Physica A 200 (1993) 570–580
North-Holland

SDI: 0378-4371(93)E0147-7

PHYSICA A

Solving optimization problems with mean field methods

Carsten Peterson[1]

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

A brief review is given for the use of feed-back artificial neural networks (ANN) to obtain good approximate solutions to combinatorial optimization problems. The key element is the mean field approximation (MFT), which differs from conventional methods and "feels" its ways towards good solutions rather than fully or partly exploring different possible solutions. The methodology, which is illustrated for the graphs bisection and knapsack problems, is easily generalized to Potts systems. The latter is related to the deformable templates method, which is illustrated with the track finding problem.

The mean field approximation is based on a variational principle, which also turns out to be very profitable when computing correlations in polymers.

1. Introduction

Many combinatorial optimization problems are NP-complete, which require state space explorations leading to $\mathcal{O}(a^N)$ computations for a system with N degrees of freedom. Different kind of heuristic methods are therefore often used to find reasonably good solutions. The artificial neural network (ANN) approach falls within this category. Whereas the use of ANN for pattern recognition and prediction problems is a non-linear extension of conventional linear interpolation/extrapolation methods, ANN in the optimization domain really brings something new to the "table". In contrast to existing search and heuristics methods the ANN approach does not fully or partly explore the different possible configurations. Rather it "feels" its way in a fuzzy manner towards good solutions. The key element in this approach is the mean field approximation (MFT) [1,2], which can be regarded as a variational scheme. The techniques involved, which results in high quality solutions, are illustrated in some detail for the graph bisection and knapsack problems. The latter requires some extra care since it contains inequality constraints. For low-dimensional geometrical problems like the traveling salesman problem (TSP)

[1] E-mail: carsten@thep.lu.se

and track finding a variant of the "pure" neural approach, *deformable templates*, is advantageous to use.

Recent efforts to use variational methods for computing polymer configurations are also discussed. These optimization problems are not combinatorial but the techniques involved are strongly related to MFT.

2. The graph bisection problem

The neural approach is particularly transparent for this problem due to its binary nature. Consider a set of N nodes to be partitioned into two halves such that the connectivity (cutsize) between the two halves is minimized (see fig. 1). The problem is mapped onto the energy function [3],

$$E[s] = -\frac{1}{2}\sum_{ij}\omega_{ij}s_i s_j - \frac{\alpha}{2}\left(\sum_i s_i\right)^2, \tag{1}$$

where for each node a binary neuron $s_i = \pm 1$ is assigned depending on whether node i is in the left or in the right position. For each pair of nodes $\omega_{ij} = 0, 1$ depending on whether i and j are connected or not. The first term in eq. (1) minimizes the connections between partitions, whereas the second penalizes unbalanced configurations ($\Sigma s_i \neq 0$). The imbalance parameter α sets the relative strength between the cutsize and the balancing term. This *global* balancing term is typical for difficult combinatorial optimization problems.

The next step is to find an efficient procedure for minimizing eq. (1) such that local minima are avoided as much as possible. Rather than doing this with the discrete updating rule $s_i = \text{sgn}\ \Sigma(\omega_{ij} - \alpha)$ in a temperature environment (simulated annealing [4]) the MFT approximation is used. This amounts to approximate the stochastic simulated annealing procedure with a set of deterministic equations,

$$v_i = \tanh\left(-\frac{\partial E[v]}{\partial v_i}\frac{1}{T}\right) = \tanh\left(\sum_j (\omega_{ij} - \alpha)\frac{v_j}{T}\right). \tag{2}$$

Here one needs to know in advance the approximate T-values – the phase

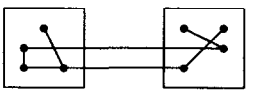

Fig. 1. A graph bisection problem.

