NASA CR-159,238

NASA Contractor Report 159238

NASA-CR-159238

1980 00 12560

ADAPTATION AND OPTIMIZATION OF A LINE-BY-LINE RADIATIVE TRANSFER PROGRAM FOR THE STAR-100 (STARSMART)

Pamela L. Rarig

SYSTEMS AND APPLIED SCIENCES CORPORATION
Riverdale, Maryland 20840

NASA Contract NAS1-15481
March 1980

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

SUMMARY

STARSMART is a highly structured, optimized and efficient infrared line-by-line radiative transfer program designed for the Langley Research Center CDC STAR-100 computer. A vector oriented PASCAL derivative language, SL/1, is used to reduce core requirements, development effort and run time. The half-word arithmetic feature, virtual memory and vector processing of the STAR-100 are employed to reduce execution time by a factor of 99 over an earlier serial code performing identical tasks. The radiative transfer formulation is analyzed in depth to produce optimum vectorized code, data structure and efficient page management schemes. In addition, the STAR-100 halfword arithmetic and input data characteristics are evaluated for conditions leading to incorrect results or loss of significance.

The computational speed and storage requirements of STARSMART are shown relative to 2 earlier vector and serial codes. Benchmarks are performed with simple and elaborate atmospheric models. The largest test case assumes a midlatitude summer atmosphere with 4 gases and 15 layers. The bandpass extends from 2075 to 2215 cm-1 and contains 4666 spectral lines with 14000 integration points. The required 979,860,000 calculations are performed in 120.88 CPU seconds.

N 80 - 21044 #

TABLE OF CONTENTS

SECTION 1 - INTRODUCTION

A line-by-line radiative transfer program (SMART) (Ref. 2) has been running on the conventional serial machines (CDC Series 6000 and CYBER 170's) at NASA Langley Research Center since the early seventies. This code calculates the infrared spectral flux at points within a given frequency interval, propagating through a non-homogeneous atmosphere. SMART is primarily used to simulate the instrument response of gas filter correlation radiometers developed to monitor air pollution.

Studies requiring large bandpasses, high spectral resolution, multiple gases and many atmospheric layers are costly to run. Furthermore, the storage limitations imposed by the core size of the serial machines have made the more finely tuned cases impossible to run without incurring significant additional costs of design and checkout of less efficient overlaid schemes. Such segmented approaches are of limited use because upcoming releases of the FORTRAN compiler are expected to delete the current overlay capability.

In principle, virtual memory allows the software designer to handle large data structures with no concern for physical memory size; and the improved computational efficiency of the vector oriented machine should provide the optimum capability to run large models at moderate cost.

Therefore an effort was undertaken to adapt and optimize the SMART serial code for use on the STAR-100. The starting point for this task was a partially completed vector code. However, analysis of this code revealed errors in logic and inefficient use of the vector capabilities. Because of these problems, it was necessary to begin again from the serial version. The approach used in this effort, the details of the programming procedures and results in terms of improved efficiency are described in the following sections.

SECTION 2 - APPROACH

The radiative transfer calculations as performed by SMART can be viewed as a two phase process. The first phase involves the processing of user parameters which define the multi-layer atmospheric model and the calculation of wavenumber dependent absorption coefficients for each layer at each user specified integration point. In the second phase the calculation of the transmissivity and emissivities is performed at each integration point. The optical path and gas concentrations in each layer as well as solar and surface effects are included in the calculations. These values are used as inputs to parametric studies of instrument response to atmospheric conditions.

Of the two phases, the first uses the most computer and peripheral time. For a virtual memory code to result in an over-all cost reduction for large cases, the first phase would have to be recast in a strongly vectorized form. The second phase would still be implemented on the serial machine where access to the more sophisticated system software allows graphic and tabular display.

The adaptation and optimization of the serial code to the STAR-100 was broken down into the following tasks:

(1) analysis of code for vectorization

(2) structuring of data and development of page management schemes

(3) selection of an appropriate language

(4) comparison of implicit and explicit I/O

(5) evaluation of computer arithmetic

(6) benchmarking and analysis of results

SECTION 3 - LANGUAGE

The selection of an appropriate language simplifies the design of an efficient algorithm by satisfying the specific needs and structure of the task. Consideration should be given to the control structures, data types, diagnostics and levels of optimization offered by the available languages. A higher order language is a software tool which enables the designer to exploit hardware features without having to introduce unclear code. Good code should be easy to read, understand, maintain and modify.

