NASA CONTRACTOR
REPORT

NASA CR-179389

AN IMPROVEMENT IN THE NUMERICAL INTEGRATION PROCEDURE USED IN THE NASA MARSHALL ENGINEERING THERMOSPHERE MODEL

By M. P. Hickey
Universities Space Research Association
4950 Corporate Drive, Suite 100
Huntsville, Alabama 35806

August 1988

Interim Report

(NASA-CR-179389) AN IMPROVEMENT IN THE NUMERICAL INTEGRATION PROCEDURE USED IN THE NASA MARSHALL ENGINEERING THERMOSPHERE MODEL Interim Report (Universities Space Research Association) 53 p CSCL 04A G3/46

N88-30200

Unclas
0166298

Prepared for
NASA, George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

TECHNICAL REPORT STANDARD TITLE PAGE

1. REPORT NO. NASA CR-179389	2. GOVERNMENT ACCESSION NO.	3. RECIPIENT'S CATALOG NO.
4. TITLE AND SUBTITLE An Improvement In the Numerical Integration Procedure Used In the NASA Marshall Engineering Thermosphere Model		5. REPORT DATE August 1988
		6. PERFORMING ORGANIZATION CODE
7. AUTHOR(S) Michael Philip Hickey		8. PERFORMING ORGANIZATION REPORT #
9. PERFORMING ORGANIZATION NAME AND ADDRESS Universities Space Research Association 4950 Corporate Drive, Suite 100 Huntsville, Alabama 35806		10. WORK UNIT NO.
		11. CONTRACT OR GRANT NO. NAS8-36639
12. SPONSORING AGENCY NAME AND ADDRESS National Aeronautics and Space Administration Washington, DC 20546		13. TYPE OF REPORT & PERIOD COVERED Contractor Report
		14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

Technical Monitor: Claude E. Green, Structures and Dynamics Lab, Science and Engineering Directorate, George C. Marshall Space Flight Center, MSFC, Alabama

16. ABSTRACT

This document contains a detailed description of a proposed replacement integration scheme for the integration of the barometric and diffusion equations in the NASA Marshall Engineering Thermosphere (MET) Model. This proposed integration scheme is based upon Gaussian Quadrature. Extensive numerical testing reveals it to be numerically faster, more accurate and more reliable than the present integration scheme (a modified form of Simpson's Rule) used in the MET model. Numerous graphical examples are provided, along with a listing of a modified form of the MET model in which subroutine INTEGRATE (using Simpson's Rule) is replaced by subroutine GAUSS (which used Gaussian Quadrature). It is recommended that the Gaussian Quadrature integration scheme, as used here, be used in the MET model.

17. KEY WORDS MET Model, Atmosphere, Density, Integration, Gaussian Quadrature, Thermosphere	18. DISTRIBUTION STATEMENT Unclassified/Unlimited

19. SECURITY CLASSIF. (of this report)	20. SECURITY CLASSIF. (of this page)	21. NO. OF PAGES	22. PRICE
Unclassified	Unclassified	52	NTIS

PREFACE

This work was performed under NASA Contract NAS8-36639 under the monitorship of Messrs. Claude E. Green and C. Kelly Hill. The author acknowledges the proof-reading of this document by Mr. Dale L. Johnson and Dr. R. E. Smith, and would also like to especially thank Mrs. Frances B. Abel for her dexterity and patience with her typing of the manuscript. Qualified requestors may obtain copies of the computer program for this modified version of the NASA Marshall Engineering Thermosphere model upon request to the Chief, Earth Science and Applications Division, Structures and Dynamics Laboratory, NASA, Marshall Space Flight Center, Alabama 35812.

TABLE OF CONTENTS

CONTRACTOR REPORT

AN IMPROVEMENT IN THE NUMERICAL INTEGRATION PROCEDURE USED IN THE NASA MARSHALL ENGINEERING THERMOSPHERE MODEL

I. INTRODUCTION

A. A PROBLEM WITH THE MET MODEL

The models which have been, and still are, used to describe the properties of the neutral atmosphere between 90 and 2500km altitude at NASA/Marshall Space Flight Center have all been based upon Jacchia's empirical models [1,2]. The former model was termed the MSFC/J70 model [3], and in the Earth Science and Applications Division of the Structures and Dynamics Laboratory the computer program used to output data from this model was the J70MM. Recently the computer code of the J70MM has been extensively modified, as described in [4], and the resultant program has been termed the NASA Marshall Engineering Thermosphere Model or the MET model.

An intermittent problem in the density output of the MSFC/J70 model was found to lie in the integration routine which the model employed [4], and this problem was apparently corrected in the MET model. The MSFC/J70 model and the MET model both use Simpson's Rule to numerically integrate the barometric and diffusion equations. Although the implementation of the integration scheme employed in the models is numerically fast, it has not always been found to be reliable in the MSFC/J70 model. This unreliability was discussed in detail in [4], where it was shown that the convergence criterion employed was not always stringent enough. In an effort to improve reliability, the MET model contained a modification in the integration scheme which ensured that above a certain altitude a predetermined minimum number of iterations were performed before the convergence criterion was employed. This modification did improve the reliability of the integration method, but further examination has now revealed it to be not as reliable as originally thought. Therefore a new integration method was sought with the requirements that it be at least as numerically fast and accurate as the Simpson's Rule employed in the MET model, but that it should be totally reliable.

One method examined, which is known as Gaussian Quadrature, was found to be more than adequate for this purpose. The rest of this report is devoted to describing the implementation of the Gaussian Quadrature in the MET model as a replacement for the Simpson's integration. Comparisons between the two different methods will be made, differences will be discussed and finally recommendations will be made.

B. THE INTEGRAND

The problem of deciding which numerical method is best to use will often depend on the behavior of the integrand one is wishing to integrate. Therefore, before discussing the integration procedure to be adopted, a short description of the behavior of the integrand is in order.

Between 90 and 105km altitude the density is computed by integrating the barometric equation while above 105km altitude it is computed by integrating the diffusion equation, as described in Jacchia [1]. The barometric equation, valid between 90 and 105km altitude is:

$$\rho(z) = \rho(90)\,\frac{\bar{M}(z)}{T(z)} \cdot \exp\left\{\frac{-1}{k}\int_{90}^{z} \frac{\bar{M}g}{T}\,dz\right\} \qquad (1)$$

while the diffusion equation, valid between 105 and 2500km altitude is:

$$n_i(z) = n_i(105) \cdot \left(\frac{T(105)}{T(z)}\right)^{1+\alpha_i} \cdot \exp\left\{\frac{-M_i}{k}\int_{105}^{z} \frac{g}{T}\,dz\right\} \cdot \qquad (2)$$

In Equation (1), ρ is the mass density, T is the temperature, $\bar{M}$ is the mean molecular weight, g is the gravitational acceleration and k is the universal gas constant. In Equation (2), n_i and M_i are respectively the number density and molecular weight of each individual atmospheric specie (N_2, O_2, O, A, He and H). Note that in the case of hydrogen Equation (2) is applied only above 500km altitude; below this altitude $n_H = 10^6\ m^{-3}$ (ie. hydrogen density is constant).

One can see that the integrand in Equation (1) is $g\bar{M}/T$ while in Equation (2) it is g/T. The empirical equations for g, T and for $\bar{M}$ below 105km altitude, can be found in Jacchia [1]. The only adjustable parameters in these equations are the exospheric temperature, T_x, and the altitude, z, and thus the integrand is only a function of T_x and z. In order to examine the behavior of the integrand two values of T_x were chosen which represent, though very approximately, extremes. The chosen values were 500K and 2500K.

The value of $g\bar{M}/T$ is displayed in figure 1(a) for altitudes between 90 and 105 km. The greatest variation of $g\bar{M}/T$ over this altitude range occurs for T_x equal to 2500K, but one can see that the variation is quite small, being typically less than 25%. Because the altitude variation of is both small and continuous, the behavior of the integrand should place no restrictions on the integration method employed to integrate the barometric equation.

The value of g/T is displayed in Figure 1(b) for altitudes between 105 and 2500km. Although g/T is a continuous function of altitude, its rapid variation at low altitudes may tend to make its numerical integration either inefficient or inaccurate. If the entire altitude region is separated into a number of sub-regions, in such a way that g/T is slowly varying in each of these sub-regions, difficulties with the numerical integration of g/T will be circumvented. An expanded representation of this region is shown in Figures 2(a) and 2(b). By a judicious choice of "sub-regioning," the altitude variation of g/T will place no restrictions on the method employed to integrate the diffusion equation. In what follows this is precisely the adopted procedure, but it is noted in passing that this procedure has not been adopted in the MET model nor the MSFC/J70 model and probably accounts for some of the problems that these two models have experienced when integrating the diffusion equation.

II. THE GAUSSIAN QUADRATURE

A. MATHEMATICAL DESCRIPTION

A concise and useful introduction to Gaussian Quadrature is given in [5] and unless explicitly required it is not repeated here.

