AD-A201 918

NAVAL POSTGRADUATE SCHOOL

Monterey, California

DTIC FILE COPY

THESIS

Transient Three-Dimensional Heat Conduction Computations Using Brian's Technique

by

John A. Watson

September 1988

Thesis Advisor: Yogendra Joshi

Approved for public release; distribution is unlimited

DTIC
ELECTE
05 JAN 1989
S D
E

89 1 05 051

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b. OFFICE SYMBOL (If applicable) 69	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	7b. ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11. TITLE (Include Security Classification)
Transient Three-Dimensional Heat Conduction Computations Using Brian's Technique

12. PERSONAL AUTHOR(S)
WATSON, JOHN A.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Master's Thesis	FROM ____ TO ____	1988 September	209

16. SUPPLEMENTARY NOTATION
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the US Government

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Transient Heat Conduction, Stability of Solutions, Finite Diferences, theses. [illegible]

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

A transient three-dimensional heat conduction code was developed using finite diferences. A stability restriction on the time step was avoided using a technique proposed by Brian. Computations from the code were validated using both the explicit technique and an available closed-form solution for small times. The maximum error was found to be within 0.019 percent for an 11 x 11 x 11 grid and time step of 17.117 seconds. The total CPU time to carry out the computations up to 3,600 seconds using Brian's technique was six times that required for the explicit technique with the same time step of 17.117 seocnds. However, as the time step was increased without altering the heometry, the CPU time using Brian's technique decreased and was less than that used in the explicit technique for time steps larger than 110 seconds. The validated code was also used in the analysis of the transient thermal response of a component on an orbiting spacecraft.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL PROFESSOR JOSHI	22b. TELEPHONE (Include Area Code) 646-3400 (408) 22c. OFFICE SYMBOL 69Ji

DD FORM 1473, 84 MAR
83 APR edition may be used until exhausted
All other editions are obsolete
SECURITY CLASSIFICATION OF THIS PAGE
U.S. Government Printing Office: 1986—606-24

UNCLASSIFIED

REPRODUCED AT GOVERNMENT EXPENSE

Approved for public release; distribution is unlimited.

Transient Three-Dimensional Heat Conduction Computations Using Brian's Technique

by

John A. Watson
Lieutenant Commander, United States Navy
B.S., New Jersey Institute of Technology, 1975

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1988

Author: John A. Watson

Approved by: Yogendra Joshi, Thesis Advisor

Anthony J. Healey, Chairman,
Department of Mechanical Engineering

Gordon E Schacher, Dean of
Science and Engineering

ABSTRACT

A transient three-dimensional heat conduction code was developed using finite differences. A stability restriction on the time step was avoided using a technique proposed by Brian. Computations from the code were validated using both the explicit technique and an available closed-form solution for small times. The maximum error was found to be within 0.019 percent for an 11 x 11 x 11 grid and time step of 17.117 seconds. The total CPU time to carry out the computations up to 3,600 seconds using Brian's technique was six times that required for the explicit technique with the same time step of 17.117 seconds. However, as the time step was increased without altering the geometry, the CPU time using Brian's technique decreased and was less than that used in the explicit technique for time steps larger than 110 seconds. The validated code was also used in the analysis of the transient thermal response of a component on an orbiting spacecraft.

Accession For
NTIS GRA&I ☒
DTIC TAB ☐
Unannounced ☐
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or Special
A-1

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

I. **INTRODUCTION** 1

A. STATEMENT OF PROBLEM 1

B. OBJECTIVES 2

II. **DEVELOPMENT OF MODEL** 3

A. MODEL CONDITIONS 3

B. EXPLICIT TECHNIQUE 4

1. Derivation of Nodal Temperature Equations 4
2. Restrictions on Time Step in the Explicit Technique 7
3. Numerical Calculations 9

C. BRIAN'S TECHNIQUE 9

1. Derivation of Nodal Temperature Equations 11
2. Solution of the Nodal Temperature Equations 17
3. Numerical Calculations 18