The STAR-100 supports three languages:

(1) STAR FORTRAN - A vectorized upward compatible version ans X3.9 FORTRAN 66

(2) META - The STAR-100 assembler language

(3) SL/1 - A vector-oriented PASCAL derivative

The first version of the vectorized code (STARSMART) was coded in STAR FORTRAN. Its similarity to the serial NOS FORTRAN made it the best language to use for a quick transliteration of the serial code onto the STAR-100. Although STAR FORTRAN was the initial target language, it does not take advantage of the powerful halfword arithmetic features of the STAR-100. Halfword operations could be performed by STAR FORTRAN callable META subroutines, but any halfword values had to remain internal to the META code. Another deficiency of STAR FORTRAN is the lack of runtime diagnostic support. For example, upon abnormal termination of a STAR FORTRAN run, the user is left with a hard to decipher hexadecimal core dump.

Exclusive programming in META was ruled out because the time required to generate and validate assembler code and the inherent difficulty in documentation were judged not to be worth the extra speed that lower-level code could give.

SL/1 is a LaRC developed PASCAL derivative (Ref. 7) which supports most of the features found in the serial version of the PASCAL compiler. Compound control constructs, explicit data typing, modularity and variable scope encourage the programmer to use structured techniques during the design phase. Use of such constructs as the DO UNTIL and IF-THEN-ELSE simplifies the code. The free format feature of SL/1 reduces the time spent keying in code and a source text reformatting utility which indents code increases the readability of the program. Halfword arithmetic, accessable via the SHORT REAL data type, provides a speedup of 2 for addition and of 4 for multiplication. The SL/1 library contains all necessary intrinsic functions (sine, cosine etc.) in halfword form.

The code produced by the SL/1 compiler takes less core and time to run. This is exhibited by the Voight profile approximation benchmarks (Table 1). The actual design and checkout time spent on the SL/1 code was less than would have been spent on comparable FORTRAN code because of the features of the SL/1 compiler.

SL/1 is actually a cross-compiler resident on a CYBER 173. This allows the user to enter code on a machine with sophisticated text editing capabilities and to compile modules interactively. In contrast, the FORTRAN user must pass source code over a data link to the STAR-100 for compilation. Use of the SL/1 compiler also reduces compilation time spent in correcting syntax errors.

During actual validation runs, the SL/1 CHECK option was turned on to generate extra code at compile time to test such error conditions as out-of-range vector and array subscripts and invalid arguments at run time. STAR FORTRAN has no comparable facility. In the case of an abnormal termination, SL/1 generates a symbolic dump and as many levels of trace back as are necessary. This feature significantly reduces the time and effort spent in the detection and correction of runtime errors.

In general, the SL/1 version of STARSMART is easier to design and implement because of the higher level of software support offered by the SL/1 compiler. The machine code generated by the SL/1 compiler runs faster and occupies less core. All of these benefits were available without losing compatibility with STAR FORTRAN. In fact, certain I/O routines which had to be written in FORTRAN during earlier releases of the SL/1 compiler were called by the SL/1 main module. As the SL/1 compiler was upgraded, these FORTRAN routines were replaced by SL/1 procedures. The current version of STARSMART is coded exclusively in SL/1.

SECTION 4 - VECTORIZATION

The computational power of the STAR-100 can only be realized with well structured vectorized code. Maximum efficiency is achieved when the code which performs CPU intensive work is treated as a series of operations on optimal length vectors. Vectors should be long enough to override the penalties of vector operation start-up times and short enough to prevent excessive disk transfers during execution. These transfers - called page faults - occur when data or code must be brought in to central memory to continue program execution.

The following assumptions were made to simplify the task of vectorization.

(1) Uniform increments between integration points would suffice because virtual memory allows large enough vectors to satisfy resolution requirements.

(2) Absorption coefficients would be kept separated with respect to gas as well as layer. Virtual memory would eliminate the questions of trade-off between storage and added flexibility.

(3) Only atmospheric layers would be considered in the model, instead of the combination of atmospheric instrument and calibration layers allowed by the serial code.

Code in the first phase of SMART containing explicit looping was studied for vectorization potential. Not all looped code lends itself to efficient use of STAR-100 vector instructions. Characteristics of vectorizable code such as the parallelism of repeated computations and independence of source and destination operands over a large data set were identified in the serial code.