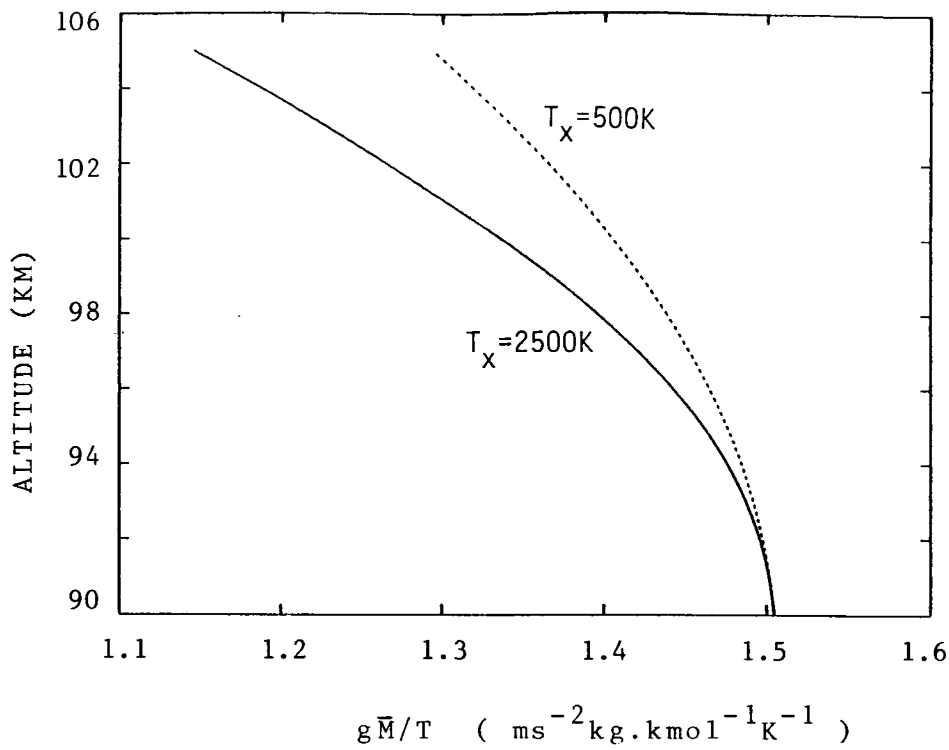


Figure 1(a) The integrand $g\bar{M}/T$

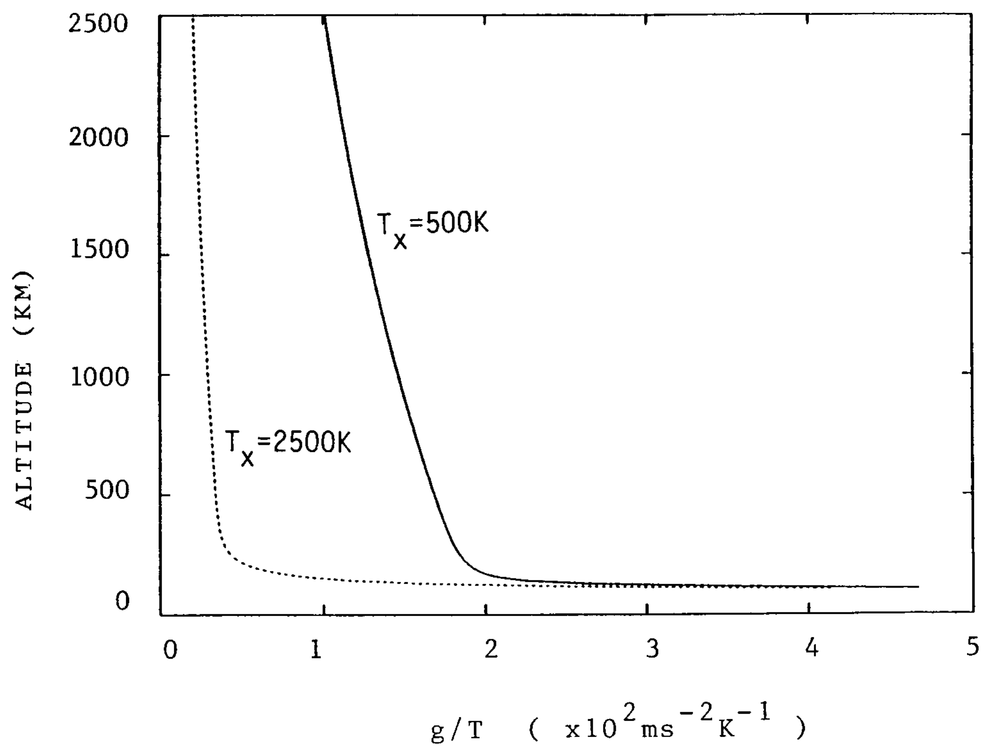


Figure 1(b) The integrand g/T

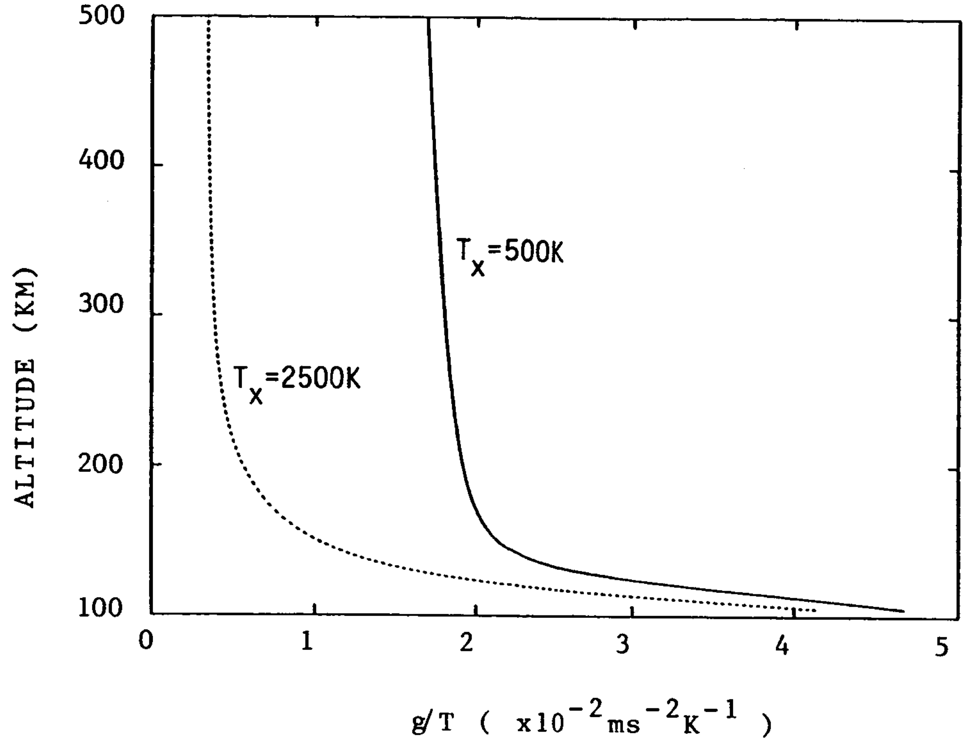


Figure 2(a) The integrand g/T below 500km altitude

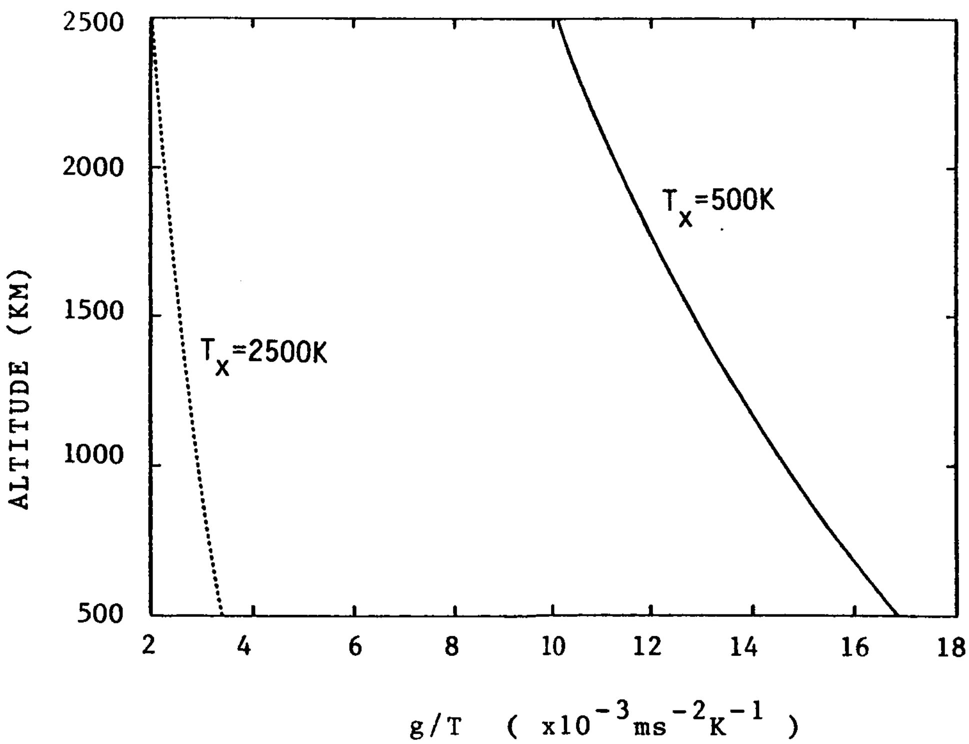


Figure 2(b) The integrand g/T above 500km altitude

Gaussian Quadrature can only be applied if integration is performed over the interval from -1 to +1, so that generally a change of variable is required if this condition is to be met. In the problem at hand integration will be between two altitudes, say z_1 and z_2 (with $z_1<z_2$), so that the following transformation will be required:

$$z = (z_2-z_1)\,\frac{(x+1)}{2} + z_1 \tag{3}$$

Thus, when $x=-1$, $z=z_1$, and when $x=+1$, $z=z_2$. Also,

$$dz = \frac{1}{2}\,(z_2-z_1)dx \tag{4}$$

Thus, the altitude variable has been changed from z to x. In Gaussian Quadrature the transformed integral is approximated by the sum of the products of the integrand (evaluated at a certain number of discrete points) and certain coefficients (associated with each of these points). Mathematically this is stated as

$$\int_{-1}^{+1} f(x)\ dx \approx \sum_{i=1}^{N} C_i f(x_i) \tag{5}$$

for some arbitrary function f. Thus, using (4) and (5),

$$\int_{z_1}^{z_2} f(z)\ dz \approx \frac{1}{2}\,(z_2-z_1) \sum_{i=1}^{N} C_i\ f(x_i) \tag{6}$$

where the integrand $f(z)$ is evaluated at certain discrete values of x which correspond to certain values of z in Equation (3). The abscissas, x_i, and the coefficients (or weight factors), C_i, for up to eight points (N=8) are given in Table 1. It is worth noting that an N-point Gaussian integration can evaluate exactly the integral of a 2N-1 degree polynomial.

Table 1. GAUSS QUADRATURE COEFFICIENTS AND ABSCISSAS

Abscissas = $\pm x_i$ Weight Factors = C_i

N	$\pm x_i$	C_i
N=2	.57735027	1.00000000
N=3	.00000000	0.88888889
	.77459667	0.55555556
N=4	.33998104	0.65214515
	.86113631	0.34785485
N=5	.00000000	0.56888889
	.53846931	0.47862867
	.90617985	0.23692689
N=6	.23861919	0.46791393
	.66120939	0.36076157
	.93246951	0.17132449
N=7	.00000000	0.41795918
	.40584515	0.38183005
	.74153119	0.27970539
	.94910791	0.12948497
N=8	.18343464	0.36268378
	.52553241	0.31370665
	.79666648	0.22238103
	.96028986	0.10122854

B. IMPLEMENTATION IN THE MET MODEL

The integral of $g\bar{M}/T$ between 90 and 105km was evaluated very accurately using only a 4-point Gaussian integration scheme. This is not too surprising if one takes a quick look at Figure 1(a), and one also remembers that this scheme will evaluate exactly the integral of a seventh-degree polynomial. The mean molecular weight, $\bar{M}$, is represented by a seventh-degree polynomial in altitude while in this small altitude range the gravitational acceleration g, a quadratic in altitude, is very slowly varying. Temperature T, is represented by a fourth-degree polynomial in altitude in this altitude range.

As discussed earlier, the large variations in g/T over the altitude range of 105 to 2500km will present a problem for most integration methods unless the whole region is separated into a number of sub-regions. It was not surprising to find, therefore, that even an eight-point Gaussian integration (which should exactly evaluate the integral of a 15-degree polynomial) could not accurately represent this integral over the full altitude range, and thus a number of sub-regions were constructed.

Within each sub-region the integral of g/T was evaluated by an N-point Gaussian integration scheme, where the number of points used in the integration, N, was determined by experimentation for each particular sub-region. After much experimentation the whole region extending from 105 to 2500km was separated into seven sub-regions. The altitude extent of each of these regions as well as the number of points, N, used in the Gaussian integration for each region is given in Table 2. For completeness, the region below 105km is also included.

Table 2. **DEFINITION OF SUB-REGIONS**

Altitude Range (km)	90-150	105-125	125-160	160-200	200-300	300-500	500-1500	1500-2500
No. of points, N, in Gaussian integration	4	5	6	6	6	6	6	6

The incorporation of the Gaussian Quadrature into the MET model was affected primarily by replacing subroutine INTEGRATE with a new subroutine named GAUSS. This new subroutine handles the setting up of the sub-intervals (ie. the altitude boundaries and the number of points required for the Gaussian Quadrature for each sub-interval), and then performs the Gaussian Quadrature in only seventeen lines of executable code! Subroutine GAUSS can be found in the Appendix along with the associated FORTRAN coding of the MET model.

III. RESULTS

In this section the integral and total mass densities are evaluated by the standard integration method employed in the MET model (using Simpson's Rule) and by the new method (Gaussian Quadrature) so that the two different integration methods can be compared. For these two methods, the accuracy and reliability of the results and the computational speeds will be compared. To ensure that the comparison is a reliable one, it would be advantageous to know the exact value of the desired quantity (the value of an integral or the total mass density). Since this is not known, a "reference" version of the MET model was written in double-precision. This reference model employed Simpson's Rule in the same way that the MET model does, but, with three exceptions. Firstly, the convergence parameter (ϵ) is set equal to 10^{-4} in the MET model, but in the double-precision reference model, it is set to 10^{-7} (it could have been set even smaller than this, but the model would then have consumed a prohibitive amount of CPU time). Secondly, in the reference model, once convergence has been achieved, one more iteration is performed and the convergence condition is tested a second time. Two consecutive positive convergence tests constitute convergence in this model. Lastly, whereas the maximum number of sub-intervals used in the Simpson's integration in the MET model is 2^{10}, the maximum number that can be used in the reference model is 2^{15}. These changes ensure that the results obtained from the reference model will be both accurate and reliable, but it should be remembered that this is achieved at the expense of CPU time.

To present results for this report, two different sets of input parameters were used in the models. The first set of input parameters, which lead to a cold atmosphere with an exospheric temperature of 597.361K, is as follows:

Date & Time:	1987, June 21, 0400 hr UT		
$F_{10.7}$ & $\overline{F}_{10.7}$:	70	A_p index:	0
Latitude:	0	Longitude:	0

The second set of input parameters lead to a hot atmosphere with an exospheric temperature of 2554.217K and is given as:

Date & Time:	1987, October 27, 1400 hr UT				
$F_{10.7}$:	400	$\overline{F}_{10.7}$:	250	A_p index:	400
Latitude:	0	Longitude:	0		

Although the exospheric temperatures for the cold and the hot atmospheres are not the same as those which were used to represent the extreme conditions shown in Figures 1 and 2, the general behavior of the integrands will not be significantly different so that the previous discussion of the integrands will still apply. A good description of these input parameters can be found in [3], and will not be repeated here.

All of the results presented here (excepting those in subsection E) are represented graphically in figures 3 through to 10. In order to maximize the sensitivity of the comparisons, a large number of data points were used in all of the figures. In the 90-105km section nearly 100 data points were generated while in the 105-2500km section some 800 data points were generated.

A. COMPARISON OF THE INTEGRALS: THE 90 TO 105KM SECTION

In this altitude range the integral of $g\overline{M}/T$ is evaluated by the three methods already discussed. The value of this integral calculated using the standard Simpson's Rule as used in the MET model is designated RS, and that calculated using the reference model (double-precision, high accuracy Simpson's Rule) is designated RD, while that calculated using Gaussian Quadrature is designated RG. This terminology will be used throughout this report.

In order to compare RS and RG, the percentage deviations of each of these from RD are calculated as:

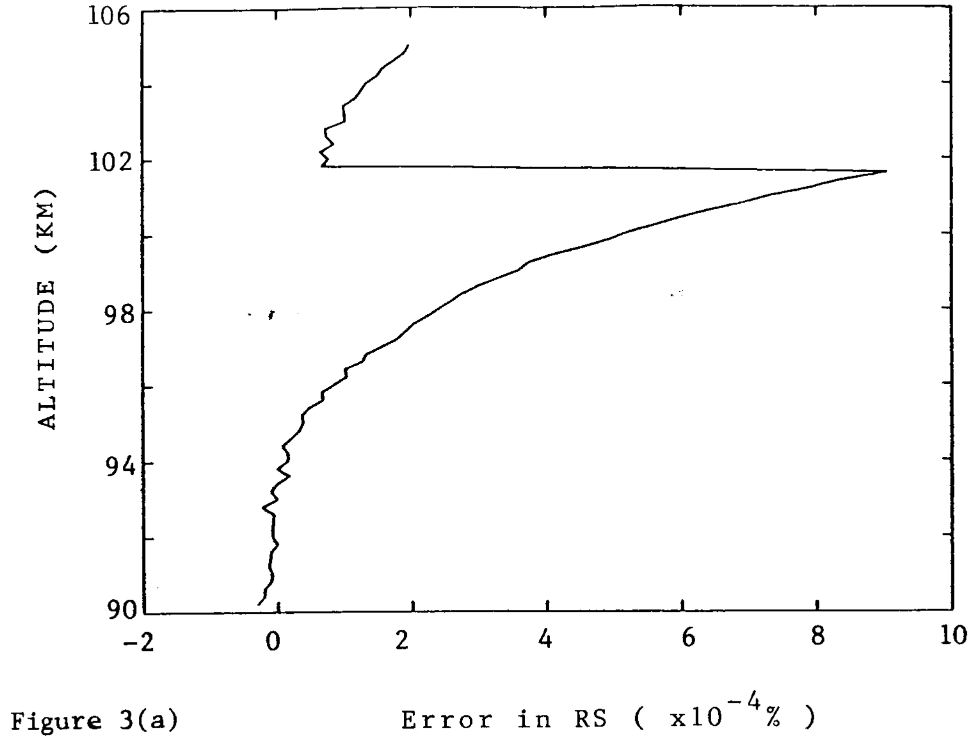
$$\% \text{ deviation} = (R/RD - 1) \times 100\% \qquad (7)$$

where R is equal to either RS or RG. Later in this report, density deviations will be calculated in a similar fashion. Assuming the high-accuracy reference model gives correct results means that these percentage deviations can then be considered as errors in the other two integration methods, and henceforth they will be referred to as such.

The error in RS as a function of altitude in the cold atmosphere is shown in Figure 3(a). At low altitudes, the errors in RS are small, typically less than about $3 \times 10^{-5}\%$ in magnitude, but they increase steadily with increasing altitude and approach a value close to $10^{-3}\%$, just below 102km altitude. Just after the maximum error is reached, the errors decrease rapidly with an increase of altitude, and the process appears to repeat itself again. The reason for this behavior is quite simple. The number of sub-intervals used in the Simpson's Rule, which calculates RS, is the same for all integration intervals extending from 90km to anywhere at or below the altitude where the maximum error is achieved.

Above that altitude the number of sub-intervals is doubled. The error is thus drastically reduced, but as one progresses to greater altitudes, the error begins to increase again. This behavior will be observed repeatedly in all of the results employing Simpson's Rule which follow, both in RS and in density.

