Note that although the number of computations required for the initial evaluation of $F(N-1, \omega)$ at time $N-1$

$$F(N-1, \omega) = \sum_{m=0}^{N-1} x(N-1-m)\, e^{-j\omega m} \tag{4}$$

depends on N, the updating (3) for each time instant $n \geq N$ requires only two multiplications and three additions regardless of N.

Although the number of computations required to update the FT at each frequency ω is invariant with N, the total number of additions and multiplications required to update the transform at all frequencies of interest K may still be a function of the transform block length. In the discrete Fourier transform (DFT) case, for example, the transform is calculated at N equally spaced frequencies spanning the Nyquist interval. In this case, the sum of the total number of additions and multiplications in the recurrence equation (3) is $4N$ which depends on N.

Relating the number of frequencies K to the transform block length N is not, however, necessary in many applications of Fourier analysis and transform. These applications include *a priori* or *a posteriori* processing of the received signal by band-limited filters with known characteristics. These filters can vary the interest in the signal transform over frequency. As a result, the spectral information may only be needed within specific frequency bands or, in general, at selected frequencies whose number is fixed independent of N. In this case, the number of additions and multiplications in (3) becomes $2K$ and $2K$, respectively, which remains invariant with increased transform block length.

III. Generalization

Equation (3) may be expressed in the z-transform domain using a transfer function $H(z, \omega)$ relating $F(n, \omega)$ and $x(n)$ as

$$H(z, \omega) = \frac{1 - e^{-j\omega N} z^{-N}}{1 - e^{-j\omega} z^{-1}}. \tag{5}$$

The impulse response $h(m, \omega)$ corresponding to $H(z, \omega)$ is $e^{-j\omega m}[U(m) - U(m-N)]$, where $U(\cdot)$ is a unit step function. If this response is modified to the form $h'(m, \omega) = e^{-j\omega m}\gamma^m$, the zeros of the transfer function lie on a circle radius γ, and (5) becomes

$$H'(z, \omega) = \frac{1 - \gamma^N e^{-j\omega N} z^{-N}}{1 - \gamma e^{-j\omega} z^{-1}}. \tag{6}$$

The resulting recurrence equation for $F(n+1, \omega)$ is then

$$F(n+1, \omega) = \gamma e^{-j\omega} F(n, \omega) + x(n+1) - \gamma^N e^{-j\omega N} x(n-N+1). \tag{7}$$

Similar to (3), the number of updating computations is independent of N. In this formulation, the functions $w(m) = 1$ and $w(m) = \gamma^m$, $0 \leq m \leq N-1$, may be viewed as rectangular and exponential data windows which are applied to the data sequence prior to computing the transform. Thus in general form

$$F(n, \omega) = \sum_{m=0}^{N-1} w(m)\, x(n-m)\, e^{-j\omega m}. \tag{8}$$

The weighting function $w(m)$ can be extended to include the sum of a finite number of geometric series, i.e.,

$$w(m) = \sum_{i=-M}^{M} c_i (\gamma_i)^m, \qquad 0 \leq m \leq N-1 \tag{9}$$

and $F(n, \omega)$ is then a summation of the corresponding Fourier updates, each calculated through (7). If we allow γ to be the complex exponential $e^{j\omega_0}$, then it can be shown that the raised cosine window (Hanning and Hamming) given by [3]

$$w(m) = \alpha + \beta \cos(\pi m/(N-1)), \qquad |m| \leq N-1 \tag{10}$$

belongs to the proposed class of windows, with $M = 1$, $c_0 = \alpha$, $c_{-1} = c_1 = \beta/2$, and $\omega_0 = \pi/(N-1)$. Other commonly used data windows may be similarly tested by identifying the corresponding parameters M, c, and γ in (9). However, since the number of computations in (9) is a monotone increasing function of M, the computational invariance property of RFT becomes less attractive with high value of M.