The computational logic of SMART consists of three major loops. The outer major loop over wavenumbers within the band-pass, an implicit DO-UNTIL implemented by a FORTRAN IF test, was translated into an explicit loop on the STAR-100. The uniform increment of integration, DW between user specified, upper and lower bounds of integration U and L, defines the number of integration points INT as

$$INT = (U-L)/DW$$

The approximate length of these vectors for a typical case is 2000. Computations involving vectors of this length enter the regime of diminishing returns on the theoretical trade-off curves of vector-length versus time on the STAR-100. At a point, the cost of paging in segments of long vectors for arithmetic operations and building temporaries begins to degrade the performance of the vector operations. Therefore, these vectors were subdivided into blocks of 250 contiguous points to allow calculation of the absorption coefficients in vector form without excessive page faulting.

Within the major loop over vector blocks of integration points were loops at each point over the gases, layers and surrounding spectral lines. These loops were too small to vectorize efficiently and were left in the STAR-100 code as explicit loops operating on the vector blocks of integration data.

At the core of these loops were the calculations of the Voigt profiles of spectral lines and Planck radiation functions. The high frequency of these calculations made it imperative to generate a highly efficient vectorized implementation of these functions. For example, the Voigt profile value is calculated at each integration point for each layer at each spectral line. The midlatitude summer model (15 layers, 4665 lines, and 14000 points) (Ref.10) requires 979,860,000 calculations.

Several candidates for an approximation to the Voigt profile were tested on the serial machines for accuracy and speed. Two algorithms, that of Drayson (Ref. 6) and that of Pierluissi (Ref. 3) were selected for adaptation to the STAR-100.

Both codes were easily translated into SL/1 as scalar functions. However, each would have to be called multiple times within each block of integration points being processed, and timing studies revealed that the serial code was faster than the STAR-100 scalar code. However, since the penalties of scalar operations on the STAR-100 offset any gains made by faster hardware, a vectorized approach to the calculation was developed. The Drayson algorithm violated the criterion of independence of source and destination operands and was rejected as a candidate for vectorization. The Pierluissi code, which is a logically simpler code, lends itself to vectorization and was rewritten. The vectorized code proved to be 3 times faster than the STAR-100 scalar code and comparable to the speed of the serial code. Moreover, when vector lengths were increased the effects of start-up time were reduced and the STAR-100 code executed in less time than the serial code. This one modification to code used many times in the course of a run resulted in a time reduction of a factor of 2 over the original SMART Voigt profile algorithm.

SECTION 5 - PAGE MANAGEMENT

Early runs made with the initial version of vectorized STARSMART code gave deceptively encouraging results. Cases dealing with bandpasses of less than 90 cm-1 in regions of sparse spectral lines executed on the order of 90 times faster in terms of raw CPU time than when run with the serial code (Table 5). However, the submittal of a full-scale problem in a dense spectral region (the 15 layer midlatitude summer atmosphere) resulted in computational catastrophe. After 15 minutes of elapsed time on the STAR-100,only one 250 member block of integration points had been processed. The program had spent nearly four-fifths of its allocated central processor time swapping portions of data in and out to disk.

This phenomenon - called thrashing - degrades machine performance to the extent that essentially no computational work can be done without a pagefault. Thrashing is usually the result of data clash or lack of program locality. Data clash occurs when the data base is structured or accessed inefficiently. Algorithms which exhibit a high degree of program locality are designed to execute over a small area of code at a time without branching to parts of the program or using parts of a data base which must be paged into memory. STARSMART code was on small pages and was in a compact modular form which minimized the program locality problem. A study of the data transfer in the program was undertaken to determine the cause of the thrashing and to correct the problem.

Three steps were then taken to decrease the amount of computer time spent in paging. The source of the thrashing was traced to the accesses of the precalculated Lorentz and Doppler halfwidth information for each line within the band pass. These values were stored in a 3-dimensional array ordered in the columnwise fashion used by FORTRAN. This ordering was changed to coincide with the rowwise data access scheme used by the SL/1 compiler. This first modification reduced the paging time incurred during the profile setup calculations.

Secondly, necessary parts of the profile information vectors were windowed in, that is brought in to memory, as subvectors stored on small pages for calculations on the current block of integration points. This technique guaranteed that any faults caused by paging current profile information in and out of central memory would be small page faults rather than the more time consuming large page faults.

The third modification required a change in the order of summation over equations governing the calculation of the absorption coefficients in order to more efficiently implement the algorithm on the STAR-100.