The error in RG as a function of altitude in the cold atmosphere is shown in Figure 3(b). The errors in RG are small



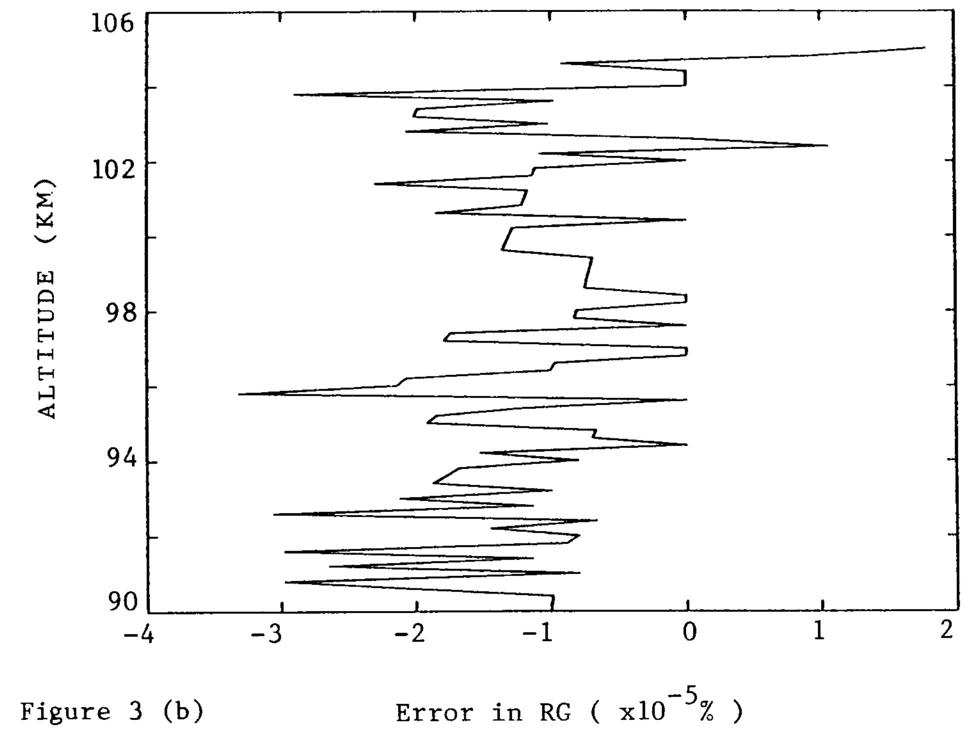


Figure 3. Error in the integral of $g\bar{M}/T$ evaluated using (a) Simpson's Rule and (b) Gaussian Quadrature for the cold atmosphere.

at all altitudes, never exceeding about $3x10^{-5}$% in magnitude. Most of this error is probably associated with single-precision round-off, because on the VAX 11/780 a precision of one part in 10^7 is typically to be expected in the representation of a single-precision real number. A comparison of figures 3(a) and 3(b) reveals that at very low altitudes (between 90 and 95km altitude) the errors in using Simpson's Rule are very similar to those arising from the use of the Gaussian Quadrature. However, at greater altitudes, the errors involved in the use of Simpson's Rule increase remarkably, while those associated with the use of the Gaussian Quadrature remain essentially small and bounded.

The results corresponding to the hot atmosphere are shown in Figures 4(a) and 4(b). There are similarities between these results and those shown in Figures 3(a) and 3(b), but once again the overall trend is that the use of the Gaussian Quadrature gives much smaller errors than through the use of Simpson's Rule. Typically more than an order of magnitude smaller.

Extensive timing tests revealed that the Gaussian integration was always faster than the Simpson's integration. The integrand was always evaluated four times in the Gaussian integration. In the Simpson's' integration the integrand was evaluated four times in the altitudes below where the maximum errors occurred (~100km), and six times for altitudes above where the maximum errors occurred. At 105km altitude the integration took twice the CPU time using Simpson's Rule, but the errors in RS were about an order of magnitude larger than those in RG. Hence, in the 90-105km altitude range the Gaussian integration can be twice as fast and ten times more accurate than the Simpson's integration.

B. COMPARISON OF THE INTEGRALS: THE 105 TO 2500KM SECTION

In the 105 to 2500km altitude range the integral of g/T is evaluated by the three methods. The error in RS as a function of altitude in the cold atmosphere is shown in Figure 5(a). One notices that at 416km and around 1350km altitude the errors in RS are extremely large, and are much larger than expected from the size of the convergence parameter (10^{-4}). These errors are not associated with round-off, but are due to the unreliability of the Simpson's' integration method as implemented in the current MSFC/J70 and NASA MET models.

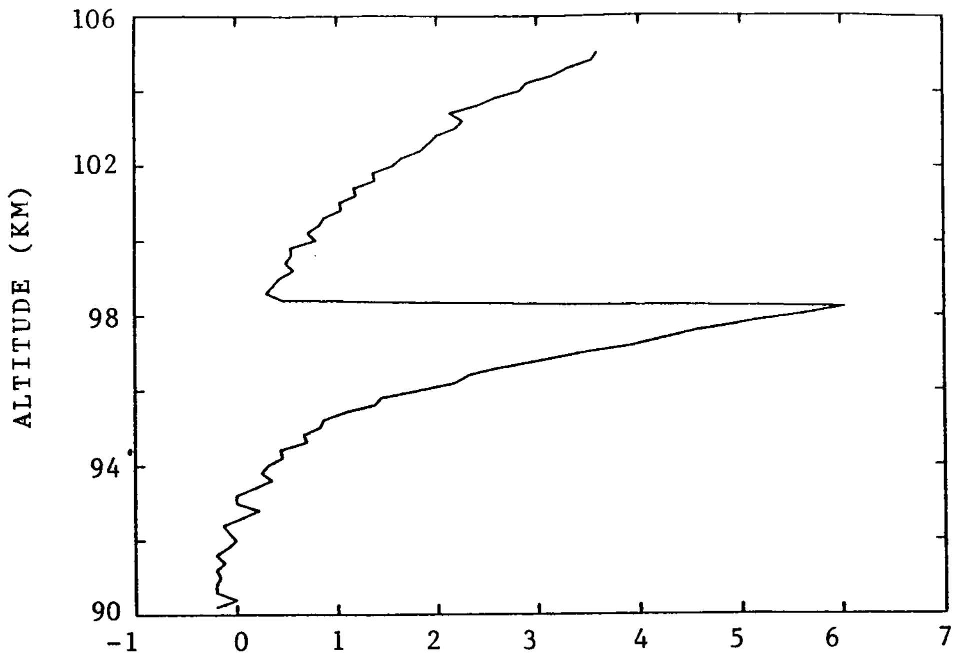


Figure 4(a) Error in RS (x10^{-4}%)

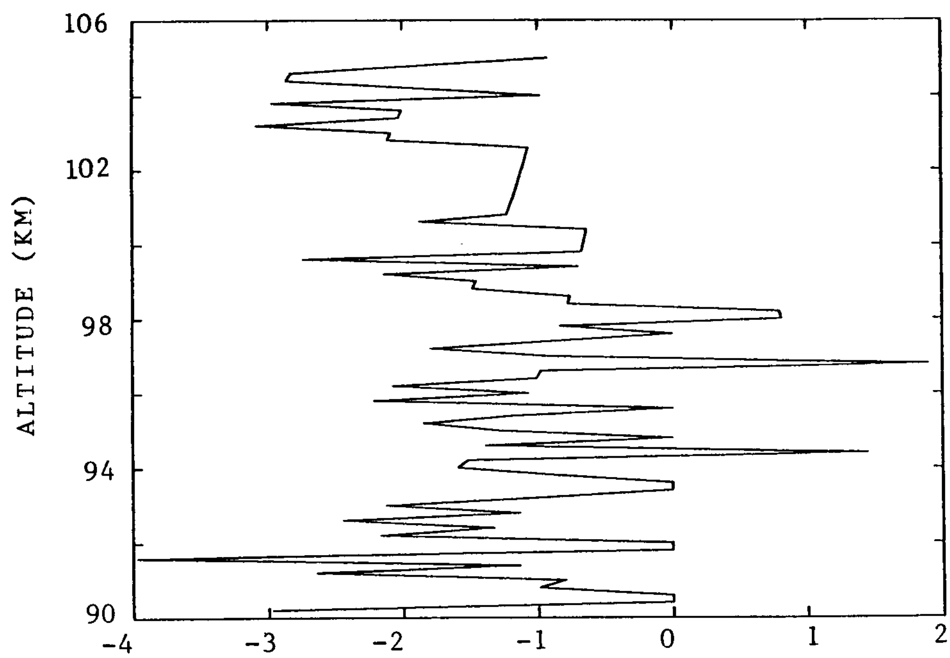


Figure 4(b) Error in RG (x10^{-5}%)

Figure 4. Error in the integral of $g\bar{M}/T$ evaluated using (a) Simpson's Rule and (b) Gaussian Quadrature for the hot atmosphere.

Figure 5(a)

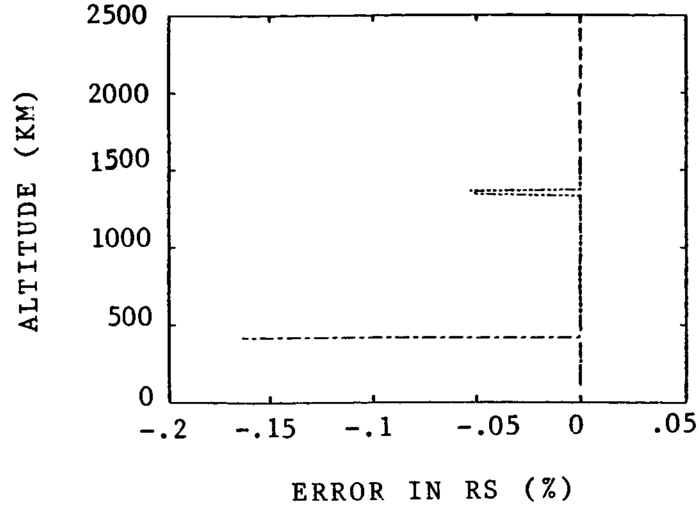


Figure 5(b)

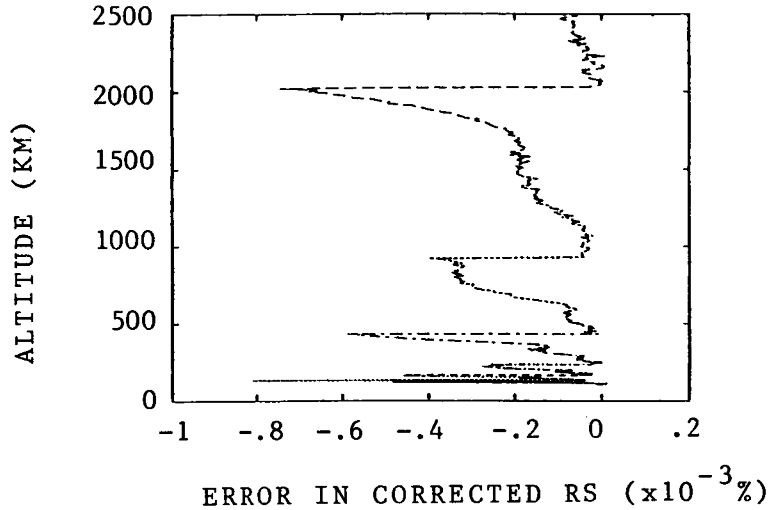


Figure 5(c)

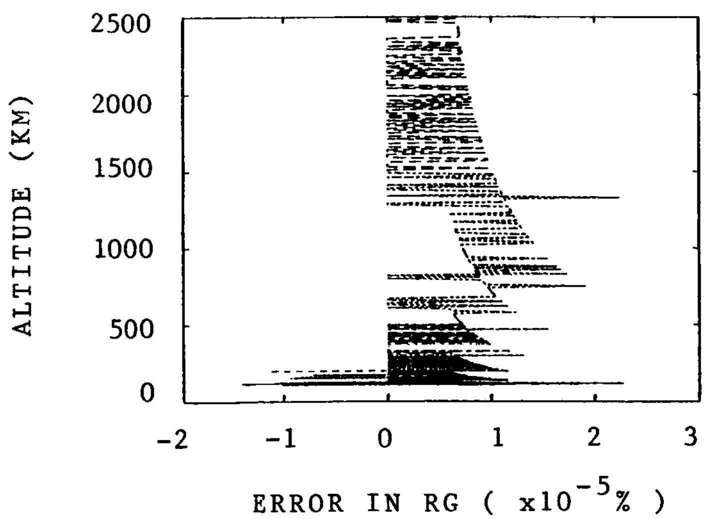


Figure 5. Error in the integral of g/T evaluated using (a) and (b) Simpson's Rule and (c) Gaussian Quadrature for the cold atmosphere. See text for details.

The unreliability of this method, which is due to an occasional and unpredictable false convergence condition in the iteration scheme, is discussed in [4]. At other altitudes, however, the Simpson's integration method appears to be reliable. In order to examine the errors associated with the reliable results, the erroneous results were changed to the same values as those of the neighboring altitudes. These errors, which were undiscernable in figure 5(a), are shown in Figure 5(b). The average errors appear to have a magnitude of 1 to 2 x 10^{-4}%, but maximum errors can be as large as 8 x 10^{-4}% in magnitude. Once the number of sub-intervals used is doubled, the errors reduce drastically to magnitudes of about 3 x 10^{-5}%.

The error in RG, as a function of altitude, in the cold atmosphere is shown in Figure 5(c). One notices that the errors associated with the Gaussian integration remain small at all altitudes, being on average less than 10^{-5}% in magnitude. The largest errors which occur are a little larger than 2 x 10^{-5}%, which is probably due mostly to round-off error. The Gaussian integration method appears to be totally reliable, showing no evidence of the type of errors which can occur with integration by Simpson's Rule (see Figure 5(a)). When the integration using Simpson's Rule is successful, the errors associated with it are on average an order of magnitude larger than those associated with the Gaussian integration.

The equivalent set of results for the hot atmosphere are shown in Figures 6(a), (b) and (c). Under these conditions the Simpson's integration appears to be more unreliable than it was for the cold atmosphere (Figure 6(a)); and once again integration by Simpson's Rule introduced errors which are on average an order of magnitude larger than these associated with the Gaussian integration.

Extensive tests which measured the amount of CPU time that was required to perform the integrations revealed that the Gaussian integration was always faster than the Simpson's integration. It was faster by a factor of approximately 2 or 3 at 200km altitude (2 in the hot atmosphere and 3 in the cold atmosphere), approximately 4.5 at 500km altitude, 7.4 at 1000km altitude increasing to about 12.5 at 2500km altitude. Thus, integration using Gaussian Quadrature will save a considerable amount of CPU time over the use of Simpson's Rule.