IV. Conclusions

A new class of windows is defined for computationally efficient recursive Fourier transform. The use of these windows allows the transform to be updated with a number of computations independent of the transform block length. The computational invariance property of the recursive FT is mainly applicable to the cases where the number of frequencies at which the transform is calculated is fixed independent of the transform block length. Both rectangular and exponential windows were shown to belong to the proposed class of windows. This class, in general, includes all windows that can be represented by a sum of geometric series. Detailed comparison between the proposed transform updating method and FFT is necessary for further realization of the computational attraction of the method. The introduced property of RFT may be extended to recursive power spectrum estimators where the transform is applied to an autocorrelation function instead of a data sequence.

References

[1] A. Papoulis, *Signal Analysis*. New York, NY: McGraw-Hill, 1977, ch. 9.

[2] P. Young, *Recursive Estimation and Time Series Analysis*. Berlin, Heidelberg, Germany: Springer-Verlag, 1984.

[3] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1975, ch. 11.

LUIZ CALÔBA

Nonconvex Optimization by Fast Simulated Annealing

HAROLD H. SZU AND RALPH L. HARTLEY

Recent advances in the solution of nonconvex optimization problems use simulated annealing techniques that are considerably faster than exhaustive global search techniques. This letter presents a simulated annealing technique, which is t/log (t) times faster than conventional simulated annealing, and applies it to a multisensor location and tracking problem.

I. Introduction

Most multidimensional optimization functions are nonconvex and, therefore, expensive to compute using exhaustive global search techniques. A recently developed solution for this type of problem is a Monte Carlo technique called simulated annealing. The simulated annealing technique arrives at a globally optimum answer by simulating the annealing of a solid into its (globally) lowest energy state by first heating the solid to its melting point and then slowly cooling the material at a rate related to its heat transport speed. From a computer science point of view, there is no reason for a computer algorithm to have a physically realizable implementation. This letter discusses a non-physically-realizable algorithm called Fast Simulated Annealing. This algorithm employs a semi-local search strategy using the (infinite variance) Cauchy probability density to generate a random process that more often allows large steps in phase space than the strictly local step strategy using a Gaussian probability. The result of allowing more of these larger steps is to permit a faster cooling schedule, which is inverse linear in time, than the cooling schedule of the conventional local annealing algorithm, which is inverse in the logarithm of time. This $t/\log t$ speed increase is similar to the $N/\log N$ improvement between the discrete Fourier transform and the Cooley–Tukey Fast Fourier transform. This research should therefore have a major impact on the computational tractability of many nonconvex optimization

Manuscript received February 5, 1987; revised June 23, 1987. This work was supported by the ONT/NOSC TWAS program and the SDI/T/IS ONR Office.

The authors are with the Naval Research Laboratory, Washington, DC 20375, USA.

IEEE Log Number 8716403.

application of this technique to the problem of locating and tracking multiple, indistinguishable targets using noisy, bearing-only data with both missed detections and false alarms.

In a convex optimization problem, one can start at any point in the function space, measure the local gradient, and take a step in any direction which is lower in altitude than the current position. Repetition of this process will assure asymptotic convergence to the minimum (i.e., optimum) solution. In a nonconvex problem, the optimization function has multiple local minima, each with different depths, for which the optimum is defined to be the global minimum. The application of local gradient techniques to nonconvex optimization creates a problem where one is caught in a local minimum with no way of determining if the local minimum is also the desired global minimum. One solution to this dilemma is to permit steps whose magnitude and direction are dependent on the local gradient and an additive random process. Further, for the algorithm to converge, the magnitude of the random component of the step size must decrease in a statistically monotonic fashion. In the physical annealing process these steps can be equated with Brownian motion of a particle, traveling at statistical velocity V, over an inter-sample time Δt. The expectation of V^2 is linearly related to the temperature of the particle. The simulated annealing community [1]–[3], therefore, refers to the "temperature" of the random process and uses the term "cooling schedule" to refer to the algorithm for monotonically reducing the temperature.

