The Mars Thermal Model (MARSTHERM) is a FORTRAN 77 program that uses the method of finite differences to compute surface and subsurface temperature variations at any given latitude throughout the martian year. For the sake of computational efficiency, the program utilizes non-constant intervals for both depth and time, increasing program execution speed with only a small loss in numerical accuracy. With only minor changes, the program should be readily adaptable to investigating the thermal behavior of most other planetary bodies.

SUBROUTINE FINDT

The subroutine FINDT determines the temperature distribution of the martian regolith as a function of depth. These temperature calculations are based upon the given physical properties and the value of the surface temperature that is passed to FINDT from the subroutine FNTEMP. The default values chosen for the surface albedo, soil thermal conductivity, density, and specific heat are the same as those adopted for the Standard Viking Thermal Model (Kieffer et al., 1977).

After initializing the variables, a basic time interval is chosen, and the depth interval for the first compartment is determined from that. At all times the time and depth increments must meet the stability criteria:

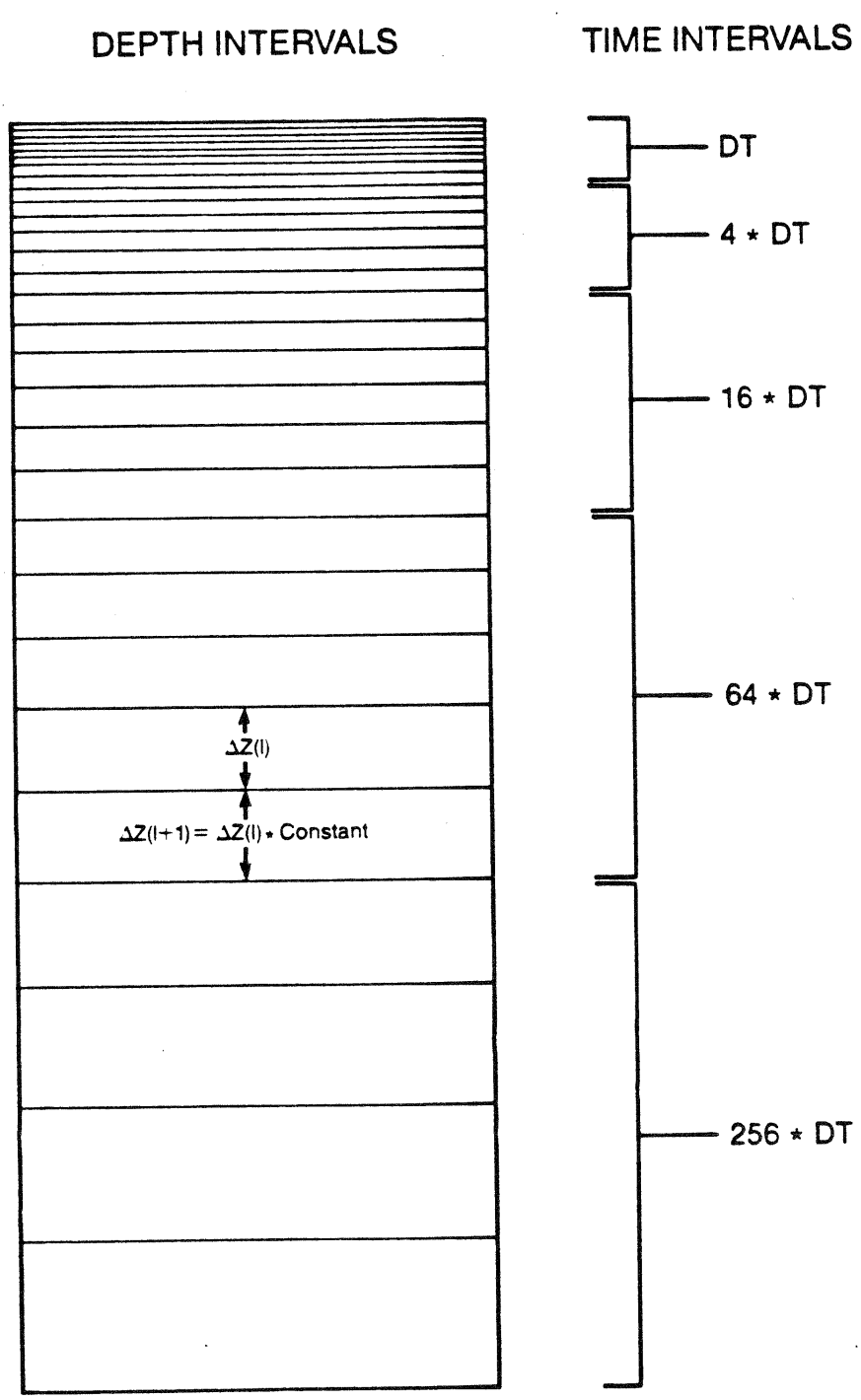
$$(K*DT)/(C*RHO*DZ^2)<=0.25 \qquad (1)$$

where K is the thermal conductivity, DT is the time increment, C is the specific heat, RHO is the density and DZ is the depth increment. [Note that some authors indicate a numerical stability criteria of 0.5 instead of 0.25. For values between 0.25 and 0.5, the solution will

converge but in an oscillatory fashion. However, for values <= 0.25, the solution is both convergent and non-oscillatory (James et al., 1967; p. 479).] Possible complications arising from an inappropriate choice of compartment size and time interval can be illustrated by considering the result of having both a very small compartment and a very long time interval between calculations. Such a combination artificially permits the accumulation of a large quantity of heat within the compartment, thus leading to an excessively high and unrealistic temperature that may be only partially relieved by conduction during the subsequent time step.

The second and third compartments are given the same thickness as the first in order to simplify the treatment of the first derivative term included in the finite-difference version of the surface boundary condition equation in subroutine FNTEMP (See equations 6 and 13). Following the lead of Kieffer et al. (1977), the remaining compartment thicknesses are chosen such that each is a multiple of the previous one (i.e., the fourth compartment is 1.13 times as thick as the third, the fifth 1.13 times as thick as the fourth, etc.).