The calculation of the absorption coefficients $k_{ijn}(w)$ at integration point w for layer j and gas i is the sum of contributions of all lines n within the interval about w expressed as:

$$k_{ij(w)} = S_{ijn} \beta_{ijn(w)}$$

where S_{ijn} is the temperature corrected line intensity value and β_{ijn} is the line shape function value for line n within the interval about w. (Ref. 2).

In discretized vector form, this summation was originally performed by a set of triple loops:

$$k_{ijn}(w) = \sum_{n=1}^{NUMN} \sum_{j=1}^{NLAY} \overline{S}_{nj}(w)\ \overline{\beta}_{nj}(w) \sum_{i=1}^{NGAS} gc_i$$

where NLAY is the number of layers in the model, NUMN is the number of included lines and NGAS is the number of active gases. The vectors $\overline{S}_{nj}$ and $\overline{\beta}_{nj}$ which contain all of the lines spanning the interval about vector block w for all active gases were fetched unnecessarily deep in the loop nest.

Accesses to the profile information vectors $\overline{S}_{nj}$ and the line shape vector $\overline{\beta}_{nj}$ in their entirety were made NUMN x NLAY times per block. This same result could be obtained by the following set of loops:

$$k_{ijn(w)} = \sum_{j=1}^{NLAY} \overline{S}_{nj}\ \overline{\beta}_{nj} \sum_{n=1}^{NUMN} \sum_{i=1}^{NGAS} gc_i$$

with accesses made to the whole vectors on the order of only NLAY per block.

The order of the two outer loops in the STARSMART code was reversed; this modification reduced the paging of profile subvectors by a factor of NUMN. Since NUMN can be a large number in dense spectral regions, this third modification made the most significant contribution in reducing the thrashing problem.

The previous modifications were incorporated and validated on a small scale by running a case for methane (Table 2) to check for degradation caused by the windowing overhead. The modifications were then tested on a large scale by the submittal of the 15 layer midlatitude summer atmosphere case. The program ran to completion in less elapsed time than the initial algorithm took to calculate values for one block (Table 3).

SECTION 6 - MACHINE PRECISION AND ACCURACY

The advantages of the STAR-100 halfword arithmetic - increased speed and decreased data base size - are diminished to some degree by the loss in precision. To assess the effects of 32-bit arithmetic on the transmission calculations, the characteristics of the data base and the computational procedure were studied in depth.

Despite its 64-bit wordlength, the STAR-100 has a full precision of only 14 significant decimal digits and a half precision of 6 or 7 digits. The CDC 6000 series machines achieve single precision results with 14 or 15 significant digits. Using STAR halfword arithmetic with estimated speedup factors of 2 and 4 for addition and multiplication and the corresponding reduction by 2 of the operand data base could lead to a more cost effective use of the machine. The tradeoffs between accuracy and economy were taken into consideration.

Errors during floating point calculations on a digital machine arise from several sources. The first source is the machine representation of any number having a fractional part not an integer power of the machine base. Such representations will have a maximum error of $\frac{1}{2}\beta^{1-t}$ where β is the base and t the number of bits in the fractional part of the word. This number, the machine epsilon, is directly related to machine precision and on the STAR-100 the halfword value is on the order of 1.0E-28. Another source of error is roundoff committed during the operand normalization and actual machine arithmetic. Errors arising at each step of a calculation can be bounded by the machine epsilon, but these round-off errors will propagate. Calculations for the solution of problems by unstable algorithms can lead to large relative errors, inaccurate solutions, and at worst, loss of significance. Another source of error related to precision is loss of significance caused by use of operands out of the range of machine representation or intrinsic system routines such as sine and cosine. These possible sources of error in the STARSMART code were studied and conditions leading to unacceptable levels of error were corrected.

The error associated with machine representation is well below the threshhold of certainty of the input parameters for STARSMART. The spectral line parameters read in from the McClatchey AFCRL tape (Ref. 9) are not known to full machine significance. The value of line location is known to $\pm$ 0.05 cm-1, while values of ground state energy and halfwidth are known only to $\pm$ 5-10%. The use of half precision did not cause loss of significance during the input of the data base.

The round-off error caused by 32-bit arithmetic used in such functions as the Voigt profile was determined to be acceptable by parallel runs of STAR-100 and 6000 series algorithms.

