SLAC-PUB-6632
SLAC/SSRL-0091
August 1994
(A/SSRL-M)

Faster Magnet Sorting with a Threshold Acceptance Algorithm*

Steve Lidia
Department of Physics, University of California, Davis, CA 95616
and
Roger Carr
Stanford Linear Accelerator Center
Stanford Synchrotron Radiation Laboratory, Stanford University, Stanford, CA 94309

We introduce here a new technique for sorting magnets to minimize the field errors in permanent magnet insertion devices. Simulated annealing has been used in this role, but we find the technique of threshold acceptance produces results of equal quality in less computer time. Threshold accepting would be of special value in designing very long insertion devices, such as long FEL's. Our application of threshold acceptance to magnet sorting showed that it converged to equivalently low values of the cost function, but that it converged significantly faster. We present typical cases showing time to convergence for various error tolerances, magnet numbers, and temperature schedules.

1. Introduction

At present, permanent magnet insertion devices usually require several hundred blocks of magnetic material. Even with strict quality control, there are errors in the magnitude and direction of the fields of these blocks, on the order of 1%. By carefully sorting the blocks into an optimum order, the errors may be canceled and the insertion device field improved substantially beyond what a random placement of blocks would yield. For a standard undulator, one wishes for RMS field errors on the order of 0.5%, and for an FEL undulator, one might require 0.1% or less. The sorting is done by measuring the magnetic moments of blocks, then feeding these values to a computerized sorting algorithm. Large scale sort optimization problems do not lend themselves to rigorous solutions, but do yield to Monte-Carlo techniques, such as simulated annealing [1]. If one uses, say, RMS magnetic field error as a merit criterion (or 'cost function'), one could sort magnets starting from a random ensemble, by trading magnets and keeping trades that resulted in improved cost function. This approach has the fatal defect that it finds only the local minimum in configuration space. It is really suitable only for the final stage of optimization, which we call 'quenching', that is done after global searching has been done to find the deepest 'valley' in configuration space.

Simulated annealing (SA) incorporates two concepts. The first is the Metropolis algorithm [2]. One executes trades, and calculates a cost function each time, as in quenching. But instead of keeping only trades that improve the cost function C, one also keeps trades that meet a Boltzmann criterion. If a unit uniform variant is less than $\exp[-(C_{new} - C_{old})/T]$, where T is a synthetic temperature, the trade is kept. This allows the sort to climb up 'hills' in configuration space, and access neighboring 'valleys'.

The other SA concept [1] is that one can 'anneal' the ensemble by lowering the temperature T. One can start at a high temperature and execute the Metropolis algorithm repetitively, until no further improvement is found. This is a way to explore a large region of configuration space. If one then lowers temperature, the region of configuration space is restricted; certain 'hills' become insurmountable. One could continue lowering temperature to zero to find the optimum; in practice, we lower temperature to a low value, then end with quenching to find the local minimum. This technique has already been used to sort magnet blocks for insertion devices [3].

The Boltzmann criterion is not the only one that might be used. Another technique is threshold acceptance (TA) [4]. In this scheme, a trade is kept if the change of cost function is no worse than a certain threshold value, which we still call

* Work supported by the US Department of Energy, Office of Basic Energy Sciences, Materials Sciences Division and the Department of Energy contract DE-AC03-76SF00515.

Presented at the International Conference on Synchrotron Radiation Instrumentation, (SRI '94), Stony Brook, NY, July 18-22, 1994

restricted, and an optimum is found. We quench at the end of this sort as well.

In the following, we show by example that TA obtains results as good as SA, but that it runs significantly faster. This is due to the extra execution cycles of the system random number generator and exponentiation functions. Dueck and Scheuer [4] note that SA explores a larger region of the configuration space than TA, but that TA converges to significantly lower values of the cost function than SA in the same number of steps. Since the time to sort an ensemble increases sharply with the number of magnets, TA would be preferable to SA for large sorting problems.

2. Cost Function

In insertion device design, one may imagine a variety of cost functions. The cost function could be as simple as the RMS deviation of the calculated field from a sine function. We use a variation of this concept. We consider only magnetic lattice geometries where the ideal dipole moment of each block aligns with either the longitudinal (z) or vertical (y) directions. Helmholtz coil measurements provide the Cartesian components of each block's dipole field. Averaging over the individual block volume, we determine the components of each block's remanent field. From the position of each block in the magnetic lattice, we know the ideal magnitude and sign of its remanent field components. We define the magnet errors as any field component except the ensemble average of the magnetization parallel to the easy axis. The cost function is then constructed by correlating these error field components with adjoining blocks. At each block location, the local cost function is calculated using