III. **VALIDATION** 20

A. SELECTED GEOMETRY AND CONDITIONS 20

B. NUMERICAL COMPUTATIONS 22

C. VALIDATION COMPUTATIONS 24

IV. **APPLICATIONS** 35

A. GENERAL 35

B. ORBITING SATELLITE COMPONENT 35

1. Geometry and Initial Conditions 35

2. Numerical Calculations 41

V. RESULTS 45

VI. CONCLUSIONS 46

VII. RECOMMENDATIONS 47

APPENDIX A EXPLICIT TECHNIQUE NODE EQUATIONS 48

APPENDIX B BRIAN'S TECHNIQUE NODE EQUATIONS 57

APPENDIX C PROGRAM EXPLICIT 85

APPENDIX D PROGRAM BRIAN 107

APPENDIX E PROGRAM VALID 187

LIST OF REFERENCES 191

INITIAL DISTRIBUTION LIST 192

LIST OF TABLES

1 Parameters Used in Validation Runs 21
2 Parameters Used in Orbiting Satellite Application 40

LIST OF FIGURES

1 Left Surface of Rectangular Block Used in the Model 5

2 Flow Chart for Program EXPLICIT 10

3 Flow Chart for Program BRIAN 19

4 Rectangular Model Showing Boundary Conditions Used in Validation Runs 21

5 Flow Chart for Program VALID 23

6 Temperature Difference Versus Time for Time Increment of 17.117 Seconds with Distance of 1 Meter Into Block 25

7 Temperature Difference Versus Time for Time Increment of 4.279 Seconds with Distance of 1 Meter Into Block 26

8 Temperature Difference Versus Time for Time Increment of 17.117 Seconds with Distance of 0.4 Meters Into Block 27

9 Temperature Difference Versus Time for Time Increment of 4.279 Seconds with Distance of 0.4 Meters Into Block 28

10 Temperature Difference Versus Time for Time Increment of 17.117 Seconds on Left Surface of Block 29

11 Temperature Difference Versus Time for Time Increment of 4.279 Seconds on Left Surface of Block 30

12 Temperature Difference Versus Time for Time Increments of 17.117 and 120 Seconds with Distance of 1 Meter Into Block 31

13 Temperature Difference Versus Time for Time Increments of 17.117 and 120 Seconds with Distance of 0.4 Meters Into Block 32

14 Temperature Difference Versus Time for Time Increments of 17.117 and 120 Seconds on Left Surface of Block 33

15 Component on a Satellite in Circular Orbit 37

16 Geometry Used in Determining Geometric Factor for Earth Thermal Radiation 38

17 Geometry Used in Determining Geometric Factor for Solar Reflected Radiation 39

18 Temperature Variation of the Top Surface Center Node 42

19 Temperature Variation of the Top Surface Center Node 43

20 Geometry of Rectangular Model 49

21 Geometry of Rectangular Model 58

LIST OF SYMBOLS

Symbol	Description
a	albedo coefficient
$A_{1_{i,j,k}}$	coefficient associated with $T^{*}_{i-1,j,k}$ in Brian's technique
$A_{2_{i,j,k}}$	coefficient associated with $T^{**}_{i,j-1,k}$ in Brian's technique
$A_{3_{i,j,k}}$	coefficient associated with $T^{n+1}_{i,j,k-1}$ in Brian's technique
$B_{1_{i,j,k}}$	coefficient associated with $T^{*}_{i,j,k}$ in Brian's technique
$B_{2_{i,j,k}}$	coefficient associated with $T^{**}_{i,j,k}$ in Brian's technique
$B_{3_{i,j,k}}$	coefficient associated with $T^{n+1}_{i,j,k}$ in Brian's technique
Bi	Biot number
C	specific heat of material
$C_{1_{i,j,k}}$	coefficient associated with $T^{*}_{i+1,j,k}$ in Brian's technique
$C_{2_{i,j,k}}$	coefficient associated with $T^{**}_{i,j+1,k}$ in Brian's technique
$C_{3_{i,j,k}}$	coefficient associated with $T^{n+1}_{i,j,k+1}$ in Brian's technique
C_{SA}	solar aspect coefficient
d	sum of satellite altitude and radius of earth
$D_{1_{i,j,k}}$	coefficient associated with T^{*} simultaneous equations in Brian's technique
$D_{2_{i,j,k}}$	coefficient associated with T^{**} simultaneous equations in Brian's technique
$D_{3_{i,j,k}}$	coefficient associated with T^{n+1} simultaneous equations in Brian's technique

