2

OFFICE OF NAVAL RESEARCH

Grant N00014-90-J-1971

R&T Code 4131001 phy

Technical Report No. 3

Data Analysis for Rotationally Resolved Spectra: A Simulated Annealing Approach

by

Julian M. Hjortshøj and Laura A. Philips

Accepted for Publication
in the
Journal of Molecular Spectroscopy

Cornell University
Department of Chemistry
Ithaca, NY 14853-1301

May 29, 1992

DATA ANALYSIS FOR ROTATIONALLY RESOLVED SPECTRA: A SIMULATED ANNEALING APPROACH

Julian M. Hjortshøj and Laura A. Philips
Department of Chemistry, Cornell University, Ithaca, New York, 14853-1301

Manuscript Contains:

46 Pages
9 Figures
8 Tables

Proposed Running Head:
Parallel Simulated Annealing

Prof. Laura Philips
Dept. of Chemistry
Cornell University
Ithaca, New York, 14853-1301

Abstract

A new method for the analysis of rotationally resolved spectra is presented. The method employs a simulated annealing algorithm with two modifications. First, the standard simulated annealing process is extended to take advantage of parallel computing. Second, rather than using a continuous random search of the parameter space, at distinct intervals new optimization processes are started at points in parameter space that have promising χ^2 values. Together these modifications make more efficient use of computer time than a standard simulated annealing approach. The technique is applied to the analysis of simulated data as well as real high resolution experimental spectra to demonstrate the effectiveness of the parallel simulated annealing algorithm.

DATA ANALYSIS FOR ROTATIONALLY RESOLVED SPECTRA: A SIMULATED ANNEALING APPROACH

Julian M. Hjortshøj and Laura A. Philips
Department of Chemistry, Cornell University, Ithaca, New York, 14853-1301

Introduction

From an examination of the available literature on high resolution spectroscopy, one can conclude that the field is limited to small to mid-sized molecules of moderate complexity. The family of accessible molecules is limited as much by the complexity of the data analysis as by the difficulty of the data acquisition. The most common approach to data analysis is to calculate a spectrum and compare it with the experimentally measured spectrum. The best simulation of the experimental spectrum is generated by the molecular parameters that best approximate the molecule under study. The problem is then reduced to an optimization problem in multi-dimensions. This optimization, however, is fraught with many difficulties. The parameter space that must be searched is populated with multiple local minima. As the spectrum becomes more complex, the computation time required for the optimization procedure for a thorough search of the parameter space becomes excessive. Alternatively, one can employ large amounts of user input, and prior knowledge of the molecule. The goal of the current work is to design an optimization procedure that requires little user input and efficiently uses computer time to find the global minimum in this optimization problem.

The solution to the problem is found in a modification of a simulated annealing algorithm. Simulated annealing has been used in a variety of applications where local minima cause difficulties in optimizations, most notably, for large scale combinatorial problems, such as the traveling salesman problem and matrix reduction.(1,2) The disadvantage of simulated annealing is that in such an exhaustive search of the parameter space, large amounts of computer time are consumed. Our modifications to the standard simulated annealing (SSA) approach, enhance the efficiency of the optimization process and employ parallel computing to decrease the requirements for computer time. The parallel simulated annealing (PSA) approach is applied to simulated data to characterize the accuracy and the efficiency of the procedure. In addition, we apply the algorithm to experimental data to demonstrate the effectiveness of the approach to the analysis of real laboratory spectra.

Simulated Annealing Procedure

Before discussing our modifications of simulated annealing, it is appropriate to briefly describe some of the salient features of the simulated annealing algorithm. Simulated annealing requires several different procedures which, for convenience will be defined here. These procedures include: 1) the cost function; 2) the move function; 3) the temperature; 4) the cooling schedule; and 5) the stop criteria.

The Cost Function: In any optimization process, one attempts to find an optimal set of parameters. The cost function is a measure of the accuracy of a given set of test parameters. A high cost is associated with a bad approximation, while a good approximation has a low cost. If we attempt to optimize a set of n parameters, then any given vector of the n parameters represents a parameter state. The set of all vectors constitutes the parameter space. For any given parameter state there is an associated cost, and the optimal parameter state will be that with the lowest cost. The cost function, then, is an n to 1 mapping from parameter vector to cost.

The Move Function: The simulated annealing routine operates by moving from one state to a randomly determined next state which is close to the first. Given a state S, the move function generates a randomly selected state, S', such that the parameters characterizing S' differ slightly from those characterizing S.

Temperature: Once a state S' is determined, the annealing algorithm does not necessarily move to this new state. At each transition from a state S to a consecutive state S' the algorithm must decide whether or not to accept the move to the new state. In order to make this decision, COST(S') is compared to COST(S). If the cost of the new state is lower than the cost of the old, it will always make the transition. If the cost of the new state is higher, the algorithm uses a probability function to determine whether or not to move to the new state. The probability function is given by:

$$P = e^{-(COST(S')-COST(S))/kT} \qquad (1)$$

where P is the probability of making the move, k is a constant, and T is the temperature, or control parameter. Note that as the temperature decreases, the algorithm is less and less likely to accept moves to states with higher costs. In the limit as T approaches 0, the probability that the routine will move to any state with a worse cost also approaches 0. If we were to start the annealing routine with a temperature of 0, then, the routine would act as a strict iterative improvement routine, only accepting moves to states with lower costs, and rapidly converging to a local minimum. The manner in which the temperature is set over the course of the optimization is critical to the effectiveness of the routine. Hence, the simulated annealing routine must employ a temperature schedule by which it controls the cooling process.

The Cooling Schedule: The annealing algorithm operates by exploring the parameter space at a series of successively lower temperatures. Each period between temperature decreases is termed an annealing epoch. An epoch consists of a series of potential moves at a given temperature, such as those described above. The epoch ends when the routine has made enough successful moves such that the probability density of the parameter space has been approximated for that temperature.(3) At the end of each epoch the temperature is multiplied by some factor slightly less than 1. The smaller this factor is, the faster the routine will cool. Particular values for the length of the epoch and the factor by which the temperature is multiplied will vary from one application to the next.

The Stop Criteria: At the end of each epoch, the routine must decide whether or not to stop. Stop criteria usually hinge on whether or not the routine is "frozen". If the routine has descended to a minimum, and the temperature is low enough that it will not be able to climb back out of the minimum, then either the optimization was successful in locating the global minimum, or it is trapped in a local minimum. In either event, there is no reason to continue. To determine whether or not to stop, the number of successful moves in the previous epoch is determined. If the number of successful moves is sufficiently small, we can assume that the routine is frozen.

With the important terms defined, it is now possible to summarize the simulated annealing algorithm. A flow chart describing the operation of a standard simulated algorithm is shown in Figure 1. The algorithm takes as input: a starting parameter state, a set of constraints, and a starting temperature. The starting parameter state is often defined by the problem. If information is already known about the parameters, then the starting point can be a good approximation of the optimal parameters. Alternatively, the parameters can be randomly generated, or a rough guess. The set of constraints can be used to limit the parameter space thereby simplifying the problem. For example, many parameters may have a limited range of probable values. In these cases, the user may opt to set bounds on the parameter space. The nature of these constraints will vary substantially from one application to the next. We choose a starting temperature at which virtually all moves are accepted.

The algorithm operates as follows. The current state S is set to the state defined by the input parameters and the temperature is set to the starting temperature. The move function is then used to determine a new state S' from S. If COST(S') is less than COST(S), S is set equal to S'. Otherwise a random number is used to determine whether or not to set S equal to S' with a probability of $e^{-\Delta COST/kT}$. If we have made enough successful moves, then an annealing epoch has ended, and we must multiply the control parameter by a set cooling factor and check the stop criteria. If the stop criteria are met, then the optimization is complete. Otherwise a new epoch is started. On completion of the optimization, the state that the routine has passed through with the lowest cost is returned as output.

Analyzing Rotationally Resolved Spectra

Our application of simulated annealing is to analyze rotationally resolved spectra. The approach is to calculate a simulation of the spectrum to compare to the experimentally determined spectrum. The parameters required to calculate a spectrum are the ground and excited state rotational constants, the angular components of the transition moment, the rotational temperature and the 0-0 frequency. Given a set of these parameters, we can calculate the spectrum that would be produced by a molecule with those parameters. The accuracy of the optimized calculated spectrum is fundamentally limited by two factors, the experimental uncertainty and the accuracy of the model used in the calculation. The model used here is a rigid asymmetric rotor. We use, as a subroutine, an algorithm that exactly diagonalizes the rigid asymmetric rotor Hamiltonian and returns the position and intensities of the allowed rotational transitions.(4) The object of the optimization is to find the calculated spectrum, and by inference, an optimized set of parameters, which best fit the experimental data.

For application to rotational analysis, two major modifications are made to the approach described above. A standard simulated annealing (SSA) approach is not ideally suited to the analysis of rotational spectra, because in order to assure that a global minimum is achieved, the amount of computer time required becomes prohibitive. The two modifications are: 1) convolution of both the experimental spectrum and the calculated spectrum with a gaussian function and 2) a novel parallel modified simulated annealing approach (PSA). The advantages of these modifications will be describe in detail below.

Convolution of the spectra substantially reduces the complexity of the parameter space. The parameter space for rotational spectra contains many deep local minima. By convolving the spectra, the local minima become less pronounced both in magnitude and frequency. Figure 2 shows one dimensional cross sections of the potential surface in a fit to an experimental spectrum of 2-fluoroethanol using spectra convolved with gaussians of two different widths. As one can see in Figures 2 and 3, when the broader gaussian convolution is used, the global minimum is both deepened and broadened, and many of the small local minima disappear. As a result, larger moves can be made and the parameter space can be searched more efficiently. The accuracy of the fit as a function of the convolution width will be discussed in detail in the discussion section.

In any optimization routine there is a tradeoff between thoroughness of search and speed of convergence. The PSA algorithm is a compromise between the rapid convergence properties of steepest descent and iterative improvement algorithms and the slow, thorough search of an SSA algorithm. The differences between PSA and SSA are described below.

