September 1991

LU TP 91-21

Rotor Neurons - Basic Formalism and Dynamics

Lars Gislén[1], Carsten Peterson[2] and Bo Söderberg[3]

Department of Theoretical Physics, University of Lund
Sölvegatan 14A, S-22362 Lund, Sweden

Neural Computation **4**, 737 (1992)

Abstract:

Rotor neurons are introduced to encode states living on the surface of a sphere in D dimensions. Such rotors can be regarded as continuous generalizations of binary (Ising) neurons. The corresponding mean field equations are derived, and phase transition properties based on linearized dynamics are given. The power of this approach is illustrated with an optimization problem – placing N identical charges on a sphere such that the overall repulsive energy is minimized. The rotor approach appears superior to other methods for this problem both with respect to solution quality and computational effort needed.

[1] larsg@thep.lu.se
[2] carsten@thep.lu.se
[3] bs@tf1.lu.se

Background

Standard McCulloch-Pitts neurons are characterized by sigmoidal updating equations

$$v_i = g(u_i) = \tanh u_i \tag{1}$$

where the local field u_i is given by

$$u_i = \sum_j \omega_{ij} v_j / T \tag{2}$$

and the inverse temperature 1/T sets the gain. The neurons are binary in the high gain (T$\rightarrow$ 0) limit. In feed-back networks [1] with a quadratic energy function in terms of binary neurons s_i,

$$E = -\frac{1}{2} \sum_{i,j} \omega_{ij} s_i s_j \ , \tag{3}$$

the iterative solutions of the mean field equations (eqs (1,2)) represent approximate minima to E for appropriate values of T, where $v_i =< s_i >_T$. In the more general case one has

$$u_i = -\frac{1}{T} \frac{\partial E}{\partial v_i} \tag{4}$$

In a series of papers, we have investigated the generalization of this approach to multi-state (Potts) neurons, which are superiour in situations where one wants only one of the s_i (v_i) to be "on" and the others "off". In effect, eq. (1) is replaced by [4]

$$v_i = \frac{e^{u_i}}{\sum_j e^{u_j}} \tag{5}$$

Such a constrained encoding turns out crucial for many optimization applications [2, 3]. Potts neurons also play a crucial role in deformable templates or so-called elastic net methods [4]. In feed-forward networks with exclusive classifications Potts neuron encoding of the output layer could be profitable [5].

In the present paper we investigate the generalization from binary neurons to the case of a continuum of states on a D-dimensional sphere, and apply the method to the problem of optimal configuration of charges on a sphere.

Rotor Neurons

Consider the general problem of minimizing an energy function $E(\vec{s}_1, \ldots, \vec{s}_N)$ with respect to a set of N D-dimensional unit vectors $\vec{s}_i$ (hereafter denoted *rotors*)

$$|\vec{s}_i| = 1 \ . \tag{6}$$

[4] Using [0,1] representation rather than the [-1,1] of eq. (1).

A locally minimal configuration must satisfy

$$\vec{s}_i = -\nabla_i E / |\nabla_i E| \; . \tag{7}$$

Local optimization consists of iterating these equations until convergence. This is in general not a very good method for finding the global minimum: the configurations easily get stuck in local minima. A more careful method is *simulated annealing* [6], where a thermal distribution $\propto \exp(-E/T)$ is simulated. The temperature T is very slowly lowered, until a stable state results. This method is very time-consuming, if a good result is desired.

For this kind of problem, we suggest a *mean field theory* (MFT) rotor approach analogous to what is used for combinatorial optimization problems [1, 2].

Mean Field Equations

Consider a thermal distribution of configurations, characterized by the *partition function* Z,

$$Z = \int e^{-E[\vec{s}]/T} ds_1 \ldots ds_N \; , \tag{8}$$

where the simplified notation ds_i means that the integration is to be performed over the directions only, and normalized such that $\int ds_i = 1$.