Figure 6(a)

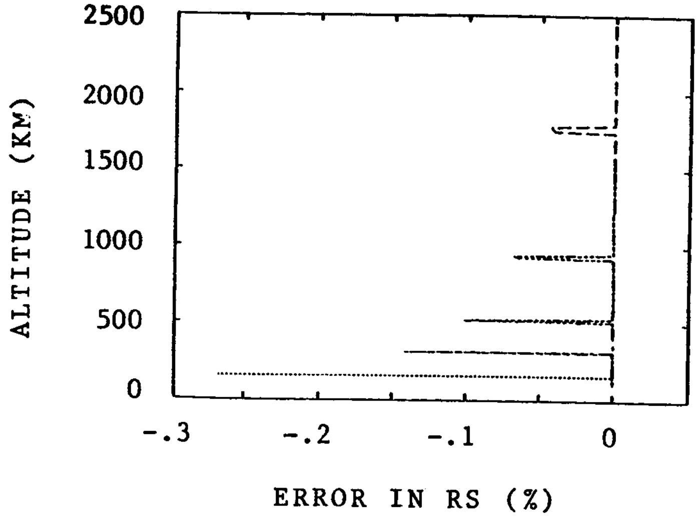


Figure 6(b)

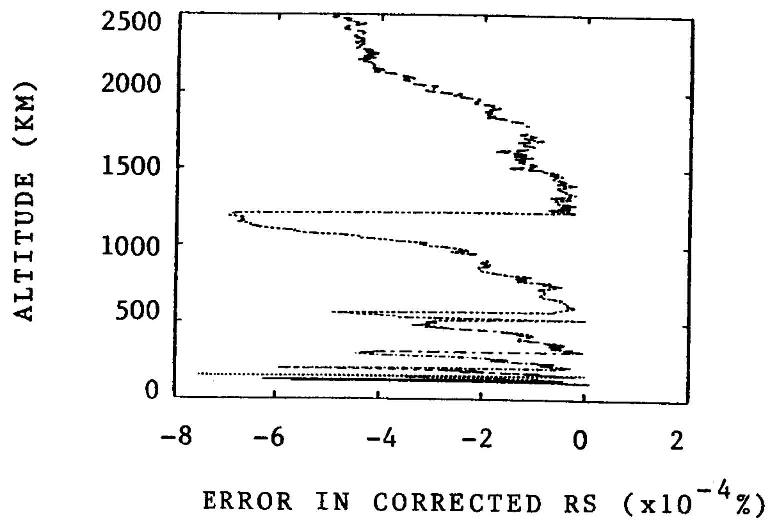


Figure 6(c)

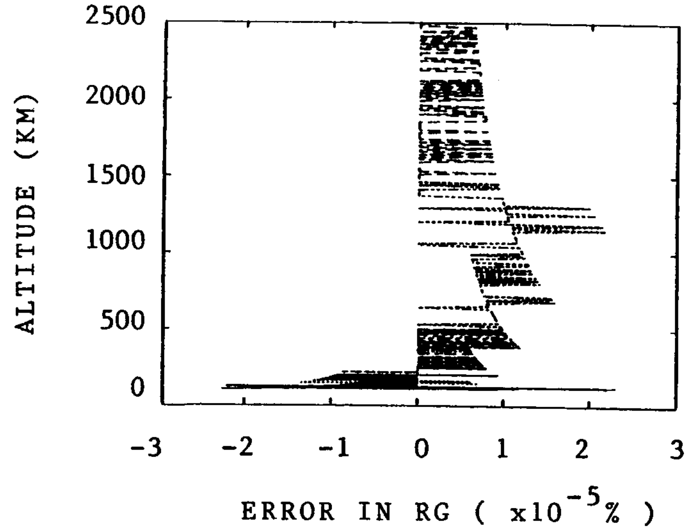


Figure 6. Error in the integral of g/T evaluated using (a) and (b) Simpson's Rule and (c) Gaussian Quadrature for the hot atmosphere. See text for details.

In the following two sub-sections, the densities calculated for the two altitude regions using Gaussian Quadrature and using Simpson's Rule are compared in a similar way that the comparisons were performed in the previous section.

C. COMPARISON OF THE DENSITIES: THE 90 TO 105KM SECTION

The errors in the density values calculated using Simpson's Rule for the cold atmosphere between 90 and 105km altitude are shown in Figure 7(a), while those associated with the Gaussian Quadrature are shown in figure 7(b). The magnitude of the maximum error associated with the use of Simpson's Rule is almost 2×10^{-3}%. Over almost half of the altitude range the magnitude of the errors associated with the use of Simpson's Rule is greater than about 2×10^{-4}%, whereas in the case of the Gaussian Quadrature these errors are "zero" (within the accuracy of single-precision arithmetic) over most (all but 2 km) of the altitude range. For the hot atmosphere, very similar results are obtained (see Figures 8(a) and 8(b)). Thus, independent of atmospheric conditions (hot or cold), the densities are calculated significantly more accurately using Gaussian Quadrature to integrate the barometric equation than by using Simpson's Rule in the 90-105km altitude region.

D. COMPARISON OF THE DENSITIES: THE 105-2500KM SECTION

The errors in the density values calculated using Simpson's Rule for the cold atmosphere between 105 and 2500km altitude are shown in Figure 9(a). The large errors at 416km and 1350km altitude are associated with the unreliability of the Simpson's integration method which gave large errors in the calculations of the integral of g/T, as previously shown in Figure 5(a). Once again, in order to examine the errors associated with reliable integration using simpson's Rule the large errors were removed from Figure 9(a), and the remaining errors in the density values were re-plotted in Figure 9(b). The magnitudes of the errors are usually smaller than 3×10^{-3}% except just above 400km altitude where they rise to almost 5.5×10^{-3}%.

The corresponding errors in the density values calculated using Gaussian Quadrature are shown in figure 9(c). These errors are negative at all altitudes, meaning that the calculated densities are smaller than they should be. The

Figure 7(a)

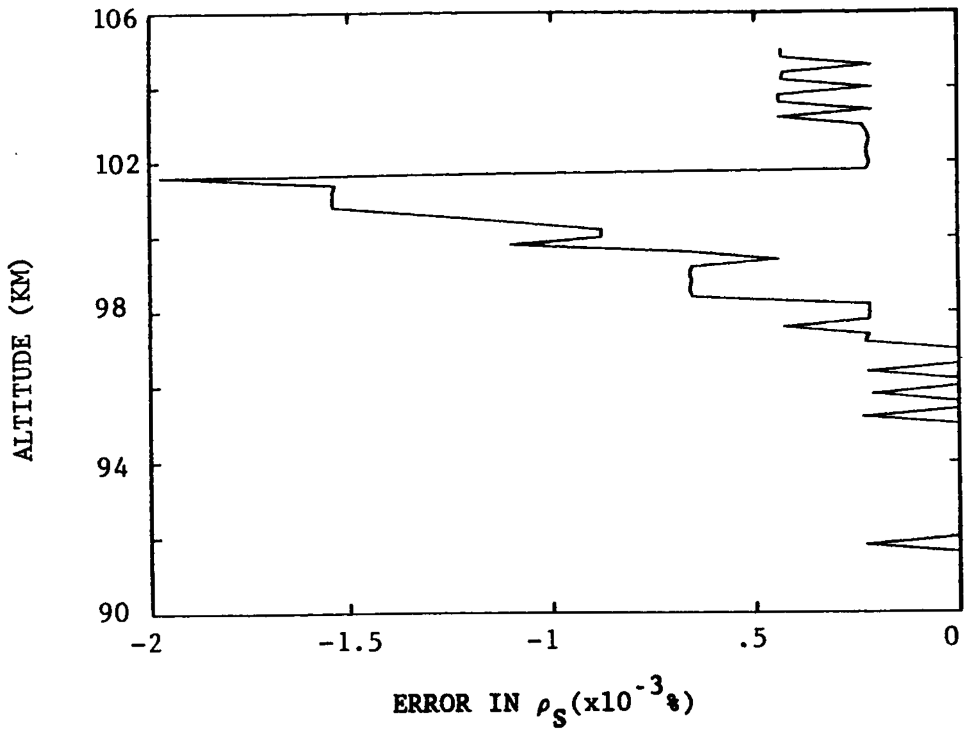


Figure 7(b)

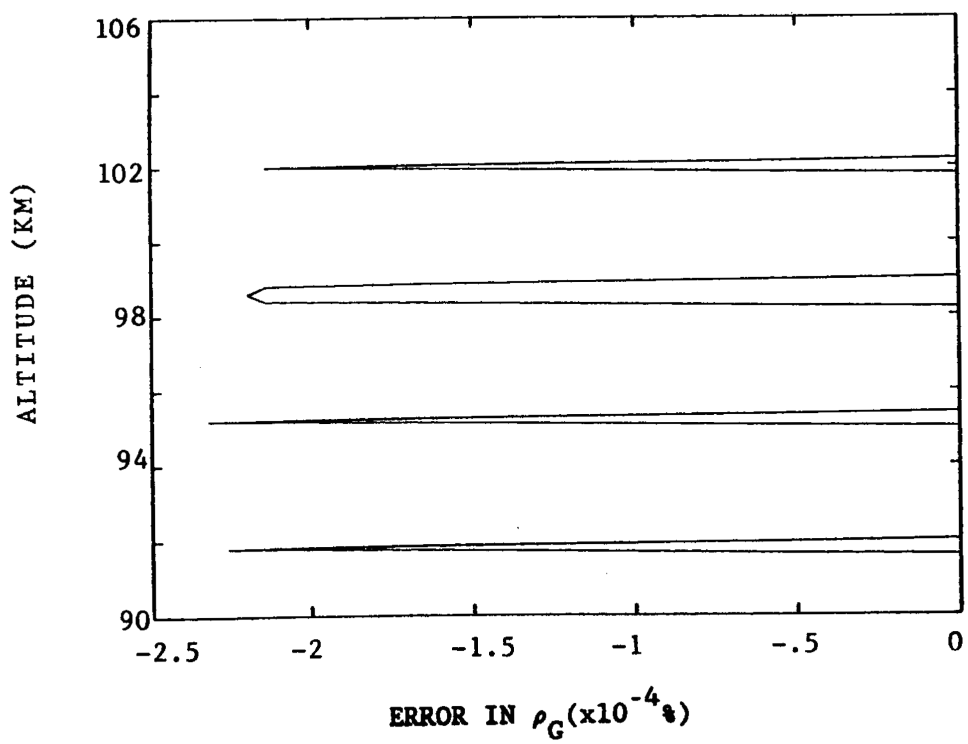


Figure 7. Error in the density evaluated using (a) Simpson's Rule and (b) Gaussian Quadrature for the cold atmosphere below 105km altitude.

Figure 8(a)

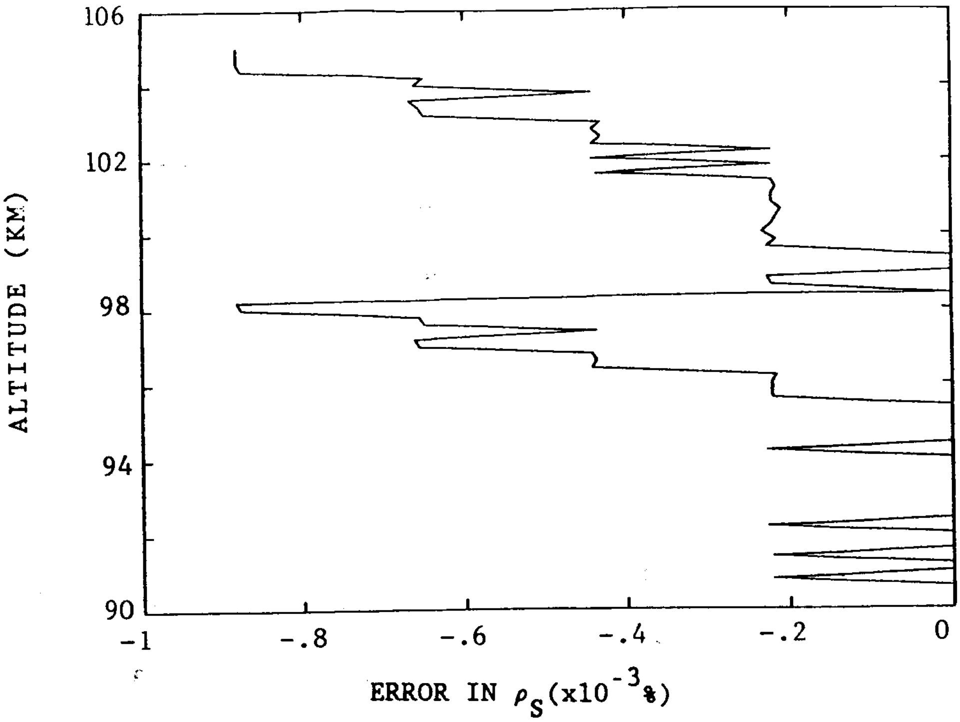


Figure 8(b)

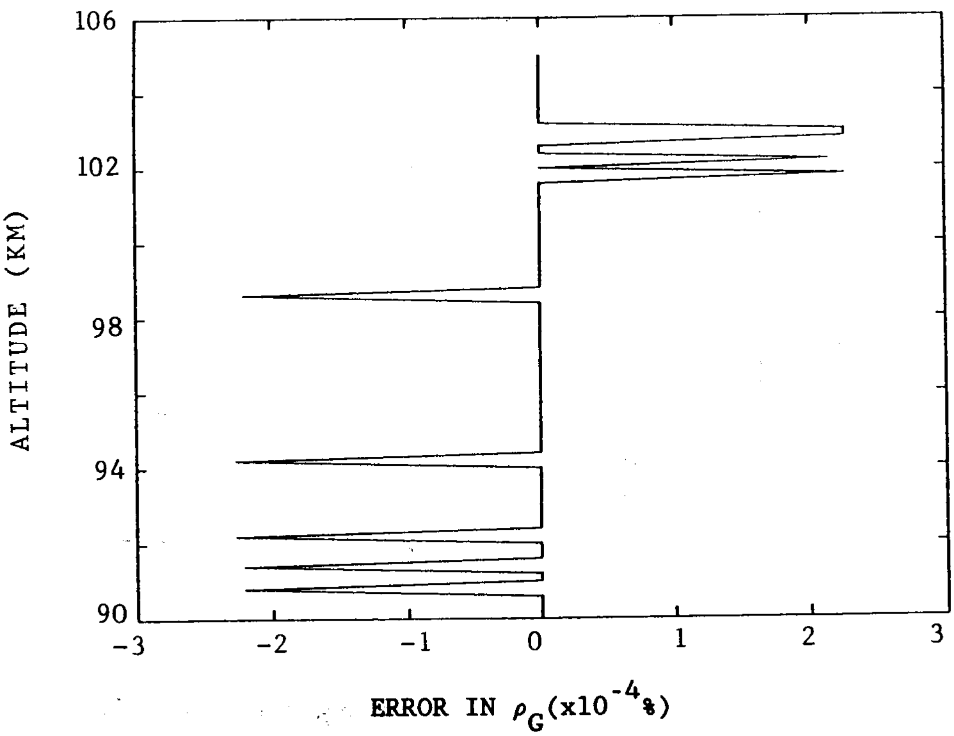


Figure 8. Error in the density evaluated using (a) Simpsons's Rule and (b) Gaussian Quadrature for the hot atmosphere below 105km altitude.