Throughout the optimization, PSA stores the three lowest-cost states it has encountered. At the end of each epoch four processes are initiated in parallel. Process #1 is started at the resulting state from the same process in the previous epoch--the same way that SSA would operate. Processes #2-4 start the epoch at the three lowest cost states. These processes act as a hybrid between simulated annealing and iterative improvement, moving around the parameter space enough that they will not get stuck in local minima, but exploring the regions around the best points found. Process #1 serves two purposes: 1) Information about the number of accepted states in process #1 is used to set the temperature and 2) Since process #1 performs a continuous random walk instead of returning to a better state at the end of each epoch, it will be able to climb larger hills. If the global minimum is on the far side of a large barrier, processes #2-4 may not be able to reach it without process #1.

Because the PSA algorithm runs four traces in parallel, these four traces must be of the same length if the algorithm is to run efficiently. Thus, the length of the epoches for each process must be fixed at the beginning of each epoch. PSA must therefore emulate the temperature behavior of SSA while using fixed rather than variable length epoches. An SSA epoch ends after it has completed enough successful moves (m) to approximate equilibrium density. After every m successful moves it multiplies the temperature by a cooling factor c. The PSA algorithm runs fixed length epoches, and keeps track of the number of successful moves (n) which occurred during each epoch. At the end of the epoch, it sets

$$\text{temp} = \text{temp} * c^{n/m} \qquad (2)$$

In our application, m is defined as 1/5 of the fixed epoch length. At high temperatures in the beginning of the optimization, n will be close to or equal to the total number of moves attempted, and hence the ratio n/m will be close to 5. As the temperature decreases, this ratio will drop to well below 1. In SSA, however if the routine is near convergence, and very few moves are accepted, the epoch will end after an arbitrary fixed number of attempted moves. PSA mimics this temperature behavior by setting temp =temp * c in cases where n/m is less than 1.

In order to implement any simulated annealing algorithm we must define a quantitative process by which we can evaluate the cost function, the move function and the stop criterion. The simulated annealing algorithm is designed to fit spectra about which we know relatively little. Therefore, we assume that we do not have information about the quantum numbers of the peaks in the experimental spectrum. In this scenario, the problem is to take a series of frequencies and intensities from the experimental spectrum, and determine how well they match another series of frequencies and intensities from a calculated spectrum. Note that we do not know which frequency in the experimental spectrum corresponds to a particular frequency in the calculated spectrum.

We define our cost function as follows. In the beginning of the optimization, the sum of the intensities of the peaks in the experimental spectrum are normalized to one. The frequencies and normalized intensities are then convolved with a gaussian (half width ≈ .1 cm^{-1}) and stored as an array. Each time a new state is generated by the move function, a calculated spectrum is generated from the parameters, and that spectrum is normalized and convolved in the same manner as the experimental spectrum. The convolved calculated spectrum is then compared to the convolved experimental spectrum by taking the sum of the squares of the differences between each point in the two spectra. An example of the two convolved spectra and the difference between them is shown in Figure 4. The difference between the calculated and experimental spectra is determined repeatedly, shifting the calculated spectrum over a range of frequencies with respect to the experimental spectrum. We keep the lowest value of the sum of the squares of the differences. In this way, we can vary the 0-0 frequency without having to recalculate the spectrum each time. Next, the sum of differences squared is normalized to approximate a χ^2 function. To normalize the sum of the differences squared we need to determine the expected sum of differences squared. In order to make this determination we have generated a spectrum with random error. Random error has been added to the calculated spectrum by randomly shifting peak frequencies and intensities using a normally distributed random number generator with the standard deviation, σ, set to the experimental error of our data acquisition process. The sum of differences squared value obtained when fitting to this generated spectrum is set to a χ^2 value of 1, thereby determining the normalization factor. In order to convert our sums of differences squared to χ^2 values, we divide by this factor at the end of each call to the cost function. The resulting χ^2 value is returned by the cost function.

The move function in the PSA generates random neighbors of a state by varying each parameter randomly within a fixed window. We set bounds on the amount each parameter can vary so that on average, the change in each parameter will change the cost function by the same amount. At the beginning of the optimization, we take partial derivatives of the cost function with respect to each parameter, $\partial C/\partial p$, at 32 randomly determined states in the parameter space. For each parameter p, given ΔC_{ave} (the average change in the cost function with the maximum change in a parameter), we can define the maximum step size Δp_{max} as follows:

$$\Delta p_{max} = \Delta C_{ave} / < | \partial C/\partial p | >. \qquad (3)$$

To generate a neighbor of a state, we add to each parameter p the product of Δp_{max} and a uniformly distributed random number between -1 and 1.

In order to determine whether or not the routine has converged, we examine the number of successful moves made by process #1 in the last epoch. If this number is sufficiently small, we can conclude that PSA has converged, and terminate the annealing stage of the algorithm. After annealing has converged, we apply a Polak-Ribierre optimization routine to the best fit it has located. With the exception of minor modifications, this routine is taken directly from Numerical Recipes.(5)

All of the computation was performed at the Cornell National Supercomputer Facility on an IBM 3090 computer. The code was fully vectorized. For the PSA algorithm, four processors were used in parallel. A flow chart for the entire PSA program is shown in Figure 5.

Results

Calculated Data

We ran the PSA routine on 10 calculated spectra with various configurations of randomly generated rotational constants as shown in Tables I. After calculating each spectrum, we randomly altered the frequencies and intensities of its peaks by our estimated experimental error in order to make it resemble experimental data as closely as possible. We added gaussian distributed random error to the frequencies with σ=0.0004 cm^{-1}. Error was added to the intensities of each peak such that σ=0.10 times the original intensity. We then simulated baseline noise by adding gaussian error to each intensity with σ=0.010 times the intensity of the tallest peak (the approximate signal-to-noise from our experimental spectrum of diflouroethane).

The initial guesses for ground state rotational constants were generated using a uniformly distributed random number in the window of +/-10% of the actual ground state values. The starting values for dipole moment angles were selected from a window of ± 20% of the actual values, and starting temperatures were taken from a window ranging from half to two times the actual temperature. Excited states rotational constants were started with values equal to the ground state rotational constants.

We allowed each of the ground state rotational constants to vary in a window of ± 10% of the initial guess. Each of the excited state rotational constants were allowed to vary from the ground state value in a window of ±5% of the initial guess for the ground state. Transition moment angles and temperature were allowed to vary within the same windows from which we took the starting values.

The starting parameters and results for these runs are represented in Tables I, II and III. Shown in Figure 6 is an example of the simulated spectrum from run #5, including the spectrum calculated from the starting parameters, and the final output spectrum from PSA. All of the above calculations used a 0.10 cm^{-1} convolution width. We repeated run #10 with identical input with the exception that the convolution width was reduced to 0.030 cm^{-1}. The results of this optimization are shown in Table IV.

In many real experimental situations, the starting guesses may be even farther from the optimized parameters than in the above calculations. To further test PSA under more strenuous conditions, we did a fit to a spectrum allowing the parameters to vary in much larger windows. The starting values for the ground state rotational constants were selected from a window of ± 25% of the actual values. Excited state rotational constants were allowed to deviate from ground states by ± 10% of the ground state values. Dipole moment angles were allowed to vary within ± 50% of the actual values, and the rotational temperature was allowed to vary from 50% to 200% of the actual value. The parameters for the initial guess were picked randomly from within these windows, with the exception of the excited state rotational constants, which were set equal to the starting ground state values. Optimizations using PSA were performed with these parameter constraints at three convolution widths: 0.040 cm^{-1}, 0.10 cm^{-1}, and 0.20 cm^{-1}. The optimizations using the two narrower convolution widths did not converge to the global minimum, but the broad convolution width was successful in locating the global minimum. The results of these three optimizations are given in Table V.

Experimental Data

PSA was used to analyze high resolution infrared spectra of 2-fluoroethanol (2FE) and difluoroethane (DFE). Although the ground and excited state rotational constants for these molecules are available from previous studies(6,7,8,9), we ran the optimizations under conditions of simulated ignorance, allowing parameters to vary in windows which would be reasonable if we knew relatively little about the molecules. Parameters for these runs are given in Tables VI and VII. Starting spectra, experimental data, and final fits are shown in Figures 7 and 8.

Discussion

The power of PSA has been demonstrated in the analysis of rotationally resolved spectra. The analysis of simulated spectra provides criteria for the evaluation of the accuracy of the analysis. Analysis of experimental data demonstrates the feasibility of PSA under real laboratory conditions. From the analysis of simulated spectra it is apparent that both the χ^2 value and the convolution width have subtle yet important effects in the optimization procedure. The implications of how we use the χ^2 value and convolution width in PSA will be addressed below. Since, PSA was developed as a compromise between thorough searching of the entire parameter space and efficient use of computer time, we will discuss the time required to optimize a fit to data as a function of the variables used in the analysis. A comparison of the simulated data and the experimental data yields an obvious discrepancy in the accuracy of the fits. To conclude, we present some possible explanations for these discrepancies, and an evaluation of the PSA approach.

χ^2 Values

There are two variables inherent in the optimization process which affect the calculation of χ^2 values: the size of the convolution array, and the convolution width. The choice of values for these parameters depends on the accuracy with which one makes an initial guess, as well as experimental parameters such as experimental error, resolution and the natural linewidth of the transition. Since the experimental error, resolution and natural linewidth will vary from one experiment to the next, the size of the convolution array, and the convolution width are user defined variables. Both of these factors, however, affect the cost function, and therefore will vary the normalization factor used to calculate the value of χ^2.

To evaluate the effect of the convolution parameters on the optimization procedure, we determined cost function values for the same fit varying the size of the array into which the spectrum was convolved, and the convolution width. We found that both factors were inversely proportional to the cost function (see Figure 9). Hence, in order to normalize for these factors, we define the χ^2 value as follows. Given that COST(Y) is the cost function of a fit Y which is within experimental error, for any state S:

$$\chi^2(S) = \frac{[COST(S)*WIDTH(S)*ARRAY(S)]}{[COST(Y)*WIDTH(Y)*ARRAY(Y)]} \quad (4)$$

where, WIDTH is the width of the convolution and ARRAY is the size of the convolution array. This renormalization of χ^2, effectively removes any dependence of χ^2 on the convolution width or the size of the convolution array.

Since the χ^2 function is constructed using experimental errors from a particular experiment, if the experimental error changes, it will obviously affect the χ^2 function. The χ^2 function is also affected by the number of peaks in the experimental spectrum.

As we can see from Table 8, χ^2 values are not well correlated with RMS deviation in peak frequencies. However, the ratio of RMS deviation to χ^2 error is partially correlated to rotational temperature. (see Table IX) Since spectra with higher temperatures tend to have more peaks, small frequency deviations in individual peaks in such spectra will have less effect on the the χ^2 value than in low temperature spectra.