The exponent range of 32-bit operands was considered because loss of significance can occur during floating point arithmetic. On the STAR-100 a floating point number is represented as c * 2**E where E is a valid signed exponent and c is a valid signed coefficient. Values are kept within the machine range by shifting the binary point and adjusting the exponent. The implicit dependence of the range of E and c upon the precision of the machine can be expressed by the following:

$$x = \pm (a_1 \beta^{-1} + a_2 \beta^{-2} + \ldots + a_t \beta^{-t}) * \beta^E$$

characterized by the number base β, precision t and an exponent range (l,u). The integers a_i (i = 1 to t) lie within O and β-1 and $1 \leq e \leq u$. (Ref. 8) When E exceeds these bounds or c vanishes during a calculation not resulting in a true zero, a loss of significance is said to occur and no digits in the computed result can be trusted.

The STAR-100 halfword exponent and coefficient bounds allow an operand range of $\pm$ 2.1e + 40 to $\pm$ 8.1e - 28. Any numbers outside of this range cannot be represented by the machine! It was discovered that the line strengths for some of the weaker spectral lines processed by STARSMART are on the order of 1.0e - 30. Thus all contributions from these lines were truncated to zero and the calculated absorption coefficients falsely indicated no absorbers in the neighborhood of weaker lines.

The transmission τ at wavenumber w and layer l is defined:

$$\tau_{w,l} = e^{(-OP_l . k_i(w) . gc_i)} \qquad \text{(Ref. 2)}$$

where OP_l is the optical path in molecules/cm2, ki(w) the absorption for gas i and gc_i the fractional gas concentration. The values of the absorption coefficients were on the order of 1.0e - 30 and the values of the optical path about 1.0 e + 24. To preserve significance the optical paths were postmultiplied by a value of 1.0e - 24 and the line strengths were premultiplied by a factor of 1.0e + 24.

These operations algebraically cancel in the preceding equation and keep the operands within the range of 32-bit operations. Comparisons between NOS and STAR-100 absorption coefficients and transmissions revealed a relative error at specific wave numbers of 0.30% for the improved algorithm versus that of 7.78% for the original problem. (Table 4)

Another possible source of error in the STARSMART calculations considered was that of invalid arguments supplied to intrinsic functions. Such functions as natural log (LN) and exponent (EXP) have narrower ranges and the use of out-of-range operands will cause unpredictable and unflagged results. Code to perform validity checking of operands and to issue diagnostics was implemented in the revised program. No current runs have caused any error diagnostics to be issued.

On the basis of the above findings, the halfword precision was kept in the STARSMART program. The only operations still performed in full precision are the I/O transfers which must be handled by the 64-bit operating system routines.

SECTION 7 - I/O HANDLING

The STAR-100 is a powerful computing machine, but its input/output handling capabilities are quite crude in comparison to those of the general purpose user oriented NOS serial machines. At times it is more expensive to store results because of costly I/O overhead than it is to recompute them. However, graphical display was one of the required products of the instrument simulation package and there is no plotting capability available on the STAR-100. Thus absorption coefficients would have to be stored for post-processing. For realistic atmospheres, the number of coefficients approaches one million and the storage, manipulation and conversion of those values have a large impact on the overall efficiency of any STAR-100 program.

The STAR-100 supports two types of I/O operations: explicit and implicit. Explicit I/O is the more familiar data transfer which is activated by READ and WRITE statements. Implicit I/O is data transfer which occurs without specific commands. Examples are the paging caused by hardware translation to get data or code during execution and the mapping of files.

Explicit I/O can be performed in two modes: formatted (coded) and unformatted (binary). Coded data transfer uses more space to represent numbers and requires conversion from the machine floating point representation to external ASCII. Binary data takes less time to process and has a higher density; however, an extra step is required to pass binary from STAR to the CYBER machines.

STARSMART initially had no I/O capability except that of listing results on the printer. During the first phase, the absorption coefficients were written out to a temporary disk file by binary WRITE commands and then converted to CDC 60-bit floating point and passed over the data channel to the NOS machines for graphical post-processing. For small cases this worked quite well; but for the large cases the operating system aborted the job because of output file overflow. The overhead caused by explicit I/O operations forced the system to extend file lengths to the maximum. The impossibility of reducing the size of the output file led to the development of an implicit I/O strategy.

Use of implicit I/O requires that the user become very familiar with the workings of the operating system and understand his file needs. The user must be responsible not only for determining the file size, but also for organizing the data and selecting the blocksize best suited to the problem. Implicit I/O avoids the file overhead caused by multiple entries of data and extra transfers through buffers which occur during explicit I/O. STARSMART was modified to invoke SL/1 system function statements which map I/O files to specific contiguous disk locations reserved by loader options. The READ and WRITE statements were replaced with references to locations within the data area which is mapped to a user file.