The stability criteria given by Eq. 1 permits an increase in the time between calculations as compartment thicknesses become larger with depth, therefore, substantial economies in time can be achieved by increasing the time intervals accordingly. The time intervals are chosen such that each is four times the duration of the previous one (i.e., DT, 4*DT, 16*DT, etc.). For a given compartment, the appropriate time interval between successive iterations is the largest of these calculated intervals that satisfies the stability criteria. Those compartments which share the same time increment constitute a


DEPTH INTERVALS
TIME INTERVALS
DT
4 * DT
16 * DT
64 * DT
256 * DT
ΔZ(I)
ΔZ(I+1) = ΔZ(I) * Constant

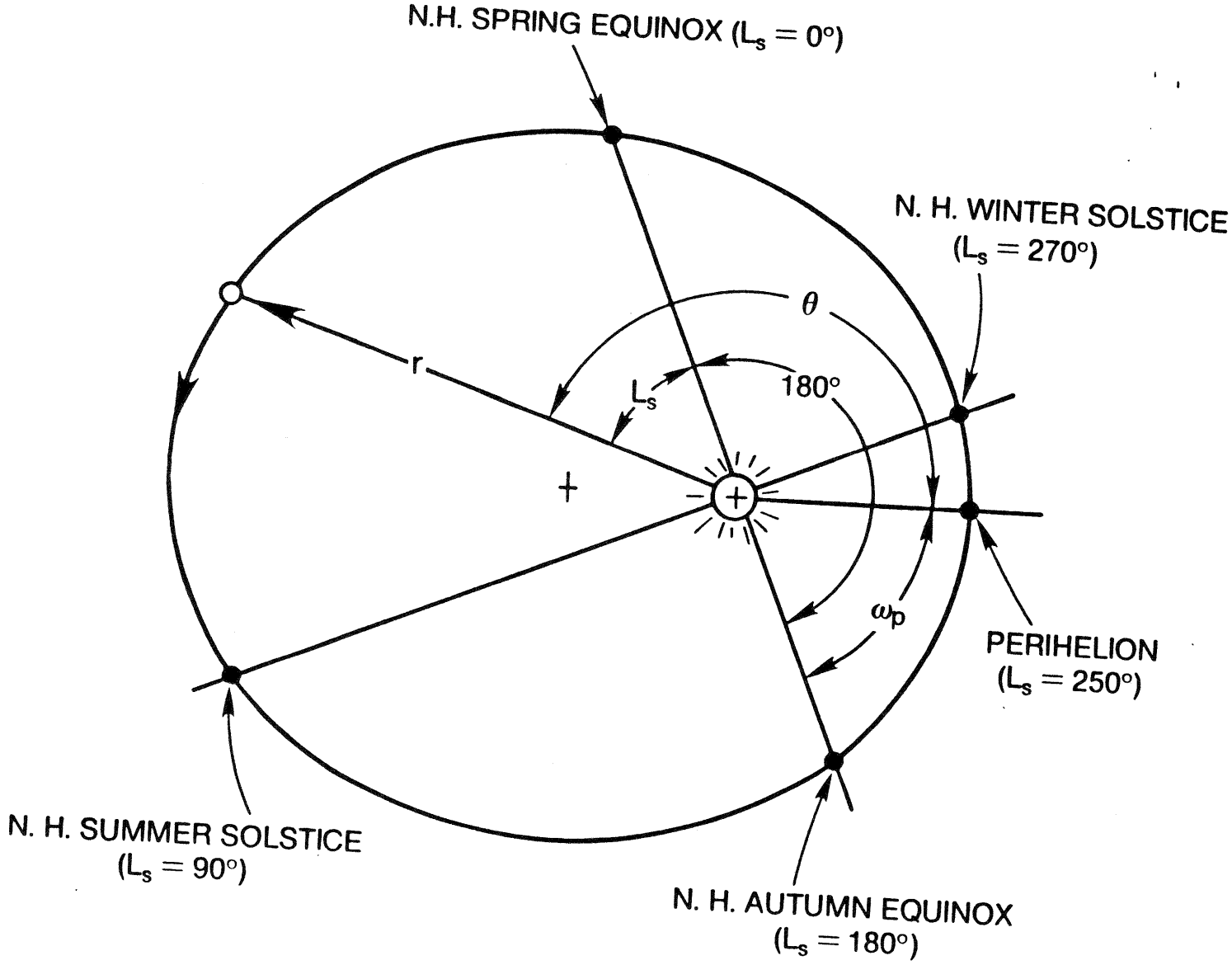
level. By assigning the time intervals in this way, the compartment temperatures in one level are always updated simultaneously with those in all higher levels. In terms of performance, when compared with a double-precision 400-compartment constant-time and -depth interval model (Appendix A), the CPU time required to run the non-constant interval model amounts to an 88% speed improvement, with a maximum discrepancy in the final numerical results of less than two percent.

Compartment temperatures are initialized by calculating the average annual equilibrium surface temperature. This temperature is calculated on the basis of the previously selected values of latitude (LAT), albedo (A, where .25 is the default value), emissivity (E), and the annually averaged solar flux given by

$$<S>=[SO/4*(1-ECCEN^2)^{1/2}]((1.5-(2*SIN(OBLIQ))/PI)-(1.5-(6*SIN(OBLIQ))/PI)*SIN^2(LAT)) \quad (2)$$

where SO is the solar flux at the semi-major axis of Mars (a), ECCEN is the eccentricity, OBLIQ is the axial inclination of Mars, and LAT is the latitude (Hoffert et al., 1981). After all compartments are initialized to this temperature, the main loop of the program is entered.