transition temperature T_c. The dynamics of eq. (2) exhibits a behaviour with two phases separated by $T = T_c$: at large enough temperatures ($T \to \infty$) the system has a trivial fixed point $v_i^{(0)} = 0$ and as $T \to 0$ fixed points $v_i^{(*)} = \pm 1$ emerge representing a specific solution (see fig. 2). The position of T_c, which depends on ω_{ij} and α, can be estimated by expanding the sigmoid function in a power series around $v_i^{(0)} = 0$ (see fig. 2). The fluctuations around $v_i^{(0)}$, $v_i = v_i^{(0)} + \epsilon_i$, satisfy

$$\epsilon_i = \frac{1}{T} \sum_j m_{ij} \epsilon_j \,, \tag{3}$$

where $m_{ij} = \omega_{ij} - \alpha$. For synchronous updating it is clear that if one of the eigenvalues of the matrix $\boldsymbol{m}/T$ in eq. (3) is >1 in absolute value, the fixed point becomes unstable and the solutions will wander away into the nonlinear region. In the case of serial updating the philosophy is the same but the analysis slightly more complicated [2]. Finding the largest eigenvalue of $\boldsymbol{m}$ is easy: multiply the matrix with itself a few times. From this procedure one reads off T_c. One can always add from an encoding point of view auxiliary terms and diagonal terms in order to modify the eigenvalues and thereby control the dynamics (i.e., avoid cyclic behaviour). This linearized dynamics analysis thus both gives us means to control dynamics and to remove one free parameter – the temperature.

For problems that are not binary, e.g. graph partition, TSP and Scheduling problems, the Ising neuron procedure above is easily generalized to Potts neurons [5,2,6,7]. The MFT equations for Potts neurons s_{ia}, satisfying $\Sigma_a s_{ia} = 1$, reads

$$v_{ia} = \frac{e^{u_{ia}}}{\Sigma_b e^{u_{ib}}} \,, \tag{4}$$

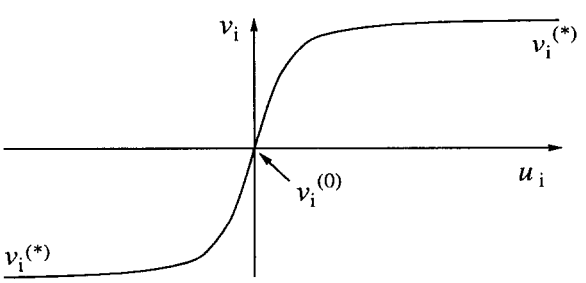

Fig. 2. Fixed points in $\tanh(u_i)$.

where $u_{ia} = (1/T)(\partial E/\partial v_{ia})$. Very competitive solution qualities have been obtained using Potts neurons for realistic scheduling problems [7].

3. The knapsack problem

The graph partition problem is characterized by an *equality* constraint, which is implemented with a polynomial penalty term. However, in many optimization problems, in particular those of resource allocation type, one has to deal with *inequalities*. One such problem category is the knapsack problem, where one has a set of N *items* i with associated *utilities* c_i and *loads* a_{ki}. The goal is to fill a "knapsack" with a subset of the items such that their total utility,

$$U = \sum_{i=1}^{N} c_i s_i \,, \tag{5}$$

is maximized, subject to a set of M load constraints,

$$\sum_{i=1}^{N} a_{ki} s_i \leq b_k \,, \qquad k = 1, M \,. \tag{6}$$

In eqs. (5), (6) s_i are binary $\{0, 1\}$ decision variables, representing whether item i goes into the knapsack or not. The variables c_i, a_{ki} and b_k that define the problem are all real numbers. We consider the most difficult problems, where a_{ki} and c_i are independent uniform random numbers on the unit interval and $b_k = N/2$ [8]. The expected number of used items in an optimal solution will then be about $N/2$. The optimal solution to such a problem contains a strong correlation between the value of c_i and the probability for s_i to be 1. Simple heuristic based on this observation often produces near-optimal solutions very fast. We will therefore also consider a class of harder problems with more homogeneous c_i distributions [8]. The extreme case is when c_i is constant, and the utility proportional to the number of items used. *Set covering* represents a special case of the general problem with random $a_{ki} \in \{0, 1\}$, and all $b_k = 1$. This is a comparatively simple problem according to the above.

