Topographic Infomax in a Neural Multigrid

James Kozloski, Guillermo Cecchi, Charles Peck, and A. Ravishankar Rao

IBM T.J. Watson Research Center, Yorktown Heights,NY 10598
{kozloski,gcecchi,cpeck,ravirao}@us.ibm.com

Abstract. We introduce an information maximizing neural network that employs only local learning rules, simple activation functions, and feedback in its functioning. The network consists of an input layer, an output layer that can be overcomplete, and a set of auxiliary layers comprising feed-forward, lateral, and feedback connections. The auxiliary layers implement a novel "neural multigrid," and each computes a Fourier mode of a key infomax learning vector. Initially, a partial multigrid computes only low frequency modes of this learning vector, resulting in a spatially correlated topographic map. As higher frequency modes of the learning vector are gradually added, an infomax solution emerges, maximizing the entropy of the output without disrupting the map's topographic order. When feed-forward and feedback connections to the neural multigrid are passed through a nonlinear activation function, infomax emerges in a phase-independent topographic map. Information rates estimated by Principal Components Analysis (PCA) are comparable to those of standard infomax, indicating the neural multigrid successfully imposes a topographic order on the optimal infomax-derived bases.

1 Introduction

Topographic map formation requires an order-embedding, by which a set of vectors X in some input space is mapped onto a set of vectors Y in some output space such that the ordering of vectors in Y, when embedded within some alternate lower-dimensional coordinate system (usually 2D) preserves as much as possible the partial ordering of vectors in X in the input space. An important additional objective of topographic map formation is that the volume defined by Y be maximized so as to avoid trivial mappings. This second objective ignores the ordering of inputs and outputs in their respective spaces and instead attempts to maximize the mutual information between X and Y, $I(X;Y)$. We observe that order embedding need not necessarily impact information, as the ordering of outputs imposes no constraint on the volume that they define.

One of the most influential approaches to topographic map formation, Kohonen's self-organizing map (SOM) [1], has its origins in another of Kohonen's algorithms, Learning Vector Quantization (LVQ) [2]. LVQ has as its stated goal density estimation, or that the number of input vectors assigned to each output vector be equalized. It aims to accomplish this as an approximation of k-means clustering, which minimizes mean squared error between input vectors and output prototypes. As such, neither algorithm guarantees the desired equalization between inputs and output vectors for nonuniform input spaces. Modifications to the traditional LVQ algorithm improve density estimation, but require ongoing adjustments of still more parameters that govern the rates at which inputs are assigned to each output vector [3]. For SOM and its many variations, these modifications are complicated by a neighborhood function which constrains the assignment of inputs to output vectors and is shrunk during learning to create a smooth mapping (for example, see [4]). While LVQ and SOM support overcomplete bases (wherein the number of outputs exceeds the number of inputs), the problem of density estimation is not addressed by adding more output vectors.

Information maximization is a well-characterized optimization that results in density estimation and equalization of output probabilities. Originally expressed for the case of multi-variate Gaussian inputs in the presence of noise [5], a significant extension of this solution accommodates input spaces of arbitrary shape for the noiseless condition [6]. The original derivations of infomax used critically sampled bases (wherein the number of outputs equals the number of inputs), either for notational simplicity [5] or by necessity [6]. While subsequent derivations of infomax [7] and related sparse-coding and probabilistic modeling strategies [8,9] incorporate overcomplete bases, none do so in the context of topographic mapping. Algorithms that maximize Shannon information rates subject to a topographic constraint [10] rely on non-locally computed learning rules and do not apply to arbitrary input spaces and the noiseless condition. Finally, because one of the main operational principles of infomax is to make outputs independent by eliminating redundancy, standard topographic mapping algorithms, which necessarily *create* dependencies between map neighbors, should seem incompatible with infomax.