The main loop first determines to what depth compartment temperatures will be updated during the current iteration. The first level compartment temperatures are recalculated with each iteration (i.e. each time step DT). Every fourth iteration these calculations are extended to include the compartments of the second level (those with a time interval of 4*DT), and every sixteenth iteration, those of the third level. This method is continued for the remaining levels until by the sixth and final level, temperatures are recalculated only


N.H. SPRING EQUINOX ($L_s = 0°$)
N. H. WINTER SOLSTICE
($L_s = 270°$)
θ
r
L_s
180°
ω_p
PERIHELION
($L_s = 250°$)
N. H. SUMMER SOLSTICE
($L_s = 90°$)
N. H. AUTUMN EQUINOX
($L_s = 180°$)

once every 1024 iterations. For a given iteration, the decision to extend the numerical calculations to a deeper level is made through the use of a multiple IF-THEN-ELSE statement. The MOD(I,4) statement finds the remainder when I is divided by four and therefore determines whether the loop has been repeated some multiple of four times. If not, only the first level is recalculated. If so, the IF statement check whether the loop has been repeated some multiple of sixteen times, and so on.

The temperatures of the compartments are found using the basic one-dimensional heat conduction equation:

$$dT/dt=k/(rho*c)*d^2T/dz^2 \qquad (3)$$

where T is the temperature of the compartment,t is time, z is the depth, k is the thermal conductivity, rho is the density and c is the specific heat. Finite difference expressions are used for the two derivatives. The expression for dT/dt is found from a Taylor expansion of T(t+dt):

$$T(t+dt)=T(t)+dt*(dT/dt)+(dt)^2/2*(d^2T/dt^2)+...$$

Ignoring all terms of order 2 or higher, we obtain:

$$dT/dt=(T(t+dt)-T(t))/dt. \qquad (4)$$

From Sundqvist and Veronis (1969) we obtained for the second derivative the following expression:

$$(dT/dz_{i+1/2}-dT/dz_{i-1/2})/(0.5*(dz_i-dz_{i-1})) \qquad (5)$$

where

$$dT/dz_{i+1/2}=(T_{i+1}-T_i)/dz_i$$

and

$$dT/dz_{i-1/2}=(T_i-T_{i-1})/dz_{i-1}$$

The expressions for the derivatives are substituted into the heat

equation which is then solved for T(t+dt) (or TEMPDT as it is called in the program) for each of the compartments. Because of the T_{i+1} term in the second derivative, the temperature in the lowest compartment cannot be calculated using this equation since there obviously is no compartment after it. To allow constant readjustment of the lower boundary, the last compartment is assigned the same temperature as the second to last following each iteration (James et al., 1967). This should cause little error provided that the temperatures in the last two compartments are approximately the same. After this last temperature calculation has been made, the time is incremented by the interval dt.

If the elapsed time is greater than or equal to the time at which a printout of the results is required, the subroutine PRINT is called. No permanent record of the results is kept, in order to reduce the required memory space. Results that are not printed out are lost after being used to compute the next set of temperatures. All the new values that have been calculated and stored in the array TEMPDT are then passed to the array TEMP, the subroutine FNTEMP is called to determine the new surface temperature, and then the loop is repeated.

SUBROUTINE FNTEMP

Subroutine FNTEMP determines the surface temperature based upon the time of day and the position of Mars in its orbit and returns that value to subroutine FINDT. The values for the orbital eccentricity and obliquity are from Ward (1979).

FNTEMP first calls the subroutine FNANOM, which computes the appropriate values of the declination and the true anomaly for that

time. With these values, FNTEMP then determines the surface temperature from the equation:

$$S(1-ALBEDO)+K*dT/dz+FF+L*dm/dt=E*STEFAN*T^4 \quad (6)$$

(Kieffer et al., 1977) where S is the solar flux at the position of Mars in its orbit, K is the thermal conductivity, dT/dz is the change in temperature with depth evaluated at the surface, FF is the downward component of atmospheric radiation, L is the latent heat of vaporization of CO_2, dm/dt is the change in the mass of the CO_2, E is the emissivity (which,after Kieffer et al., (1977), is taken to be 1.0), and STEFAN is the Stefan-Boltzmann constant.

The incident flux is found from the equation $S=S_0\cos(i)$, where i is the angle of incidence of the incoming sunlight (measured from the zenith), and where S_0 is given by:

$$S_0=L/(4*pi*R^2) \quad (7)$$

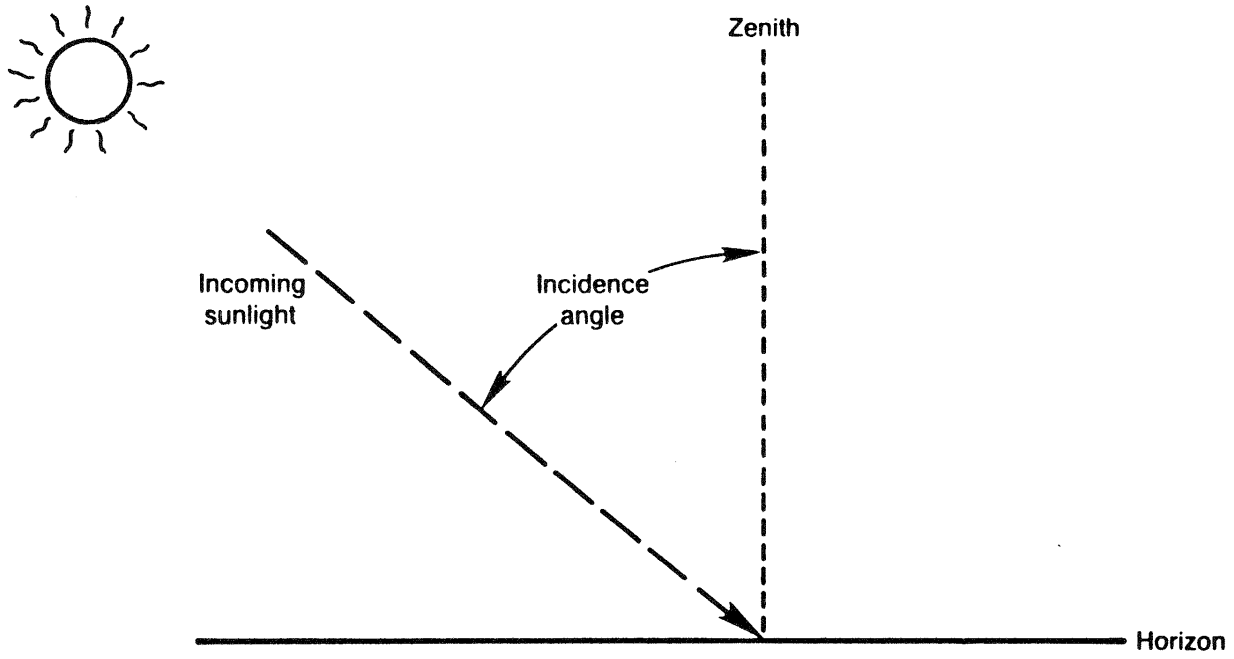
where L is the luminosity of the sun and R is the instantaneous orbital distance of Mars. In the program, the constant SO is equal to S_0 evaluated at a distance equivalent to the semimajor axis, a, of Mars. Thus the solar flux at any other point in the planet's orbit is given by the expression $SO*a^2/R^2$. The ratio a^2/R^2 is called ORBIT in the program, and is given by:

$$ORBIT=(1+ECCEN*COS(ANOM))^2/(1+ECCEN^2)^2 \quad (8)$$

where ECCEN is the orbital eccentricity and ANOM is the true anomaly (Haymes, 1971, p.500). The cosine of the incidence angle (called SUNPOS in the program) is given by the equation:

$$SUNPOS=SIN(LAT)SIN(DEC)+COS(LAT)COS(DEC)COS(H) \quad (9)$$

where LAT is the latitude, DEC is the declination and H is the hour angle (Barkstrom,1981). H is calculated from the expression:


Zenith
Incoming
sunlight
Incidence
angle
Horizon

H=PI*(ABS(1-(T-DAY*SOL)/12*HOUR)) (10)

where T is the time elapsed, DAY is the number of days elapsed, SOL is the length of a martian day and HOUR is the length of a martian hour. The value of H at sunrise and sunset (called HRZN) is also calculated. When the sun is at the horizon, the angle of incidence is 90 degrees; therefore, since cos(90)=0, Eq. 9 can be solved for HRZN giving:

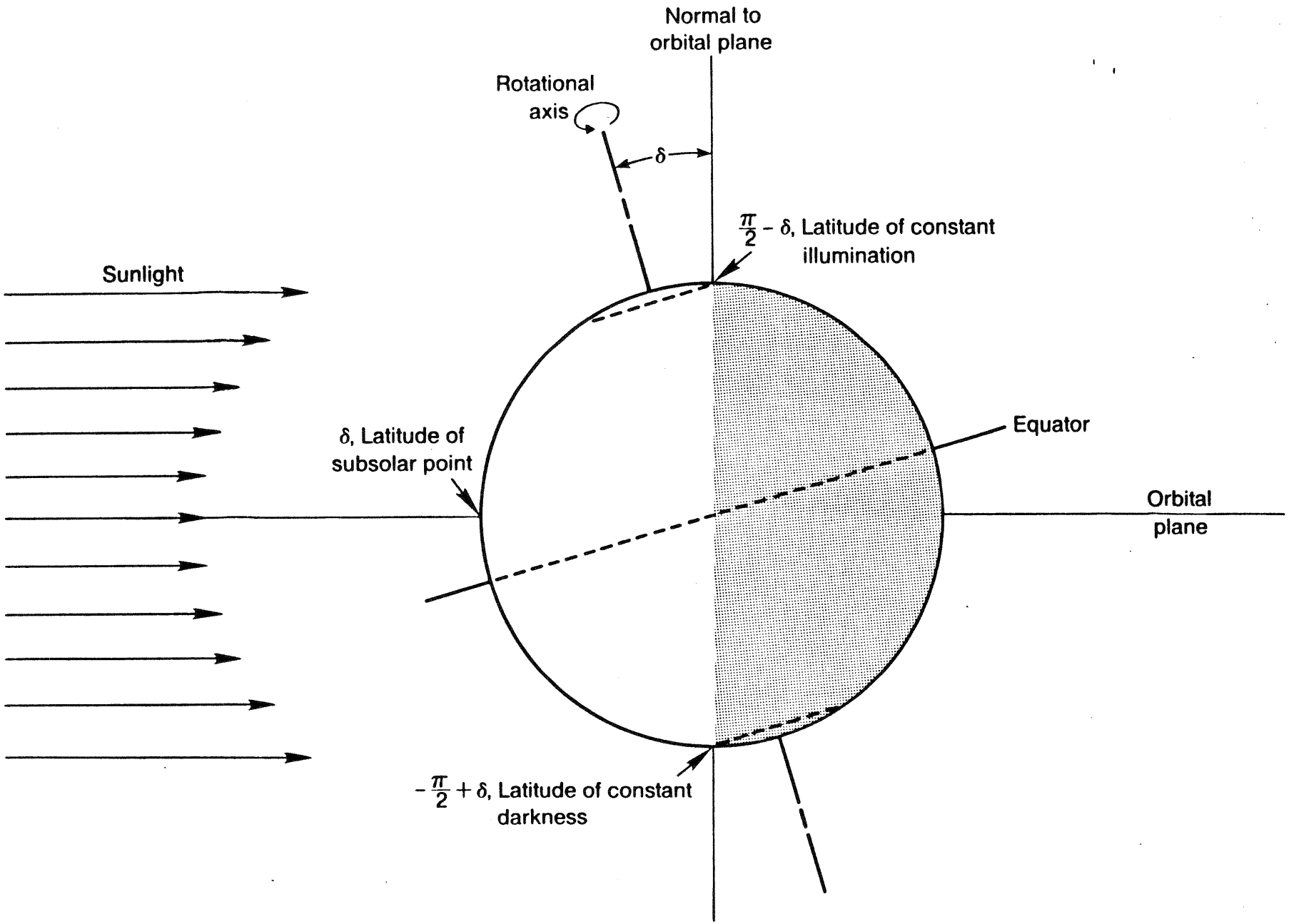
LAT-β for f_{ac}

LAT+β for T_s

HRZN=ACOS(-TAN(DEC)*TAN(LAT)) (11)