We map the problem onto an energy function E [8],

$$E = -\sum_{i=1}^{N} c_i s_i + \alpha \sum_{k=1}^{M} \Phi\Big(\sum_{i=1}^{N} a_{ki} s_i - b_k\Big) \,, \tag{7}$$

where Φ is a penalty function ensuring that the constraint in eq. (6) is fulfilled. The coefficient α governs the relative strength between the utility and

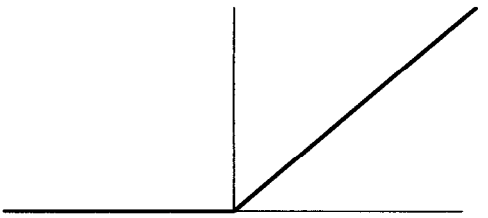

Fig. 3. The penalty function $x\theta(x)$.

constraint terms. An appropriate choice of $\Phi(x)$ is $x\Theta(x)$ (see fig. 3). The slope of Φ is implicitly given by α in eq. (7). Minimizing eq. (7) is done with the MFT equations. Due to the non-polynomial form of the constraint, some special care is required here. The derivative $\partial E/\partial v_i$ in eq. (2) is replaced by a difference[#1]

$$\frac{\partial E}{\partial v_i} \to -c_i + \alpha \sum_{k=1}^{M} \left[\Phi\left(\sum_{j=1}^{N} a_{kj} v_j - b_k \right) \Big|_{v_i=1} - \Phi\left(\sum_{j=1}^{N} a_{kj} v_j - b_k \right) \Big|_{v_i=0} \right]. \qquad (8)$$

The MFT equations are solved iteratively by annealing in T with a more or less automized scheme [8].

In ref. [8] this ANN approach is compared with other approaches. For small problem sizes it is feasible to use an exact algorithm, branch and bound, for comparison. For larger problem sizes, one is confined to other approximate methods, such as simulated annealing [4], greedy heuristics and simplex based on linear programming [9].

With the branch and bound (BB) tree search technique one can reduce the number of computational steps as compared to exhaustive search by proceeding down a search tree, checking bounds on constraints or utility for entire subtrees, thereby avoiding unnecessary searching. In particular for non-homogeneous problems, this method is accelerated by ordering the c_i's according to magnitude: $c_1 > c_2 > \cdots > c_N$. Greedy heuristics (GH) is a simple and fast approximate method for a non-homogeneous problem. Proceeding from larger to smaller c_i one collects every item not violating constraints. This method scales like NM. Simulated annealing (SA) [4] is easily implemented in terms of single-spin flips, subject to the constraints. This method also scales like NM times the number of iterations needed for thermalization and cooling. Linear programming (LP) based on the simplex method [9] is not designed to solve discrete problems like the knapsack one. It does apply, however, to a modified problem with $s_i \in [0, 1]$. For the ordered non-homogeneous knapsack

[#1] In fact, this trick can be used to kill self-couplings for any energy function, not necessarily with inequality constraints.

Table I
Comparison of performance and CPU time consumption for the different algorithms on a $N = M = 30$ problem. The CPU consumption refers to a DEC3100 workstation.

Algorithm	c_i = rand[0, 1]		c_i = rand[0.45, 0.55]		$c_i = 0.5$	
	perf.	CPU time	perf.	CPU time	perf.	CPU time
BB	1	16	1	1500	1	1500
NN	0.98	0.80	0.95	0.70	0.97	0.75
SA	0.98	0.80	0.95	0.80	0.96	0.80
LP	0.98	0.10	0.93	0.25	0.93	0.30
GH	0.97	0.02	0.88	0.02	0.85	0.02