For simplicity, consider first a single integral $I = \int H(\vec{s}) ds$, with H an arbitrary function. It can be rewritten as

$$\int H(\vec{s}) ds = \int g(\vec{v}) \delta(\vec{s} - \vec{v}) ds d\vec{v} \propto \int H(\vec{v}) e^{\vec{u}\cdot(\vec{s}-\vec{v})} ds d\vec{v} d\vec{u} \; . \tag{9}$$

Performing the $\vec{s}$ integral, one is left with

$$I \propto \int H(\vec{v}) e^{-\vec{v}\cdot\vec{u} + F(u)} d\vec{v} d\vec{u} \; , \tag{10}$$

where $u = |\vec{u}|$ and $F(u)$ is defined by

$$F(u) = \log \int e^{\vec{u}\cdot\vec{s}} ds \; . \tag{11}$$

This evaluates to

$$F(u) = \log I_{(D-2)/2}(u) - \frac{D-2}{2} \log u + const. \tag{12}$$

where I_n are modified Bessel functions.

Repeating this trick N times with the multiple integral Z, we obtain

$$Z \propto \int e^{-E[\vec{v}]/T - \sum_i \vec{v}_i \cdot \vec{u}_i + \sum_i F(u_i)} d\vec{v}_1 d\vec{u}_1 \ldots d\vec{v}_N d\vec{u}_N \; . \tag{13}$$

D	$g(u)$	$g(u),\ u \to 0$	$g(u), u \to \infty$
1	$\tanh u$	u	$1 - 2e^{-2u}$
2	$I_1(u)/I_0(u)$	$u/2$	$1 - 1/2u$
3	$\coth u - 1/u$	$u/3$	$1 - 1/u$
4	$I_2(u)/I_1(u)$	$u/4$	$1 - 3/2u$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
D	$I_{D/2}(u)/I_{(D-2)/2}(u)$	u/D	$1 - (D-1)/2u$

Table 1: $g(u)$ for different dimensions D.

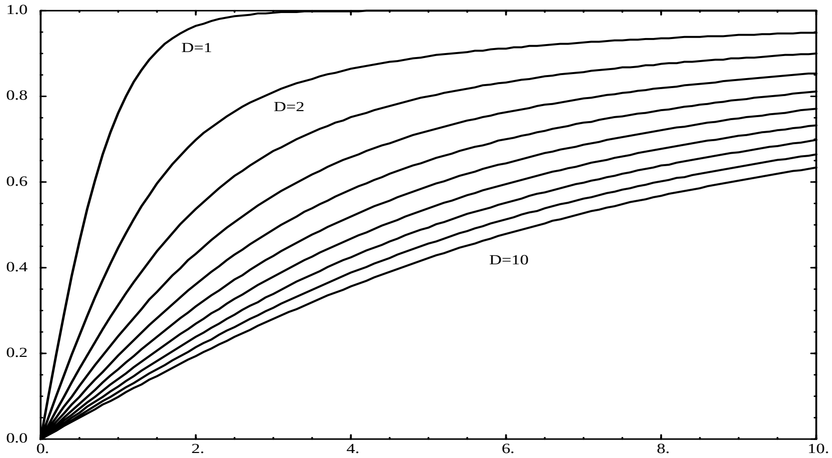

Figure 1: $g(u)$ for different dimensions D.

Next, we seek a saddlepoint of the effective potential in the exponent of eq. (13), by demanding vanishing derivatives, leading to the *mean field* equations

$$\vec{u}_i = -\frac{1}{T}\nabla_i E(\vec{v}) \tag{14}$$

$$\vec{v}_i = \vec{g}(\vec{u}_i) \equiv \hat{u}_i g(u_i) \equiv \hat{u}_i F^{'}(u_i) \tag{15}$$

where $\hat{u}_i = \vec{u}_i/u_i$. They give $\vec{v}_i$ as the average of $\vec{s}_i$ in the *local field* $\nabla_i E(\vec{v})$. In table 1 $g(u)$ is listed for different values of D. When $D = 1$, eq. (1) is recovered. The corresponding curves are shown in fig. 1. In the large D limit, the shape of g is given by

$$\lim_{D\to\infty} g(Du) = \frac{\sqrt{4u^2+1}-1}{2u} \tag{16}$$

We regard this system as a neural network, with $\vec{v}_i$ as generalized neurons, $\vec{g}$ as a generalized