Two IF statements are also included to set HRZN equal to PI if the latitude is such that the sun never rises and to set HRZN equal to 0 if the latitude is such that the sun never sets. If the value of the hour angle is greater than that of HRZN (i.e. the sun is below the horizon) the solar flux is automatically set equal to zero. The net insolation at the martian surface is found from the equation INSOL=S*(1-ALBEDO), where the value chosen for ALBEDO is defined by the user (but which has a default value of 0.25 (Kieffer et al., 1977)). If the surface temperature falls below 149 K, CO_2 begins to condense from the atmosphere, whereupon the albedo jumps instantaneously to 0.65 (Kieffer et al., 1977). The value for FF (which has been dubbed the fudge factor) is also calculated at this time. As noted earlier, FF accounts for heat that is radiated from the atmosphere. It is set equal to two percent of the value of the insolation at noon (Kieffer, et al, 1977).

The heat that is conducted from (or to) the martian surface ($k*dT_1/dz$) is calculated using a finite difference approximation for the derivative. As before, Taylor expansions are used for the temperatures in the second and third compartments. The expressions are:


Normal to
orbital plane
Rotational
axis
δ
$\frac{\pi}{2} - \delta$, Latitude of constant
illumination
Sunlight
δ, Latitude of
subsolar point
Equator
Orbital
plane
$-\frac{\pi}{2} + \delta$, Latitude of constant
darkness

$$T(z+dz)=T(z)+dz*dT_1/dz+dz^2/2*d^2T_1/dz^2+...$$

$$T(z+2*dz)=T(z)+(2*dz)*dT_1/dz+(2*dz)^2/2*d^2T_1/dz^2+...$$

Substituting T_1 for T(z), T_2 for T(z+dz), and T_3 for T(z+2*dz) gives:

$$T_2=T_1+dz*dT_1/dz+dz^2/2*d^2T_1/dz^2+...$$

$$T_3=T_1+(2*dz)*dT_1/dz+(2*dz)^2/2*d^2T_1/dz^2+... \quad (12)$$

By multiplying the first equation by four, subtracting the second and ignoring all terms of third order or higher, one obtains:

$$dT_1/dz=(4*T_2-T_3-3*T_1)/(2*dz). \quad (13)$$

The surface boundary condition (Eq. 6) is then solved for the surface temperature using Newton's method. The values of the equation and its derivative when all terms are pulled to one side and an initial value of the surface temperature are sent to subroutine NEWTON. The process is repeated with the new values calculated by subroutine NEWTON for the surface temperature until the estimated value given by Newton's method is sufficiently close to the preceeding value to assume that it is correct (the program uses a difference of 0.00001). The program then checks whether the latent heat term will be involved. If so, the temperature at the surface is set to 149 degrees, because once the CO_2 snow begins to form, the surface will remain at that temperature until all the CO_2 finally sublimates away. To determine the mass of CO_2 that either condenses or sublimates during the time interval DT, the surface energy balance is solved for dm (kg CO_2 m^{-3}). This amount is then added to the total mass of CO_2 that has already accumulated. Control is then passed back to the subroutine FINDT along with the most recently calculated value of the surface temperature.

SUBROUTINE FNANOM

Subroutine FNANOM calculates the declination of the sun in the sky and the true anomaly of the planet's position in its orbit. It also calculates the corresponding aerocentric longitude for identifying the orbital position of Mars at any given instant. The true anomaly can be found from the equation:

$$TAN(ANOM/2)=((1+ECCEN)/(1-ECCEN))^{1/2}*TAN(EA/2) \quad (14)$$

where ANOM is the true anomaly, ECCEN is the eccentricity and EA is the eccentric anomaly (Haymes, 1971). The eccentric anomaly is found by using Newton's method to solve the equation:

$$MEAN=EA-ECCEN*SIN(EA) \quad (15)$$

where MEAN (the mean anomaly) is used as the initial value of the eccentric anomaly (Barkstrom, 1981). The mean anomaly is found from the equation:

$$MEAN=(2*PI/PERIOD)*(T-TAU) \quad (16)$$

where T is the current time and TAU is the time of perihelion in seconds (Haymes, 1971). Since no value of TAU is specified in the program, the calculations begin with Mars at perihelion.

After both the eccentric and true anomalies have been calculated, the declination of the sun can be found from:

$$SIN(DEC)=SIN(OBLIQ)*SIN(ANOM+OMEGA-PI) \quad (17)$$

where DEC is the declination of the sun, OBLIQ is the obliquity of Mars (the angle between its axis of rotation and the perpendicular to the plane of orbit), and OMEGA is the angular distance between the northern hemisphere autumn equinox and perihelion (Hoffert et al., 1981).

L_s, the aerocentric longitude of the sun, is given by:

LS=(ANOM+PI+OMEGA)*180/PI-(REVS)*360 (18)

The term 180/PI merely converts the L_s from radians to degrees, and subtracting the number of orbital revolutions (REVS) multiplied by 360 keeps L_s between zero and 360 degrees. The values of the true anomaly, declination, and L_s are then passed back to the subroutine FNTEMP.

SUBROUTINE NEWTON

This subroutine solves an equation for a given variable using Newton's method. The value of the equation after all terms have been placed on one side, the value of its derivative, and an initial value for the variable are sent to the subroutine. The subroutine then estimates a new value using the equation:

GUESS=VALUE-X/DX (19)

where X is the equation and DX is the derivative of the equation. If the estimated value is close enough to the initial value (the program currently uses 1E-5), the variable REPEAT is assigned the value of false so that the procedure will not be repeated. The value of the variable is then set equal to the guessed value and that value is returned to the calling subroutine.