$$\begin{aligned} \mathrm{Cost}_n = \{ & (2B_{x_n} + B_{x_{n+1}} + B_{x_{n-1}})^2 \\ & + (2E_{y_n} + E_{y_{n+1}} + E_{y_{n-1}})^2 \\ & + (2E_{z_n} + E_{z_{n+1}} + E_{z_{n-1}})^2 \}^{1/2} \end{aligned}$$

where E is the corresponding error field in the Cartesian component of the nth block. The global cost function is then the ensemble average of the local cost function. We use this same cost function for SA, TA, and quenching. During execution, however, we only calculate for those blocks which are traded and their immediate neighbors. This prevents recalculation of the entire global cost function every time two blocks are traded, and greatly speeds execution.

The behavior of both SA and TA depend upon several parameters. The first is the number of magnet blocks in the set. Second is the schedule ratio between temperatures of successive iterations of the algorithm. Third is the relative error between values of the cost function in successive iteration cycles that must be achieved before the iteration completes. This parameter was used to determine when the algorithm has completed both a search at a given temperature and when it has completed all iteration cycles at the global level.

We first calculate the total cost function and then perform a number of random block trades to determine an average fluctuation of the cost function. This fluctuation is then doubled to give an initial temperature. A desired starting point for each algorithm is a temperature for which the ratio of accepted to attempted block trades is 80%. For TA, doubling the initial fluctuation was found to produce this acceptance ratio. But for SA, which has a tougher acceptance criteria, the temperature is repeatedly doubled until the acceptance ratio equals or exceeds 80%. The iteration cycles begin by reducing the temperature by the given schedule ratio factor, and then performing many random block trades until the relative change in the cost function has a magnitude no larger than the given relative error.

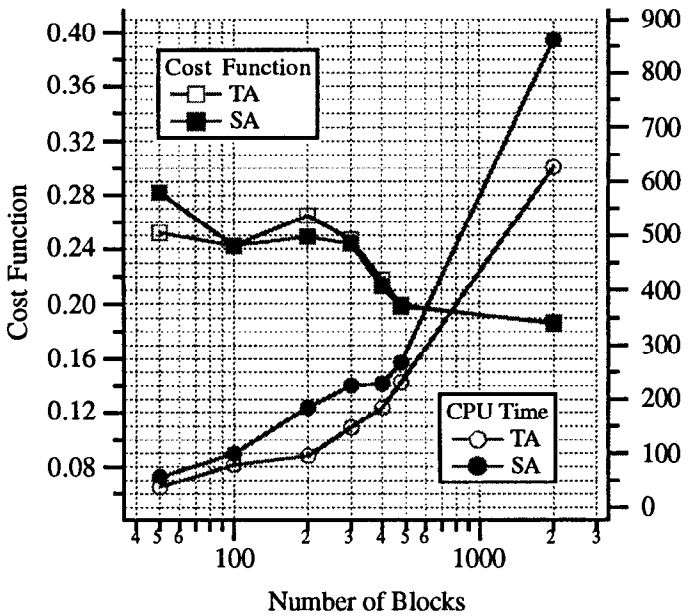

Figure 1 - Variation of cost function and CPU time with number of magnet blocks, with schedule ratio of 0.80 and relative error of 0.001.

We allowed each algorithm to complete its global search before local minimization by quenching. Two indicators measure the resulting performance of the algorithm: the final value of the cost function before quenching; and the amount of CPU time it took to complete the global search. For this paper we used data randomly taken from a set of

478 blocks, with approximately gaussian distributions of remanent field and angular declination from the ideal easy axis, and a uniform distribution of azimuthal angles. The mean remanent field was 1.234 Tesla and the error (standard deviation) in the remanent field was 1.5%. The error in angular declination was approximately 1.5°. For the cases with 2000 magnet blocks a truncated gaussian distribution of blocks was generated with identical errors. The algorithms were implemented in FORTRAN on a DEC Alpha AXP 3000/600 running VMS.

4. Results and Discussion

In Figures 1, 2, and 3 we plot the variation of the cost function and CPU time with numbers of blocks, relative error, and schedule ratio, respectively. Each point in these plots is an average over five runs, where identical sets of initial randomization seed values was used for both TA and SA.

The final value of the cost function before quenching is nearly identical for SA and TA. However, the CPU time is much different. On the average, TA completes its global search faster than SA with comparable optimization of the cost function.