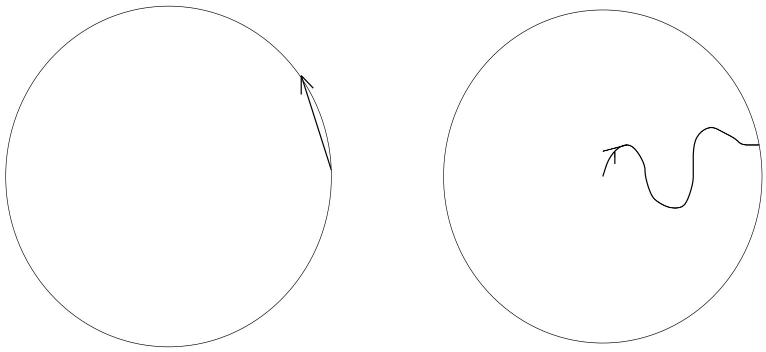

Figure 2: **(a)**. Elementary moves on the unit sphere. **(b)**. Evolution of a MF rotor initialized close to the center, for $T < T_c$.

sigmoid and $\vec{u}_i$ as inputs (local fields). The obvious dynamics consists in iterating eqs. (14,15). The performance of the neural network thus defined will depend on T; this is to be discussed next.

Critical Temperature Estimation

From eq. (15) we infer that for small u_i, $\vec{v}_i \approx \vec{u}_i / D$. Making the simplifying assumption that E is rotationally invariant, we can approximate E for small $\vec{v}$ with a quadratic form

$$E = E_0 - \frac{1}{2} \sum_{i,j} \omega_{ij} \vec{v}_i \cdot \vec{v}_j + O(v^2) \tag{17}$$

But then $\vec{v}_1 = \ldots = \vec{v}_N = 0$ is a fixpoint of the updating eqs. (14,15). Linearizing these for small v, we obtain

$$\vec{v}_i = \frac{1}{DT} \sum_j \omega_{ij} \vec{v}_j \tag{18}$$

If the temperature is higher than the *critical temperature*

$$T_c = \frac{1}{D} \max(\lambda_{\max}, -\lambda_{\min}) \ , \tag{19}$$

where $\lambda_{\min/\max}$ are the extreme eigenvalues of ω, this trivial fixpoint is stable under *synchronous* updating, and the system is in a symmetric phase. For a lower T, it becomes unstable, and the mean fields $\vec{v}_i$ will be repelled by the origin. For low enough temperature they will stabilize close to the sphere $\vec{v}_i^2 = 1$ (cf. fig. 2). The dynamics is thus very different from that of local optimization and simulated annealing, where moves take place on the surface.

For *serial* updating, things are similar, although T_c is different. In the special case of a constant self-coupling $\omega_{ii} = \mu$, we have instead

$$T_c = \frac{1}{D} \max(\lambda_{\max}, -\mu) \tag{20}$$

Thus, for a large set of energy-functions, we can estimate T_c in advance. A good strategy is then to initialize the system close the symmetry-point, with T close to T_c, and slowly anneal while the system settles. When finally a stable state is reached, a possible solution to the minimization problem is extracted.

For both types of updating, it turns out advantageous to be able to adjust the self-coupling to achieve maximum stability. This is done by adding a term $-\frac{\beta}{2}\sum_i v_i^2$ to the energy. For a more detailed discussion of β and T_c, the reader is referred to [2].

Placing Charges on a Sphere

We now turn to the specific problem of the equilibrium configuration of N equal charges on a sphere ($D = 3$). With Coulomb interaction between the charges the energy function is given by

$$E = \frac{1}{2\sqrt{2}} \sum_{i \neq j} \frac{1}{(1 - \vec{s}_i \cdot \vec{s}_j)^{\frac{1}{2}}} - \frac{\beta}{2} \sum_i \vec{s}_i^2 \tag{21}$$

where we for dynamical reasons have added a β-term as discussed above. With this energy function the local field $\vec{u}_i$ (cf. eq. (14) is given by

$$\vec{u}_i = \frac{1}{T} \left(-\frac{1}{2\sqrt{2}} \sum_{i \neq j} \frac{\vec{v}_j}{(1 - \vec{v}_i \cdot \vec{v}_j)^{\frac{3}{2}}} + \beta \sum_i \vec{v}_i \right) \tag{22}$$

