluded) to a local maximum of $v(\lambda)$. Each local maximum is attracted from within the attraction interval between the nearest left local minimum (or -∞) and the nearest right local minimum (or $+\infty$).

We can use these ideas to stabilize given eigenvalues for an isospectral flow by considering

$$\dot{H} = [H, \Omega] + v'(H) \qquad (7)$$

Since the first term is isospectral the eigenvalues still follow (6), and converge to local maxima of $v(\lambda)$, from within the attraction intervals described above. While converging, the contribution of the second term vanishes. Therefore, when started from an initial state H_0, H converges to the same solution as would be obtained by $\dot{H} = [H, \Omega]$ with as starting point, the equilibrium obtained by $\dot{H} = v'(H)$, when started at H_0. For example, if we want to stabilize the eigenvalues 0 and 1 in (2), we can choose $v(\lambda) = -\frac{1}{2}\lambda^2(\lambda - 1)^2$, resulting in $v'(H) = -H(H - I)(2H - I)$.

2.3 Regaining local implementability

The calculation on the network of the powers of H in $v'(H)$ would need some further sophistication. We can however introduce modifications which reduce the calculations again to a uniform law of the form (1).

We assume that a new vector $s(t)$, is available in the nodes, which evolves randomly in R^n. We now replace the term $v'(H)$ by the term $ss^T v'(H) + v'(H)ss^T$, which is of the form (1). To understand this modification, we consider the linear symmetric matrix space operation $\gamma : X \rightarrow \gamma(X) = ss^T X + Xss^T$. This operation is positive semidefinite with null space $\{X \in R^{n \times n} \mid X^T = X, Xs = 0\}$. Therefore, while $\dot{H} = v'(H)$ performs a steepest ascent of trace($v(H)$), $\dot{H} = \gamma(v'(H)) = ss^T v'(H) + v'(H)ss^T$ performs a non steepest ascent. Because γ has eigenvalues 0, the flow wouldn't necessarily find a local minimum of trace($v'(H)$) if s were a constant vector. It would converge to a point H where $v'(H)s = 0$. However, when s keeps varying in R^n and doesn't settle in a smaller subspace, $v'(H)s$ can only stay 0 if $v'(H)$ goes to 0. This finally assures again that the eigenvalues converge to local maxima of $v(\lambda)$. One can also prove that the corresponding attraction intervals for the eigenvalues are unchanged, and that the combination of the new term $\gamma(v'(H))$ with isospectral terms, ensures convergence to the same points as with the term $v'(H)$.

The calculation of the terms $ss^Tv'(H) + v'(H)ss^T$ requires the calculation of H^ks for k up to the degree of $v'(H)$. We can halve this effort, by replacing these terms by $v_1(H)ss^Tv_2(H)+v_2(H)ss^Tv_1(H)$ where $v(x) = v_1(x)v_2(x)$. This gives almost the same results.

Finally, if we omit the noise source and let p take over its role, the only difference is that $p(t)$ does not span R^n. However, the principal eigenvalue will still be stabilized by (7), but deviations of the other eigenvalues are only partly restored.

2.4. Application to learning in neural networks

As mentioned above, the principal eigenvector of H instantaneously turns towards a vector (or pattern) p, available in the nodes. This is interesting behavior for learning in neural associative memory. Consider for example a piecewise linear Hopfield type model

$$\begin{aligned} \dot{x} &= -x + Ay + b \\ y &= f(x) \end{aligned} \tag{8}$$

where $x \in R^n$ is the state, $y \in R^n$ is the output $b \in R^n$ is the external bias, $A \in R^{n\times n}$ is the symmetric weight matrix and $f : R^N \rightarrow R^N$:

$$\begin{aligned} f_i(x) &= -1 \text{ if } x_i < -1 \\ f_i(x) &= x_i \text{ if } -1 < x_i < 1 \\ f_i(x) &= 1 \text{ if } x_i > 1 \end{aligned}$$

We will first consider $b = 0$, and later assign a role to b. If a binary output pattern (with $y_i = \pm 1$) is an eigenvector of the weight matrix with eigenvalue larger than 1, it is stable. When A changes gradually to make it an eigenvector, the pattern will even become stable before.

Other dynamical learning rules can be regarded as exploiting the same idea. For example, in a simple Hebbian learning learning, like $\dot{A} = pp^T$, the principal eigenvector of A tends to p, when p is kept constant. However, this method leads to positive definite weight matrices with a tendency to stabilize many more patterns than intended.

Another rule,

$$\dot{A} = -(A - \lambda I)pp^T - pp^T(A - \lambda I) \tag{9}$$

where $\lambda > 1$, is related to linear perceptron learning and minimizes $||(Ap-\lambda p)||^2$. Also here there is a tendency to stabilize many more patterns than intended.

With the isospectral flow we can initialize A with k eigenvalues $\lambda > 1$ and $n-k$ eigenvalues $-\tau < 0$, and let A follow $\dot{A} = [A,[A,pp^T]]$, or equivalently, from $(A-\lambda I)(A+\tau I) = 0$,

$$\dot{A} = -(A - \lambda I)pp^T(A + \tau I) - (A + \tau I)pp^T(A - \lambda I) \tag{10}$$

which avoids the need of A^2p. Note the similarity with (9), for large τ.

If p is a teaching signal, spanning a k-dimensional subspace $\mathcal{P}$ of R^n, and the largest eigenvalue of A has multiplicity k, then A converges to a matrix with eigenvalue λ in that subspace and $-\tau$ in its orthogonal complement. If $\tau/(\lambda - 1)$ is large enough, the binary patterns in $\mathcal{P}$ and no others will be stable. The flow converges to a variant of a weight instantiation studied in [3] for the related LSSM model (Linear Systems in Saturated Mode).

The initial value of A can be a diagonal matrix, possibly perturbed by letting (10) run for a short time with a random p, to avoid initial conditions leading to unstable equilibria.

A simple realization is reached by a rule $\dot{H} = pp^T - Hp(Hp)^T$, preserving the eigenvalues 1 and -1 of H ($H^2 = I$) and defining $A = H + aI$, where $a > 0$.

To stabilize the eigenvalues we can add the stabilizing terms, studied in the previous sections. With a random signal $s(t)$, this also avoids the problems with special initial conditions.

Numerous modifications can be thought of. We only discussed rules for which exact results can be proved. However, for an application in neural network learning, the isospectrality is not at all a necessary property. The main benefit is that by preserving the negative eigenvalues, the number of spurious stable states, remains limited. Moreover, a pattern needn't become an eigenvector to be stable. With these observations in mind, one could try to find well performing rules with less exact foundations. One could also implement the learning rule as taking place simultaneously with the memory operation, but on a slower time scale. The role of p could then be played by the output y. To learn a new pattern, y can be forced to it, by supplying a strong bias b. The system adapts to make b unnecessary for the stabilization of the pattern. If during operation, the system remains most of the time at or near an equilibrium, the memory contents wouldn't be destroyed by the constant presence of the learning mechanism.

References

1. R.W. Brockett, "Dynamical systems that sort lists, diagonalize matrices, and solve linear programming problems", in *Linear Algebra and its Applications* **146** (1991) 79-91.

2. J. Dehaene and J. Vandewalle, "Locally implementable matrix flows and their application to neural network learning.", report ESAT-SISTA-1992-26, K.U.Leuven.

3. J.H. Li, A.N. Michel and W. Porod, "Analysis and synthesis of a class of neural networks: linear systems operating on a closed hypercube", in *IEEE Trans. Circuits Syst* **CAS-36** (1989) 1405-1422.