Figure 9(a)

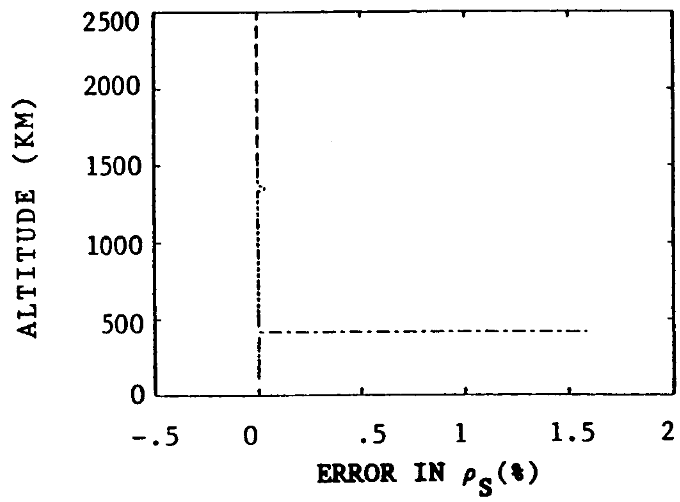


Figure 9(b)

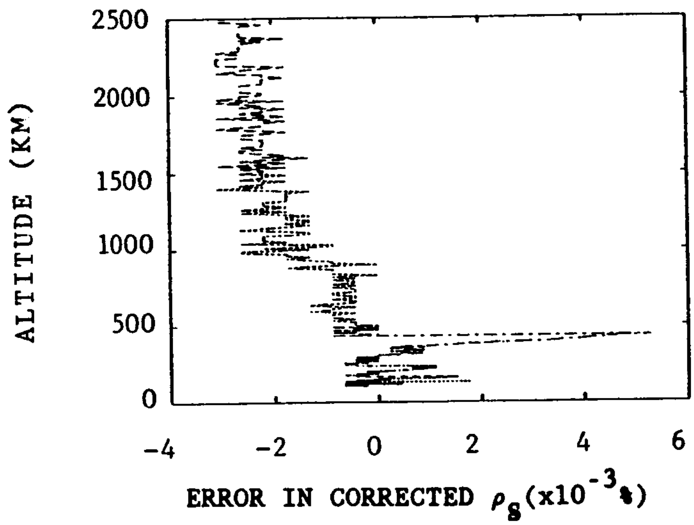


Figure 9(c)

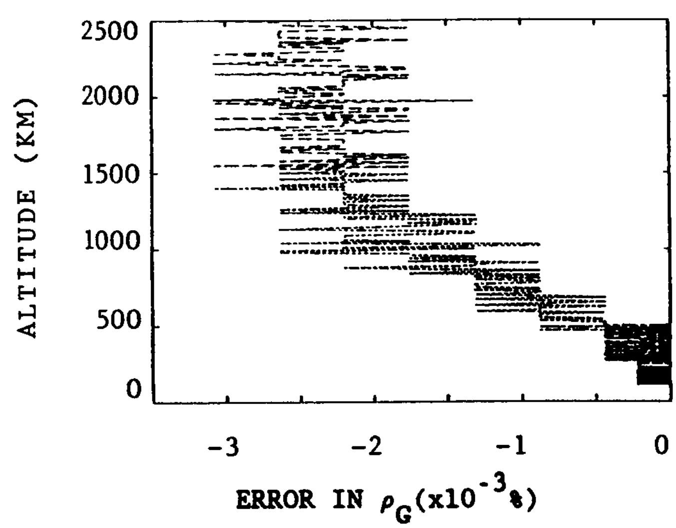


Figure 9. Error in the density evaluated using (a) and (b) Simpson's Rule (c) Gaussian Quadrature for the cold atmoshere above 105km altitude. See text for details.

magnitudes of the errors also generally increase with increasing altitude, being less than about 2.5 x 10^{-4}% at low altitudes and rising to values of about 3 x 10^{-3}% at the highest altitudes. A comparison of Figures 9(c) and 9(b) reveals that at the highest altitudes (say, greater than 1000km) the errors arising from the two integration methods (Simpson's Rule and Gaussian Quadrature) are much the same, providing that one overlooks the errors of about 3 x 10^{-2}% centered around 1350km altitude associated with the unreliability of the Simpson's integration method employed in the MET model. At low altitudes, however, the densities calculated using the Gaussian Quadrature will always be accurate, usually much more accurate than those calculated using Simpson's Rule.

The corresponding set of results for the hot atmosphere are shown in Figures 10(a), (b) and (c). In the hot atmosphere the Simpson's integration method appears to be considerably more unreliable than it is in the cold atmosphere (compare Figures 9(a) and 10(a)). A comparison of figures 10(c) and 9(c) shows that the Gaussian Quadrature gives much more accurate density values in the hot atmosphere than in the cold atmosphere. In the hot atmosphere the magnitudes of these errors are usually less than 4 x 10^{-4}%, so that the Gaussian Quadrature calculates densities significantly more accurately than does the Simpson's integration method at all altitudes in the hot atmosphere.

E. A NUMERICAL EXAMPLE

Here, the density value output from the standard MET model using Simpson's Rule, ρ_S, the modified MET model using Gaussian Quadrature, ρ_G, and the high-precision reference model (double precision, smaller tolerance, etc.), ρ_R, are given. These results are for the hot atmosphere as defined by the input parameters given on page 10, and for an altitude of 400km:

$$\rho_S = 3.3844014 \times 10^{-11} \mathrm{kgm}^{-3}$$

$$\rho_G = 3.3844164 \times 10^{-11} \mathrm{kgm}^{-3}$$

$$\rho_R = 3.3844164 \times 10^{-11} \mathrm{kgm}^{-3}$$

Figure 10(a)

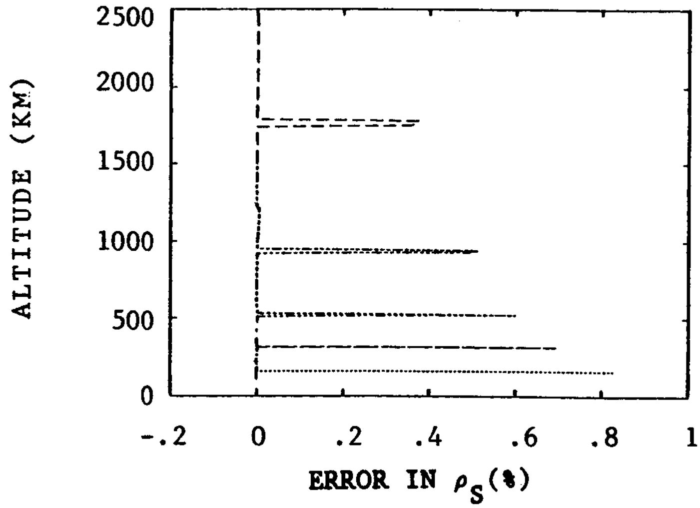


Figure 10(b)

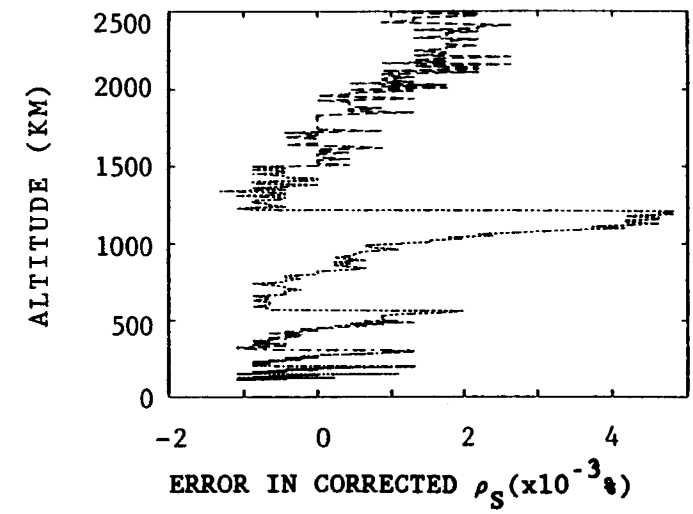


Figure 10(c)

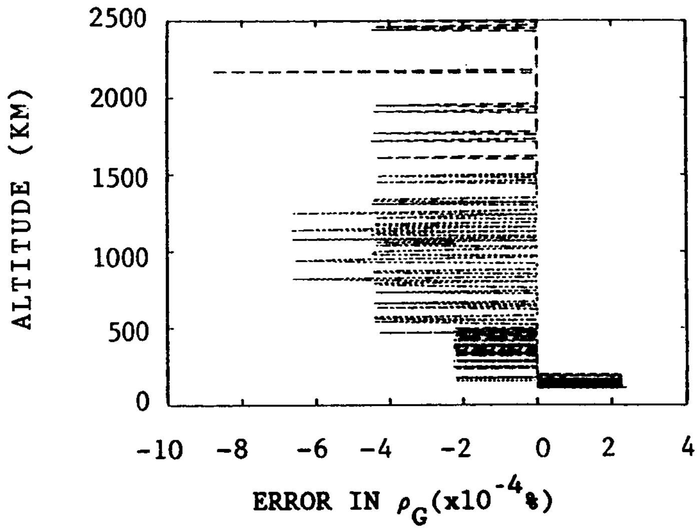


Figure 10. Error in the diensity evaluated using (a) and (b) Simpson's Rule and (c) Gaussian Quadrature for the hot atmosphere above 105km altitude. See text for details.

Thus, the error in the density value calculated with the MET model is about $-4.43 \times 10^{-4}\%$ while the corresponding error with the modified MET model which uses Gaussian Quadrature is zero.

IV. DISCUSSION

The evaluation of the integral of $g\bar{M}/T$ between 90 and 105km altitude and g/T between 105 and 2500km altitude has been performed by two different methods. The first of these methods employs a modified form of Simpson's Rule, basically as used in the MSFC/J70 and NASA MET models. The second method is Gaussian Quadrature. Extensive comparisons of the results of using these two methods have revealed that integration by Gaussian Quadrature is significantly numerically faster than by Simpson's Rule and that the integration by Gaussian Quadrature is overall significantly more accurate than by Simpson's Rule. It was also found that at certain altitudes the form of Simpson's Rule used in the MET model becomes unreliable, where errors in the results become significantly larger than expected from the convergence criterion used in the iterative scheme of the method. This unreliability occurred more often in the hot atmosphere. No such unreliability was evident in any of the results which involved Gaussian Quadrature.

Extensive comparisons of the densities calculated using the two methods revealed, not surprisingly, that those resulting from the use of Gaussian Quadrature were overall significantly more accurate than those resulting from the use of Simpson's Rule. Also, reliability of the calculated densities was a problem with Simpson's Rule, but not with the Gaussian Quadrature. Any program which is written to calculate densities will run faster if integration is done using Gaussian Quadrature, however, just how much faster it will run, than an equivalent program which integrates using Simpson's Rule, will depend on the nature of the particular program involved. When Gaussian Quadrature was used, the program which was used to generate the data files, which are plotted in Figures 7(a) through to 10(c), ran only very slightly faster at low altitudes and ran five times faster above 2000km altitude.

To all intents and purposes the results obtained by using the Gaussian Quadrature are as accurate and reliable as those obtained from the "reference" model. One must recall that this "reference" model employed the same form of Simpson's Rule that is used in the MET model, but it is written in double-precision. It has a convergence parameter that is one-thousand times smaller than that used in the MET model, and requires completion of two consecutive positive convergence tests before convergence is accepted. It is no wonder that integration using this reliable version of Simpson's Rule took anywhere from about 30 to 400 times longer than by using Gaussian Quadrature.

One should also note that the choice of sub-interval parameters which are given in Table 2 could possibly be improved upon or modified to suit ones individual needs. For general use, however, the values given in Table 2 are recommended because they do not compromise on accuracy, and the Gaussian Quadrature using these values still requires less computational time than the standard method (Simpson's Rule) which is currently used in the MSFC/J70 and MET models.

Finally, it should be noted that if integration is performed between two different altitudes (say z_1, and z_2) and the fractional errors associated with this integration are greater at one altitude (z_1) than the other (z_2), it does not necessarily follow that when the densities are calculated the fractional error in the density at altitude z_1 will be greater than that at z_2. This is because the density calculation involves integration errors being multiplied by numbers which are altitude dependent.

V. CONCLUSIONS AND RECOMMENDATIONS

Although the reliability of the integration scheme used in the MSFC/J70 model was improved in the MET model, the results presented in this report have shown that it is still not as reliable as desired. An alternative integration scheme, based on Gaussian Quadrature, has been substituted for the standard integration scheme used in the MET model and has been found to be totally reliable. At low altitudes, it generally gives significantly more accurate density results than does the standard MET model, and uses less computer time to do it. At high altitudes, it generally gives results which have a similar accuracy to those of the standard MET model, but it uses only a fraction of the computer time to do it.

Due to the reliability, the accuracy, and the speed of this new integration scheme, which uses Gaussian Quadrature, it is recommended that this new scheme replace the older, more unreliable scheme, based upon Simpson's Rule, as presently used in the MET model. If this is done, it is emphasized that the integration parameters given in Table 2 should be adhered to. This being the case, it is further recommended that the program listed in the Appendix, which is also named the NASA Marshall Engineering Thermosphere (MET) Model, and which contains the essentials of this new integration scheme in subroutine GAUSS, should replace the current MET program.

REFERENCES

1. Jacchia, L. G.: New Static Models of the thermosphere and Exosphere with Empirical Temperature Profiles. Smithsonian Astrophysical Observatory Special Report 313, May 6, 1970.

2. Jacchia, L. G.: Revised Static Models of the Thermosphere and Exosphere with Empirical Temperature Profiles. Smithsonian Astrophysical Observatory Special Report 332, May, 1971.

3. Johnson, D. L. and Smith, R. E.: The MSFC/J70 Orbital Atmospheric Model and the Data Bases for the MSFC Solar Activity Prediction Technique. NASA TM-86522, November 1985.