problem this gives solutions with a set of leading 1's and a set of trailing 0's, with a window in between containing real numbers. Augmented by greedy heuristics for the elements in this window, fairly good solutions emerge. The simplex method scales like N^2M^2. In table I the NN, SA, GH and LP approaches are compared with the exact BB for $N = M = 30$ non-homogeneous and homogeneous problems. As expected, LP and in particular GH benefit from non-homogeneity both quality- and CPU-wise while for homogeneous problems the NN algorithm is the winner. For larger problem sizes it is not feasible to use the exact BB algorithm. The best we can do is to compare the different approximate approaches, NN, SA and LP. The conclusions from problem sizes ranging from 50 to 500 are the same. The real strength in the NN approach is best exploited for more homogeneous problems.

In summary, the MFT approach is very competitive as compared to other approximate methods for the hard homogeneous knapsack problems, both with respect to solution quality and time consumption. It is also compares very well with exact solutions for problem sizes where these are accessible. Being able to find good approximate solutions to difficult knapsack problems opens up several application areas within the field of resource allocation.

4. Deformable templates

For low-dimensional geometric optimization problems it is often advantages to use the deformable templates approach, thereby reducing the number of degrees of freedom. This approach was successfully applied to TSP [10]. A similar approach is also profitable for the particle physics track finding problem [11–13].

Track finding is the assignment problem of connecting signal points with smooth curves. In particle physics it has been successfully approached with a "pure" neural approach [14,15] with encouraging results. The basic idea is to

assign neurons $s_{ij} = \pm 1$ between signals i and j to encode whether these are connected or not. An energy function is constructed in terms of s_{ij} such that smooth tracks with no bifurcations correspond to minima. In its original form this approach requires N^2 degrees of freedom for N signals. This assumes full potential connectivity. In reality this is however never the case due to the locality of the problem. Another local neural network inspired approach is to have a rotor (xy-model) neuron [17] associated with each signal. Even though the neural approach [14–16] works amazingly well, it may not be the optimal way to proceed for the following reasons:

(1) It only solves the combinatorial optimization part of the problem; assigns signals to tracks. One still has to make a fit to obtain the track parameters.

(2) The neural approach is more general than what is needed for this problem since the parametric form of the tracks are known in advance – straight lines or helices. Any powerful algorithm should take advantage of this prior knowledge of the parametric form. This is the way a human being would do it. To illustrate this point we show in fig. 4a signal points generated from 15 straight tracks with 100% noise. The human method of spotting the tracks in the noisy environment is by inspecting the image in fig. 4 by holding it up and look for straight lines[#2].

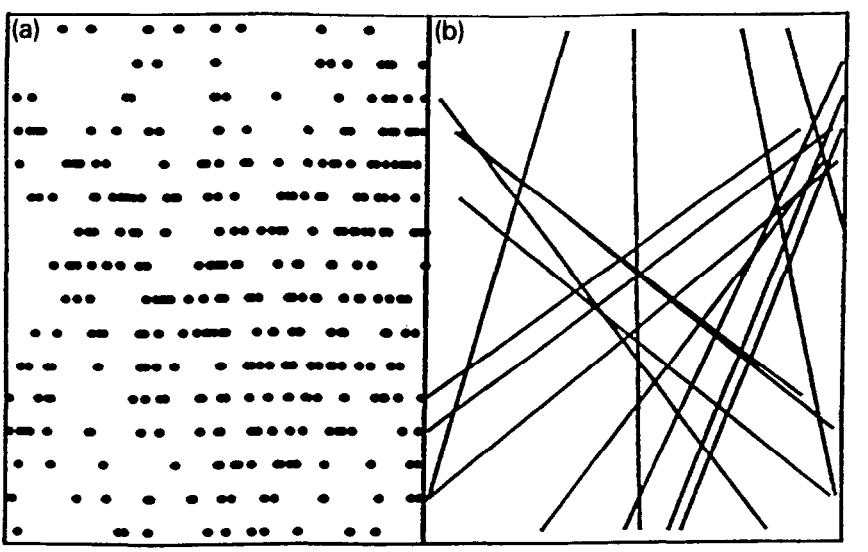

Fig. 4. (a) Signal points from 15 generated straight tracks with 100% noise. (b) The corresponding solution. From ref. [13].