Several data structures to represent the absorption coefficients generated by a problem with NLAY layers, NGAS gases and NWAVE integration points were tested. The most obvious structure, that of a 3-dimensional array of dimension NLAY x NGAS x NWAVE indexed by layer number, gas number and wavenumber was not the most efficient because the references to the structure were not over optimal length vectors. The final structure used was a 3-dimensional array of dimension (NWAVE/blocksize) x (NGAS x NLAY) x blocksize which was selected after small scale tests were run. (Table 2) Since this representation referenced the optimal length blocksize vectors through the gases and layers in a rowwise order, the amount of pagefaulting was reduced.

Although the mapped file had to be processed by an SL/1 routine which placed it in binary format for system conversion; the cost of post-processing the mapped file was small in comparison to the overhead used to generate the large binary file during calculation.

SECTION 8 - RESULTS

Optimization of STARSMART code has reduced the time and cost of runs to calculate absorption coefficients for large atmospheric models. For example, a 15-layer atmospheric model spanning the 140 cm-1 bandpass of the Pressure Modulated Radiometer now executes 56 times faster than when processed by the original STAR-100 code. Indications of better utilization of the powerful STAR-100 central processing unit can be seen in the system activity summaries at the end of user printouts. The number of large and small page faults and the percent of CPU time spent in paging have been reduced significantly.

Elapsed clock time and turnaround time also were reduced. Results which formerly required 2 to 3 days to process now return to the user in less than a day.

From a software design viewpoint, the STARSMART code is a considerable improvement over the initial codes. STARSMART takes advantage of the features offered by the SL/1 compiler to present clear formatted source in a well-structured modular form. The "ruggedized" code prevents executions with mismatched parameters and issues diagnostics to flag potential problems.

SECTION 9 - CONCLUSIONS

Proper utilization of the STAR-100 has resulted in a generalized program to calculate and store absorption coefficients which executes much faster than the original STAR-100 code and faster still than the comparable CDC 6000 series code. The steps taken during the adaption and optimization of the code have been described.

Although virtual memory remedied the problems of core limitation experienced on the serial machines, it was not a panacea. A straightforward translation of an efficient serial code often performs poorly on the vector machine because the designer has neglected to consider the side-effects of virtual memory on vectorization and data structure.

In the case of STARSMART, operations taken for granted on a serial machine, such as I/O and array accessing, caused an inordinate amount of paging on the STAR-100. To overcome these adverse effects a study on the impact of virtual memory had to be undertaken.

Not all problems are suited to the use of halfword arithmetic. Thus, the equations describing the model, as well as the input parameters, were studied carefully during the evaluation of half-word arithmetic. Fortunately the STARSMART code could use half-word arithmetic and take advantage of the resulting speedups in operations and reduction in data base size.

The selection of SL/1 resulted in faster debugging and check-out, as well as faster execution time than could have been provided by the other available languages. SL/1 provides the control structures necessary for clear quality code. STARSMART is easier to read and modify than the original codes; this additional clarity makes the program a better tool for other researchers in this area of atmospheric work to use.

The STAR-100 code shows an improvement in time and cost over the serial code when used for larger problems. The greatest advantages are gained in regions of dense spectral lines. It is these cases which are not even possible to run on the serial machines. Thus STARSMART has expanded the range of user definable models which can be realized on the available machines.

REFERENCES

1. Lambiotte, Jules J., Jr.: Effect of Virtual Memory on Efficient Solution of Two Model Problems. NASA TMX-3512, July 1977.

2. Casas, Joseph C. and Campbell, Shirley A.: A Modular Radiative Transfer Program for Gas Filter Correlation Radiometry. NASA CR-2895, October 1977.

3. Pierluissi, Vanderwood and Gomez: Fast Calculational Algorithm for the Voigt Profile. J.Q.S.R.T.Vol. 18, ppg. 555-558, 1977.

4. Twitty, Jerold T.: Comparison of Various Computational Methods for the Calculation of the Voigt Profile Function. Old Dominion Research Foundation Technical Report PGSTR-AP78-7, May 1978.

5. Campbell, S. and Casas, J.: A Line-by-line Radiative Transfer Program for the STAR-100. STAR-100 Workshop, Langley Research Center, October 3-4, 1978.