4. Hickey, M. P.: The NASA Marshall Engineering Thermosphere Model. NASA CR-179359, July 1988.

5. Stark, P. A.: Introduction to Numerical Methods. The MacMillon Company, N.Y., 1970.

```
C**************************************************************************
C*                                                                        *
C*                 The  Marshall  Space  Flight  Center                   *
C*               Marshall Engineering  Thermosphere Model                 *
C*        ________________________________________________________        *
C*                                                                        *
C*                              written by                                *
C*                                                                        *
C*                              Mike Hickey                               *
C*                Universities Space Research Association                 *
C*                          NASA / MSFC , ED44                            *
C*                          Tel. (205) 544-5692                           *
C*                                                                        *
C*   This program is a driving program for the following subroutines :-  *
C*                                                                        *
C*                          ATMOSPHERES                                   *
C*                          SOLSET                                        *
C*                          TIME                                          *
C*                          J70                                           *
C*                                                                        *
C*  The atmospheric model is a modified Jacchia 1970 model and is given in *
C*  the subroutine J70.  All of the other subroutines were designed to    *
C*  allow flexible use of this model so that various input parameters could *
C*  be varied within a driving program with very little software development.*
C*  Thus, for example, driving routines can be written quite easily to    *
C*  facilitate the plotting of output as line or contour plots.  Control is *
C*  achieved by setting the values of four switches in the driving program, *
C*  as described in subroutine ATMOSPHERES.                               *
C*                                                                        *
C**************************************************************************

        REAL*4  INDATA (12) , OUTDATA (12) , AUXDATA (5)

        CHARACTER*1  SWITCH (4)

        CALL LIB$INIT_TIMER

C  Set all switches to 'Y' so that only one particular calculation is performed

                SWITCH (1) = 'Y'
                SWITCH (2) = 'Y'
                SWITCH (3) = 'Y'
                SWITCH (4) = 'Y'

        CALL ATMOSPHERES ( INDATA, OUTDATA, AUXDATA, SWITCH )

C  Now type output data

        Type *,' All output in MKS units'
        Type *,' '
        Type *,' Exospheric temperature = ', OUTDATA (1),' K'
        Type *,' Temperature            = ', OUTDATA (2),' K'
        Type *,' N2 number density      = ', OUTDATA (3),' /m3'
        Type *,' O2 number density      = ', OUTDATA (4),' /m3'
        Type *,' O number density       = ', OUTDATA (5),' /m3'
        Type *,' A number density       = ', OUTDATA (6),' /m3'
        Type *,' He number density      = ', OUTDATA (7),' /m3'
        Type *,' H number density       = ', OUTDATA (8),' /m3'
        Type *,' Average molecular wt.  = ', OUTDATA (9)
        Type *,' Total mass density     = ', OUTDATA (10),' kg/m3'
        Type *,' Log10 mass density     = ', OUTDATA (11)
        Type *,' Total pressure         = ', OUTDATA (12),' Pa'
        Type *,' Local grav. acceln.    = ', AUXDATA (1),' m.sec-2'
        Type *,' Ratio specific heats   = ', AUXDATA (2)
        Type *,' Pressure scale-height  = ', AUXDATA (3),' m'
        Type *,' Specific heat cons. p  = ', AUXDATA (4),'m2.sec-2.K-1'
        Type *,' Specific heat cons. v  = ', AUXDATA (5),'m2.sec-2.K-1'
        Type *,' '
```

```
CALL LIB$SHOW_TIMER

STOP
END
```