SUBROUTINE PRINT

This subroutine prints out the table of results. It can be modified to print out any of the variables in the program. The first time the subroutine is called, the headings for the table of results are also printed, and then the value of FIRST is set to false, to prevent the headings from being printed again. After printing out the

values requested, it calculates the time for the next printout and returns control to subroutine FINDT.

References

Barkstrom, B., 1981. What Time Does the Sun Rise and Set, Byte, pp. 94-114.

Haymes, R.C., 1971. Introduction to Space Science, John Wiley and Sons, Inc., p. 500.

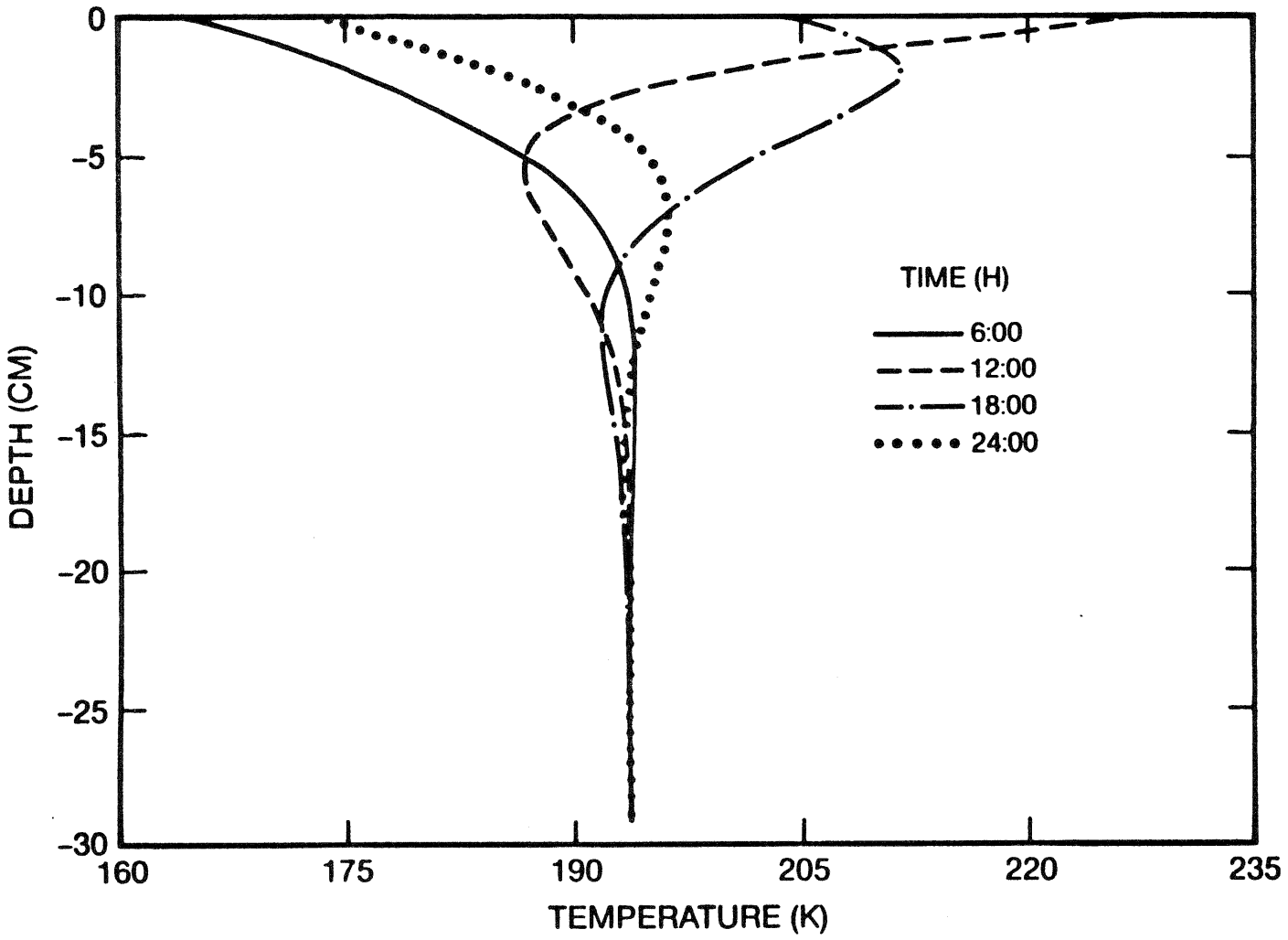
Hoffert, M.I. et al., 1981. Liquid Water on Mars: An Energy Balance Climate Model for CO_2/H_2O Atmospheres, Icarus 47, pp. 112-129.