[#2] The reader is encouraged to hold up fig. 4 to verify this.

(3) The number of degrees of freedom needed to solve an N signal problem is large even with the connectivity restrictions imposed in refs. [15,16]. For a problem with N signals and M tracks one should only need $\mathcal{O}(M)$ degrees of freedom.

(4) As demonstrated in ref. [13] the neural approach is somewhat sensitive to noise. Again with prior knowledge of the parametric form one should be more robust with respect to noise.

All these issues are accounted for in the deformable templates approach [11,13]. The strategy is to try to match the observed events to simple parametrized models – *templates* – the form of which reflects the a priori knowledge about the possible track shapes – helices passing through the origin (collision point). In addition, the formalism allows for the data points corresponding to noise to be unmatched[#3].

The algorithm works in two steps. First a Hough transform [19] is used to give initial conditions for the templates and to specify the number of templates required. Given a set of M templates defined by the parameters $\boldsymbol{\pi}_a = (\theta_a, \kappa_a, \gamma_a)$, $a = 1, \ldots, M$, from the Hough transform, the elastic arms method takes over and resolves ambiguities and makes detailed fits to the signals. The latter is based on minimizing

$$E[v_{ia}; \boldsymbol{\pi}_a] = \sum_{i,a} (M_{ia} - \lambda) v_{ia} , \tag{9}$$

where: (i) $\boldsymbol{\pi}_a$ are parameters defining the ath track template ($a = 1, \ldots, M$), (ii) $\boldsymbol{x}_i$ is the position of the ith signal ($i = 1, \ldots, N$), (iii) M_{ia} is a measure of the (squared) distance between the ith signal and the ath template, (iv) v_{ia} is the ath component of a Potts-like decision neuron – it is 1 if the ith point is assigned to the ath arm, and zero otherwise, and subject to constraint $\Sigma_a v_{ia} = 0, 1$ (in other words, given i there should be *at most* one a such that $v_{ia} = 1$. This allows for noise signals not being assigned to any track), and (v) λ is a penalty for the non-assignment of a signal point. In effect it introduces a distance cutoff such that signals with no templates within the distance $\sqrt{\lambda}$ tend to be assigned.

We want to minimize $E[v_{ia}, \boldsymbol{\pi}_a]$ with respect to $\boldsymbol{\pi}_a$ subject to the modified *Potts* constraint $\Sigma_a v_{ia} = 0, 1$. Note that with eq. (9) the problem is encoded in two kinds of variables – the "neurons" v_{ia} and the template coordinates $\boldsymbol{\pi}_a$. A Boltzmann distribution is introduced in order to avoid local minima,

$$P[v_{ia}, \boldsymbol{\pi}_a] = \frac{e^{-\beta E[v_{ia}; \boldsymbol{\pi}_a]}}{Z} . \tag{10}$$

[#3] The mechanism involved is closely related to redescending M-estimators used in robust statistics [18].

Summing over allowed configurations for $v_{i\alpha}$, one obtains the marginal probability

$$P_M[\boldsymbol{\pi}_a] = \frac{1}{Z} e^{-\beta E_{\text{eff}}[\boldsymbol{\pi}_a]}, \tag{11}$$

where we have introduced the effective energy E_{eff}. Gradient descent on E_{eff} gives for $\Delta\pi_a^{(k)}$ $(k = 1, 2, 3)$

$$\Delta\pi_a^{(k)} = -\eta \frac{\partial E_{\text{eff}}}{\partial \pi_a^{(k)}} = -\eta \sum_i \hat{V}_{ia} \frac{\partial M_{ia}}{\partial \pi_a^{(k)}}, \tag{12}$$

$$\hat{V}_{ia} = \frac{e^{-\beta M_{ia}}}{e^{-\beta\lambda} + \sum_b e^{-\beta M_{ib}}}. \tag{13}$$

In the $T \to 0$ limit these equations are nothing but the *k means* clustering updating equations (apart from the noise factor).