Here we present a network capable of performing the infomax optimization over an arbitrary input space (in our case, natural images) for the noiseless condition, using either critically sampled or overcomplete bases, while simultaneously creating a topographic map with either a phase-dependent or a phase-independent order embedding. Changes to learning rates or neighborhood sizes are not required for convergence. These capabilities derive from a novel neural multigrid, configured to estimate Fourier modes of the infomax antiredundancy learning vector using feed-forward, lateral, and feedback connections. Section 2 introduces the infomax network from which ours derives [11], which we have termed a *Linsker network*. In subsequent sections we present, step by step, each of the modifications to the Linsker network necessary to achieve these capabilities. Section 3 shows how to implement a multilayer Linsker network with feedback, which generates topography, but fails to achieve infomax. Section 4 refines the feedback network and introduces the neural multigrid to address low frequency redundancy reduction. Section 5 shows how an overcomplete basis finally achieves the infomax optimum in a neural multigrid. Section 6 demonstrates that a modified multigrid can achieve the infomax optimum for a critically sampled basis, provided a phase-independent order embedding is specified. Finally section 7 discusses the experimental results in the context of a new principle of information maximization which we term *topographic infomax*.

2 A Three-Stage Infomax Network

A Linsker network comprises three stages. Stage one selects a vector $\widehat{x}$ from the input ensemble, $\widehat{x} \in X$, and computes the input vector $x = q^{-1/2}(\widehat{x} - x_0)$, where x_0 is an input bias vector that continually adapts with each input according to $\varDelta x_0 = \beta_{x_0}[\widehat{x} - x_0]$,[1], and $q^{-1/2}$ is the pre-computed whitening matrix, where $q = \langle(\widehat{x} - x_0)(\widehat{x} - x_0)'\rangle$. Pre-whitening of inputs is not required for infomax, but speeds convergence. For results shown here, X was a set of image segments drawn at random from natural images [12].

Stage two learns the input weight matrix C and computes $u \equiv Cx$, where each element u_i is the linear output of a stage two unit i to a corresponding stage three unit. In addition, each stage two unit computes an element of the output vector y, such that $y_i = \sigma(u_i)$, where $\sigma(\cdot)$ denotes a nonlinear squashing function. We used the logistic transfer function for $\sigma(\cdot)$ as in [6]: $y = 1/1 + e^{-(u+w_0)}$, where w_0 is an output bias vector that continually adapts with each input according to $\varDelta w_0 = \beta_{w_0}[1 - 2y]$. The ensemble of all network output vectors is then $y \in Y$, and the objective to maximize $I(X;Y)$. Because the network is deterministic, maximizing $I(X;Y)$ is equivalent to maximizing $H(Y)$, since $I(X;Y) = H(X) + H(Y) - H(Y|X)$, $H(X)$ is fixed, and, for the noiseless condition, $H(Y|X) = 0$.

Let us now consider stage three, whose outputs comprise a learning vector with the same dimension as u. When applied in stage two to learning C, this learning vector yields the anti-redundancy term $(C')^{-1}$ of Bell and Sejnowski (ie., the inverse of the transpose of the input weight matrix) [6,11]. Consider the chain rule $H(Y) = \sum_{i=1}^{n} H(\{y_i\}) - \sum_{i=1}^{n} I(\{y_i\}; \{y_{i-1}\}, \ldots, \{y_1\})$. Then, maximizing $H(Y)$ is achieved by constraining the entropy of the elements y_i (as expressed by the first sum) while minimizing their redundancy (as expressed by the second). In a Linsker network, the constraint imposed by $\sigma(\cdot)$ and the learning vector produced by stage three perform these functions, and are sufficient to guide C learning to maximize $H(Y)$. Thus, the learning vector produced by stage three is responsible for redundancy minimization. We refer to it (and its subsequent derivatives) as the *anti-redundancy learning vector*, hereafter denoted ψ.