6. Drayson, S. R.: Rapid Computation of the Voigt Profile. J.Q.S.R.T. Vol. 16, ppg. 611-614 1976.

7. Knight, J. C., et. al.: "SL/1 Manual". Analysis and Computation Division, Programming Techniques Branch, June 1979.

8. Forsythe, G. E., Malcolm, M. A. and Moler, C. M.: Computer Methods for Mathematical Computations. Chpt. 2. Prentice-Hall, Inc. 1977.

9. McClatchey, R. A. et. al.: AFCRL Atmospheric Absorption Line Parameters Compilation. AFCRLTR-73-0096, January 1973.

10. Park, J. H.: Atlas of Infrared Absorption Lines. NASA CR-2925, November 1977.

TABLE 1

EFFECTS OF PRECISION AND
COMPILER OPTIMIZATION
UPON COMPUTATIONAL SPEED AND STORAGE
OF SELECTED APPROXIMATIONS OF
THE VOIGT PROFILE

ALGORITHM	TIMING (CPU SECONDS)	STORAGE (WORDS) CODE	DATABASE
Drayson(1)	0.1669	--	--
Pierluissi(1)	0.1568	--	--
Pierluissi 64-Bit Math	0.0560	258	1325
W/Check Option(2)	0.0683	726	1325
Pierluissi 32-Bit Math	0.0558	251	843
W/Check Option(2)	0.0729	762	343
Pierluissi 32-Bit Math Generalized Regions	0.0568	254	843

(1) 101 multiple calls to scalar routines as compared to succeeding cases performed as single calls to process vectors of length $|v| = 101$.

NOS scalar timing for both methods was 0.05 seconds

(2) Extra runtime code generated by the compiler for diagnostic purposes caused timing and storage overhead.

TABLE 2

EFFECTS OF PAGING AND I/O MANAGEMENT
UPON A SIMPLE ATMOSPHERIC MODEL

METHANE CASE

GASES = 1 LAYERS = 1 SPECTRAL LINES = 740
MAXIMUM DENSITY = 84 4200 - 4490 CM-1 BANDPASS
INTEGRATION POINTS = 7250

	CRU'S	PAGE FAULTS LARGE	PAGE FAULTS SAMLL	% OF CRU'S SPENT PAGING
ORIGINAL ALGORITHM (3-D ARRAY EXPLICIT I/O)	26.04	4	22	4.46
3-D ARRAY IMPLICIT I/O LARGE PAGES	49.50	291	116	56.5
3-D ARRAY(1) IMPLICIT I/O SMALL PAGES	13.45	4	911	14
2-D ARRAY(2) IMPLICIT I/O SMALL PAGES	12.95	6	117	11

(1) Selected Data Structure

(2) This data structure was not selected because the number of absorption coefficients calculated by large models generally exceeds the maximum vector length of 65,534 enforced by SL/1.

TABLE 3

RESULTS OF OPTIMIZATION OF
STARSMART CODE

CASE I. MIDLATITUDE SUMMER ATMOSPHERE (SMALL)

GASES = 4   LAYER = 15   SPECTRAL LINES = 211

INTEGRATION POINTS = 2500   BANDPASS 2000-2025 cm^{-1}

INCREMENT = 0.01 cm^{-1}

	CRU'S[1]	PAGE FAULTS		%CRU'S
		LARGE	SMALL	PAGING
ORIGINAL ALGORITHM	15.774	6	820	10.7
IMPROVED ALGORITHM	14.412	3	388	5.4

CASE II. MIDLATITUDE SUMMER ATMOSPHERE (LARGE)

GASES = 4   LAYERS = 15   SPECTRAL LINES = 4666

INTEGRATION POINTS = 14000   BANDPASS 2075-2215 cm^{-1}

MAXIMUM DENSITY OF SPECTRAL LINES = 645   INCREMENT = 0.01 cm^{-1}

	CRU'S	PAGE FAULTS		%CRU'S
		LARGE	SMALL	PAGING
ORIGINAL[2] ALGORITHM	6751	42840	39984	81.8
IMPROVED ALGORITHM	151	698	3356	61.8
FINE TUNED[3] ALGORITHM	105	350	2540	30.9

(1) CRU is the standard Computer Resource Unit used for accounting purposes.

(2) This algorithm only processed 1/56th of the desired bandpass. The extrapolated cost is 6751 CRU's.

(3) The array dimensions were set very close to the minimums required for the bandpass.