with the corresponding updating equation (see table 1)

$$\vec{v}_i = \hat{u}_i(\coth u_i - 1/u_i) \tag{23}$$

The critical temperature T_c for this system in serial updating mode is, for reasonable β-values (cf. eq. (19))

$$T_c = (\beta + \tfrac{1}{2\sqrt{2}})/3 \tag{24}$$

The β-term also affects the low temperature behaviour, controlling the tendency for the system to remain in a local minimum. A good final configuration, with the charges uniformly distributed over the sphere, should be stable. This means that the updated $\vec{u}_i$'s should satisfy

$$\vec{u}_i(t) \cdot \vec{v}_i(t-1) > 0 \tag{25}$$

A necessary condition for accomplishing this can be derived for large N. The result is

$$\beta > \beta_0 \approx \frac{N^{3/2}}{4} \tag{26}$$

The role of β is thus twofold, it controls the phase transition *and* the dynamical behaviour at low temperatures. Equipped with prior estimates of T_c and β_0, the algorithm for a given problem size can take the following "black box" form

1. Compute T_c and β_0 according to eqs. (24,26).
2. Initialize with $v_i^\alpha = 0.01\cdot$ rand[-1,1].
3. Update all $\vec{v}_i$'s in turn according to eqs. (22,23).
4. Decrease the temperature, $T \to 0.95 \cdot T$.
5. Repeat from step 3, until the saturation [2] $\sum_i \vec{v}_i^2/N > 0.99$.
6. Extract configuration by setting $\vec{s}_i = \hat{v}_i$.

Using this prescription we have computed configurations of 3, 5, 10, 20, 30 and 100 charges, respectively. In fig. 3 the time consumption as a function of problem size is shown. As expected, the MFT rotor algorithm scales asymptotically like N^2.

As for the quality of the solution, the MFT rotor model gives the correct solutions where these are explicitly known ($N = 2, 3, 4, 6$). For larger problems we have compared the solutions with those from the gradient descent (GD) and simulated annealing (SA) algorithms. In GD the basic move is a perpendicular move of size $1/N$. The same step size was used in SA with the annealing schedule $T \to 0.95 \cdot T$. These algorithms were clocked when the energy was within 1% of the MFT rotor result. The time consumption to achieve this is shown in fig. 3. In fig. 4 the evolution of the rotors for an $N = 32$ problem is shown.

Comparing the MFT rotor approach with the conventional ones we find that for the MFT rotor algorithm the number of sweeps needed to reach a satifactory solution is practically independent of problem size, while for the other methods it is (with optional step size) roughly proportional to the problem size. As for quality, the final energies obtained by the MFT rotor approach were always equal to or lower than the ones obtained with the other approaches.

We have also run the MFT rotor algorithm for a $D = 3$ system, where we substituted the appropriate sigmoid in eq. (23) with a properly scaled tanh-function

$$v_i = \hat{u}_i \tanh(u_i/3) \; . \tag{27}$$

We find that the algorithm performs as well (if not better) with respect to the number of sweeps needed to obtain a good solution. The reason for this investigation is that this sigmoid is more natural in a VLSI implementation.