$\dot{E}_{ALBEDO}$	rate of energy transfer into a control volume from albedo flux
$\dot{E}_{COND}$	rate of energy transfer into a control volume from conduction
$\dot{E}_{CONV}$	rate of energy transfer into a control volume from convection
$\dot{E}_{EARTH}$	rate of energy transfer into a control volume from earth thermal radiation
$\dot{E}_{FLUX}$	rate of energy transfer into a control volume from imposed flux
$\dot{E}_{GEN}$	rate of energy generation
$\dot{E}_{G_{i,j,k}}$	rate of energy generation per unit volume
$\dot{E}_{IN}$	rate of energy transfer into a control volume
$\dot{E}_{OUT}$	rate of energy transfer out of a control volume
$\dot{E}_{RAD}$	rate of energy transfer into a control volume from radiation heat transfer
$\dot{E}_{SOLAR}$	rate of energy transfer into a control volume from solar flux
$\dot{E}_{ST}$	rate of increase of stored within a control volume
F_A	geometric factor for albedo flux
F_E	geometric factor for earth thermal radiation
Fo	Fourier number
h	heat transfer coefficient
i	x-coordinate of node
j	y-coordinate of node
k	z-coordinate of node thermal conductivity of material

P	period of revolution for earth satellite
q''	imposed heat flux
R_1	$\left(\frac{\Delta x}{\Delta y}\right)^2$
R_2	$\left(\frac{\Delta x}{\Delta z}\right)^2$
T_∞	ambient temperature
T^n	temperature at previous time level n
$T^{n+\frac{1}{2}}$	temperature at half-time step interval
T^*	temperature at $n + \frac{1}{2}$ time level in the x-direction
T^{**}	temperature at $n + \frac{1}{2}$ time level in the y-direction
T^{n+1}	temperature at current time level
U_1	R_1Fo
U_2	R_2Fo
U_3	$2R_1Fo$
U_4	$2R_2Fo$
U_5	$2Fo$
Y_1	$Fo\Delta x^2/k$
α	solar absorptivity
Δt	time increment
Δx	distance increment in x-direction
Δy	distance increment in y-direction
Δz	distance increment in z-direction
ε	emissivity; absorptivity of earth flux
μ	gravitational constant

ρ	mass density
σ	Stefan-Boltzmann constant
ϕ_S	solar flux
ϕ_E	earth thermal radiation flux

The subscripts below have the following meaning:

1	left face
2	right face
3	bottom face
4	top face
5	front face
6	back face

X_1	$-2Fo\sigma\varepsilon_1 R_1 \Delta y/k$
X_2	$-2Fo\sigma\varepsilon_2 R_1 \Delta y/k$
X_3	$-2Fo\sigma\varepsilon_3 R_2 \Delta z/k$
X_4	$-2Fo\sigma\varepsilon_4 R_2 \Delta z/k$
X_5	$-2Fo\sigma\varepsilon_5 \Delta x/k$
X_6	$-2Fo\sigma\varepsilon_6 \Delta x/k$
V_1	$FoBi_1 R_1^{\frac{1}{2}}$
V_2	$FoBi_2 R_1^{\frac{1}{2}}$
V_3	$FoBi_3 R_1^{\frac{1}{2}}$
V_4	$FoBi_4 R_1^{\frac{1}{2}}$
V_5	$FoBi_5$
V_6	$FoBi_6$
W_1	$2FoR_1 \frac{\Delta y}{k}(q_1'' + \varepsilon_1 F_E \phi_E + \alpha_1 C_{SA_1} \phi_S + \alpha_1 a F_A \phi_S)$

W_2 $\quad 2FoR_1\frac{\Delta y}{k}(q''_2 + \varepsilon_2 F_E \phi_E + \alpha_2 C_{SA_2}\phi_S + \alpha_2 aF_A \phi_S)$