We have no effective technique for consistently normalizing for changes in experimental error, temperature and peak density, and hence the χ^2 values we produce for fits to experimental data are limited by this inherent inaccuracy. Thus, the actual numerical value calculated for χ^2 from the PSA routine should be used with caution. We should note, however, that none of these errors will change during the course of an optimization, so while our cost function is not technically an accurate χ^2 value, its effectiveness as a cost function for PSA is unaffected.

Convolution Width

It may be argued that the broad convolution used in PSA represents a substantial loss of information, and to be sure, this is true. The resolution of the data acquisition process is 0.004 cm^{-1} (7 MHz), while we convolve the spectrum to 0.1 cm^{-1}. The information lost , however, is not necessarily information that might be useful to the fitting routine. Not only does the use of wide convolution greatly simplify the potential surface, it also hides many phenomena which may appear in the experimental spectrum but which are not included in the model, such as centrifugal distortion and rotational fine structure resulting from vibrational mode coupling. While wide convolution of spectra creates a broad, shallow bottomed global minimum and thus increases the uncertainty associated with resulting parameters, it allows us to get a good approximate fit which can then be used to assign quantum numbers to individual peaks. Once quantum numbers are assigned, there are a number of other fitting routines which are more efficient to determine the most accurate fit to the data.

We have seen that narrowing convolution width in a fit can, in fact, produce a better approximation of the correct rotational constants for a molecule, but it is not always feasible to use narrower convolutions. In experiments where less is known about the molecule, and as a result the search space is much larger, PSA is no longer effective in efficiently finding the global minimum with a narrow convolution width. It would be more effective to start with a very broad convolution width to find the vicinity of the global minimum, and then to perform a second optimization with a reduced search space and a narrower convolution width to find a solution from which quantum numbers could be assigned. This approach of successively decreasing the convolution width in the spectrum has been effectively used in the analysis of rotationally resolved spectra in previous studies.(10)

Computation Time

The average time taken to run the ten optimizations described above was 1:10:04 running on four processors on an IBM 3090 supercomputer. During this time the optimizations consumed an average of 3:58:10 CPU hours. Running in serial, on a machine about 1/10 the speed of the 3090, such as a workstation or small mainframe, the code would take roughly 40 hours to run. There are a number of ways to make PSA converge more quickly, and hence shorten the run time, and it is likely that the ten optimizations described above could have been completed successfully in less time, but the faster PSA converges, the greater the chance it will get stuck in a local minimum. By shortening run time excessively, we would compromise the reliability of the algorithm.

Analysis of Experimental Data

The fits we obtained for our experimental data were noticeably worse than those obtained for the simulated spectra, both in terms of χ^2 values and in terms of errors in rotational constants compared to literature values. The decreased accuracy of the analysis is most likely due to errors extant in the experimental spectra which we could not quantify, and which we therefore could not include in the simulated spectra.

Known properties of the experimental spectrum that are not in the model used to generated spectra, are rotational fine structure from vibrational mode coupling, and non-Boltzmann population distribution.(6,10,11)

Furthermore, since the data for a given spectrum is taken over the course of two or three days, it is also likely that the rotational temperature in the molecular beam varied slightly from one day to the next, making it impossible to match the temperature over the entire spectrum exactly. As mentioned above, the decrease in resolution caused by the convolution procedure can effectively hide some of these problems and allow us to fit the data without dealing with these problems explicitly.

Conclusions

The PSA approach is an effective technique for the analysis of rotationally resolved spectra. Although some input variables must be set by the user, the amount of user input required is minimal over the course of the optimization process. After a few decisions are made, including the limitations set on the parameter space, the starting temperature, the convolution width and the starting point, the optimization proceeds independently. Therefore, PSA can be used on the analysis of a spectrum in which relatively little is known about the molecular parameters. As long as the convolution width is sufficiently large and the cooling is sufficiently slow, optimization to the global minimum occurs with high probability. Furthermore, the computer time required for the optimization can be obtained from a common laboratory workstation. Thus, PSA is an ideal compromise between efficiency of optimization and thorough searching of parameter space for the analysis of rotationally resolved spectra.

Acknowledgement: This work is supported by: The National Institute of Health under grant #08-R9N527039A; The Office of Naval Research under grant #N00014-90-J-1971. The authors would like to gratefully acknowledge H. Li and H. M. Caffey for their computational assistance.

References

(1) S. Kirkpatrick, C. D. Gelatt, Jr., M. P. Vecchi, *Science*, **220**, 671-680 (1983).

(2) J. M. Stern, Technical Report, Cornell Computational Optimization Project, Report #90-1, (1990).

(3) R. H. J. M. Otten, L. P. P. van Ginneken, The Annealing Algorithm, §4.1,§8.2, Kluwer Academic Publishers, Boston,1989.

(4) Subroutine is a modified version of a program written by A. Maki, and furnished to us by G. T. Fraser, N.I.S.T, Gaithersburg, MD, 20899.

(5) W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling, Numerical Recipes, §10.6, Cambridge University Press, Cambridge, 1986.

(6) C. L. Brummel, S. W. Mork, L. A. Philips, *J. Chem. Phys.*, **95**, 7041-7053 (1991).

(7) H. Takeo, C. Matsumura, *J. Chem. Phys.*, **84**, 4205-4210 (1986).

(8) S. S. Butcher, R. A. Cohen, T. C. Rounds, *J. Chem. Phys.*, **54**, 4123-4124 (1971).

(9) H. Dreizler, D. Steffek, Z. *Naturforsch*, **36a**, 1239-1241 (1981).

(10) L. A. Philips, D.H. Levy, *J. Chem. Phys.*, **85**, 1327-1332 (1986); L.A. Philips, D.H. Levy, *J. Chem. Phys.*, **89**, 85-90 (1988).

(11) A.M. de Souza, D. Kaur, D. S. Perry, *J. Chem. Phys.*, **88**, 4569 (1987); A. McIlroy, D.J. Nesbitt, *J. Chem. Phys.*, **92.**, 2229 (1990); B. H. Pate, K.K. Lehmann, G. Scoles, *J. Chem. Phys..*, **95**, 3891-916 (1991).

Figure Captions

1. The SSA algorithm is displayed above in flow-chart form. SSA begins by setting current state (S) to the starting state, and temperature (T) to the starting temperature. A new state S' is generated by the MOVE function, and the cost of this state is determined. If the cost of S' is lower than the cost of S, then we set S = S', otherwise the move is made if a uniform random number over [0,1] is less than $e^{-(COST(S')-COST(S))/kT}$, as described in Eq. (1). After we have moved from one state to another enough times to approximate the equilibrium density of the potential surface for that temperature, we multiply the temperature by some cooling factor (c) slightly less than one. If we determine that the optimization has frozen, we end the run.

2. One dimensional cross-sections of potential surfaces are shown, which were created by evaluating the cost function for a range of excited state C rotational constant values, while leaving all other parameters fixed. The curve represented by (——) shows the potential surface when the spectra are convolved with a 0.005 cm^{-1} half-width. The curve represented by (- - -) is the surface with half-width set to 0.05 cm^{-1}.

3. Top Left: A) two gaussian functions with σ=0.02. B) two gaussian *functions with σ=0.06. Both pairs are separated by* 0.40 cm^{-1} *from center to* center.

Bottom Left: The difference in intensity squared (ΔI^2) error functions measuring the difference between (——) and (- - -) are normalized for convolution width. The total normalized error ($\Sigma \Delta I^2 \times \sigma$) for the narrow gaussians is 0.564190, and the total error for the wide gaussians is 0.564181. Note that for large deviations, total normalized error for wide and narrow gaussian functions are virtually identical.

Top Right: The same gaussian functions as displayed on the left are shown with the exception that the pairs are split by 0.08 cm^{-1} from center to center

Bottom Right: The error function for the top right panel are plotted. The total error for the narrow gaussians is 0.564190, while total error for the wide gaussians is 0.202442. Note that for smaller deviations in frequency, the error function of the wide gaussian functions drops substantially, while the total error for the narrow gaussian function remains almost entirely unchanged.

4. A) An experimental spectrum of 2-Fluoroethanol convolved with 0.05 cm^{-1} half-width. B) The results of a fit to 2-Fluoroethanol, also convolved with 0.05 cm^{-1} half-width. C) The magnitude of intensity difference for A and B. The cost function is generated using the area under the square of this function.

5. A) The PSA algorithm is displayed in flow-chart form. PSA begins by setting current state (S) to the starting state, and temperature (T) to the starting temperature. The maximum step sizes for the move function are set inversely proportional to the partial derivatives of each parameter averaged over the potential surface. Four processes run in parallel. At the beginning of each epoch, processes #2-4 are started from the best 3 states located so far, while process #1 is started on the results of process #1 from the previous epoch. If the routine has frozen, the best state is optimized using a Polak-Ribere minimization, otherwise the temperature is reduced as described in Eqn. (2). B) The algorithm for the PSA epoch is displayed in flow chart form

Unlike SSA, the PSA epoch runs for a fixed # of attempted moves. A new state S' is generated by the move function, as described in the text. The parameters in S' are then used to generate a spectrum, which is convolved and compared to the convolved laboratory spectrum by the cost function. If COST(S') < COST(S) then S is set to S'. Otherwise the move is made if a uniform random number over [0,1] is less than $e^{-(COST(S')-COST(S))/kT}$, as described in Eq. (1). A running comparison is made to save the best three states to begin the epoch.

6. The results of run #7 are displayed. A) The simulated spectrum with errors added. B) The spectrum generated from the starting state input to PSA. C) The spectrum generated from the results of PSA optimization.

7. PSA optimization of 2-Fluoroethanol. A) The experimental spectrum of 2-Fluoroethanol. B) The spectrum generated from the starting state given to PSA. C) The spectrum generated from the results of PSA optimization. Note that the 2-Fluoroethanol spectrum exhibits rotational fine structure (6) which does not inhere in the model used to calculate spectra, so the match to the experimental spectrum is not ideal.

8. PSA optimization of Difluoroethane. A) The experimental spectrum of Difluoroethane. B) The spectrum generated from the starting state given to PSA. C) The spectrum generated from the results of PSA optimization.