How does this algorithm work? At a high starting temperature a set of initial template arms are placed according to the Hough transform values for the parameters $\boldsymbol{\pi}_a$. The templates compete for the signals by means of Gaussian distributions of width $T = 1/\beta$ centered around the arm positions (see fig. 5). Initially each arm can attract many signals. The relative importance of the different signals is measured by the Potts factor (eq. (13)). As the temperature is lowered the different arms are attracted more exclusively only to nearby signals.

As discussed in connection with eq. (9), λ governs the amount of noise points or *outliers* the algorithm allows for. It enters the Potts factor (eq. (13)) much like an extra "null" component contributing to the denominator. For $\lambda \to \infty$ no signals are ignored and the "null" component vanishes. For finite values of λ the domain of attraction of the arms is cut off (for small T) at a distance $\sqrt{\lambda}$.

The performance of the algorithm has been successfully tested with simulated data from the CERN DELPHI TPC detector. The arms do not confuse one another, even when passing close, or crossing each other. The elastic arms approach is very similar to human processing for this kind of recognition problem. A human looks for helices in a global way and then makes fine-tuning adjustments.

5. A mean field approach to correlations in polymers

The MFT approximation is a variational approach – one approximates the true energy E with a trial one $E_0(\alpha_i)$ and optimizes with respect to the

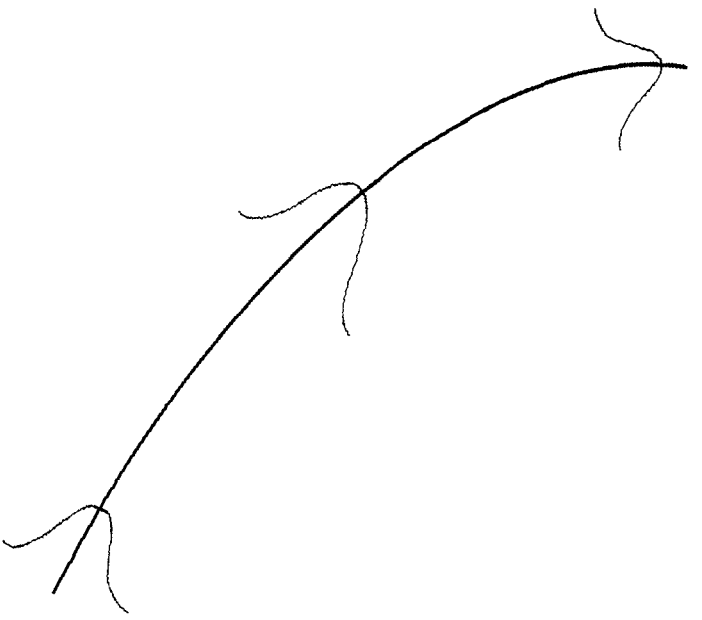

Fig. 5. An elastic arm at temperature T.

variational parameters α_i. This procedure can be used in a wide range of situations. Inspired by ref. [20], such a variational algorithm for computing correlations in polymers has been devised [21]. An N-atom polymer with nearest neighbour harmonic oscillator bonds and Coulombic self-repulsion is described by the energy

$$E = \frac{1}{2}\sum_i r_{i,i+1}^2 + \sum_{i<j} \frac{1}{r_{ij}} . \tag{14}$$

In a variational approach one makes the ansatz

$$\frac{E_0}{T} = \frac{1}{2}\sum_{ij} G_{ij}^{-1}(\boldsymbol{x}_i - \boldsymbol{a}_i)\cdot(\boldsymbol{x}_j - \boldsymbol{a}_j) . \tag{15}$$

The exact free energy $F = -T \log Z$ of the polymer is then approximated by

$$\hat{F} + F_0 + \langle E - E_0 \rangle_0 \geq F , \tag{16}$$

where $F_0 = -T \log Z_0$, and $\langle \ \rangle_0$ refers to averages with respect to the trial

Table II
$\langle r_{ee}^2 \rangle^{1/2}$ in units of $r_0 = 6$ Å as computed with the variational method (V) and Monte Carlo (MC). The errors originating from the Monte Carlo runs are estimated to be $\mathcal{O}(0.1\%)$.