TABLE 4

PERCENT RELATIVE ERROR FOR TRANSMISSIONS AT INTEGRATION POINTS

LAYER	w = 2077: $\left\lvert\frac{\tau_\ell^{*(1)}-\tau_\ell}{\tau_\ell}\right\rvert \times 100$	w = 2077: $\left\lvert\frac{\tau_\ell^{*(2)}-\tau_\ell}{\tau_\ell}\right\rvert \times 100$	w = 2075: $\left\lvert\frac{\tau_\ell^{*(1)}-\tau_\ell}{\tau_\ell}\right\rvert \times 100$	w = 2075: $\left\lvert\frac{\tau_\ell^{*(2)}-\tau_\ell}{\tau_\ell}\right\rvert \times 100$
1	1.07	0	0.69	0.50
2	1.87	0	1.01	0.50
3	2.6	0	1.50	0.70
4	3.03	0	2.00	0.80
5	3.88	0.15	2.52	1.06
6	3.96	0.25	2.87	1.07
7	4.5	0.41	3.07	1.24
8	5.0	0.69	3.24	1.24
9	5.89	0.12	3.41	1.25
10	6.32	0.20	3.58	1.25
11	6.90	0.34	3.72	1.25
12	7.08	0.23	3.91	1.25
13	7.36	0.27	3.91	1.25
14	7.55	0.37	4.07	1.09
15	7.78	0.30	4.07	1.09

(1) τ_ℓ^* is total transmission at top of layer ℓ for original algorithm.

(2) τ_ℓ^* is total transmission at top of layer ℓ for improved algorithm.

τ_ℓ is total transmission at top of layer ℓ for serial algorithm.

TABLE 5

COMPARISON OF STAR-100 AND
CDC 6000 SERIES EXECUTION
OF RADIATIVE TRANSFER CODE

CASE I. 15 LAYER MIDLATITUDE SUMMER ATMOSPHERE
4 GASES 2075-2175 CM-1 (Ref. 5)

	CDC	STAR-100
CPU SECS.	2898.8	29.2
COST FACTOR	(CDC/STAR)	7
CPU FACTOR	(CDC/STAR)	99

CASE II.[1] 6 LAYER N_2O CASE 2075-2215 CM-1

CPU SECS.	5122	92.6
CLOCK TIME	3 HOURS	1.5 MINUTES
TURNAROUND TIME	2 DAYS	OVERNIGHT
COST FACTOR	(CDC/STAR)	1.69
CPU FACTOR	(CDC/STAR)	55

(1) The cost and CPU factors for CASE II are less because the case deals with a region of denser spectral lines and the overhead of storing the calculated absorption coefficients.

1. Report No. NASA CR-159238	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Adaptation and Optimization of a Line-by-Line Radiative Transfer Program for the STAR-100 (STARSMART)		5. Report Date March 1980
		6. Performing Organization Code
7. Author(s) Pamela Livingston Rarig		8. Performing Organization Report No. R-SAL-12/79-02
9. Performing Organization Name and Address Systems and Applied Sciences Corp. 6811 Kenilworth Ave., Suite 500 Riverdale, MD 20840		10. Work Unit No.
		11. Contract or Grant No. NAS1-15481
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546		13. Type of Report and Period Covered Contractor Report
		14. Sponsoring Agency Code
15. Supplementary Notes Langley Technical Monitor: Dr. H. D. Orr, III		

16. Abstract SMART, a program to calculate upwelling infrared radiation, has been modified to operate efficiently on the STAR-100 at NASA Langley Research Center. The modified software has processed specific test cases, both large and small, significantly faster than the initial STAR-100 code. For example, a midlatitude summer atmospheric model executed in less than 2% of the time originally required on the STAR-100. Furthermore, the optimized program performed extra operations to save the calculated absorption coefficients which the earlier program did not attempt. Cost and time reductions of 2 and 55 respectively over the comparable serial version of SMART residing on the CDC 6000 series computers have been achieved during runs of average atmospheric models. Some of the blessings and pitfalls of virtual memory and vector processing and the strategies used to avoid loss of accuracy and computing power are discussed in this paper. Results from the vectorized code, in terms of speed, cost and relative error with respect to serial code solutions are encouraging. Tables of benchmarks performed during phases of the optimization are presented.

17. Key Words (Selected by Author(s)) Radiative Transfer, Atmospheric Models, STAR-100 Computer Program, Line-by-Line Calculations, Virtual Memory		18. Distribution Statement Unclassified - Unlimited	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 33	22. Price* $4.50