W_3 $\quad 2FoR_2\frac{\Delta z}{k}(q''_3 + \varepsilon_3 F_E \phi_E + \alpha_3 C_{SA_3}\phi_S + \alpha_3 aF_A \phi_S)$

W_4 $\quad 2FoR_2\frac{\Delta z}{k}(q''_4 + \varepsilon_4 F_E \phi_E + \alpha_4 C_{SA_4}\phi_S + \alpha_4 aF_A \phi_S)$

W_5 $\quad 2Fo\frac{\Delta x}{k}(q''_5 + \varepsilon_5 F_E \phi_E + \alpha_5 C_{SA_5}\phi_5 + \alpha_5 aF_A \phi_S)$

W_6 $\quad 2Fo\frac{\Delta x}{k}(q''_6 + \varepsilon_6 F_E \phi_E + \alpha_6 C_{SA_6}\phi_S + \alpha_6 aF_A \phi_S)$

ACKNOWLEDGMENTS

The author would like to express his deep gratitude to Assistant Professor Y. Joshi, his thesis advisor, for his guidance and advice in the endeavor.

I. INTRODUCTION

A. STATEMENT OF PROBLEM

Three-dimensional transient heat conduction computations involving finite difference techniques can be costly and time consuming. In the explicit technique, there is a maximum time increment that can be used before the solution becomes unstable [Ref. 1]. This time increment results from an upper limit on the allowable values of the Fourier number [Ref. 2]. The implicit technique, on the other hand, requires more complicated computations than the explicit technique due to matrix inversion operations [Ref. 2]. Even the storage capability of a modern computer can be taxed when dealing with storage and manipulation of very large matrices [Ref. 3]. Both the explicit and implicit techniques can become expensive when a large number of nodes is required [Ref. 4].

For two-dimensional transients, an alternative technique is the alternating direction implicit (ADI) method. Calculations at each time step require only the handling of tridiagonal matrices [Ref. 5]. A direct extension of the ADI technique to three-dimensional problems, however, has been found to have only conditional stability [Ref. 5]. A different algorithm by Brian is unconditionally stable and still requires the handling of only tridiagonal system of equations, thus avoiding the more complex matrix manipulation of the implicit approach [Ref. 3].

B. OBJECTIVES

The objectives of this study were:

1. To develop a FORTRAN computer program employing Brian's technique for three-dimensional transient heat conduction.

2. To validate the FORTRAN computer program using the explicit technique with identical geometric configuration, boundary, and initial conditions. Comparisons with closed-form solutions available for a semi-infinite geometry were also made.

3. To employ the above technique in the application of temperature computations involving an orbiting satellite.

This study was also intended to be a basis for future satellite heat transfer analysis and computations involved with Project Orion, the Naval Postgraduate School's multi-disciplinary space research program.

All objectives were achieved.

II. DEVELOPMENT OF MODEL

A. MODEL CONDITIONS

The model is a three-dimensional rectangular block with geometric dimensions expressed in Cartesian coordinates. A numerical temperature solution only permits the determination of the temperature distribution at discrete points or nodes [Ref. 2]. The method of finite differences requires that a nodal equation be written for each node in the block. The number of nodes in any coordinate direction is based on the distance increment in the same direction.

The nodal temperature equation for each node was obtained using a conservation of energy equation within a control volume. Incopeia and DeWitt [Ref. 2] present the general form of the conservation of energy equation as

$$\dot{E}_{IN} - \dot{E}_{OUT} + \dot{E}_{GEN} = \dot{E}_{ST} \tag{1}$$

The nodal temperature equations were developed using equation 1. In the following sections, an equation for a node on the left surface of the rectangular block away from corners and edges is derived for both the explicit and Brian techniques. Equations for all other nodes are derived in an identical fashion. The explicit technique equations are tabulated in Appendix A and the Brian technique equations are tabulated in Appendix B.

B. EXPLICIT TECHNIQUE

1. Derivation of Nodal Temperature Equations

Figure 1 shows the left surface of the rectangular block with associated boundary conditions. Equation 1 is rewritten assuming that all the energy components are into the control volume as

$$\dot{E}_{IN} + \dot{E}_{GEN} = \dot{E}_{ST} \tag{2}$$

where the energy generation term is assumed to be zero.