In a Linsker network, local learning rules and a linear activation function iterating over a single set of lateral connections compute ψ_i for each unit i of a single stage three layer. Units in this layer are connected by the weight matrix $\widehat{Q}$, whose elements undergo Hebbian learning such that $\widehat{Q} \to Q \equiv \langle uu'\rangle$. For a given input presentation, feed-forward and lateral connections modify elements of an auxiliary vector v at each iteration t according to $v_t = v_{t-1} + u - \alpha\widehat{Q}v_{t-1}$. To learn $\widehat{Q}$ locally, we set $v_0 = u$ and use the learning rule $\varDelta\widehat{Q} = \beta_Q[v_0 v_0{}' - \widehat{Q}]$. Regardless of initial v, and assuming $\widehat{Q} = Q$ and the scalar α is chosen so that v converges, by Jacobi $\alpha v_\infty = Q^{-1}u$. The constraint $0 < \alpha < 2/Q_+$ must be satisfied for v to converge, where Q_+ is the largest eigenvalue of Q [11]. In Linsker's network, α is computed by a heuristic [5]. We devised a dynamic, local computation of

[1] Note that all learning rates in the network are constant, and denoted by β with subscript. For this and subsequent learning rates, we used $\beta_{x_0} = 0.0001$, $\beta_{w_0} = 0.0021$, $\beta_Q = 0.0007$, and $\beta_C = 0.0021$.

Q_+ based on power iteration, from which α can be computed precisely. Let e represent an activity vector propagated through the lateral network $\widehat{Q}$ in the absence of the normal stage three forcing term, $v_{t-1}+u$, such that $e_t = -\alpha\widehat{Q}e_{t-1}$. Precalculating $\alpha = 1/\|e_t\|$ for each t ensures $\|e\| \to Q_+$ and $\alpha \to 1/Q_+$, thus satisfying the convergence criterion of v. In practice, a finite number of iterations are sufficient to approximate αv_∞,[2] and therefore anti-redundancy learning for a given input weight C_{ij} can depend on the locally computed element of the learning vector, $\psi_i = \alpha v_i$, and the local input to stage two, x_j. The final infomax learning rule for the network is then $\Delta C = \beta_C[(\psi + 1 - 2y)x']$ [11]. Use of a standard Linsker network produces the expected infomax result reliably with a fixed number of Jacobi iterations and learning rates (Fig. 1A).

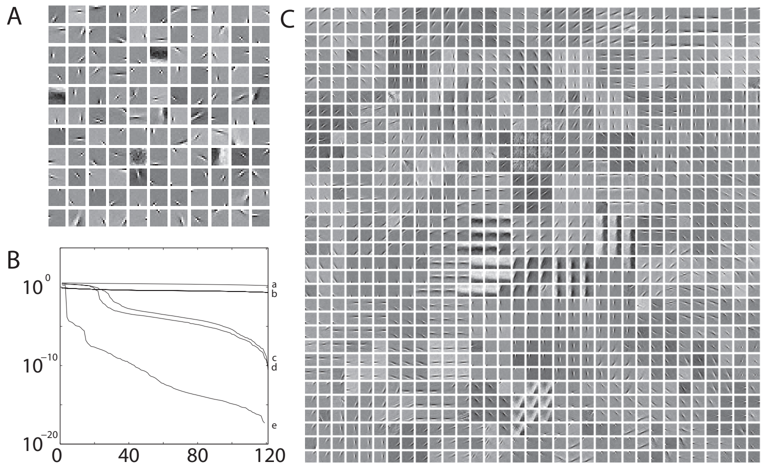

Fig. 1. A: Learned bases of Linsker network (standard infomax). This and all subsequent figures derive from training over a set of publicly available natural images (http://www.cis.hut.fi/projects/ica/data/); B: PCA of outputs u (eigenvalues of Q shown on a continuous function). Curves labeled a and b result from A and C, and show maximization of the output volume (successful infomax). Curves labeled c, d, and e, result from multigrid infomax over low to high frequency modes, infomax from feedback, and multigrid infomax over low frequency modes only, and show failed infomax. C: Learned bases of overcomplete, topographic infomax.