9. A) (**x**) is the cost of the same fit evaluated using different array sizes for the convolution array. (—) is COST(S)*ARRAY(S)/ARRAY(Y) as described in

Eq. (4). B) (**x**) is the cost of the same fit evaluated using different half-widths for the convolution. (——) is COST(S)*WIDTH(S)/WIDTH(Y) as described in Eq. (4).

Table Captions

Table I - Ten sets of parameters used to generate the simulated spectra on which PSA was run.

Table II -Parameters produced by PSA fits to the spectra represented in Table I.

Table III - Errors in the values given in Table II, expressed as percentages of the values given in Table I.

Table IV - Results of a repeat of run #10 with narrower (0.03 cm^{-1}) convolution. Note that the errors for this run are substantially smaller than for run #10 with wider convolution, as represented in Tables I, II and III.

Table V - Results of a run given a larger parameter space. The parameter space for run #11 is roughly 2000 times the size of the parameter spaces in runs #1-10.

Table VI -

[a]Literature values taken from reference (6)

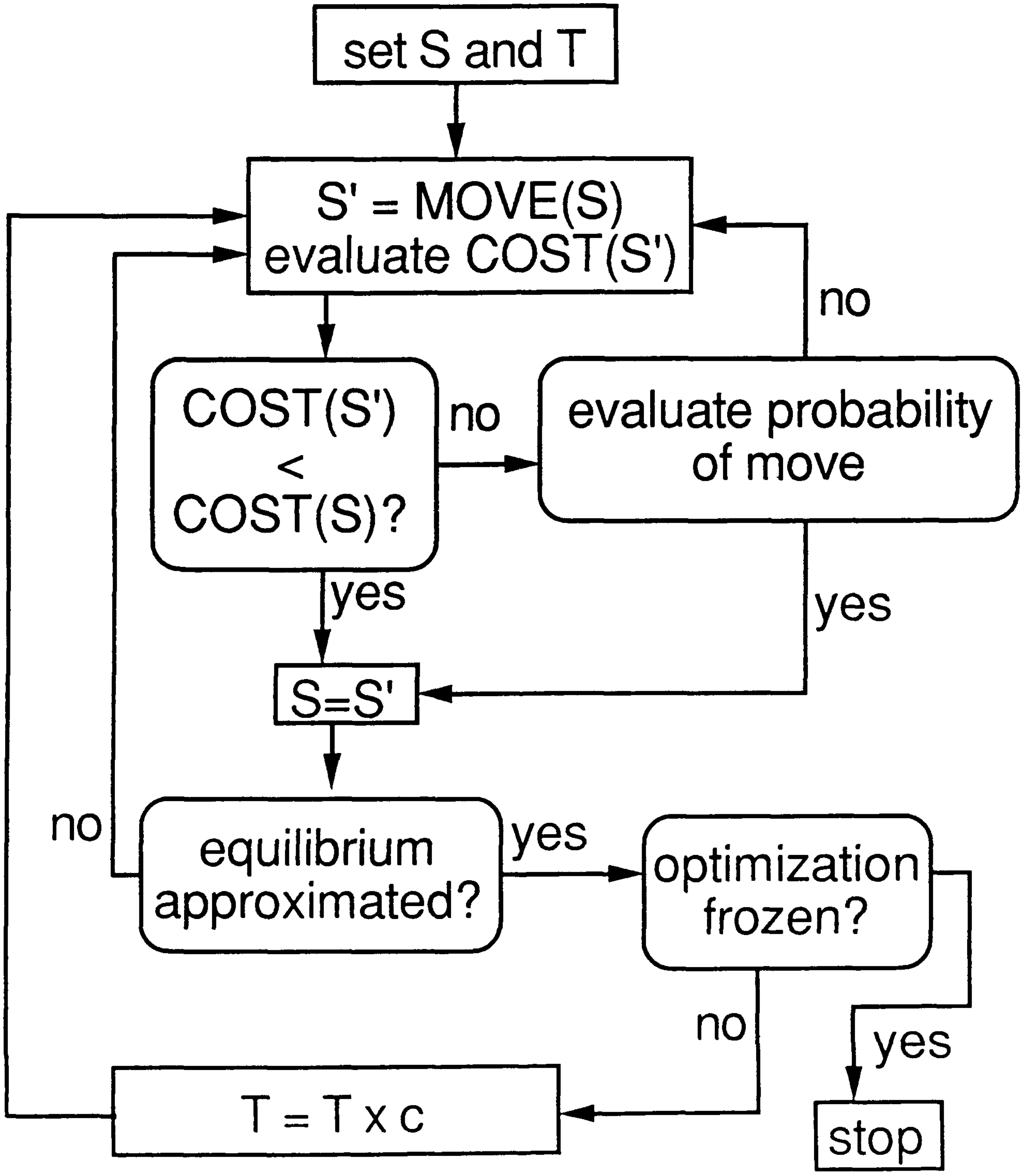
Table VII -

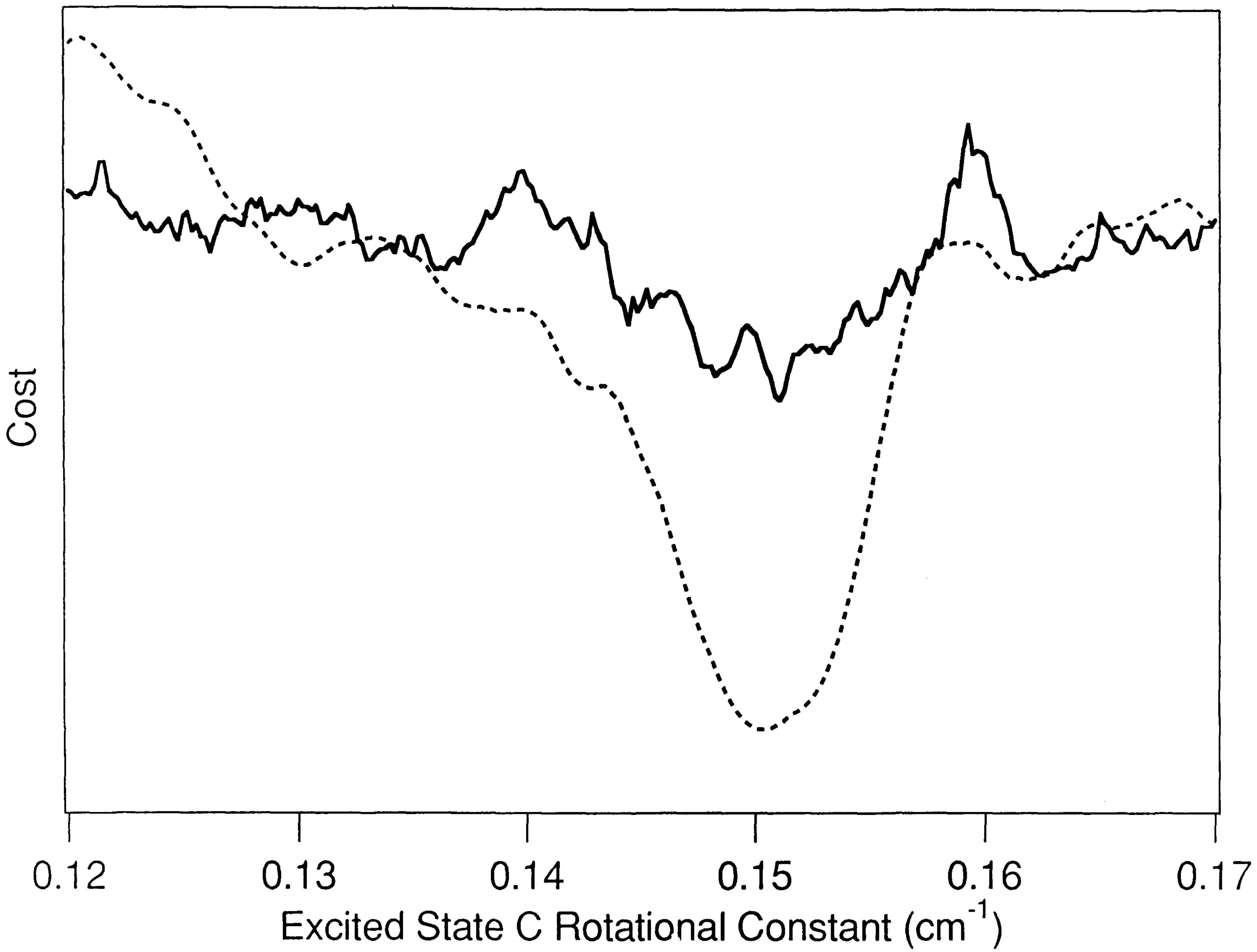
[a]Literature values averaged from 3 microwave studies of Difluororethane from references (7,8,9).