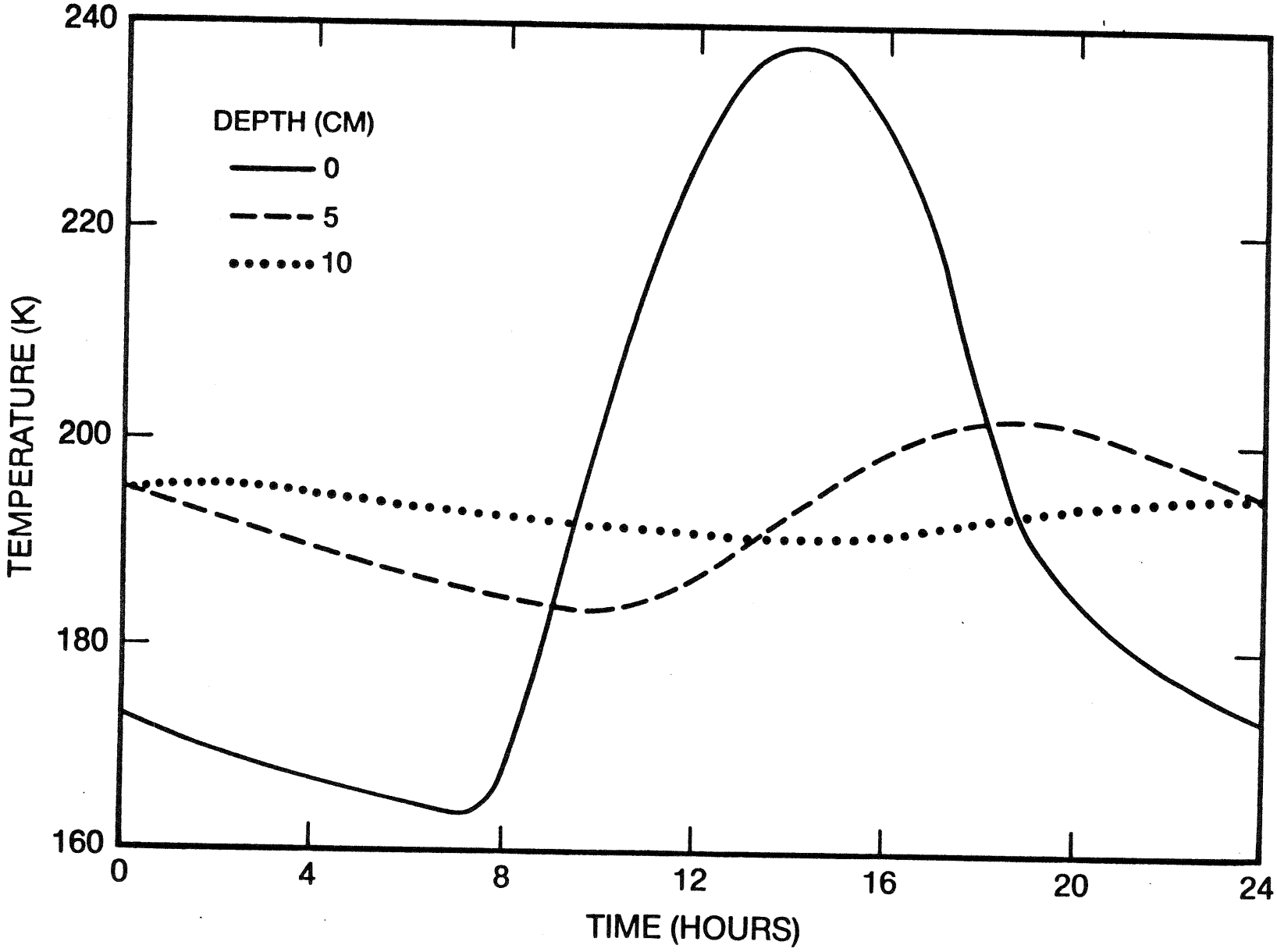
James, M.L. et al., 1967. Applied Numerical Methods for Digital Computation with FORTRAN, International Textbook Company, pp. 479-484.

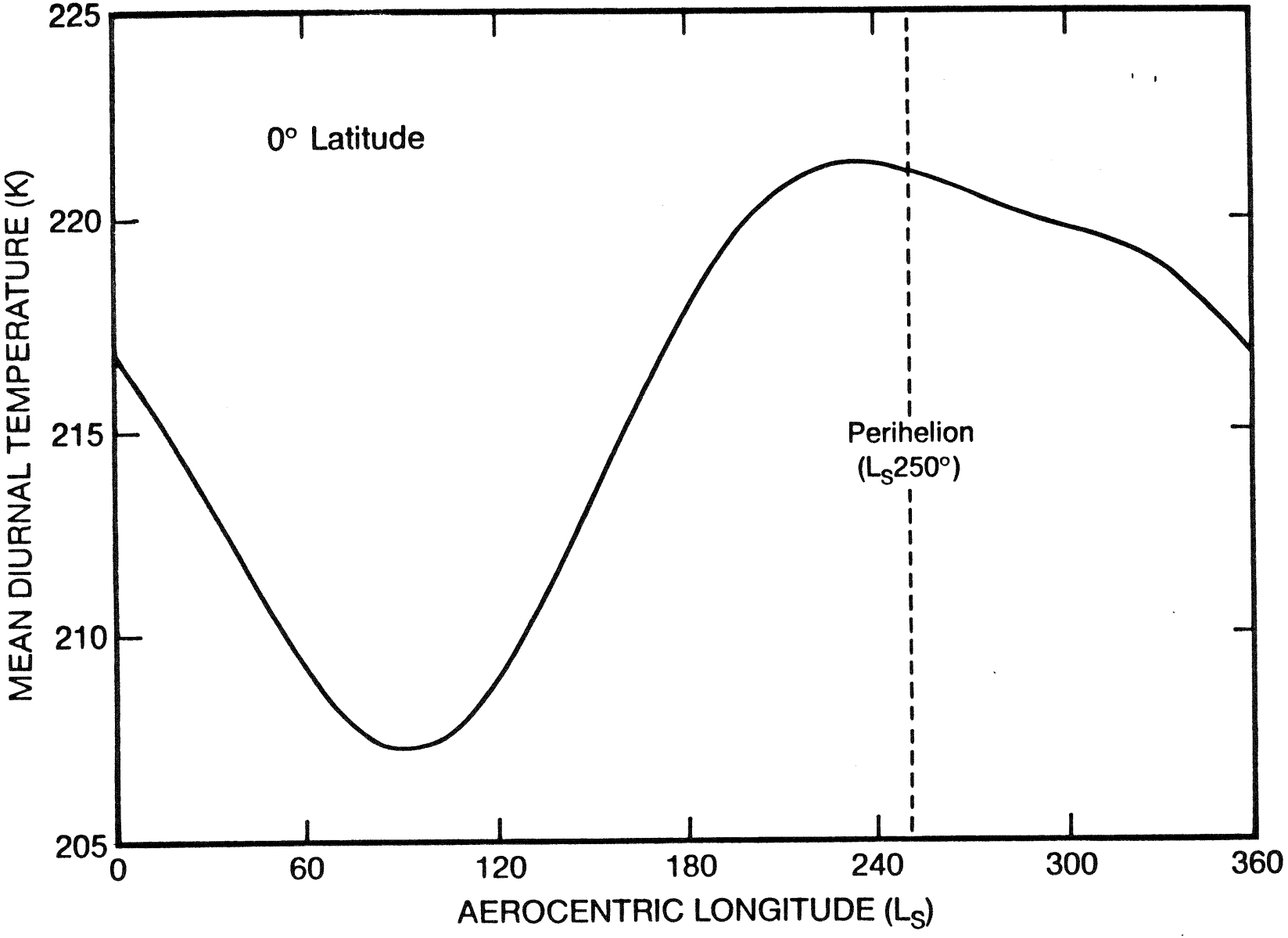
Kieffer, H.H. et al., 1977. Thermal and Albedo Mapping of Mars During the Viking Primary Mission, J. Geophy. Res. 82, pp. 4249-4291.