3 Infomax from Feedback

Jacobi iteration in stage three of a Linsker network aims to solve the equation $\alpha Qv = u$. We observe that certain linear transformations of this equation likewise yield the infomax anti-redundancy term. For example, stage three might include a second layer, or *grid* (denoted h_1), in which a fixed, symmetric, full rank weight

[2] For most problems, we found 4 Jacobi iterations to be sufficient.

matrix S linearly transforms u, yielding a new auxiliary vector $u^{h_1} = Su = SCx$. As in a Linsker network, units in h_1 are connected by a lateral weight matrix, $\widehat{Q}^{h_1}$, that undergoes Hebbian learning such that $\widehat{Q}^{h_1} \rightarrow Q^{h_1} \equiv \langle u^{h_1} {u^{h_1}}' \rangle = SQS'$. Jacobi iteration in h_1 proceeds similarly as above, $v_t^{h_1} = v_{t-1}^{h_1} + u^{h_1} - \alpha^{h_1} \widehat{Q}^{h_1} v_{t-1}^{h_1}$. To recover the infomax anti-redundancy term $(C')^{-1}$, a derivation similar to that of Linsker yields: $I = {Q^{h_1}}^{-1} SQS'$, $(S')^{-1} = {Q^{h_1}}^{-1} SCxx'C'$, $(S')^{-1}(C')^{-1} = {Q^{h_1}}^{-1} SCxx'$, and $(C')^{-1} = S'{Q^{h_1}}^{-1} SCxx'$. Hence, $(C')^{-1} = S\alpha^{h_1} \langle v_\infty^{h_1} x' \rangle$, and anti-redundancy learning for a given input weight now depends on an element of the feedback learning vector $\psi = S'\alpha^{h_1} v^{h_1}$ computed in the stage three's input layer and the corresponding local input to stage two.

Theoretically, the proposed feedback network is equivalent to a Linsker network, and should therefore not change the infomax optimization it performs. In practice, however, the choice of S can influence the optimization dramatically, since Jacobi iteration is used to estimate $\alpha^{h_1} v_\infty^{h_1}$. If, for example, S represents a low-pass convolution filter and thus low frequency modes are emphasized in both u^{h_1} and Q^{h_1},[3] Jacobi iteration in h_1 can fail to provide a solution that is accurate enough for infomax to succeed, given some fixed number of Jacobi iterations, since low frequency modes of the solution are notoriously slow to converge [13]. In fact, we observed a failure of this network to achieve infomax (Fig. 1B[d]). However, we consistently observed a stable topographic ordering of the outputs in the 2D coordinate system of the network after each failed infomax optimization, suggesting that errors derived from incomplete convergence of low frequency Fourier modes in ψ are sufficient to generate a topographic map. Next we set out to eliminate these errors.

4 A Neural Multigrid

The second layer of the feedback infomax network described in the preceding section aims to solve the equation $\alpha^{h_1} Q^{h_1} v^{h_1} = u^{h_1}$ by means of Jacobi iteration. It fails to do so accurately because of the dominant low frequency modes of the problem, rendering ψ unsuitable for anti-redundancy learning that depends on accurate solutions in these modes. Hence infomax fails. We explored using multigrid methods to better estimate this solution, since multigrid methods in general speed convergence and accuracy of Jacobi iteration by decomposing it into a series of iterative computations performed sequentially over a set of grids, each solving different Fourier modes of the original problem [13]. Multigrid casts the problem into a series of smaller and smaller grids, such that low frequency modes in the original problem can converge quickly and accurately in the form of high frequency modes in a restricted problem. The multigrid method we implement here in a neural network is nested iteration, though the network design can easily accommodate other multigrid methods such as "V-cycle" and "Full Multigrid" [13]. Similar to the feedback network described in the previous section, Jacobi iteration in h_1 aims to solve $\alpha^{h_1} Q^{h_1} v^{h_1} = u^{h_1}$, but now only after

[3] We used a low-pass, 2D gaussian kernel with SD=0.85.

v^{h_1} is initialized with the result of a preceding series of nested iterative computations over a set of smaller grids wherein lower frequency modes of the solution have already been computed.