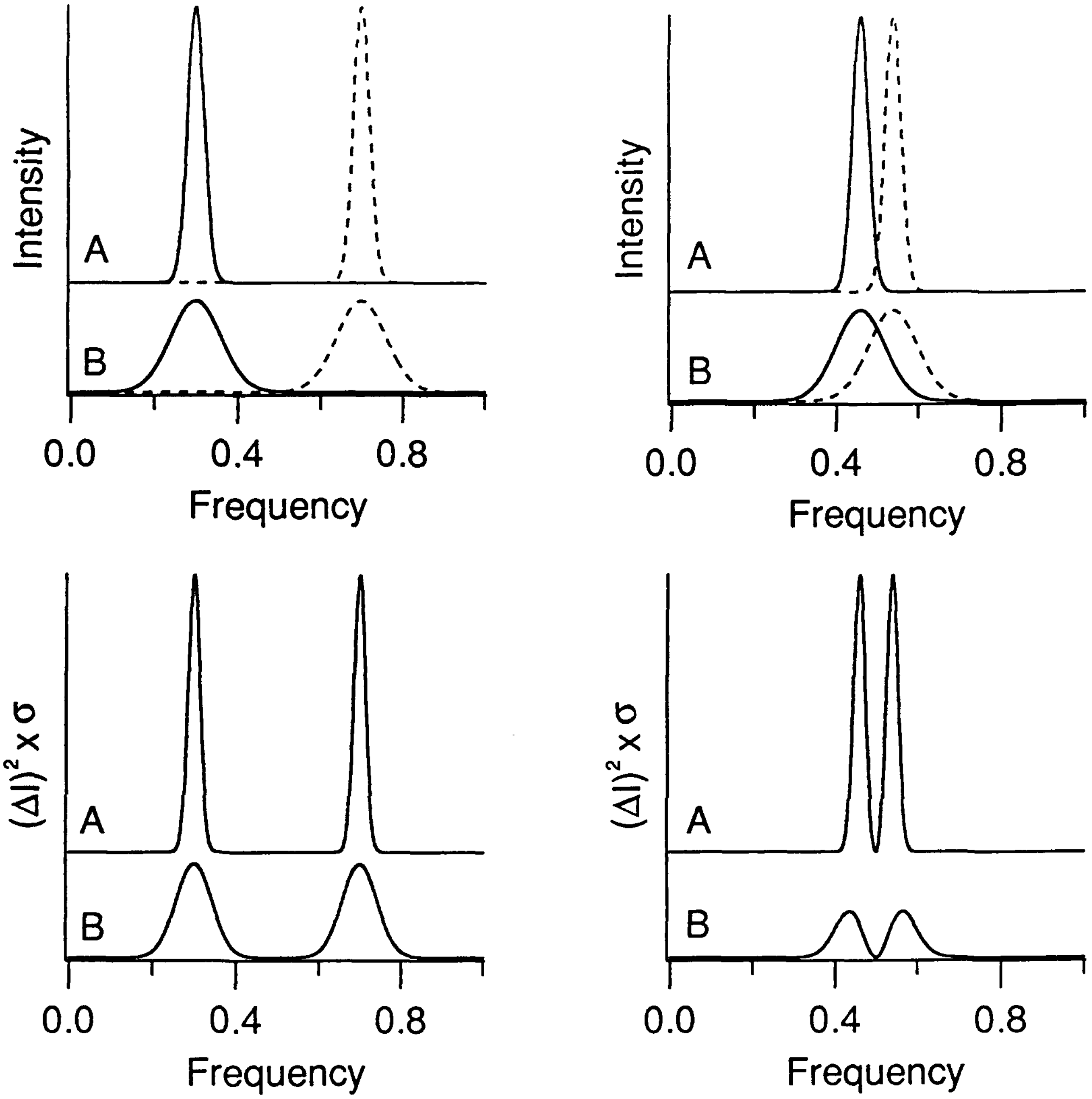
Table VIII - χ^2, RMS deviation, RMS/χ^2, and Temperature from Runs #1-10 are displayed. Note that the ratio of RMS to χ^2 is partially correlated with temperature.

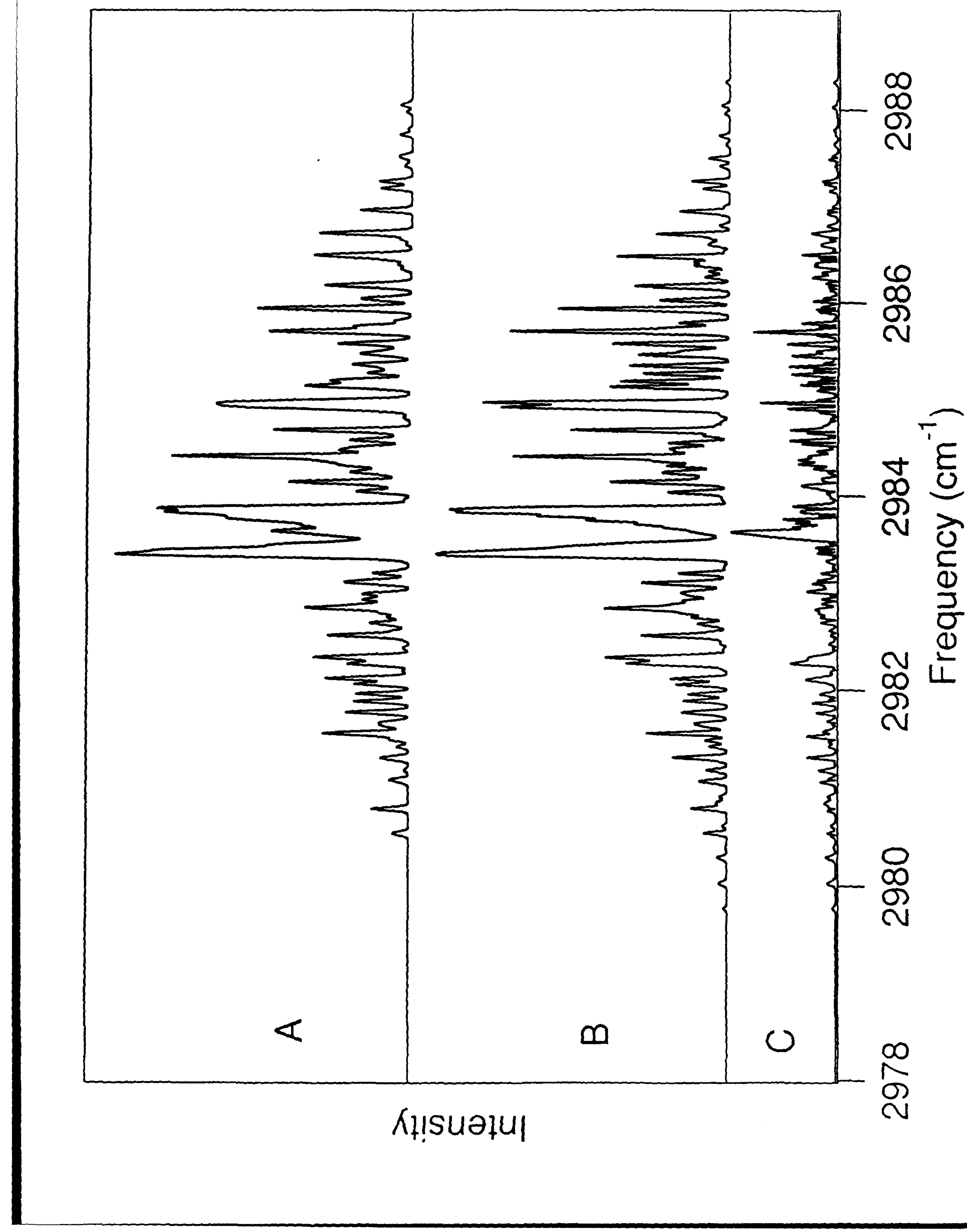
List of Tables

Simulated Annealing

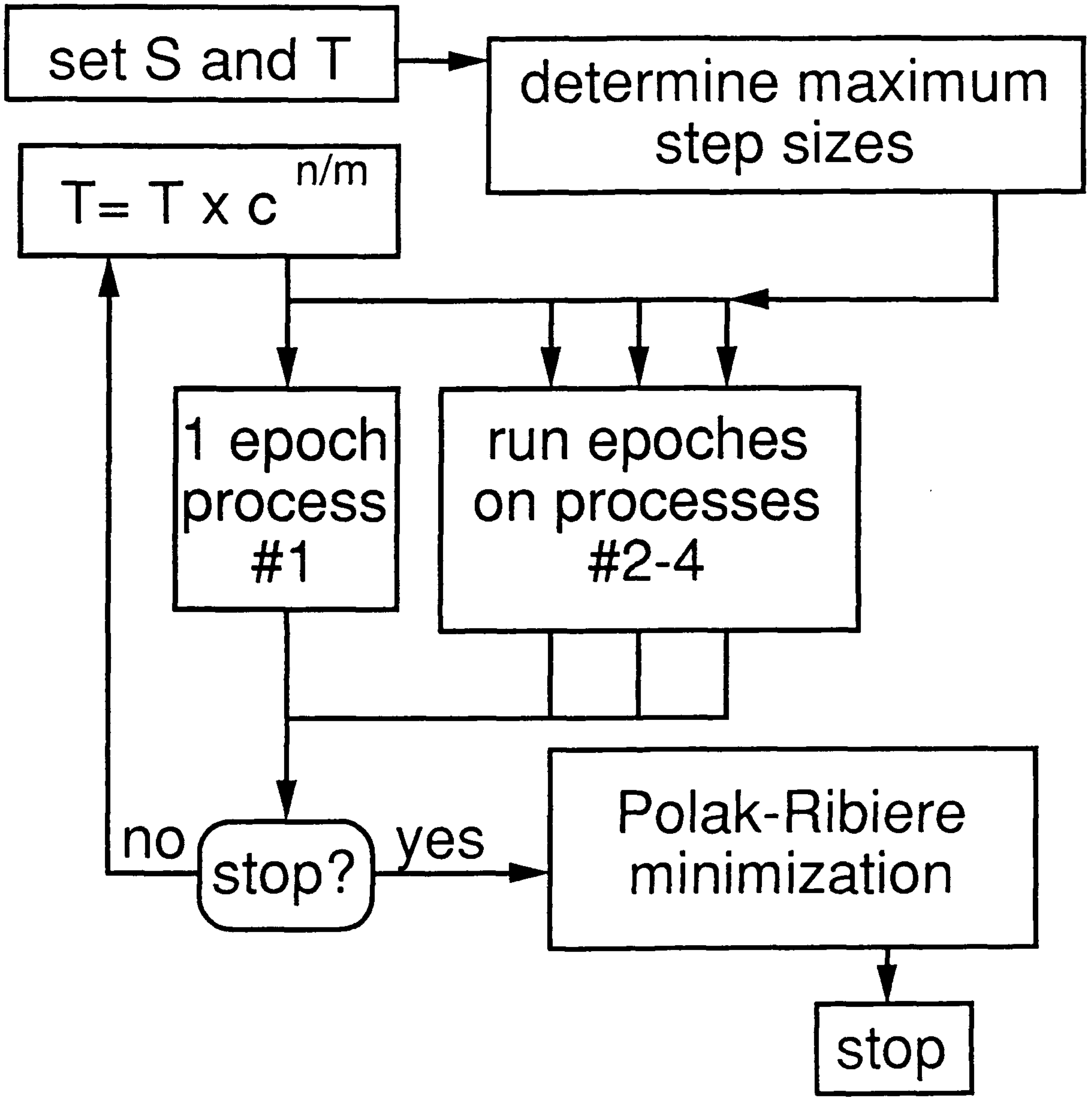



Cost
0.12
0.13
0.14
0.15
0.16
0.17
Excited State C Rotational Constant (cm^{-1})