Using finite difference notation, the energy storage term is

$$\dot{E}_{ST} = \rho C \, \Delta x \frac{\Delta y}{2} \Delta z \left(\frac{T^{n+1}_{i,j,k} - T^{n}_{i,j,k}}{\Delta t} \right) \tag{3}$$

The energy inflow term, $\dot{E}_{IN}$, is, in general, composed of convection, conduction, radiation, and imposed heat flux terms. For the development of the explicit technique, the radiation term was neglected. In the explicit technique, all involved nodal temperatures on the left side of equation 2 are evaluated at the prior time step. The convection term is thus

$$\dot{E}_{CONV} = h \, \Delta x \, \Delta z \left(T_{\infty} - T^{n}_{i,j,k} \right) \tag{4}$$

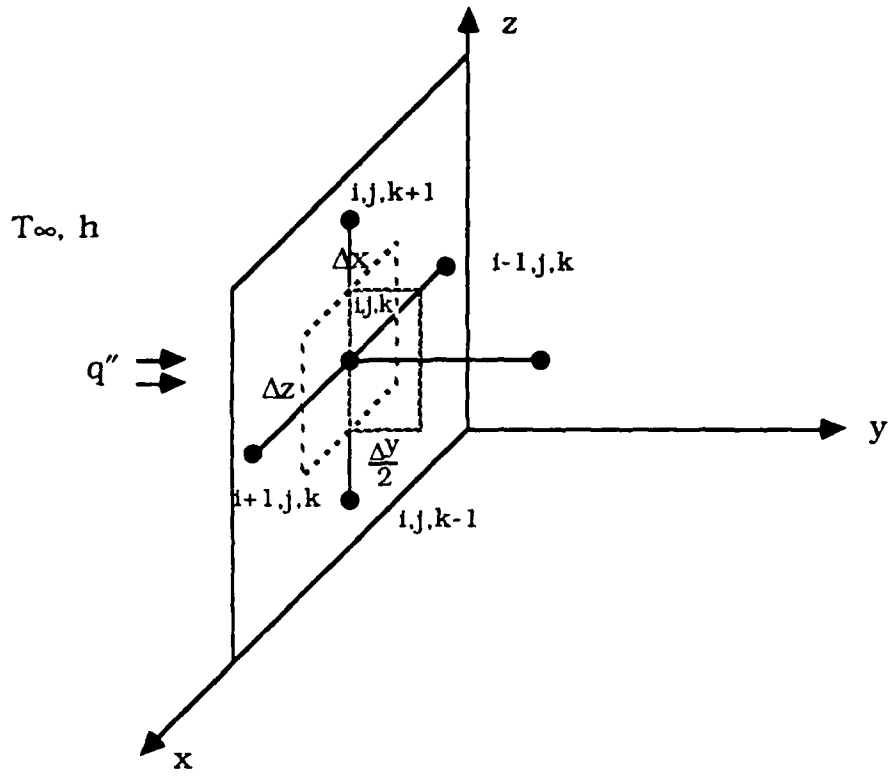

Figure 1. **Left Surface of Rectangular Block Used in the Model**

The net conduction term will have contributions from all three coordinate directions and is

$$\dot{E}_{COND} = k\frac{\Delta y}{2}\Delta z\frac{T^n_{i-1,j,k} + T^n_{i+1,j,k} - 2T^n_{i,j,k}}{\Delta x} + k\Delta x\Delta z\frac{T^n_{i,j+1,k} - T^n_{i,j,k}}{\Delta y}$$
$$+ k\Delta x\frac{\Delta y}{2}\frac{T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}}{\Delta z} \tag{5}$$