The set of grids, h_k, is enumerated by the set of wavelengths of the Fourier modes of the problem that each solves, for example $k \in \{1, 2, 4, \ldots\}$. The iterative computation performed in each grid is similar to that in a Linsker network, now denoted $v_t^{h_n} = v_{t-1}^{h_n} + u^{h_n} - \alpha^{h_n} \widehat{Q}^{h_n} v_t^{h_n}, n \in k$. As in traditional multigrid methods, we chose powers of two for the neural multigrid wavelengths, such that if the Linsker network and grid h_1 are 11×11 layers, grid h_2 is a 5×5 layer, and h_4 a 2×2 layer. Feed-forward connections initially propagate and restrict each u^{h_n} to each lower dimensional grid h_{2n}, such that $u^{h_{2n}} = S^{h_n} u^{h_n} \forall n \in k$, where S^{h_n} denotes the restriction operator (in our neural multigrid, a rectangular feed-forward weight matrix) from grid h_n to h_{2n}. As in traditional multigrid methods, restriction here applies a stencil (in our neural multigrid, a neighborhood function), such that the restriction from grid h^n to h^{2n} is $u_{x',y'}^{h_{2n}} = \frac{1}{16}[4u_{x,y}^{h_n} + u_{x+1,y+1}^{h_n} + u_{x+1,y-1}^{h_n} + u_{x-1,y+1}^{h_n} + u_{x-1,y-1}^{h_n} + 2(u_{x,y+1}^{h_n} + u_{x,y-1}^{h_n} + u_{x+1,y}^{h_n} + u_{x-1,y}^{h_n})]$, where (x, y) are coordinates in h_n, and (x', y') are the corresponding transformed coordinates in h_{2n}. Jacobi iteration proceeds within coarse grids first, followed by finer grids. Feedback propagates and smoothly interpolates the result of a coarse grid iteration, $\alpha^{h_{2n}} v^{h_{2n}}$, to the next finer grid, where it replaces v^{h_n} prior to Jacobi iteration within the finer grid: $v^{h_n} \leftarrow S^{h_n \prime} \alpha^{h_{2n}} v^{h_{2n}}$. In this way, higher frequency mode iteration refines the solution provided by lower frequency mode iteration. The process continues until $\alpha^{h_1} v^{h_1}$ is computed by iteration at the second layer of stage three, and finally ψ is derived from feedback to stage three's input layer as described in the previous section.

While restriction and interpolation of activity vectors in the neural multigrid are easily accomplished through feed-forward and feedback connections described by S^{h_n} and $S^{h_n \prime}$, how is Q^{h_n} computed for each grid using only local learning rules? Consider the problem of restricting to a coarser grid the matrix Q^{h_n}, defined for all multigrid methods as $Q^{h_{2n}} = S^{h_n} Q^{h_n} S^{h_n \prime}$ [13]. By substitution, $Q^{h_{2n}} = S^{h_n} \langle u^{h_n} u^{h_n \prime} \rangle S^{h_n \prime} = \langle S^{h_n} u^{h_n} u^{h_n \prime} S^{h_n \prime} \rangle = \langle u^{h_{2n}} u^{h_{2n} \prime} \rangle$. Hence, any restricted matrix $Q_{h_{2n}}$ can be computed by Hebbian learning over a lateral weight matrix $\widehat{Q}^{h_{2n}}$ such that $\widehat{Q}^{h_{2n}} \to Q^{h_{2n}} \equiv \langle u^{h_{2n}} u^{h_{2n} \prime} \rangle$.[4]

In the preceding experiments with feedback infomax networks, the *failure* to compute low frequency modes of the solution to $\alpha^{h_1} Q^{h_1} v^{h_1} = u^{h_1}$ was responsible for a topographic ordering (since infomax learning was unable to eliminate low frequency spatial redundancy in the map). Next, we hypothesized that *limiting* our solution to these same low frequency modes could have a comparable topographic influence while providing a means to complete the infomax optimization. Minimizing redundancy in low frequency Fourier modes is equivalent to minimizing redundancy *between* large spatial regions of the map. In the absence of competing anti-redundancy effects from other Fourier modes, redundancy *within* these large regions should therefore increase, as units learn