SUBROUTINE ATMOSPHERES (INDATA, OUTDATA, AUXDATA, SWITCH)

```
C*************************************************************************
C*                         DESCRIPTION:-                                 *
C*                         ------------                                  *
C*                                                                       *
C*  Calculate atmospheric data in single precision using subroutine J70  *
C*  and J70SUP.                                                          *
C*                                                                       *
C*                         SUBROUTINES:-                                 *
C*                         ------------                                  *
C*                                                                       *
C*      TIME, SOLSET, GMC, J70 and J70SUP                                *
C*                                                                       *
C*                           INPUT:-                                     *
C*                           ------                                      *
C*  ------------ all single precision, either through   ---------------  *
C*  ------------ subroutines or from main driver prog.  ---------------  *
C*                                                                       *
C*  INDATA (1) -- altitude               = Z                             *
C*    ..   (2) -- latitude               = XLAT                          *
C*    ..   (3) -- longitude              = XLNG                          *
C*    ..   (4) -- year (yy)              = IYR                           *
C*    ..   (5) -- month (mm)             = MN                            *
C*    ..   (6) -- day (dd)               = IDA                           *
C*    ..   (7) -- hour (hh)              = IHR                           *
C*    ..   (8) -- mins (mm)              = MIN                           *
C*    ..   (9) -- geomagnetic index      = IGEO_IND                      *
C*    ..   (10)-- solar radio noise flux= F10                            *
C*    ..   (11)-- 162-day average F10    = F10B                          *
C*    ..   (12)-- geomagnetic activity index = GI=AP                     *
C*                                                                       *
C*                                                                       *
C*                           OUTPUT:-                                    *
C*                           -------                                     *
C*                                                                       *
C*     NOTE :  All output in MKS units                                   *
C*     -------------------------------                                   *
C*                                                                       *
C*  --------------------- all single precision  ---------------------    *
C*                                                                       *
C*                                                                       *
C*  OUTDATA (1) -- exospheric temperature (K)                            *
C*    ..    (2) -- temperature at altitude Z                             *
C*    ..    (3) -- N2 number density  (per meter-cubed)                  *
C*    ..    (4) -- O2 number density  (      ..       )                  *
C*    ..    (5) -- O  number density  (      ..       )                  *
C*    ..    (6) -- A  number density  (      ..       )                  *
C*    ..    (7) -- He number density  (      ..       )                  *
C*    ..    (8) -- H  number density  (      ..       )                  *
C*    ..    (9) -- average molecular weight                              *
C*    ..    (10)-- total density                                         *
C*    ..    (11)-- log10 ( total density )                               *
C*    ..    (12)-- total pressure ( Pa )                                 *
C*                                                                       *
C*  AUXDATA (1) -- gravitational acceleration ( m/s-s )                  *
C*    ..    (2) -- ratio of specific heats                               *
C*    ..    (3) -- pressure scale-height ( m )                           *
C*    ..    (4) -- specific heat at constant pressure                    *
C*    ..    (5) -- specific heat at constant volume                      *
C*                                                                       *
C*                                                                       *
C*                          COMMENTS:-                                   *
C*                          ---------                                    *
C*                                                                       *
C*  SWITCH(1) -- if Y(es), date and time are input from terminal through *
C*                         subroutine TIME once only                     *
C*  SWITCH(2) -- if Y(es), solar/magnetic activity are input from terminal *
C*                         through subroutine SOLSET once only           *
C*  SWITCH(3) -- if Y(es), only ONE altitude value is input from terminal *
C*                         through main calling program                  *
C*  SWITCH(4) -- if Y(es), only ONE latitude AND longitude are input from *
C*                         terminal through main calling program         *
C*                                                                       *
```

```
C*   ATMOSPHERES written by Mike Hickey ( USRA, NASA/ED44 )                   *
C*                        Tel: (205) 544-5692                                 *
C*         ------   January-April 1987  -------                               *
C*****************************************************************************

        EXTERNAL TIME

        DIMENSION  AUXDATA (5)

        INTEGER  HR

        REAL*4  LAT, LON, INDATA (12), OUTDATA (12)

        CHARACTER*1  SWITCH (4)

        PARAMETER PI = 3.14159265

C ----------------------------------------------------------------------------
C       This next section is only executed on the first call to ATMOSPHERES

                DO WHILE ( CALL. EQ. 0.0 )

C  SECTION A:-
C  ----------

        IF ( SWITCH(1). EQ. 'Y' ) THEN

        CALL TIME ( IYR, MON, IDA, HR, MIN, SWITCH(1) )
        INDATA (4) = FLOATJ (IYR)
        INDATA (5) = FLOATJ (MON)
        INDATA (6) = FLOATJ (IDA)
        INDATA (7) = FLOATJ (HR)
        INDATA (8) = FLOATJ (MIN)

        END IF

C  SECTION B:-
C  ----------

        IF ( SWITCH(2). EQ. 'Y' ) THEN

        CALL SOLSET ( IGEO_IND, F10, F10B, GI, SWITCH(2) )
        INDATA (9) = FLOATJ (IGEO_IND)
        INDATA (10) = F10
        INDATA (11) = F10B
        INDATA (12) = GI

        END IF

C  SECTION C:-
C  ----------

        IF ( SWITCH(3). EQ. 'Y' ) THEN

        TYPE *,' Input altitude, km'
        ACCEPT *, INDATA (1)
        Z = INDATA (1)

        END IF

C  SECTION D:-
C  ----------

        IF ( SWITCH(4). EQ. 'Y' ) THEN

        TYPE *,' Input latitude and longitude, degrees'
        ACCEPT *, ( INDATA(I), I= 2,3 )
        LAT = INDATA (2)
        LON = INDATA (3)
        RLT = INDATA (2) * PI / 180.     !  geographic latitude, radians

        END IF
```

```
        CALL = 1.0
                END DO

C       End of first executable section
C ------------------------------------------------------------------------

C  The following depend on the values of the switches

C****
C* SECTION 1:-

                IF ( SWITCH(1). NE. 'Y' ) THEN

        IYR = JINT ( INDATA (4) )
        MON = JINT ( INDATA (5) )
        IDA = JINT ( INDATA (6) )
        HR  = JINT ( INDATA (7) )
        MIN = JINT ( INDATA (8) )
        CALL TIME ( IYR, MON, IDA, HR, MIN, SWITCH(1) )

                END IF


C*****
C* SECTION 2:-

                IF ( SWITCH(2). NE. 'Y' ) THEN

        IGEO_IND = JINT ( INDATA (9) )
        F10      = INDATA (10)
        F10B     = INDATA (11)
        GI       = INDATA (12)
        CALL SOLSET ( IGEO_IND, F10, F10B, GI, SWITCH(2) )

                END IF

C*****
C* SECTION 3:-


                IF ( SWITCH(3). NE. 'Y' ) THEN

        Z = INDATA (1)

                END IF


C*****
C* SECTION 4:-

                IF ( SWITCH(4). NE. 'Y' ) THEN

        LAT = INDATA (2)
        LON = INDATA (3)
        RLT = INDATA (2) * PI / 180.     ! geographic latitude, radians

                END IF


C  All setting-up complete.

                CALL J70 ( INDATA, OUTDATA )
                CALL J70SUP ( Z, OUTDATA, AUXDATA )


        RETURN

        ENTRY ATMOS_ENT ( DUMMY )
        CALL = DUMMY
        RETURN

        END
```

```
      SUBROUTINE TIME ( IYR, MON, IDA, HR, MIN, SWITCH )

C*****************************************************************************
C*                                                                           *
C*        This subroutine sets up time of year and day                       *
C*                                                                           *
C*                   INPUTS/OUTPUTS:                                         *
C*                                                                           *
C*  IYR  =  year ( 2 digits )                                                *
C*  MON  =  month                                                            *
C*  IDA  =  day of month                                                     *
C*  HR   =  hour of day                                                      *
C*  MIN  =  minutes                                                          *
C*                                                                           *
C*       Written by Mike Hickey, USRA                                        *
C*****************************************************************************

      DIMENSION  IDAY ( 12 )

      INTEGER  HR

      CHARACTER*1  SWITCH

      DATA IDAY / 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 /

      PARAMETER PI = 3.14159265

C-----------------------------------------------------------------------------
C ------     If  SWITCH = Y(es)  then input data and time from terminal ------
C-----------------------------------------------------------------------------

              IF ( SWITCH.EQ.'Y'. OR. SWITCH.EQ.'y' )  THEN

      TYPE *, ' Input date and time of date? ( yy,mm,dd,hh,mm ) '
      ACCEPT *, IYR, MON, IDA, HR, MIN

              END IF

C ----------------------------------------------------------------------------

      IF (  JMOD (IYR,4) .EQ. 0 )  THEN
              IF ( JMOD (IYR,100) .NE. 0 ) IDAY ( 2 ) = 29
      ELSE
              IDAY ( 2 ) = 28
      END IF

      DAYTOT = 0.0

              DO  1  I = 1, 12
      DAYTOT = DAYTOT + FLOATJ ( IDAY ( I )  )
1             CONTINUE

              IF ( MON. GT. 1 )              THEN

      KE = MON - 1
      ID = 0
              DO 2  I = 1, KE
      ID = ID + IDAY (I)
2             CONTINUE

      ID = ID + IDA
                                             ELSE
      DD = IDA
              END IF

      RETURN
      END
```

```fortran
      SUBROUTINE SOLSET ( IGEO_IND, F10, F10B, GI, SWITCH )

C***********************************************************************
C*                                                                     *
C* This subroutine simply calls for a setup of the solar-activty and auroral *
C* activity indices.                                                   *
C*                                                                     *
C*                      INPUTS/OUTPUTS:                                *
C*                                                                     *
C*  IGEO_IND = geomagnetic index                                       *
C*  F10      = solar radio noise flux                                  *
C*  F10B     = 162-day average F10                                     *
C*  GI       = geomagnetic activity index                              *
C*                                                                     *
C*     Written by Mike Hickey, USRA                                    *
C***********************************************************************

      CHARACTER*1  SWITCH

      IGEO_IND = 2

C ----------------------------------------------------------------------
C ---- If SWITCH = Y(es) then input geomagnetic indices from terminal ----
C_______________________________________________________________________

              IF ( SWITCH.EQ.'Y'. OR. SWITCH.EQ.'y' ) THEN

C     TYPE *, ' Input geomagnetic index ( 1-KP, 2-AP ) '
C     ACCEPT *, IGEO_IND

      TYPE *, ' Input solar radio noise flux ( F10 = 0-400 ) '
      ACCEPT *, F10

      TYPE *, ' Input 162-day average F10 ( F10B = 0-250 ) '
      ACCEPT *, F10B

C             IF ( IGEO_IND . EQ. 2 )         THEN
C
C     TYPE *, ' Input geomagnetic activity index ( GI = 0-400 ) '
C
C                                             ELSE
C
C     TYPE *, ' Input geomagnetic activity index ( GI = 0-9 ) '
C
C             END IF

      TYPE *,' Input AP index ( AP = 0 - 400 ) '

      ACCEPT *, GI

              END IF

C _______________________________________________________________________


      RETURN
      END
```

```
      SUBROUTINE J70SUP ( Z, OUTDATA, AUXDATA )

C*************************************************************************
C*                                                                       *
C*                         DESCRIPTION:-                                 *
C*                         ------------                                  *
C*                                                                       *
C*  J70SUP calculates auxilliary variables which are output in array     *
C*  AUXDATA, given data input from J70 which are contained in array OUTDATA. *
C*                                                                       *
C*                         INPUT DATA:-                                  *
C*                         -----------                                   *
C*                                                                       *
C*   Z  —  altitude (km)                                                 *
C*  TZZ —  temperature at altitude z    =  OUTDATA (2)                   *
C*      —  N2 number density            =     ..   (3)                   *
C*      —  O2  ..      ..               =     ..   (4)                   *
C*      —  O   ..      ..               =     ..   (5)                   *
C*      —  A   ..      ..               =     ..   (6)                   *
C*      —  He  ..      ..               =     ..   (7)                   *
C*      —  H   ..      ..               =     ..   (8)                   *
C*  EM  —  average molecular weight     =     ..   (9)                   *
C* DENS —  total density                =     ..   (10)                  *
C*  P   —  total pressure               =     ..   (12)                  *
C*                                                                       *
C*                         OUTPUT DATA:-                                 *
C*                         ------------                                  *
C*                                                                       *
C*  G  —  gravitational acceleration          =  AUXDATA (1)             *
C* GAM —  ratio of specific heats             =  AUXDATA (2)             *
C*  H  —  pressure scale-height               =  AUXDATA (3)             *
C*  CP —  specific heat at constant pressure  =  AUXDATA (4)             *
C*  CV —  specific heat at constant volume    =  AUXDATA (5)             *
C*                                                                       *
C*  Written by Mike Hickey, USRA                                         *
C*************************************************************************

      REAL*4  OUTDATA (12), AUXDATA (5), H

      G = 9.80665 / ( ( 1. + Z / 6.356766E3 )**2 )

      H = OUTDATA (12) / ( G * OUTDATA (10) )

      SUM1 = OUTDATA (3) + OUTDATA (4)
      SUM2 = 0.0
              DO 1  I = 5, 8
      SUM2 = SUM2 + OUTDATA (I)
1             CONTINUE

      GAM = ( 1.4 * SUM1 + 1.67 * SUM2 ) / ( SUM1 + SUM2 )

      CV = G * H / (   ( GAM - 1.0 ) * OUTDATA (2)    )

      CP = GAM * CV

      AUXDATA (1) = G
      AUXDATA (2) = GAM
      AUXDATA (3) = H
      AUXDATA (4) = CP
      AUXDATA (5) = CV

      RETURN
      END
```

```
                    SUBROUTINE J70 ( INDATA, OUTDATA )

C**************************************************************************
C**                                                                     **
C**                  J70 developed from J70MM by                        **
C**                       Mike P. Hickey                                **
C**           Universities Space Research Association                   **
C**                            at                                       **
C**           NASA / Marshall Space Flight Center, ED44,                **
C**               Huntsville, Alabama, 35812, USA.                      **
C**                    Tel. (205) 544-5692                              **
C**                                                                     **
C** INPUTS:        through the subroutine calling list                  **
C**                                                                     **
C** OUTPUTS:       through the subroutine calling list                  **
C**                                                                     **
C**                                                                     **
C**                        INPUT DATA:                                  **
C**                                                                     **
C**          Z    —  altitude                    = INDATA (1)           **
C**          XLAT — latitude                     = INDATA (2)           **
C**          XLNG — longitude                    = INDATA (3)           **
C**          IYR — year (yy)                     = INDATA (4)           **
C**          MN — month (mm)                     = INDATA (5)           **
C**          IDA — day  (dd)                     = INDATA (6)           **
C**          IHR — hour (hh)                     = INDATA (7)           **
C**          MIN — mins (mm)                     = INDATA (8)           **
C**          I1 — geomagnetic index              = INDATA (9)           **
C**          F10  —  solar radio noise flux      = INDATA (10)          **
C**          F10B —  162-day average F10          = INDATA (11)         **
C**          GI —  geomagnetic activity index = INDATA (12)             **
C**                                                                     **
C**                                                                     **
C**                        OUTPUT DATA:                                 **
C**                                                                     **
C**          T— exospheric temperature        = OUTDATA (1)             **
C**          TZZ— temperature at altitude Z = OUTDATA (2)               **
C**          A(1)— N2 number density          = OUTDATA (3)             **
C**          A(2)— O2 number density          = OUTDATA (4)             **
C**          A(3)— O  number density          = OUTDATA (5)             **
C**          A(4)— A  number density          = OUTDATA (6)             **
C**          A(5)— He number density          = OUTDATA (7)             **
C**          A(6)— H  number density          = OUTDATA (8)             **
C**          EM— average molecular weight     = OUTDATA (9)             **
C**          DENS— total density              = OUTDATA (10)            **
C**          DL— log10 ( total density )      = OUTDATA (11)            **
C**          P— total pressure                = OUTDATA (12)            **
C**                                                                     **
C**     NB.  Input through array 'INDATA'                               **
C**          Output through array 'OUTDATA'                             **
C**************************************************************************

      DIMENSION  A ( 6 )

      REAL*4  INDATA ( 12 ), OUTDATA ( 12 )
      PARAMETER RGAS = 8.31432E3             !  J/kmol-K
      PARAMETER BFH = 440.0

C Calcultions performed for only one latitude , one longitude
C and one altitude

C
C** Set parameters to INDATA values
C
      Z    = INDATA (1)
      XLAT = INDATA (2)
      XLNG = INDATA (3)
      IYR  = JINT ( INDATA (4) )  + 1900
      MN   = JINT ( INDATA (5) )
      IDA  = JINT ( INDATA (6) )
      IHR  = JINT ( INDATA (7) )
      MIN  = JINT ( INDATA (8) )
      I1   = JINT ( INDATA (9) )
      F10  = INDATA (10)
```

```
      F10B = INDATA (11)
      GI   = INDATA (12)


      CALL TME ( MN , IDA , IYR , IHR , MIN , XLAT , XLNG , SDA ,
     .        SHA , DD , DY )

      CALL TINF ( F10 , F10B , GI , XLAT , SDA , SHA , DY , I1 , TE )

      CALL JAC ( Z , TE , TZ , A(1) , A(2) , A(3) , A(4) , A(5) , A(6) ,
     .        EM , DENS , DL )


              DENLG = 0.
              DUMMY = DL
              DEN   = DL

      IF ( Z .LE. 170. )  THEN
              CALL SLV ( DUMMY , Z , XLAT , DD )
              DENLG = DUMMY
      END IF

C
C** 'Fair' helium number density between base fairing height ( BFH ) and 500 km
C

      IF ( Z. GE. 500. )  THEN
                      CALL SLVH ( DEN , A(5) , XLAT , SDA )
                      DL = DEN
      ELSE IF ( Z .GT. BFH )  THEN
                      DHEL1 = A ( 5 )
                      DHEL2 = A ( 5 )
                      DLG1 = DL
                      DLG2 = DL
              CALL SLVH ( DLG2 , DHEL2 , XLAT , SDA )
                      IH = Z
              CALL FAIR5 ( DHEL1 , DHEL2 , DLG1 , DLG2 , IH , FDHEL , FDLG )
                      DL = FDLG
                      A ( 5 ) = FDHEL
      END IF

                      DL = DL + DENLG
                      DENS = 10.**DL
                      XLAT = XLAT * 57.29577951


C  Fill OUTDATA array
      OUTDATA (1) = TE
      OUTDATA (2) = TZ

              DO 80 I = 1, 6
      OUTDATA (I+2) = 1.E6 * ( 10. ** A(I) )
80            CONTINUE

      OUTDATA (9) = EM
      OUTDATA (10) = DENS * 1000.
      OUTDATA (11) = DL
      P = OUTDATA (10) * RGAS * TZ / EM
      OUTDATA (12) = P

      RETURN
      END
```

```
      SUBROUTINE TME ( MN , IDA , IYR , IHR , MIN , XLAT , XLNG ,
                       SDA , SHA , DD , DY )

C**************************************************************************