N:	20	40	80	160
V	20.33	46.17	105.3	237.5
MC	19.83	45.67	100.7	230.1
diff.	2.5%	2.9%	4.6%	3.2%

Boltzmann distribution. The parameters G_{ij} and $\boldsymbol{a}_i$ are to be determined such that the variational free energy $\hat{F}$ is minimized. G_{ij} can be expressed as the product of a matrix with its $G_{ij} = \Sigma_{\mu=1}^{N-1} z_{i\mu} z_{j\mu} = \boldsymbol{z}_i \cdot \boldsymbol{z}_j$. Equations for a local extremum of $\hat{F}$ are obtained by differentiation with respect to $\boldsymbol{z}_i$ and $\boldsymbol{a}_i$. The computational demand scales roughly as N^3, with a constant of proportionality several orders of magnitude smaller than that of a reasonable statistics Monte Carlo (MC). The results for the important end-to-end correlations $\langle r_{ee}^2 \rangle^{1/2}$ are impressive when compared to the CPU demanding MC method (see table II).

References

[1] J.J. Hopfield and D.W. Tank, Biol. Cybern. 52 (1985) 141.
[2] C. Peterson and B. Söderberg, Int. J. Neural Syst. 1 (1989) 3.
[3] C. Peterson and J.R. Anderson, Complex Syst. 2 (1988) 59.
[4] S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi, Science 220 (1983) 671.
[5] I. Kanter and H. Sompolinsky, J. Phys. A 20 (1987) L673.
[6] L. Gislén, B. Söderberg and C. Peterson, Int. J. Neural Syst. (1989) 167.
[7] L. Gislén, B. Söderberg and C. Peterson, Neural Comput. 4 (1992) 808.
[8] M. Ohlsson, C. Peterson and B. Söderberg, Neural Comput. 5 (1993) 295.
[9] W.P. Press, B.P. Flannery, S.A. Teukolsky and W.T. Vettering, Numerical Recipes, The Art of Scientific Computing (Cambridge Univ. Pres, Cambridge, 1986).
[10] R. Durbin and D. Willshaw, Nature 326 (1987) 689.
[11] A. Yuille, K. Honda and C. Peterson, Particle tracking by deformable templates, in: Proc. 1991 IEEE INNS Inter. Joint Conf. on Neural Networks, vol. 1, Seattle, WA (July 1991) pp. 7–12.
[12] M. Ohlsson, C. Peterson and A. Yuille, Comput. Phys. Commun. 71 (1992) 77.
[13] M. Gyulassy and H. Harlander, Comput. Phys. Commun. 66 (1991) 31.
[14] B. Denby, Comput. Phys. Commun. 49 (1988) 429.
[15] C. Peterson, Nucl. Instrum. Methods A 279 (1989) 537.
[16] G. Stimpfl-Abele and L. Garrido, Comput. Phys. Commun. 64 (1991) 46.
[17] L. Gislén, B. Söderberg and C. Peterson, Neural Comput. 4 (1992) 727.
[18] P.J. Huber, Robust Statistics (Wiley, New York, 1981).
[19] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis (Wiley, New York, 1973).
[20] J.-P. Bouchaud, M. Mezard, G. Parisi and J.S. Yedida, J. Phys. A 24 (1991) L1025.
[21] B. Jönsson, C. Peterson and B. Söderberg, A variational approach to correlations in polymers, LU TP 92–24, Phys. Rev. Lett., in press.