[4] Any optimization over a matrix A requiring a solution to $Ax = b$ can be implemented using a neural multigrid if, and only if, A is strictly a function of b.

based on a smooth, interpolated anti-redundancy learning vector derived from coarse grids only. In fact, the topographic effect was pronounced. Again, however, the network failed to achieve infomax (Fig. 1B[e]), now because high frequency modes of the solution were neglected by our partial multigrid. We therefore devised a network that gradually incorporates the iterative computations of each grid of the neural multigrid, from coarse grids to fine, into the computation of ψ.

Initially, iteration proceeded only at the two coarsest grids, with m inputs presented to the network in this configuration.[5] Infomax learning proceeded with feedback vectors computed by interpolating the solutions from the coarsest grid through each multigrid layer, then feeding the result back to stage three's input layer, where finally ψ was computed. Subsequent layers in the neural multigrid were activated one at a time at intervals of m input presentations. The number of input presentations for which a grid h_n had been active was p^{h_n}, and only when $p^{h_n} \geq m \forall n \in k$ were all Fourier modes of the solution to $\alpha^{h_1} Q^{h_1} v^{h_1} = u^{h_1}$ present in ψ, and the multigrid complete. Prior to this, ψ was computed in the partial multigrid as a linear combination of the feedback vectors from the two finest active grids, h_a and h_b, fed back through all intervening layers: $\psi = S \prod_{n=1}^{a} S^{h_n \prime} [\beta^{h_a} \alpha^{h_a} v^{h_a} + (1 - \beta^{h_a}) S^{h_b \prime} \alpha^{h_b} v^{h_b}]$, where $\beta^{h_a} = p^{h_a}/m \in [0, 1]$.

Gradual incorporation of each grid of the neural multigrid resulted in a topographic map, but failed to achieve infomax (Fig. 1B[c]), suggesting a more radical approach was required to recover an infomax solution.

5 Overcomplete Topographic Infomax

To recover an infomax solution within a topographic map, we first devised a network that uses a neural multigrid to compute an anti-redundancy learning vector based on pooled outputs from nine separate critically sampled bases (each initially embedded in an 11×11 grid as above). Each basis therefore represented a separate infomax problem, and each was then re-embedded within a single "overcomplete" 2D grid (now 33×33, denoted h_{oc}) as follows: $x \leftarrow r + 3x$, $y \leftarrow s + 3y$, where (r, s) represents a unique pair of offsets applied to each 11×11 grid's corresponding coordinates in order to embed it within the 33×33 grid, $r \in [0, 2]$, $s \in [0, 2]$. The pooling of elements from each basis was achieved by restricting the output of h_{oc} to the neural multigrid's first layer, such that $u^{h_1} = S_{oc} u^{h_{oc}}$, where h_1 is an 11×11 grid, and S_{oc} is a rectangular feed-forward weight matrix.[6]

The computation of ψ proceeded as above, with grids incorporated into the multigrid gradually, from coarse to fine. After the multigrid was completely active, a set of nine, independent lateral networks within the overcomplete grid became active. Each lateral network included only connections between those

[5] We used $m = 2,000,000$.

[6] We scaled the previous low-pass, 2D gaussian kernel proportionally to SD=2.55 in order to create the new restriction matrix, S_{oc}.

units comprising a single critically sampled basis, and the overcomplete grid's lateral network thus comprised overlapping, periodic, lateral connections. The results of iteration over each of these independent Linsker networks represented nine separate solutions to nine fully determined problems $Qv = u$. These solutions were gradually combined with the feedback learning vector from the multigrid as described in the previous section, and yielded an infomax solution wherein each critically sampled basis was co-embedded in a single topographic map (Fig. 1B[b],C).