The flux term will be

$$\dot{E}_{FLUX} = q''\Delta x\Delta z \tag{6}$$

Substituting equations 3 through 6 into equation 2 produces

$$k\frac{\Delta y}{2}\Delta z\frac{T^n_{i-1,j,k}+T^n_{i+1,j,k}-2T^n_{i,j,k}}{\Delta x}+k\Delta x\Delta z\frac{T^n_{i,j+1,k}-T^n_{i,j,k}}{\Delta y}$$

$$+k\Delta x\frac{\Delta y}{2}\frac{T^n_{i,j,k-1}+T^n_{i,j,k+1}-2T^n_{i,j,k}}{\Delta z}+k\Delta xz\left(T_\infty-T^n_{i,j,k}\right) \tag{7}$$

$$+q''\Delta xz=\rho C\Delta x\frac{\Delta y}{2}\Delta z\left(\frac{T^{n+1}_{i,j,k}-T^n_{i,j,k}}{\Delta t}\right)$$

The temperature at the next time level, n+1, is next expressed in terms of temperatures at the previous time level, n. To accomplish this, terms in equation 7 are rearranged.

Multiplying each term in equation 7 by $\frac{\Delta x}{k\Delta y\Delta z}$ yields

$$\tfrac{1}{2}\left(T^n_{i-1,j,k}+T^n_{i+1,j,k}-2T^n_{i,j,k}\right)+\left(\frac{\Delta x}{\Delta y}\right)^2\left(T^n_{i,j+1,k}-T^n_{i,j,k}\right)$$

$$+\tfrac{1}{2}\left(\frac{\Delta x}{\Delta z}\right)^{2^2}\left(T^n_{i,j,k-1}+T^n_{i,j,k+1}-2T^n_{i,j,k}\right) \tag{8}$$

$$+\frac{h}{k}\Delta x\frac{\Delta x}{\Delta y}\left(T_\infty-T^n_{i,j,k}\right)+q''\frac{\Delta x^2}{k\Delta y}=\tfrac{1}{2}\frac{\rho C}{k\Delta t}\Delta x^2\left(T^{n-1}_{i,j,k}-T^n_{i,j,k}\right)$$

Equation 8 is simplified using Biot and Fourier numbers, which are written as $\frac{h}{k}\Delta x$ and $\frac{k\Delta t}{\rho C\Delta x^2}$, respectively. The terms $\left(\frac{\Delta x}{\Delta y}\right)^2$, $\left(\frac{\Delta x}{\Delta z}\right)^2$, and $\left(\frac{\Delta x}{\Delta y}\right)^2$ are identified as R_1, R_2, and $R_1^{\frac{1}{2}}$, respectively. Equation 8 now becomes

$$\tfrac{1}{2}\left(T^{n}_{i-1,j,k} + T^{n}_{i+1,j,k} - 2T^{n}_{i,j,k}\right) + R_1\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ \tfrac{1}{2}R^{2}\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right) \tag{9}$$

$$+ BiR_1^{\frac{1}{2}}\left(T_\infty - T^{n}_{i,j,k}\right) + q''R_1\frac{\Delta y}{k} = \frac{1}{2Fo}\left(T^{n+1}_{i,j,k} - T^{n}_{i,j,k}\right)$$

Rearranging terms, the temperature at the current time level is expressed as

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + F_o\left(T^{n}_{i-1,j,k} + T^{n}_{i+1,j,k} - 2T^{n}_{i,j,k}\right) + 2F_oR_1\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2F_oR_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right) \tag{10}$$

$$+ 2F_oB_iR_1^{\frac{1}{2}}\left(T_\infty - T^{n}_{i,j,k}\right) + 2F_oR_1q''\frac{\Delta y}{k}$$

In a similar manner, the temperature equations for each node are derived.

2. Restrictions on Time Step in the Explicit Technique

In the derivation of the explicit technique equations, the Fourier number was expressed as

$$Fo = \frac{k\Delta t}{\rho C\Delta x^2} \tag{11}$$

The stability of the temperature solution is dependent on the time increment [Ref. 2]. Rearranging equation 10 to isolate the node of interest at both the current and previous time levels yields

$$T^{n+1}_{i,j,k} = Fo\left(T^{n}_{i-1,j,k} + T^{n}_{i+1,j,k}\right) + 2FoR_1 T^{n}_{i,j+1,k} + FoR_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1}\right)$$

$$+ 2FoB_i R_1^{\frac{1}{2}} T_\infty + 2FoR_1 q'' \frac{\Delta y}{k} \tag{12}$$

$$+ \left(1 - 2Fo - 2FoR_1 - 2FoR_2 - 2FoBiR_1^{\frac{1}{2}}\right) T^{n}_{i,j,k}$$