The cooling schedule is critical to the performance of the algorithm. For a given random process, cooling at too fast a rate will likely "freeze" in a nonglobal minimum. Cooling at a too slow rate, while reaching the desired minimum, is a waste of computational resources. The technical problem then becomes deriving the fastest cooling schedule that will guarantee convergence to the global minimum. With this understanding, this letter will use the term "cooling schedule" as synonymous with "fastest cooling schedule."

If the Gaussian random process is used, as in the Conventional Simulated Annealing (CSA), it has been proved [4] that the cooling schedule $T(t)$ must be inversely proportional to the logarithm of time t in order to guarantee convergence to the global minimum. This relatively slow convergence is due to the bounded variance of the Gaussian process which constrains the neighborhood of successive samples. This bounded variance random walk is called a local search strategy. If, on the other hand, one uses an infinite variance, Cauchy random process, the cooling schedule has been proved [2] to be sped up to be inversely proportional to time t. We call this new class of algorithms a semi-local search strategy because they permit occasionally long steps (the so-called Lévy flights) far from the neighborhood of the previous sample.

II. Proofs of the Cooling Schedules for FSA and CSA

There are a number of similarities in the proofs of the cooling schedules for the FSA and CSA algorithms in D-dimensional vector spaces. For the convenience of comparison, the proofs will be demonstrated in parallel. In locating the minimum, one must start at some position or state in a D-dimensional space, evaluate the function at that state, and generate the next state vector. The FSA and CSA algorithms are different in that FSA uses a Cauchy distribution and CSA uses a Gaussian distribution in their respective state-generating functions. Both the CSA and FSA algorithms will use as its next state either the current state vector or the next state vector provided its incremental cost increase is less than the time-dependent noise bound which is temperature- (and time-) dependent.

The FSA algorithm requires that *state generating* be infinitely often in time (i.o.t.) whereas the CSA requires the *state visiting* be i.o.t. At some cooling temperature $T_c(t)$ at time t, let the state-generating probability of being within a specific neighborhood be lower bounded by g_t. Then the probability of not generating a state in that neighborhood is upper bounded by $1 - g_t$. To insure a globally optimum solution for all temperatures, a state in an arbitrary neighborhood must be able to be generated i.o.t., which however does not imply ergodicity which requires actual visits i.o.t. To prove that a specific cooling schedule maintains the state generation i.o.t., it is easier to prove the *negation* of the *converse*, namely the *impossibility* of *never* generating a state in the neighborhood after an arbitrary time t_0. Mathematically this is equivalent to stating that the sum of the g_t terms is infinite. One can now verify cooling schedules in a D-dimensional neighborhood $|\Delta x_0|$ and arbitrary time t_0. Among the various Lévy distributions (including Cauchy, Holtzmach, and Gaussian) there are two different classes, local (as in CSA) and semi-local (as in FSA). There exists an initial temperature T_0 and for $t > 0$, such that

CSA

$$T_a(t) = T_0/\log(t) \tag{1a}$$

$$g_t \approx \exp\left(\frac{|\Delta x_0|^2}{T_a(t)}\right) T_a(t)^{D/2} \tag{2a}$$

$$\sum_{t=t_0}^{\infty} g_t \geq \exp(\log(t)) = \sum_{t=t_0}^{\infty} \frac{1}{t} = \infty. \tag{3a}$$

FSA

$$T_c(t) = T_0/t \tag{1b}$$

$$g_t \approx \frac{T_c(t)}{(T_c^2 - |\Delta x_0|^2)^{(D+1)/2}} \approx \frac{T_0}{t|\Delta x_0|^{D+1}} \tag{2b}$$

$$\sum_{t=t_0}^{\infty} g_t \approx \frac{T_0}{|\Delta x_0|^{D+1}} \sum_{t=t_0}^{\infty} \frac{1}{t} = \infty. \tag{3b}$$