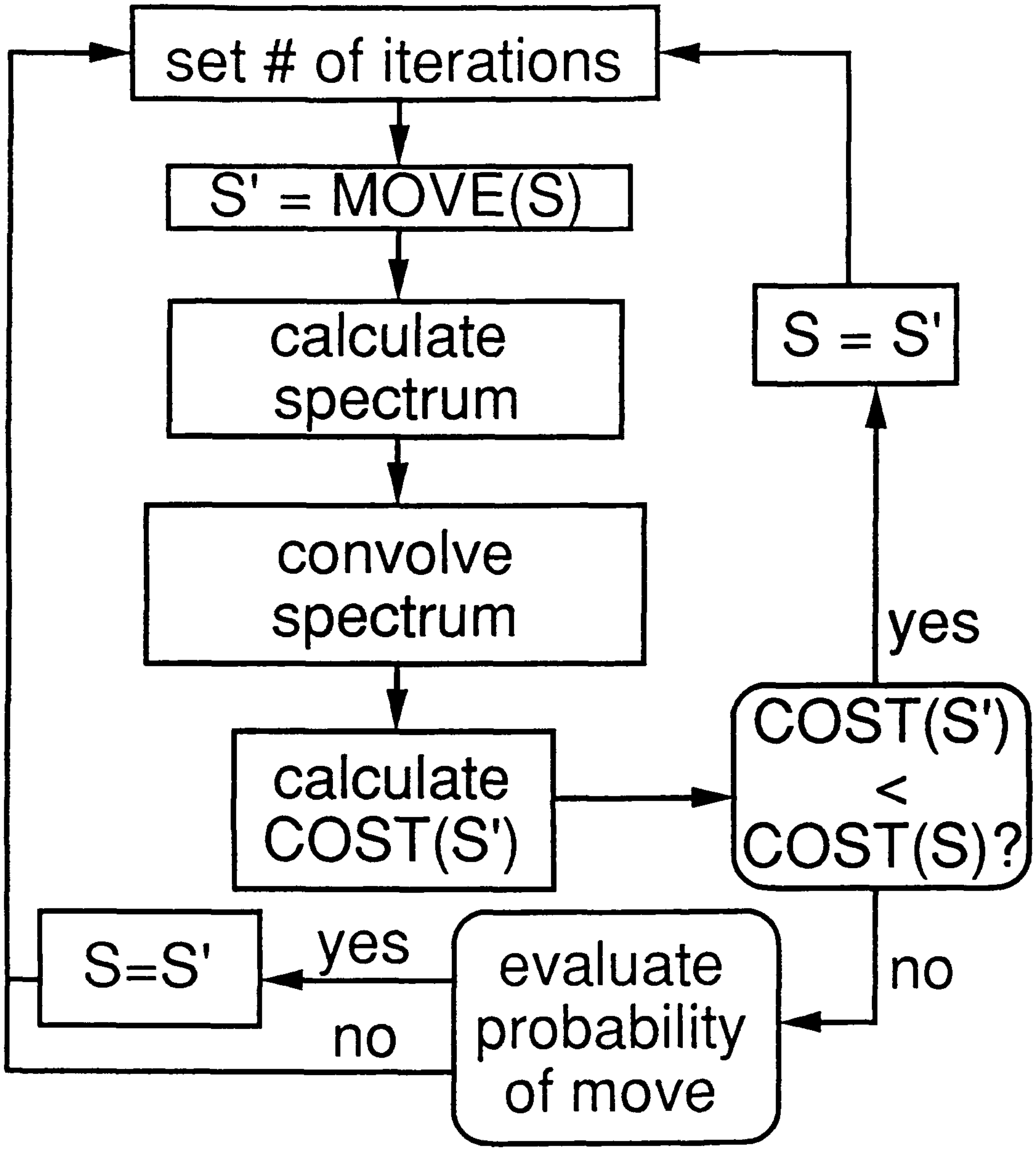

Intensity
A
B
0.0
0.4
0.8
Frequency
Intensity
A
B
0.0
0.4
0.8
Frequency
$(\Delta I)^2 \times \sigma$
A
B
0.0
0.4
0.8
Frequency
$(\Delta I)^2 \times \sigma$
A
B
0.0
0.4
0.8
Frequency


A
B
C
Intensity
2978
2980
2982
2984
2986
2988
Frequency (cm^{-1})

Parallel Simulated Annealing


set S and T
determine maximum step sizes
T= T x c $^{n/m}$
1 epoch process #1
run epoches on processes #2-4
no
stop?
yes
Polak-Ribiere minimization
stop

1 PSA Epoch:



set # of iterations

S' = MOVE(S)

calculate spectrum

convolve spectrum

calculate COST(S')

COST(S') < COST(S)?

yes

S = S'