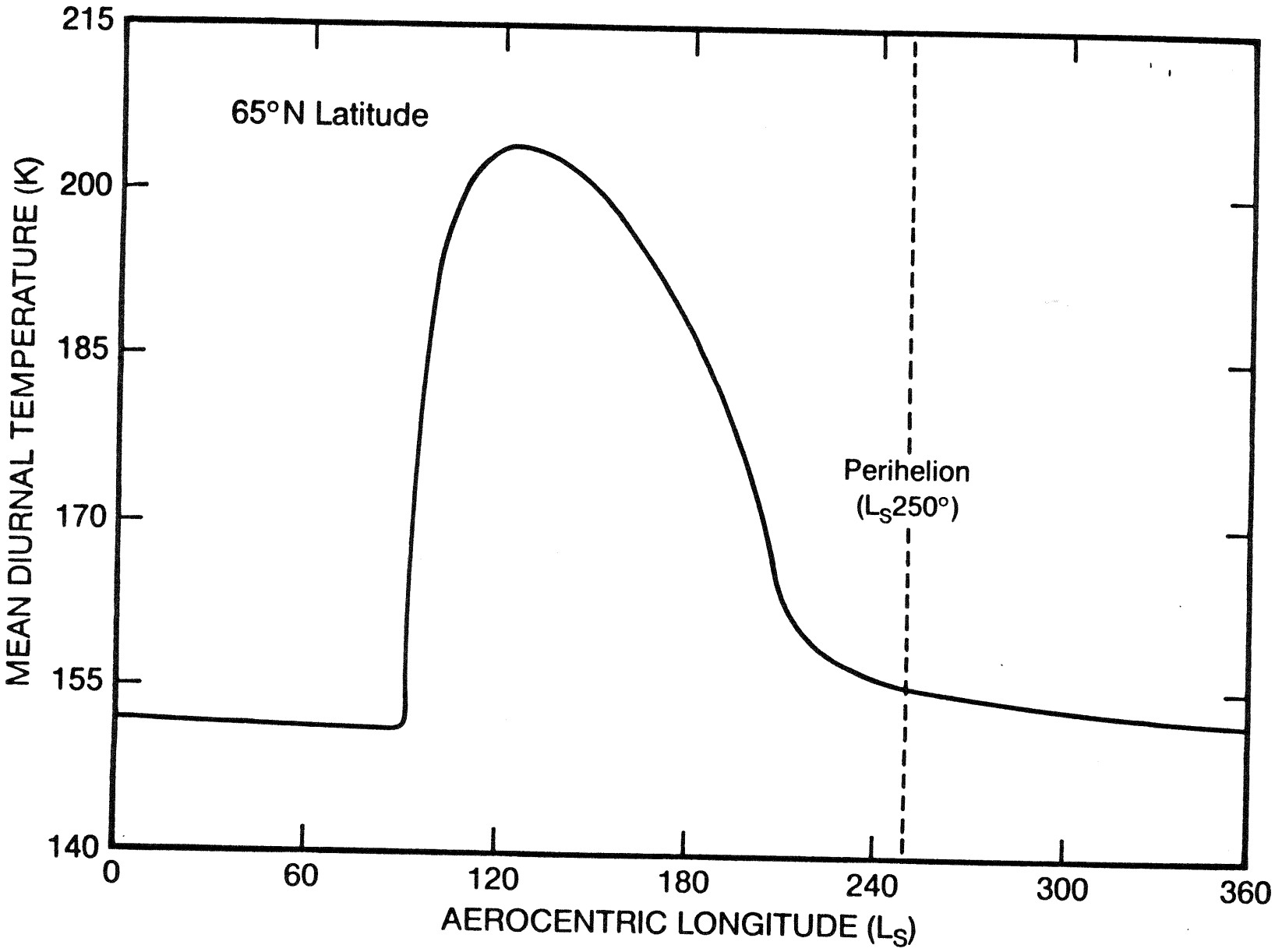
Sundqvist, H. and G. Veronis, 1969, A Simple Finite-Difference Grid with Non-Constant Intervals. Tellus, 22

Ward, W.R., 1979. Present Obliquity Oscillations of Mars: Fourth-Order Accuracy in Orbital e and I, J. Geophys. Res. 84, pp. 237-241.


0
-5
-10
-15
-20
-25
-30
DEPTH (CM)
160
175
190
205
220
235
TEMPERATURE (K)
TIME (H)
6:00
12:00
18:00
24:00


240
220
200
180
160
TEMPERATURE (K)
DEPTH (CM)
0
5
10
0
4
8
12
16
20
24
TIME (HOURS)


225
220
215
210
205
0° Latitude
Perihelion
(L_S250°)
MEAN DIURNAL TEMPERATURE (K)
0
60
120
180
240
300
360
AEROCENTRIC LONGITUDE (L_S)


65°N Latitude
MEAN DIURNAL TEMPERATURE (K)
215
200
185
170
155
140
0
60
120
180
240
300
360
Perihelion
(L_S250°)
AEROCENTRIC LONGITUDE (L_S)