III. Generic Nonconvex Problem of Resolving Ambiguities in Unsupervised Learning

An example of a nonconvex ambiguity resolution is the bearing fix problem. Given a known set of identical passive sensors at locations $\{(\xi_i, \eta_i), i = 1, \cdots, n\}$, each sensor estimates the direction (bearing) to a set of indistinguishable targets at locations $\{(x_j\ y_j), j = 1, \cdots, m\}$. The bearing from the ith sensor and the jth target is of the form $\theta_{ij} + \Delta\theta_{ij}$, where θ_{ij} are the true bearings and the parameters $\Delta\theta_{ij}$ are identically distributed random variables. The nonconvex nature of this problem is demonstrated in Fig. 1, where

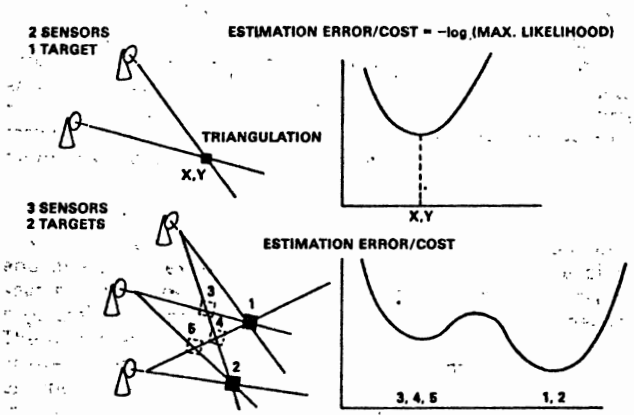

Fig. 1. Target location from direction-finding data: ambiguity in multiple sensors and targets.

local minima (false or ghost targets 3, 4, and 5) appear at every intersection of two or more bearings. The minima in the neighborhood of targets are generally lower but depend on the number of sensors and targets, and their relative geometry.

Mathematically, one can consider this process as deriving a set of vectors x_i, from the origin to the estimated location of ith target, given a set of line vector L_j passing the jth sensor with the measured bearing. The functional form of the conditional probability of x_i given L_j is identical to the conditional probability of L_j given x_i. The latter problem is, given a set of positions, to derive the course and speed of the various targets. The conditional probability is given by

$$p(x_i|L_j) = (2\pi\sigma_{ij}^2)^{-1/2} \exp(-d^2(x_i, L_j)/2\sigma_{ij}^2) = p(L_j|x_i) \tag{4}$$

where the distance $d(x_i, L_j)$ and its associated variance σ_{ij}^2 relate to the angular errors in the bearing fix problem and off-track distance errors in the tracking problem. The maximization of the appropriate maximum likelyhood is equivalent to the minimization of

its exponent, namely, the cost function is

$$C = -\log \prod_{\text{Lines}}^{n} \sum_{\text{Points}}^{m} p(L_i|x_i). \quad (5)$$

Fig. 2 is the results of a more real-world simulation. Here, there is some probability of wild bearing, which is flat distributed over

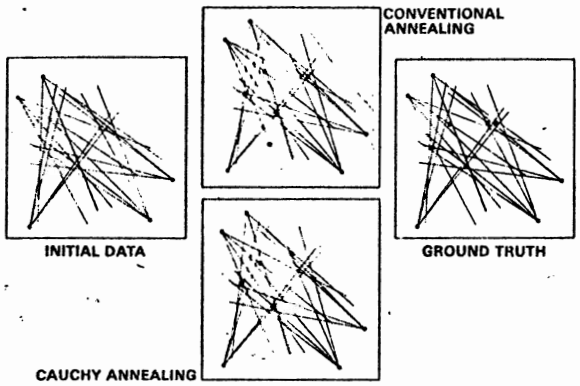

Fig. 2. Bearing-fix problem with missed detectors and wild bearings.