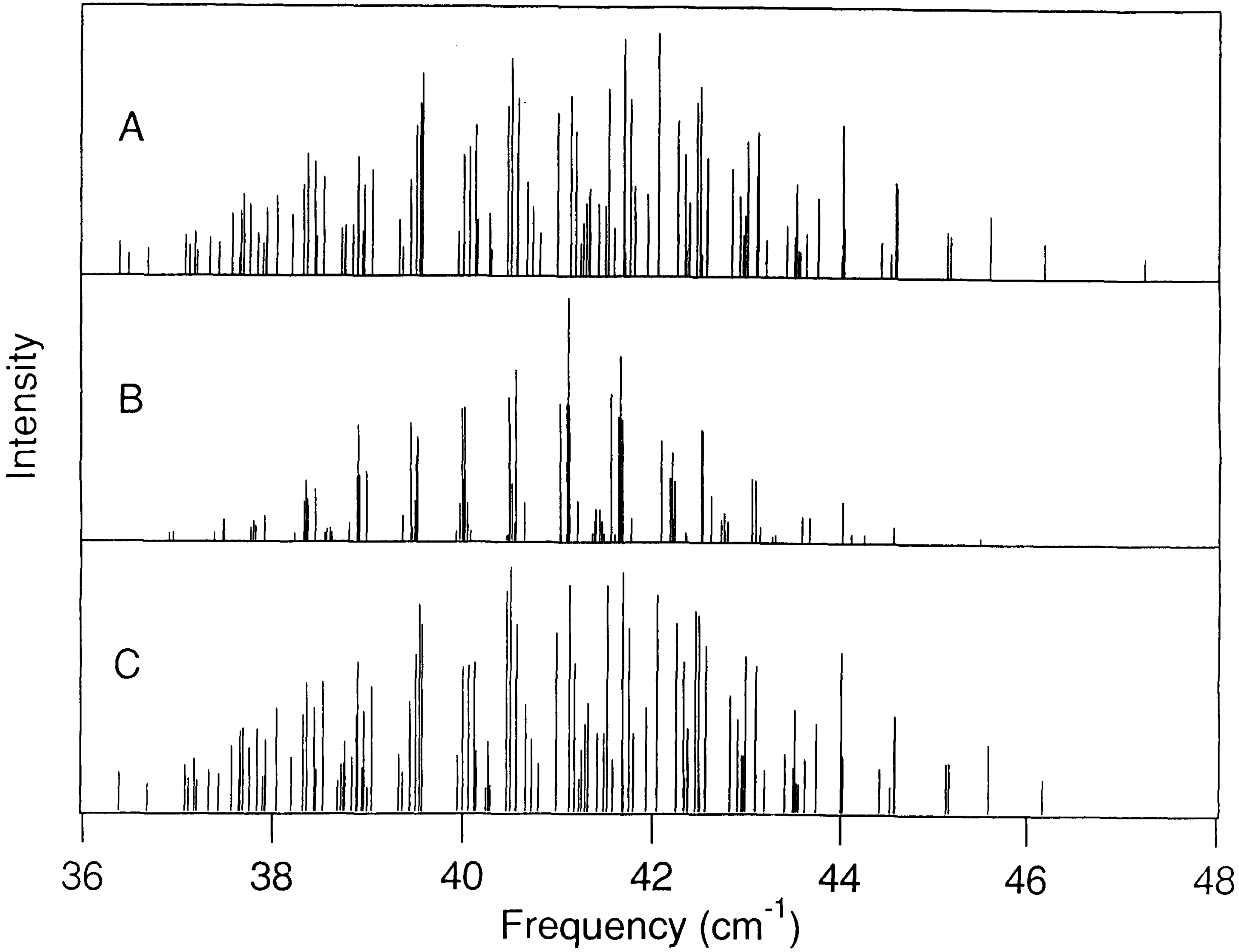
no

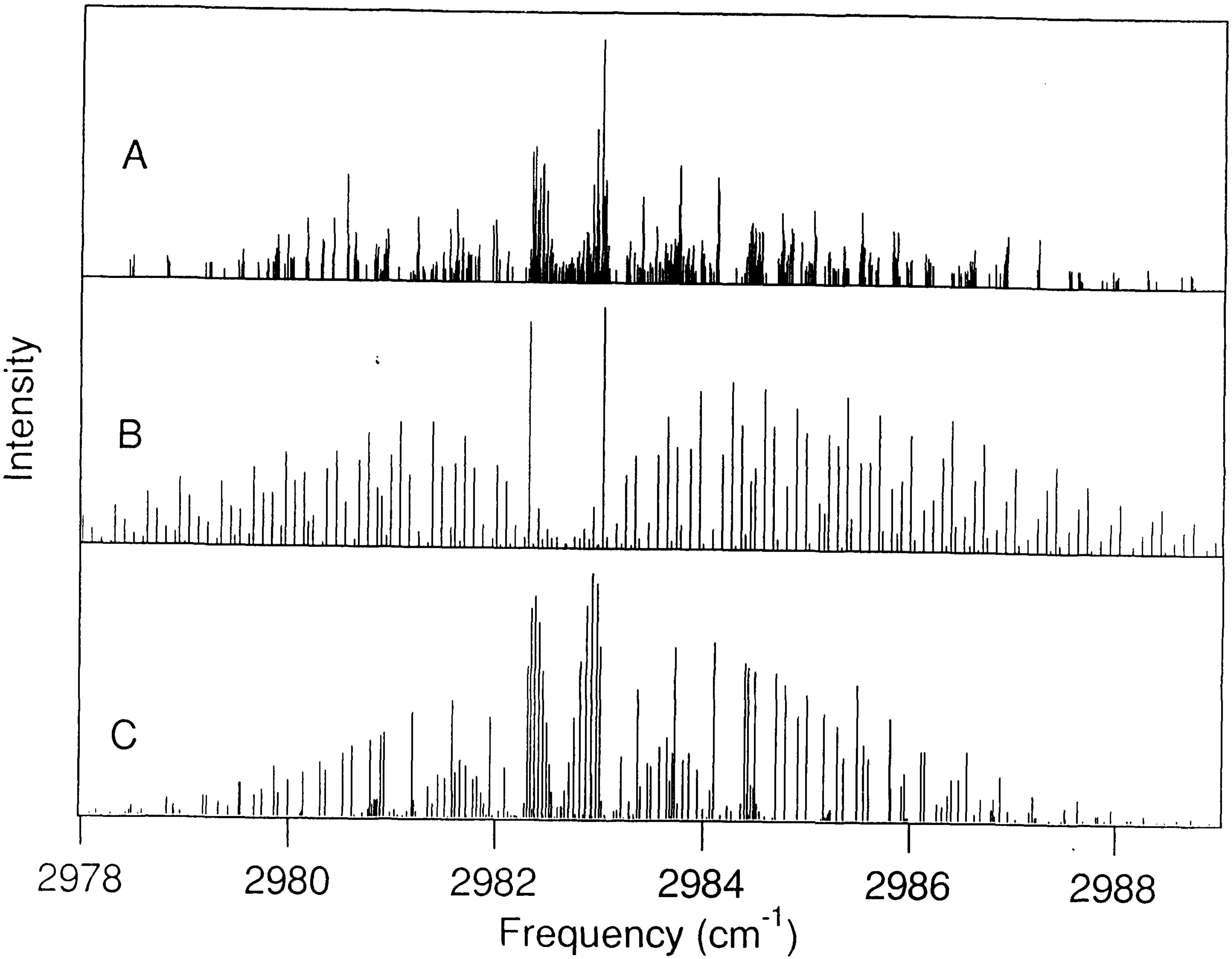
evaluate probability of move

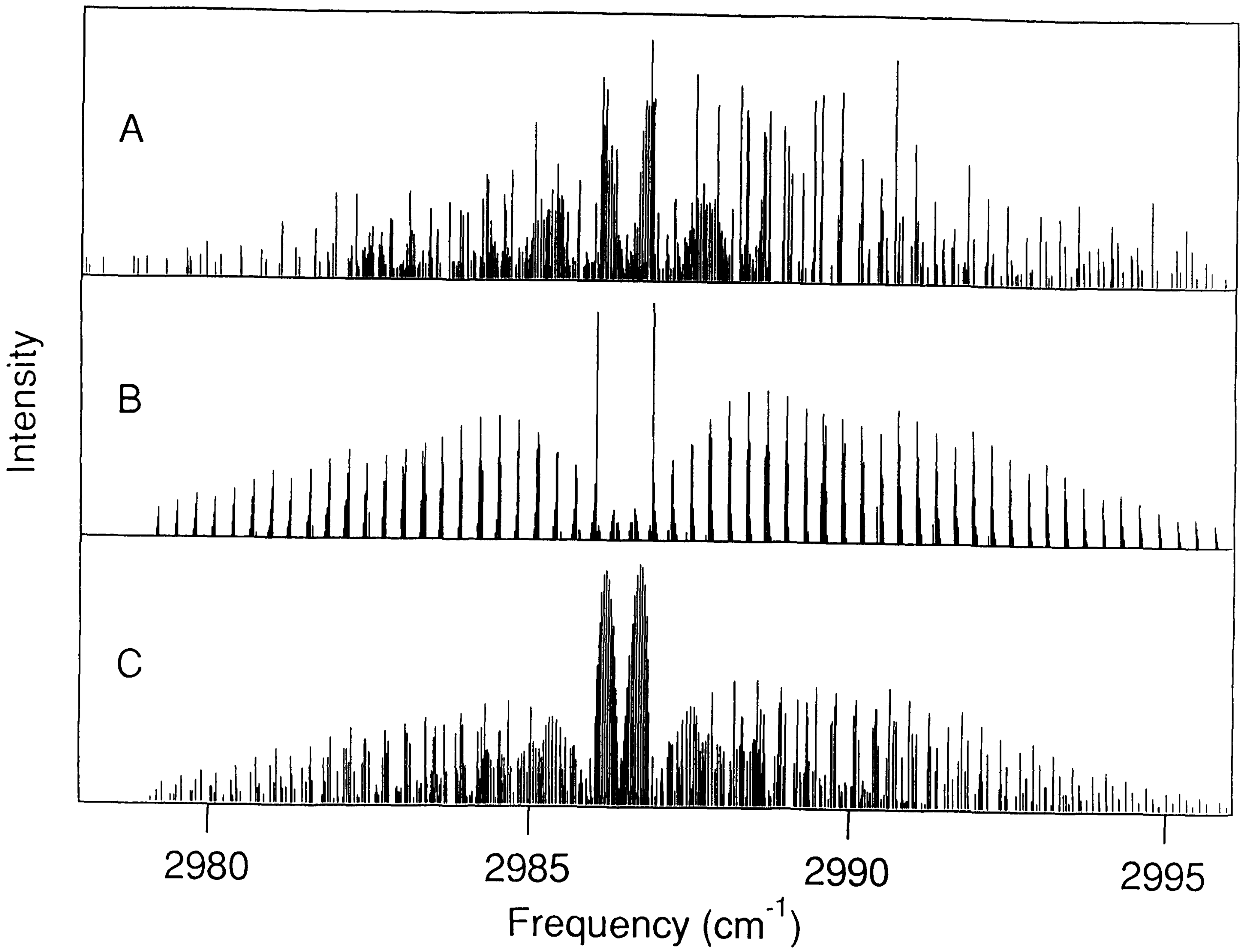
yes

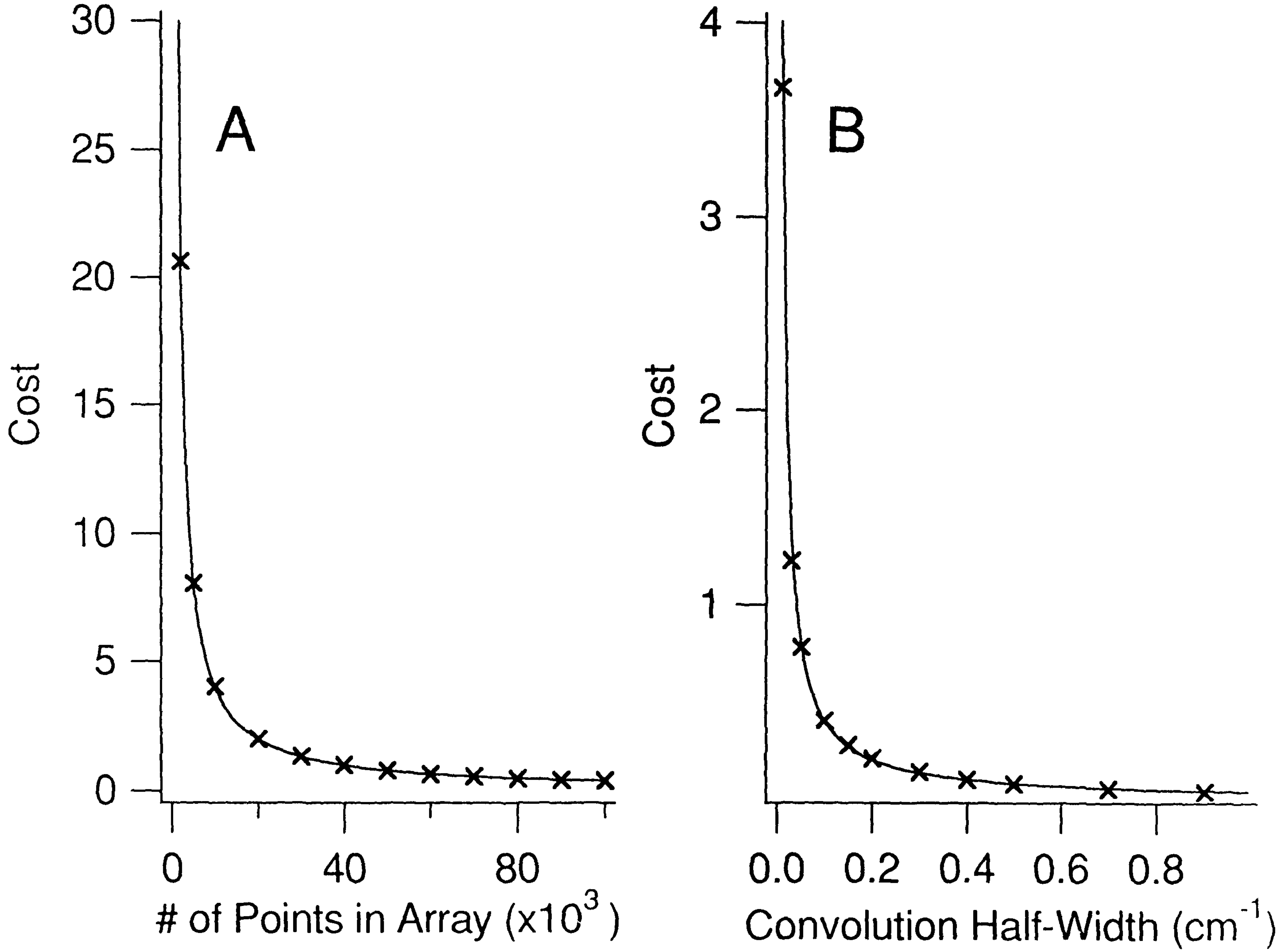
S=S'

no


A
B
C
Intensity
36
38
40
42
44
46
48
Frequency (cm^{-1})


A
B
C
Intensity
2978
2980
2982
2984
2986
2988
Frequency (cm^{-1})


A
B
C
Intensity
2980
2985
2990
2995
Frequency (cm^{-1})


A
Cost
30
25
20
15
10
5
0
0
40
80
# of Points in Array (x10^{3})
B
Cost
4
3
2
1
0.0
0.2
0.4
0.6
0.8
Convolution Half-Width (cm^{-1})

Parameters for Simulated Spectra

Run Number	A''	B''	C''	A'	B'
	(cm^{-1})	(cm^{-1})	(cm^{-1})	(cm^{-1})	(cm^{-1})
1	0.645455	0.460125	0.263068	0.639878	0.452087
2	0.292634	0.179285	0.159544	0.300411	0.180556
3	0.396232	0.385118	0.186761	0.407835	0.383043
4	0.495038	0.347534	0.270012	0.491552	0.357123
5	0.582688	0.309138	0.190527	0.579643	0.309986
6	0.473862	0.322565	0.190892	0.469058	0.319835
7	0.691545	0.294061	0.259242	0.720883	0.293749
8	0.343778	0.253228	0.208633	0.351268	0.243826
9	0.181767	0.148911	0.122387	0.173430	0.141700
10	0.467185	0.345608	0.150506	0.449992	0.340996