The coefficient associated with the previous time level at the node of interest must be greater than or equal to zero [Ref. 2]. This implies

$$1 - 2Fo\left(1 + R_1 + R_2 + BiR_1^{\frac{1}{2}}\right) \geq 0 \tag{13}$$

Rearranging terms, it follow that, for stability,

$$Fo \leq \frac{1}{2\left(1 + R_1 + R_2 + BiR_1^{\frac{1}{2}}\right)} \tag{14}$$

Limitations on Fo were derived for each different node. The maximum allowable Fo was chosen as the least of all these values. Since the Fourier number is dependent on Δt, Δx, and material properties, it follows from equation 11 that the maximum time increment will be dependent on the distance increment. As Δx decreases, the maximum time increment must also decrease to have equation 14 hold true [Ref. 1].

3. Numerical Calculations

A computer program (program EXPLICIT) was written to evaluate the transient temperature distribution within the block using the explicit technique. Figure 2 is a flow-chart description of the program. A complete listing of program EXPLICIT is provided in Appendix C.

C. BRIAN'S TECHNIQUE

The Brian technique provides unconditional stability, allowing large time increments for transient three-dimensional computations [Ref. 3]. As mentioned earlier, the explicit and fully implicit techniques may become unsuitable for computations involving a large number of nodes or long transient times. The explicit technique relies on small time increments for stability, thus increasing the amount of computer time required for calculations [Ref. 5]. The implicit technique requires inverting large matrices that may create problems with storage capability and computer time [Ref. 4].

Another possible technique, the alternating direction implicit (ADI), is applicable to both two- and three-dimensional problems. While the ADI method is unconditionally stable for two-dimensional cases, instability occurs for large time increments in three-dimensional problems [Refs. 4, 5].

The Brian technique modifies the ADI method to provide unconditional stability and minimize computer time [Ref. 3].