2π. Further, the probability of detection of a target by a sensor is limited. The five sensors are fixed at random locations on a circle and the five targets are randomly located in an inscribed square. The experimental parameters are standard deviation of bearing errors 2.5 degrees, probability of wild bearing 0.05, and probability of detection 0.95. The intermediate results in Fig. 2 are after 100 iterations of the Cauchy annealing algorithm [5]. Fig. 3 is an example of the tracking problem where the various locations are first associated with a set of straight lines and the direction of each line defines a course and the average velocity along the line determines the speed.

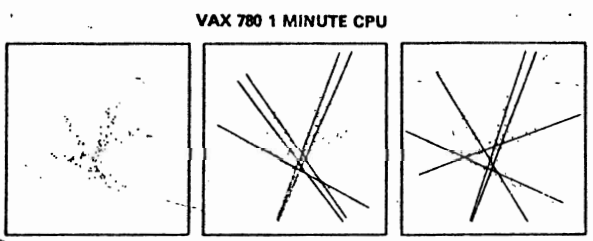

Fig. 3. Track-correlation problem.

Acknowledgment

The authors would like to thank F. Polkinghorn for his generous help throughout the course of this research and S. Gardner for his thoughtful discussions on this subject.

References

[1] R. L. Hartley and H. H. Szu, "Generalized Simulated Annealing," submitted to *J. Stat. Phys.*

[2] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculation by fast computing machines," *J. Chem. Phys.*, vol. 21, pp. 1087–1092, June 1953.

[3] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, May, 13, 1983.

[4] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distribution and Bayesian restoration in images," *IEEE Trans. Patt. Anal. Mach. Int.*, vol. PAMI-6, no. 6, pp. 721–741, Nov. 1984.

[5] H. H. Szu and R. L. Hartley, "Fast simulated annealing," *Phys. Lett. A*, vol. 122, nos. 3, 4, pp. 157–162, Jan. 1987.

A Class of Bandpass and Bandstop Butterworth Digital Filters

N. AHMED, A. WEBSTER, AND B. ARMSTRONG

This letter introduces a classs of bandpass and bandstop Butterworth digital filters that are simple to implement using existing software for low-pass and high-pass filters.

I. Introduction

Robust software that implements a powerful class of low-pass (LPDES) and high-pass (HPDES) filters is available; e.g., see [1], [2]. To illustrate, the LPDES software is listed in Table 1. This routine implements an even-order *Butterworth low-pass* filter as a *cascade* of *second-order sections* of the form

$$H_i(z) = \frac{a_k(1 + 2z^{-1} + z^{-2})}{1 + b_k z^{-1} + c_k z^{-2}} \quad (1)$$

where the a_k, b_k, and c_k are constants. The overall transfer function of the cascaded structure is given by

$$G(z) = \prod_{i=1}^{K} H_i(z) \quad (2)$$

where K is the number of stages.

We now show that very simple *preprocessing* and *postprocessing* with respect to $G(z)$ in (2) leads to *bandpass* filters. The basic idea is to use modulation properties [3].

II. Approach

We consider the configuration in Fig. 1(a), where ω_c is the center frequency of the desired bandpass filter, and T is the sampling interval. It is apparent the multiplication by the complex exponentials $e^{\pm jn\omega_c T}$ can be viewed as two simple stages of processing and postprocessing.

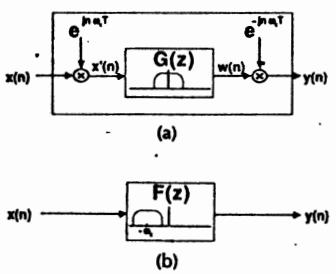

Fig. 1. Low-pass to bandpass transformation.

Using elementary properties of the Z-transform with respect to the variables in Fig. 1(a), it is straightforward to show that

$$\frac{Y(e^{j\omega T})}{X(e^{j\omega T})} = F(e^{j\omega T}) = G[e^{j(\omega + \omega_c)T}]. \quad (3)$$

Manuscript received December 23, 1986.

The authors are with the Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM 87131, USA.

IEEE Log Number 8714543.

0018-9219/87/1100-1540$01.00 © 1987 IEEE