Run Number	C'	T	μ_A	μ_B	μ_C
	(cm^{-1})	(K)			
1	0.273244	5.28903	0.888871	0.349631	0.296085
2	0.162969	14.22520	0.869417	0.484707	0.095780
3	0.179620	4.71573	0.221728	0.830584	0.510849
4	0.256965	4.67846	0.988759	0.149512	0.001247
5	0.181010	13.76580	0.153013	0.774748	0.613476
6	0.197009	5.43611	0.507746	0.216616	0.833829
7	0.264794	4.35626	0.651408	0.739291	0.170634
8	0.218166	12.37900	0.887586	0.360470	0.286797
9	0.121808	3.56478	0.951796	0.039762	0.304142
10	0.144638	3.96290	0.984210	0.173267	0.036180

Output from PSA

Run Number	A''	B''	C''	A'	B'
	(cm^{-1})	(cm^{-1})	(cm^{-1})	(cm^{-1})	(cm^{-1})
1	0.643217	0.459794	0.263360	0.637986	0.451860
2	0.291581	0.178969	0.159939	0.299505	0.180190
3	0.395844	0.386259	0.186489	0.407782	0.383814
4	0.494682	0.347559	0.270127	0.491347	0.356891
5	0.583153	0.309264	0.189892	0.580438	0.309833
6	0.473526	0.322959	0.191314	0.468743	0.320417
7	0.691579	0.294068	0.259501	0.720704	0.293768
8	0.343111	0.252891	0.208462	0.350801	0.243602
9	0.180629	0.148419	0.122267	0.172248	0.141418
10	0.468441	0.345974	0.150443	0.450219	0.341937

Run Number	C'	T	μ_A	μ_B	μ_C
	(cm^{-1})	(K)			
1	0.273444	5.25183	0.890958	0.359591	0.277287
2	0.163302	14.02077	0.868156	0.482753	0.115127
3	0.179421	4.78813	0.208002	0.831579	0.514987
4	0.257120	4.84683	0.992801	0.119176	0.011967
5	0.180910	13.55277	0.015681	0.782984	0.621844
6	0.197232	5.34898	0.486779	0.300063	0.820371
7	0.264936	4.38413	0.645669	0.740744	0.185500
8	0.217971	12.45320	0.886620	0.357309	0.293660
9	0.121709	3.58339	0.952073	0.087244	0.293165
10	0.144515	4.08328	0.980218	0.195241	0.032447

Errors in PSA Output

Run Number	A'' (%)	B'' (%)	C'' (%)	A' (%)	B' (%)	C' (%)
1	0.347	0.0720	0.111	0.296	0.0501	0.0731
2	0.360	0.176	0.248	0.302	0.203	0.204
3	0.0979	0.296	0.146	0.0131	0.201	0.111
4	0.0719	0.00729	0.0424	0.0417	0.0649	0.0603
5	0.0798	0.0409	0.333	0.137	0.0495	0.0553
6	0.0710	0.122	0.221	0.0671	0.182	0.113
7	0.00488	0.00237	0.100	0.0249	0.00650	0.0537
8	0.194	0.133	0.0820	0.132	0.0919	0.0894
9	0.626	0.331	0.0982	0.682	0.199	0.0816
10	0.269	0.106	0.0421	0.0504	0.276	0.0855
MEAN	0.212	0.129	0.142	0.175	0.132	0.0928

Run Number	T (%)	μ_A (%)	μ_B (%)	μ_C (%)	χ^2	RMS (cm^{-1})
1	0.703	0.235	2.05	6.35	0.822	$2.75x10^{-3}$
2	1.44	0.145	0.403	20.2	0.306	$4.99x10^{-3}$
3	1.54	6.19	0.120	0.810	0.307	$2.77x10^{-3}$
4	3.60	0.409	20.3	860.	0.699	$1.42x10^{-3}$
5	1.55	89.8	1.06	1.36	0.808	$2.01x10^{-2}$
6	1.60	4.13	38.5	1.61	0.511	$3.03x10^{-3}$
7	0.640	0.881	0.196	8.71	0.842	$1.61x10^{-3}$
8	0.599	0.109	0.877	2.39	0.244	$5.61x10^{-3}$
9	0.522	0.0290	119.	3.61	0.254	$3.23x10^{-3}$
10	3.04	0.406	12.7	10.3	1.06	$3.90x10^{-3}$
MEAN	1.52	10.2	19.6	91.5	0.585	$4.95x10^{-3}$

Run #10: 0.03 cm^{-1} Convolution

	Actual Value	Fit Value	Error (%)
A"(cm^{-1})	0.467185	0.467094	0.0195
B"(cm^{-1})	0.345608	0.345569	0.0112
C"(cm^{-1})	0.150506	0.150548	0.0277
A' (cm^{-1})	0.449992	0.449737	0.0567
B' (cm^{-1})	0.340996	0.341036	0.0115
C' (cm^{-1})	0.144638	0.144664	0.0176
T (K)	3.962900	4.069376	2.69
μ_A	0.984210	0.977289	0.703
μ_B	0.173267	0.209044	20.6
μ_C	0.036180	0.034741	3.98
χ^2	1.12		
RMS (cm^{-1})	7.24×10^{-4}		

Run #11: Larger Parameter Space			
	Actual Value	Fit Value	Error (%)
A"(cm^{-1})	0.645455	0.644854	0.0930
B"(cm^{-1})	0.460125	0.461424	0.282
C"(cm^{-1})	0.263068	0.262927	0.0535
A' (cm^{-1})	0.634301	0.633824	0.0752
B' (cm^{-1})	0.444049	0.444589	0.122
C' (cm^{-1})	0.283421	0.283526	0.0369
T (K)	5.289030	5.271796	0.326
μ_A	0.888871	0.885534	0.375
μ_B	0.349631	0.369878	5.79
μ_C	0.296085	0.281104	5.06
χ^2	1.23		
RMS (cm^{-1})	3.51x10^{-3}		

PSA Analysis of 2-Fluoroethanol

	Starting Value	Range	Fit Value	Literature Value[a]	Error (%)
A"(cm^{-1})	0.510000	0.48-0.57	0.527650	0.52964	0.375
B"(cm^{-1})	0.155000	0.15-0.19	0.181523	0.18048	0.579
C"(cm^{-1})	0.155000	0.12-0.16	0.148301	0.15105	1.82
A' (cm^{-1})	0.510000	A" ±0.05	0.530663	0.53223	0.294
B' (cm^{-1})	0.155000	B" ±0.05	0.180587	0.17944	0.639
C' (cm^{-1})	0.155000	C" ±0.05	0.147550	0.15087	2.20
T (K)	8.000000	3.0-10.0	3.994539	na	na
μ_A	0.000000	fixed	0.000000	na	na
μ_B	0.000000	fixed	0.000000	na	na
μ_C	1.000000	fixed	1.000000	na	na
χ^2	4.27				
RMS (cm^{-1})	1.59×10^{-2}				

PSA Analysis of Difluoroethane

	Starting Value	Range	Fit Value	Literature Value[a]	Error (%)
A"(cm^{-1})	0.590000	0.55-0.65	0.578768	0.577813	0.165
B"(cm^{-1})	0.150000	0.13-0.17	0.165418	0.167221	1.08
C"(cm^{-1})	0.150000	0.13-0.17	0.147045	0.146192	0.583
A' (cm^{-1})	0.590000	A" ±0.05	0.577941	na	na
B' (cm^{-1})	0.150000	B" ±0.05	0.165606	na	na
C' (cm^{-1})	0.150000	C" ±0.05	0.146887	na	na
T (K)	15.000000	3.0-23.0	13.100225	na	na
μ_A	0.000000	0.0-1.0	0.456695	na	na
μ_B	0.000000	fixed	0.000000	na	na
μ_C	1.000000	0.0-1.0	0.889623	na	na
χ^2	7.62				
RMS (cm^{-1})	2.15x10^{-2}				

RMS Deviation vs. χ^2				
Run Number	χ^2	RMS (cm^{-1})	RMS/χ^2 (cm^{-1})	T (K)
1	0.822	$2.75 x 10^{-3}$	0.00335	5.28903
2	0.306	$4.99 x 10^{-3}$	0.0163	14.22520
3	0.307	$2.77 x 10^{-3}$	0.00904	4.71573
4	0.699	$1.42 x 10^{-3}$	0.00203	4.67846
5	0.808	$2.01 x 10^{-2}$	0.0249	13.76580
6	0.511	$3.03 x 10^{-3}$	0.00593	5.43611
7	0.842	$1.61 x 10^{-3}$	0.00191	4.35626
8	0.244	$5.61 x 10^{-3}$	0.0230	12.37900
9	0.254	$3.23 x 10^{-3}$	0.0127	3.56478
10	1.06	$3.90 x 10^{-3}$	0.00369	3.96290

TECHNICAL REPORT DISTRIBUTION LIST - GENERAL

Office of Naval Research (2)*
Chemistry Division, Code 1113
800 North Quincy Street
Arlington, Virginia 22217-5000

Dr. James S. Murday (1)
Chemistry Division, Code 6100
Naval Research Laboratory
Washington, D.C. 20375-5000

Dr. Robert Green, Director (1)
Chemistry Division, Code 385
Naval Air Weapons Center
Weapons Division
China Lake, CA 93555-6001

Dr. Elek Lindner (1)
Naval Command, Control and Ocean
Surveillance Center
RDT&E Division
San Diego, CA 92152-5000

Dr. Bernard E. Douda (1)
Crane Division
Naval Surface Warfare Center
Crane, Indiana 47522-5000

Dr. Richard W. Drisko (1)
Naval Civil Engineering
Laboratory
Code L52
Port Hueneme, CA 93043

Dr. Harold H. Singerman (1)
Naval Surface Warfare Center
Carderock Division Detachment
Annapolis, MD 21402-1198

Dr. Eugene C. Fischer (1)
Code 2840
Naval Surface Warfare Center
Carderock Division Detachment
Annapolis, MD 21402-1198

Defense Technical Information
Center (2)
Building 5, Cameron Station
Alexandria, VA 22314

* Number of copies to forward