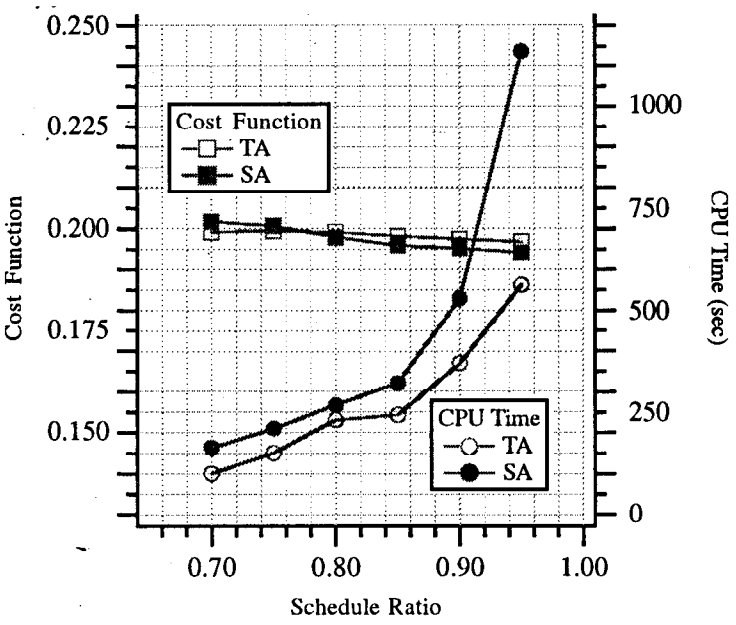

Figure 2 - Variation of cost function and CPU time with schedule ratio, with 478 blocks and relative error of 0.001.

The performance difference between TA and SA is particularly sensitive to variations in relative error and schedule ratio. These two parameters determine the extent and speed with which TA or SA explore different regions of the available configuration space. A schedule ratio close to unity ensures that the temperature is lowered slowly, while a low relative error allows for greater range of exploration at a given temperature.

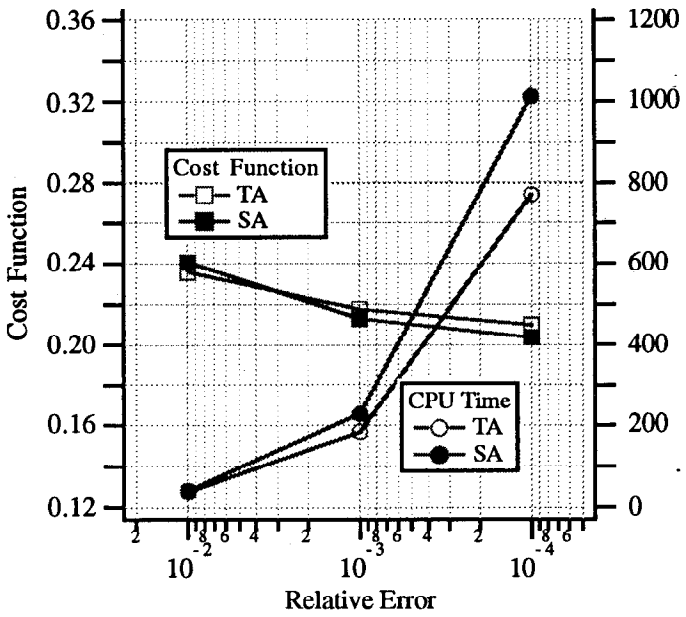

Figure 3 - Variation of cost function and CPU time with relative error, with 400 blocks and schedule ratio 0.80.

The cost function and RMS error before sorting was 0.59 and 1.5%, respectively. To achieve a better than 0.5% RMS error, a sort would have to find a cost function before quenching less than 0.2. This requires a schedule ratio of at least 0.90 and a relative error less than 0.001. TA outperforms SA in this region. In our design [5], a final cost function of 0.193 before quenching resulted in an RMS error of 0.3%. A schedule ratio of 0.90 and a relative error during the threshold accepting of 0.0005, and a relative error during quenching of 0.0001 were used. Most cases of interest will require sorting parameters as stringent as these, making TA an attractive alternative to SA.

5. Conclusion

We have introduced a new technique for sorting large sets of magnet blocks for insertion device optimization. We have shown that the average behavior of the TA algorithm is to converge to an equal level of quality as SA, but at a faster rate. This new technique may prove especially useful in the optimization of long insertion devices or FEL's.

6. References

[1] S. Kirkpatrick, et. al., Science 220 (1983) 671.

[2] N. Metropolis. et. al., J. Chem. Phys. 21 (1953) 1087.

[3] A.D. Cox and B.P. Youngman, SPIE Proc. 582 (1985) 91.

[4] G. Dueck and T. Scheuer, J. Comp. Phys. 90 (1990) 161.

[5] R. Carr and S. Lidia, SPIE Proc. 2013 (1993) 56.