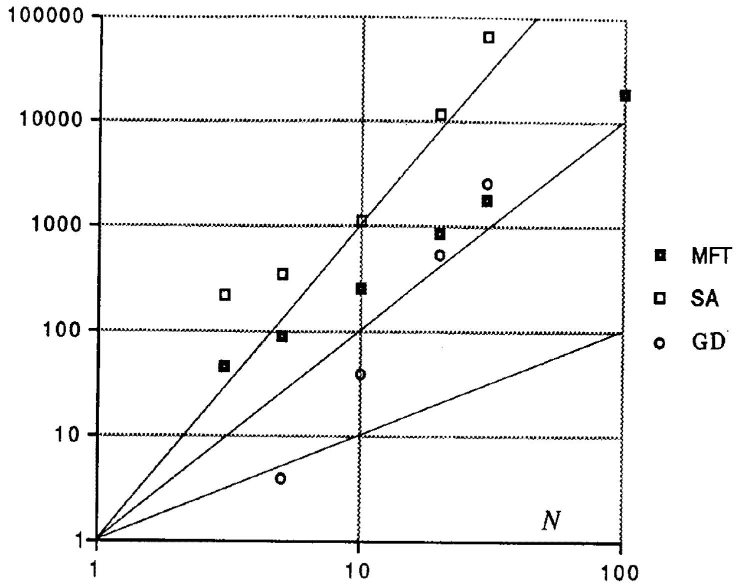

Figure 3: Time consumption as a function of problem size (N) for the MFT rotor (MFT), gradient descent (GD) and simulated annealing (SA) algorithms. Both axes are logarithmic. The three lines correspond to N, N^2 and N^3 respectively.

Summary

Formalism and dynamics for D-dimensional feed-back rotor neurons have been developed. For $D = 1$ one recovers the conventional sigmoidal updating equations. As a first test bed for this approach in higher dimensions we applied the method to the problem of finding optimal charge configuration on a $D = 3$ sphere. The performance of the rotor method seems to be superior to that of other approaches for this problem.

Other potential problem areas of more practical use are e.g. curve detection in the early vision system [7], or the reconstruction of tracks from signals [8]. The $D > 1$ updating equations can of course also be used in feed-forward multi-layered networks.

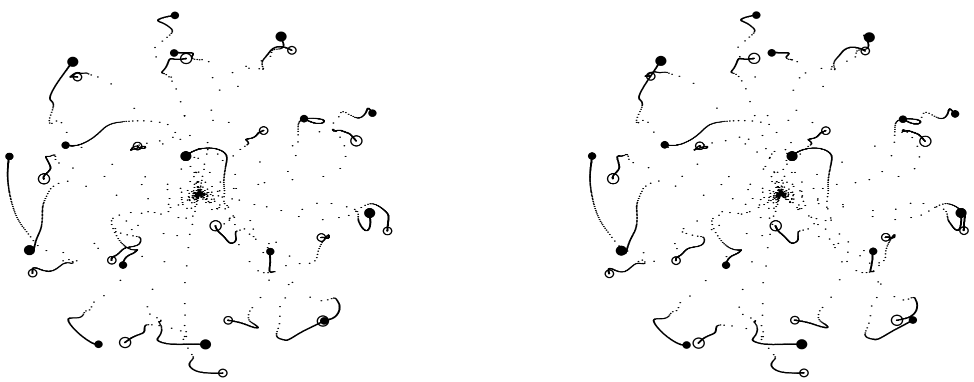

Figure 4: Evolutions of rotors for a N=32 problem as T decreases. Open and filled dots represent charges placed in front and the back of the sphere respectively. The two graphs are generated to provide a stereo effect.

References

[1] J.J. Hopfield and D.W. Tank, "Neural Computation of Decisions in Optimization Problems", *Biological Cybernetics* **52**, 141 (1985).

[2] C. Peterson and B. Söderberg, "A New Method for Mapping Optimization Problems onto Neural Networks", *International Journal of Neural Systems* **1**, 3 (1989).

[3] C. Peterson, "Parallel Distributed Approaches to Combinatorial Optimization", *Neural Computation* **2**, 261 (1990).

[4] R. Durbin and G. Willshaw, "An Analogue Approach to the Travelling Salesman Problem Using an Elastic Net Method", *Nature* **326**, 689 (1987).

[5] L. Lönnblad, C. Peterson and T. Rögnvaldsson, "Using Neural Networks to Identify Jets", *Nuclear Physics* **B 349**, 675 (1991).

[6] S.Kirkpatrick, C.D.Gelatt and M.P.Vecchi, "Optimization by Simulated Annealing", *Science* **220**, 671 (1983).

[7] See e.g. S.W. Zucker, A. Dobbins and L. Iverson, "Two Stages of Curve Detection Suggest Two Styles of Visual Computation", *Neural Computation* **1**, 68 (1990).

[8] C. Peterson, "Neural Networks and High Energy Physics", *Proc. of International Workshop on Software Engineering, Artificial Intelligence and Expert Systems for High Energy and Nuclear Physics, Lyon Villeurbanne, France, March , 1990*, eds. D. Perret-Gallix and W. Woijcik, Editions du CRNS (Paris 1990).