6 Absolute Redundancy Reduction

Next, we aimed to generate a phase-independent order embedding in our topographic map using the neural multigrid. Drawing upon the work of Hyvärinen et al., [14] we reasoned that certain nonlinear transformations of the inputs to the neural multigrid might produce a topographic influence independent of phase selectivity in output units, as has been observed in primary visual cortex. Unlike Hyvärinen et al., we maintained the goal of maximizing mutual information between the input layer and the *first* layer of our multilayer network.

The nonlinear transformation applied was the absolute value, such that $u^{h_1} = S|u|$. Given this transformation, feedback from the multigrid was no longer consistent with anti-redundancy learning at the input layer of stage three. We reasoned that the input weights to unit i, $C_{ij} \forall j$, should be adjusted to eliminate redundancy in multigrid units, which derives from pooled absolute activation levels over i. C_{ij} must then be modified in a manner dependent on each unit i's contribution to this redundancy, i.e., its absolute activation level. Hence, at each unit i, the multigrid feedback vector was multiplied by ω_i prior to its incorporation into ψ, where ω_i takes the value 1 if $u_i \geq 0$ and -1 otherwise.

The computation of ψ was modified during learning as above, with grids incorporated into the multigrid gradually, from coarse to fine. In the end, the network employed a linear combination of the multigrid feedback vector and the infomax anti-redundancy vector αv, such that locally computed elements of ψ were determined by $\psi_i = \beta \alpha v_i + (1 - \beta)\omega_i \sum_k S'_{ik} \alpha^{h_1} v_k^{h_1}$.[7]

The results show that a phase-independent topographic map does result from nonlinearly mapping the problem $\alpha Qv = u$ onto a gradually expanded neural multigrid (Fig. 2A), and that infomax was readily achieved in these experiments (Fig. 2B[b]). Interestingly, when multigrid inputs were transformed in this way, the constraints imposed by initial learning, biased topographically by the partial multigrid, were not severe enough to prevent infomax from emerging within the map, even for a critically sampled basis. Finally, we applied the nonlinear mapping to an overcomplete basis as above and found that phase-independent, overcomplete, topographic infomax was readily achieved (Fig. 2B[b],C).

[7] We limited β to 0.25 in order to maintain a smooth mapping, given that infomax emerged readily at this weighting, and did not require $\beta \to 1$.

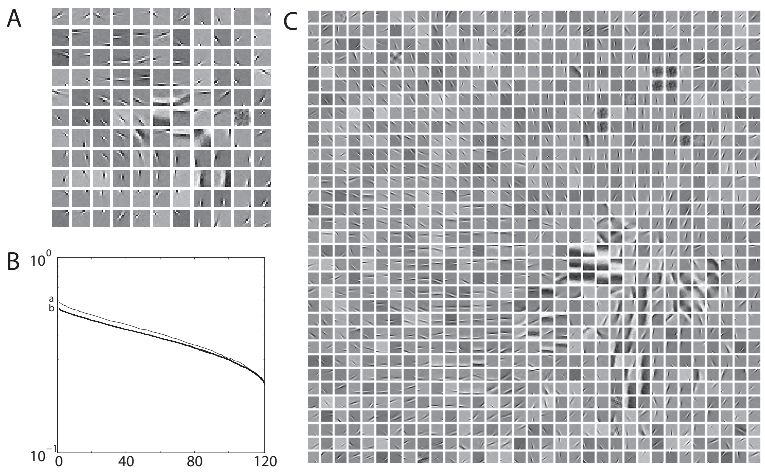

Fig. 2. A: Learned bases of phase-independent, topographic infomax; B: PCA of outputs u (eigenvalues of Q shown on a continuous function). Curves labeled a and b result from Linsker network (standard infomax) and from phase-independent topographic infomax (A and C overlapping), and show maximization of the output volume (successful infomax). C: Learned bases of phase-independent, overcomplete, topographic infomax.