C** Subroutine 'TME' performs the calculations of the solar declination **
C** angle and solar hour angle.                                          **
C**                                                                      **
C** INPUTS:  MN   = month                                                **
C**          IDA  = day                                                  **
C**          IYR  = year                                                 **
C**          IHR  = hour                                                 **
C**          MIN  = minute                                               **
C**          XMJD= mean Julian date                                      **
C**          XLAT= latitude ( input-geocentric latitude )                **
C**          XLNG= longitude ( input-geocentric longitude, -180,+180 )   **
C**                                                                      **
C** OUTPUTS: SDA = solar declination angle (rad)                         **
C**          SHA = solar hour angle (rad)                                **
C**          DD  = day number from 1 JAN.                                **
C**          DY  = DD / tropical year                                    **
C**     Modified by Mike Hickey, USRA                                    **
C**************************************************************************

      DIMENSION IDAY(12)

      DATA IDAY / 31,28,31 ,30,31,30 ,31,31,30 ,31,30,31 /
      PARAMETER YEAR = 365.2422
      PARAMETER A1 = 99.6909833 , A2 = 36000.76892
      PARAMETER A3 = 0.00038708 , A4 = 0.250684477
      PARAMETER B1 = 0.0172028  , B2 = 0.0335     , B3 = 1.407
      PARAMETER PI = 3.14159265 , TPI = 6.28318531
      PARAMETER PI2 = 1.57079633 , PI32 = 4.71238898
      PARAMETER RAD_DEG = 0.017453293

      XLAT = XLAT / 57.29577951
      YR = IYR

      IF ( JMOD(IYR,4) .EQ. 0 ) THEN
         IF ( JMOD(IYR,100) .NE. 0 ) IDAY(2) = 29  ! Century not a leap year
      ELSE
         IDAY(2) = 28
      END IF
            ID = 0
      IF ( MN. GT. 1 ) THEN
         DO  20  I = 1 , MN-1
            ID = ID + IDAY(I)
20       CONTINUE
      END IF
            ID = ID + IDA
            DD = ID
            DY = DD/YEAR
C
C** Compute mean Julian date
C
      XMJD = 2415020. + 365. * ( YR - 1900. ) + DD
                      + FLOATJ (  ( IYR - 1901 ) / 4 )
C
C** Compute Greenwich mean time in minutes GMT
C
      XHR = IHR
      XMIN = MIN
      GMT = 60 * XHR + XMIN
      FMJD = XMJD - 2435839. + GMT / 1440.
C
C** Compute Greenwich mean position - GP ( in rad )
C
      XJ = ( XMJD - 2415020.5 ) / ( 36525.0 )
      GP = AMOD ( A1 + A2 * XJ + A3 * XJ * XJ + A4 * GMT , 360. )

C
C** Compute right ascension point - RAP ( in rad )
C
C** 1st convert geocentric longitude to deg longitude -  west neg , + east
```

```
      IF ( XLNG .GT. 180. ) XLNG = XLNG - 360.

      RAP = AMOD ( GP + XLNG , 360. )

C
C** Compute celestial longitude - XLS ( in rad ) - - zero to 2PI
C
      Y1 = B1 * FMJD
      Y2 = 0.017202 * ( FMJD - 3. )
      XLS = AMOD ( Y1 + B2 * SIN(Y2) - B3 , TPI )
C
C** Compute solar declination angle - SDA ( in rad )
C
      B4 = RAD_DEG * ( 23.4523 - 0.013 * XJ )
      SDA = ASIN ( SIN ( XLS ) * SIN ( B4 ) )
C
C** Compute right ascension of Sun - RAS ( in rad ) - - zero to 2PI
C
      RAS = ASIN ( TAN ( SDA ) / TAN ( B4 ) )
C
C** Put RAS in same quadrant as XLS
C
      RAS = ABS ( RAS )
      TEMP = ABS ( XLS )

      IF ( TEMP.LE.PI .AND. TEMP.GT.PI2 )  THEN
             RAS = PI - RAS
      ELSE IF ( TEMP.LE.PI32 .AND. TEMP.GT.PI )  THEN
             RAS = PI + RAS
      ELSE IF ( TEMP.GT.PI32 )  THEN
             RAS = TPI - RAS
      END IF
             IF ( XLS. LT. 0. ) RAS = -RAS

C
C** Compute solar hour angle - SHA ( in deg ) - -
C
      SHA = RAP * RAD_DEG - RAS
      IF ( SHA.GT.PI ) SHA = SHA - TPI
      IF ( SHA.LT.-PI ) SHA = SHA + TPI


      RETURN
      END
```

```
      SUBROUTINE TINF ( F10 , F10B , GI, XLAT, SDA , SHA , DY , I1 , TE )

C*****************************************************************************
C** Subroutine 'TINF' calculates the exospheric temperature according to    **
C** L. Jacchia SAO 313, 1970                                                **
C**                                                                         **
C**   F10 = solar radio noise flux ( x E-22 Watts / m2 )                    **
C**   F10B= 162-day average F10                                             **
C**   GI  = geomagnetic activity index                                      **
C**   LAT = geographic latitude at perigee ( in rad )                       **
C**   SDA = solar declination angle ( in rad )                              **
C**   SHA = solar hour angle                                                **
C**   DY  = D / Y  ( day number / tropical year ) ; 1                       **
C**   I1  = geomagnetic equation index  ( 1--GI=KP , 2--GI=AP )             **
C**   RE  = diurnal factor  KP, F10B, AVG                                   **
C**                                                                         **
C**      CONSTANTS -- C = solar activity variation                          **
C**                -- BETA , etc = diurnal variation                        **
C**                -- D = geomagnetic variation                             **
C**                -- E = semiannual variation                              **
C**                                                                         **
C**  Modified by Mike Hickey, USRA                                          **
C*****************************************************************************

      PARAMETER PI = 3.14159265 , TPI = 6.28318531
      PARAMETER XM = 2.5 , XNN = 3.0
C
C**  Ci are solar activity variation variables
C
      PARAMETER C1 = 383.0 , C2 = 3.32 , C3 = 1.80
C
C**  Di are geomagnetic variation variables
C
      PARAMETER D1 = 28.0 , D2 = 0.03 , D3 = 1.0 , D4 = 100.0 , D5 = -0.08
C
C**  Ei are semiannual variation variables
C
      PARAMETER E1 = 2.41 , E2 = 0.349 , E3 = 0.206 , E4 = 6.2831853
      PARAMETER E5 = 3.9531708 , E6 = 12.5663706 , E7 = 4.3214352
      PARAMETER E8 = 0.1145 , E9 = 0.5 , E10 = 6.2831853
      PARAMETER E11 = 5.9742620 , E12 = 2.16

      PARAMETER BETA = -0.6457718 , GAMMA = 0.7504916 , P = 0.1047198
      PARAMETER RE = 0.31
C
C** solar activity variation
C
      TC = C1 + C2 * F10B + C3 * ( F10 - F10B )
C
C** diurnal variation
C
      ETA    = 0.5 * ABS ( XLAT - SDA )
      THETA  = 0.5 * ABS ( XLAT + SDA )
      TAU    = SHA + BETA + P * SIN ( SHA + GAMMA )

      IF ( TAU. GT. PI ) TAU = TAU - TPI
      IF ( TAU. LT.-PI ) TAU = TAU + TPI

      A1 = ( SIN ( THETA ) )**XM
      A2 = ( COS ( ETA ) )**XM
      A3 = ( COS ( TAU / 2. ) )**XNN
      B1 = 1.0 + RE * A1
      B2 = ( A2 - A1 ) / B1
      TV = B1 * ( 1. + RE * B2 * A3 )
      TL = TC * TV
C
C** geomagnetic variation
C
      IF ( I1.EQ.1 ) THEN
              TG = D1 * GI + D2 * EXP(GI)
      ELSE
              TG = D3 * GI + D4 * ( 1 - EXP ( D5 * GI ) )
      END IF
```

```
C** semiannual variation
C
      G3 = 0.5 * ( 1.0 + SIN ( E10 * DY + E11 ) )
      G3 = G3 ** E12
      TAU1 = DY + E8 * ( G3 - E9 )
      G1 = E2 + E3 * ( SIN ( E4 * TAU1 + E5 ) )
      G2 = SIN ( E6 * TAU1+ E7 )
      TS = E1 + F10B * G1 * G2
C
C** exospheric temperature
C
      TE = TL + TG + TS

      RETURN
      END
```

```
      SUBROUTINE JAC ( Z , T , TZ , AN , AO2 , AO , AA , AHE , AH , EM ,
     .                 DENS , DL )

C*****************************************************************************
C**                                                                         **
C**  Subroutine 'JAC' calculates the temperature TZ , the total density DENS **
C**  and its logarithm DL, the mean molecular weight EM, the individual     **
C**  specie number densities for N, O2, O, A, HE and H ( each preceded with **
C**  an 'A' ) at altitude Z given the exospheric temperature T.             **
C**  This subroutine uses the subroutine 'GAUSS' and the function           **
C**  subprograms 'TEMP' and 'MOL_WT'.                                       **
C**                                                                         **
C**   Rewritten by Mike Hickey, USRA                                        **
C*****************************************************************************

      DIMENSION  ALPHA(6) , EI(6) , DI(6) , DIT(6)
      REAL*4  MOL_WT

      PARAMETER AV = 6.02257E23
      PARAMETER QN   = .78110
      PARAMETER QO2  = .20955
      PARAMETER QA   = .009343
      PARAMETER QHE  = 1.289E-05
      PARAMETER RGAS = 8.31432
      PARAMETER PI   = 3.14159265
      PARAMETER T0   = 183.

      GRAVITY ( ALTITUDE ) = 9.80665 / ( ( 1. + ALTITUDE / 6.356766E3 )**2 )


      DATA ALPHA / 0.0 , 0.0 , 0.0 , 0.0 , -.380 , 0.0 /
      DATA EI / 28.0134 , 31.9988 , 15.9994 , 39.948 , 4.0026 , 1.00797 /


      TX = 444.3807 + .02385 * T - 392.8292 * EXP ( -.0021357 * T )
      A2 = 2. * (T-TX) / PI
      TX_T0 = TX - T0
      T1 =  1.9 * TX_T0 /   35.
      T3 = -1.7 * TX_T0 / ( 35.**3 )
      T4 = -0.8 * TX_T0 / ( 35.**4 )
      TZ = TEMP ( Z , TX , T1 , T3 , T4 , A2 )


C**     SECTION 1
C**     ---------

      A = 90.
      D = AMIN1 ( Z , 105. )

C  Integrate  gM/T  from  90  to  minimum of Z or 105 km :-

      CALL GAUSS ( A, D, 1, R, TX , T1 , T3 , T4 , A2 )

C  The number 2.1926E-8 = density x temperature/mean molecular weight at 90 km.

      EM = MOL_WT ( D )
      TD = TEMP ( D , TX , T1 , T3 , T4 , A2 )

      DENS = 2.1926E-8 * EM * EXP( -R / RGAS ) / TD

      FACTOR = AV * DENS
      PAR = FACTOR / EM
      FACTOR = FACTOR / 28.96


C  For altitudes below and at 105 km calculate the individual specie number
C  densities from the mean molecular weight and total density.

      IF ( Z. LE. 105 )  THEN

      DL = ALOG10 ( DENS )
      AN  = ALOG10 ( QN * FACTOR )
```

```
        AA  = ALOG10 ( QA * FACTOR )
        AHE = ALOG10 ( QHE * FACTOR )
        AO  = ALOG10 ( 2. * PAR * ( 1.-EM / 28.96 ) )
        AO2 = ALOG10 ( PAR * ( EM * ( 1.+QO2 ) / 28.96-1. ) )
        AH  = 0.
C
C** Return to calling program
C
        RETURN


        END IF


C**     SECTION 2  :  This section is only performed for altitudes above 105 km
C**     ---------

C  Note that having reached this section means that D in section 1 is 105 km.
C  ----

C  Calculate individual specie number densities from the total density and mean
C  molecular weight at 105 km altitude.

                DI(1) = QN * FACTOR
                DI(2) = PAR * (EM * (1.+QO2) / 28.96-1.)
                DI(3) = 2. * PAR * (1.- EM / 28.96)
                DI(4) = QA * FACTOR
                DI(5) = QHE * FACTOR

C  Integrate  g/T  from  105 km  to  Z km :-

        CALL GAUSS ( D, Z, 2, R, TX , T1 , T3 , T4 , A2 )

                DO  41  I = 1 , 5
        DIT(I) = DI(I) * ( TD / TZ ) **(1.+ALPHA(I)) * EXP( -EI(I) * R / RGAS)
                IF ( DIT(I). LE. 0. ) DIT(I) = 1.E-6
41              CONTINUE



C**  This section calculates atomic hydrogen densities above 500 km altitude.
C**  Below this altitude , H densities are set to 10**-6.

C**     SECTION 3
C**     ---------

        IF ( Z .GT. 500. )  THEN

                A1 = 500.
                S = TEMP ( A1 , TX , T1 , T3 , T4 , A2 )

        DI(6) = 10.** ( 73.13 - 39.4 * ALOG10 (S) + 5.5 * ALOG10(S) *ALOG10(S))

        CALL GAUSS ( A1, Z, 7, R, TX , T1 , T3 , T4 , A2 )

                DIT(6) = DI(6) * (S/TZ) * EXP ( -EI(6) * R / RGAS )

        ELSE

                DIT (6) = 1.0

        END IF


C   For altitudes greater than 105 km , calculate total density and mean
C   molecular weight from individual specie number densities.

                       DENS=0
                DO  42  I = 1 , 6
                       DENS = DENS + EI(I) * DIT(I) / AV
 42             CONTINUE
```

```
EM = DENS * AV / ( DIT(1)+DIT(2)+DIT(3)+DIT(4)+DIT(5)+DIT(6) )
DL = ALOG10 (DENS)

AN  = ALOG10(DIT(1))
AO2 = ALOG10(DIT(2))
AO  = ALOG10(DIT(3))
AA  = ALOG10(DIT(4))
AHE = ALOG10(DIT(5))
AH  = ALOG10(DIT(6))


RETURN
END
```

```
      FUNCTION  TEMP ( ALT , TX , T1 , T3 , T4 , A2 )

C*****************************************************************************
C**                                                                         **
C**  Function subprogram 'TEMP' calculates the temperature at altitude ALT  **
C**  using equation (10) for altitudes between 90 and 125 km and equation   **
C**  (13) for altitudes greater than 125 km , from SAO Report 313.          **
C**                                                                         **
C**  Written by Mike Hickey, USRA                                           **
C*****************************************************************************

      PARAMETER BB = 4.5E-6

            U = ALT - 125.
      IF ( U .GT . 0. )  THEN
            TEMP = TX  +  A2 * ATAN ( T1 * U * ( 1. + BB * (U**2.5)) / A2 )
      ELSE
            TEMP = TX  +  T1 * U  +  T3 * (U**3)  +  T4 * (U**4)
      END IF

      END



      REAL FUNCTION MOL_WT*4 ( A )

C*****************************************************************************
C**                                                                         **
C**  Subroutine 'MOL_WT' calculates the molecular weight for altitudes      **
C**  between 90 and 105 km according to equation (1) of SAO report 313.     **
C**  Otherwise, MOL_WT is set to unity.                                     **
C**                                                                         **
C**  Written by Mike Hickey, USRA                                           **
C*****************************************************************************

      DIMENSION  B (7)

      DATA B / 28.15204 , -0.085586, 1.284E-4, -1.0056E-5, -1.021E-5,
              1.5044E-6, 9.9826E-8 /

      IF ( A. GT. 105. ) THEN

            MOL_WT = 1.

      ELSE

            U = A - 100.
            MOL_WT = B (1)
      DO  1  I = 2 , 7

      MOL_WT = MOL_WT  +  B (I) * U ** ( I-1 )

 1    CONTINUE

      END IF

      END
```

```
      SUBROUTINE GAUSS ( Z1 , Z2 , NMIN , R , TX , T1 , T3 , T4 , A2 )
C*****************************************************************************
C**  Subdivide total integration-altitude range into intervals suitable for  **
C**  applying Gaussian Quadrature , set the number of points for integration **
C**  for each sub-interval , and then perform Gaussian Quadrature.            **
C**       Written by Mike Hickey, USRA, NASA/MSFC, ED44, July 1988.           **
C*****************************************************************************

      REAL*4   ALTMIN (9) , C(8,6), X(8,6), MOL_WT
      INTEGER  NG (8) , NGAUSS , NMIN , J

      GRAVITY ( ALTITUDE ) = 9.80665 / ( ( 1. + ALTITUDE / 6.356766E3 )**2 )

      DATA  ALTMIN / 90., 105., 125., 160., 200., 300., 500., 1500., 2500. /
      DATA    NG   / 4 , 5  ,  6  ,  6  ,  6  ,  6  ,  6  , 6 /

C  Coefficients for Gaussian Quadrature ...

      DATA  C / .5555556 , .8888889 , .5555556 , .0000000 , ! n=3
     .          .0000000 , .0000000 , .0000000 , .0000000 , ! n=3
     .          .3478548 , .6521452 , .6521452 , .3478548 , ! n=4
     .          .0000000 , .0000000 , .0000000 , .0000000 , ! n=4
     .          .2369269 , .4786287 , .5688889 , .4786287 , ! n=5
     .          .2369269 , .0000000 , .0000000 , .0000000 , ! n=5
     .          .1713245 , .3607616 , .4679139 , .4679139 , ! n=6
     .          .3607616 , .1713245 , .0000000 , .0000000 , ! n=6
     .          .1294850 , .2797054 , .3818301 , .4179592 , ! n=7
     .          .3818301 , .2797054 , .1294850 , .0000000 , ! n=7
     .          .1012285 , .2223810 , .3137067 , .3626838 , ! n=8
     .          .3626838 , .3137067 , .2223810 , .1012285 / ! n=8

C  Abscissas for Gaussian Quadrature ...

      DATA  X / -.7745967 ,  .0000000 ,  .7745967 ,  .0000000 , ! n=3
     .           .0000000 ,  .0000000 ,  .0000000 ,  .0000000 , ! n=3
     .          -.8611363 , -.3399810 ,  .3399810 ,  .8611363 , ! n=4
     .           .0000000 ,  .0000000 ,  .0000000 ,  .0000000 , ! n=4
     .          -.9061798 , -.5384693 ,  .0000000 ,  .5384693 , ! n=5
     .           .9061798 ,  .0000000 ,  .0000000 ,  .0000000 , ! n=5
     .          -.9324695 , -.6612094 , -.2386192 ,  .2386192 , ! n=6
     .           .6612094 ,  .9324695 ,  .0000000 ,  .0000000 , ! n=6
     .          -.9491079 , -.7415312 , -.4058452 ,  .0000000 , ! n=7
     .           .4058452 ,  .7415312 ,  .9491079 ,  .0000000 , ! n=7
     .          -.9602899 , -.7966665 , -.5255324 , -.1834346 , ! n=8
     .           .1834346 ,  .5255324 ,  .7966665 ,  .9602899 / ! n=8

              R  =  0.0

      DO   2   K = NMIN , 8

              NGAUSS = NG (K)
              A      = ALTMIN (K)
              D      = AMIN1 ( Z2 , ALTMIN (K+1) )
              RR   = 0.0
              DEL = 0.5 * ( D - A )
              J   = NGAUSS - 2

      DO   1   I = 1 , NGAUSS

              Z = DEL * ( X(I,J) + 1. ) + A
      RR = RR + C(I,J) * MOL_WT(Z) * GRAVITY(Z) / TEMP ( Z,TX,T1,T3,T4,A2 )

1     CONTINUE

              RR = DEL * RR
              R  = R + RR
              IF ( D .EQ. Z2 )  RETURN

2     CONTINUE

      RETURN
      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      SUBROUTINE SLV ( DEN , ALT , XLAT , DAY )

C***************************************************************************
C** Subroutine 'SLV' computes the seasonal-latitudinal variation of density  **
C** in the lower thermosphere in accordance with L. Jacchia, SAO 332, 1971.  **
C** This affects the densities between 90 and 170 km.  This subroutine need  **
C** not be called for densities above 170 km, because no effect is observed. **
C**                                                                           **
C** The variation should be computed after the calculation of density due to **
C** temperature variations and the density ( DEN ) must be in the form of a  **
C** base 10 log.  No adjustments are made to the temperature or constituent  **
C** number densities in the region affected by this variation.               **
C**                                                                           **
C**                        DEN    = density (log10)                           **
C**                        ALT    = altitude (km)                             **
C**                        XLAT   = latitude (rad)                            **
C**                        DAY    = day number                                **
C**                                                                           **
C***************************************************************************

C** initialize density (DEN) = 0.0
C
       DEN = 0.0
C
C** check if altitude exceeds 170 km
C
       IF ( ALT. GT. 170. ) RETURN
C
C** compute density change in lower thermosphere
C
       Z = ALT - 90.
       X = -0.0013 * Z * Z
       Y = 0.0172 * DAY + 1.72
       P = SIN (Y)
       SP = ( SIN (XLAT) ) **2
       S = 0.014 * Z * EXP (X)
       D = S * P * SP
C
C** check to compute absolute value of 'XLAT'
C
       IF ( XLAT. LT. 0. ) D = -D
       DEN = D

       RETURN
       END
```

```
      SUBROUTINE SLVH ( DEN , DENHE , XLAT , SDA )

C***************************************************************************
C** Subroutine 'SLVH' computes the seasonal-latitudinal varaition of the    **
C** helium number density according to L. Jacchia, SAO 332, 1971.  This    **
C** correction is not important below about 500 km.                        **
C**                                                                         **
C**                     DEN   = density (log10)                             **
C**                     DENHE = helium number density (log10)               **
C**                     XLAT  = latitude (rad)                              **
C**                     SDA   = solar declination angle (rad)               **
C***************************************************************************

       D0 = 10. ** DENHE
       A = ABS ( 0.65 * ( SDA / 0.40909079 ) )

       B  = 0.5 * XLAT
C
C** Check to compute absolute value of 'B'
C
       IF ( SDA. LT. 0. ) B = -B
C
C** compute X, Y, DHE and DENHE
C
       X  = 0.7854 - B
       Y  = ( SIN (X) ) ** 3
       DHE= A * ( Y - 0.35356 )
       DENHE = DENHE + DHE
C
C** compute helium number density change
C
       D1 = 10. ** DENHE
       DEL= D1 - D0
       RHO= 10. ** DEN
       DRHO = ( 6.646E-24 ) * DEL
       RHO  = RHO + DRHO
       DEN  = ALOG10 (RHO)

       RETURN
       END
```

```
      SUBROUTINE FAIR5 ( DHEL1 ,DHEL2 ,DLG1 ,DLG2 ,IH ,FDHEL ,FDLG )

C******************************************************************************
C** This subroutine fairs between the region above 500 km, which invokes the **
C** seasonal-latitudinal variation of the helium number density ( subroutine **
C** SLVH ), and the region below, which does not invoke any seasonal-        **
C** latitudinal variation at all.                                            **
C**                                                                          **
C** INPUTS:  DHEL1 = helium number density before invoking SLVH              **
C**          DHEL2 = helium number density after invoking SLVH               **
C**          DLG1  = total density before invoking SLVH                      **
C**          DLG2  = total density after invoking SLVH                       **
C**          IH    = height  ( km ) -- INTEGER                               **
C**          IBFH  = base fairing height ( km ) -- INTEGER                   **
C** OUTPUTS: FDHEL = faired helium number density                            **
C**          FDLG  = faired total density                                    **
C**                                                                          **
C** Written by Bill Jeffries, CSC, Huntsville, AL.                           **
C**            ph. (205) 830-1000, x311                                      **
C******************************************************************************

      DIMENSION CZ ( 6 )
      DATA CZ / 1.0, 0.9045085, 0.6545085, 0.3454915, 0.0954915, 0.0 /
      PARAMETER IBFH = 440

C  Height index
      I = ( IH - IBFH ) /10 + 1
C  Non-SLVH fairing coefficient
      CZI = CZ ( I )
C  SLVH fairing coefficient
      SZI = 1.0 - CZI
C  Faired density
      FDLG = ( DLG1 * CZI ) + ( DLG2 * SZI )
C  Faired helium number density
      FDHEL = ( DHEL1 * CZI ) + ( DHEL2 * SZI )

      RETURN
      END
```