7 Discussion

The infomax algorithm implemented here in a modified multi-layered Linsker network maximizes $I(X;Y)$, the mutual information between network inputs and outputs. Rather than doing so directly, however, the novel network configuration uses first a partial neural multigrid to induce spatial correlations in its output, then a complete multigrid to perform the infomax optimization. The manner in which the multigrid guides infomax to a specific topographically organized optimum constitutes a new principle of information maximization, which we refer to as topographic infomax.

The problem solved by the partial neural multigrid first is that of eliminating redundancy in low frequency Fourier modes of the output. While solving this problem for these modes and no others, the network operates as if these modes contain *all* output redundancy, which they clearly do not. Redundancy between large regions of the output map is minimized, even though doing so increases redundancy in higher frequency modes (ie., between individual units). In the completed multigrid, iteration in coarse grids precedes iteration in finer grids for any input, and thus higher frequency redundancy reduction is heavily constrained by any previous minimization of redundancy in lower frequency modes. For standard infomax, redundancy minimization can be achieved by redundancy reduction in any or all modes simultaneously. Topographic infomax instead aims to first eliminate low frequency redundancy and thus imposes a topographic order on the output map.

The constraints imposed by low frequency redundancy reduction can prevent infomax from emerging in the completed multigrid. Two approaches to relaxing these constraints and recovering the infomax solution have yielded topographic infomax: first, the use of an overcomplete basis, and second, the use of a phase-independent order-embedding. We anticipate that many parallels between the network configuration employed here and those observed in biological structures such as primate primary visual cortex remain to be drawn, and that topographic infomax represents a mechanism by which these structures, constrained developmentally by local network connection topologies, can achieve quantities of mutual information between inputs and outputs comparable to what is achieved in more theoretical, fully-connected networks of equal dimension.

Acknowledgments

We thank Ralph Linsker and John Wagner for many helpful discussions.

References

1. Kohonen, T.: Self-Organizing Maps. Berlin: Springer-Verlag (1997)
2. Kohonen, T.: Learning Vector Quantization. Neural Networks 1, Supplement 1 (1988) 303
3. Desieno, D.: Adding a Conscience to Competitive Learning. Proc. Int. Conf. on Neural Networks **I** (1988) 117-124
4. Bednar, J. A., Kelkar, A., Miikkulainen, R.: Scaling Self-Organizing Maps to Model Large Cortical Networks. Neuroinformatics **2** (2004) 275-302
5. Linsker, R.: Local Synaptic Learning Rules Suffice to Maximise Mutual Information in a Linear Nnetwork. Neural Computation **4** (1992) 691-702
6. Bell, A. J., Sejnowski, T. J.: An Information-Maximisation Approach to Blind Separation and Blind Deconvolution. Neural Computation **7** (1995) 1129-1159
7. Shriki, O., Sompolinsky, H., Lee, D.D.: An Information Maximization Approach to Overcomplete and Recurrent Representations. 12th Conference on Neural Information Processing Systems (2000) 87-93
8. Olshausen, B. A., Field, D. J.: Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1. Vision Research **37** (1996) 3311-3325
9. Lewicki, M. S., Sejnowski, T. J.: Learning Overcomplete Representations. Neural Computation **12** (2000) 337-365
10. Linsker R.: How to Generate Ordered Maps by Maximizing the Mutual Information between Input and Output Signals. Neural Computation **1** (1989) 402-411
11. Linsker, R.: A Local Learning Rule that Enables Information Maximization for Arbitrary Input Distributions. Neural Computation **9** (1997) 1661-1665
12. Hyvärinen, A., Hoyer, P. O.: A Two-Layer Sparse Coding Model Learns Simple and Comlex Cell Receptive Fields and Topogrpahy From Natural Images. Vision Research **41** (2001) 2413-2423
13. Briggs, W.L., Henson, V.E., McCormick, S.F.: A Multigrid Tutorial, Phildelphia, PA: Society for Industrial and Applied Mathematics
14. Hyvärinen, A., Hoyer, P.O., Inki, M.: Topographic Independent Component Analysis. Neural Computation **13** (2001) 1527-1528