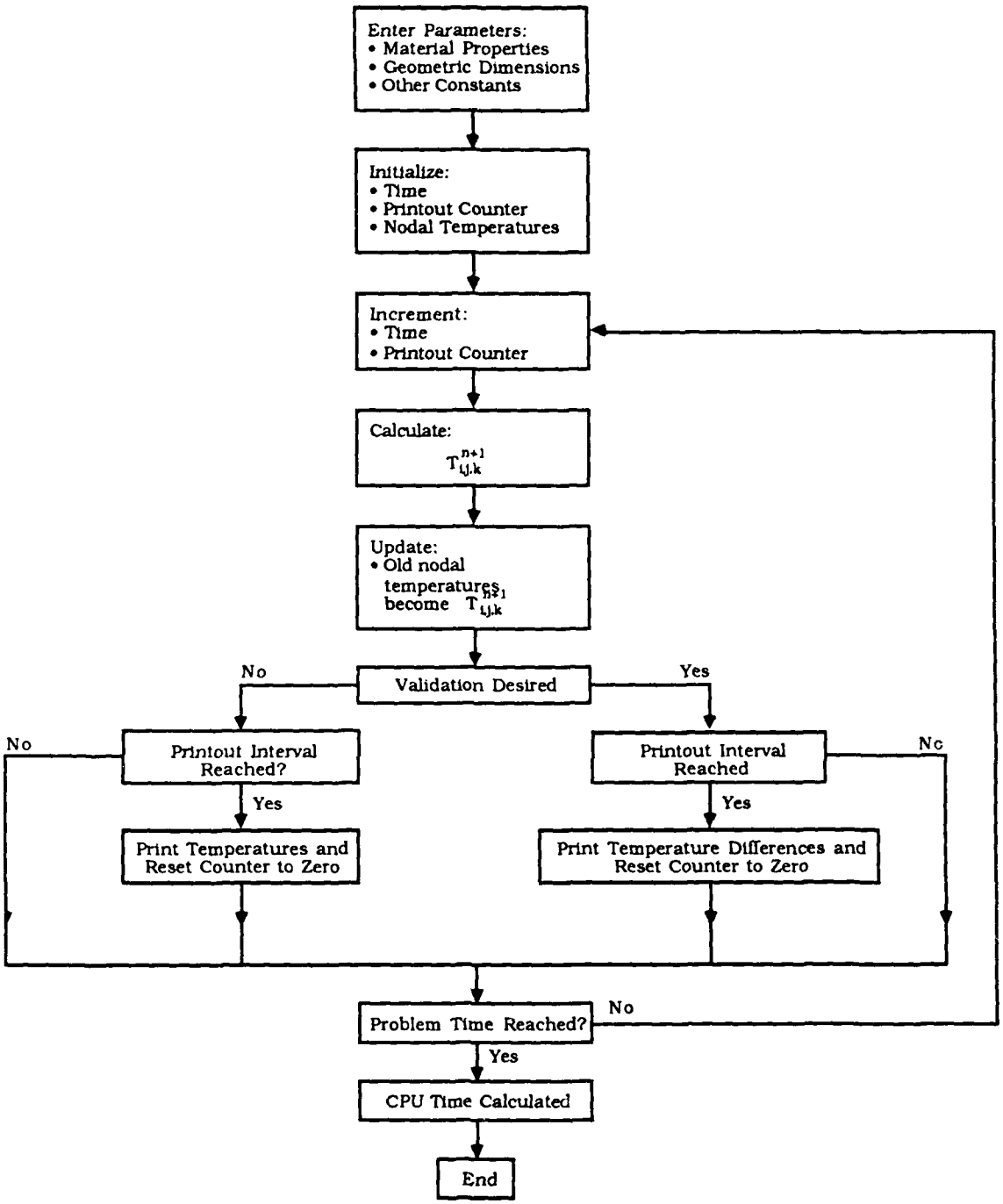

Figure 2. **Flow Chart for Program EXPLICIT**

1. Derivation of Nodal Temperature Equations

Figure 1 shows the left surface of the rectangular block with associated boundary conditions. Equation 2 is again the starting point for nodal temperature equation derivation and is

$$\dot{E}_{IN} + \dot{E}_{GEN} = \dot{E}_{ST} \tag{2}$$

where the energy generation term is again assumed to be zero.

The nodal temperature equation derivation for Brian's technique uses half-time steps for each coordinate direction. This differs from the explicit technique, in which a whole-time increment is used and spatial coordinate direction is not considered.

In Brian's technique, the first half-time level, $n + \frac{1}{2}$, treats temperatures in the x direction as unknown with temperatures in the y and z directions known from the previous time level, n. Solving a tridiagonal system of simultaneous equations, the temperatures at the $n + \frac{1}{2}$ level are determined.

Temperatures in the y direction are then determined using x-direction temperatures at the $n + \frac{1}{2}$ time level and z-direction temperatures at the M time level. Similarly, temperatures in the z-direction at the current n+1 time level are determined using $n + \frac{1}{2}$ temperatures from the x and y directions.

The above procedure is demonstrated for a node on the left surface away from corners and edges.

The energy storage term is

$$\dot{E}_{ST} = \rho C \Delta x \frac{\Delta y}{2} \Delta z \left(\frac{T^{n+\frac{1}{2}}_{i,j,k} - T^{n}_{i,j,k}}{\frac{\Delta t}{2}} \right) \tag{15}$$

The energy inflow term, $\dot{E}_{IN}$, is composed of convection, conduction, radiation, and any imposed external heat fluxes. As a first step, only the nodal temperatures associated with heat conduction and energy storage in the x-direction are evaluated at the $n + \frac{1}{2}$ time level. The convection term is thus

$$\dot{E}_{CONV} = h \Delta x \Delta z \left(T_{\infty} - T^{n}_{i,j,k} \right) \tag{16}$$

The net conduction term will have contributions from all three coordinate directions and is

$$\begin{aligned} \dot{E}_{COND} = {} & k \frac{\Delta y}{2} \Delta z \frac{T^{n+\frac{1}{2}}_{i-1,j,k} + T^{n+\frac{1}{2}}_{i+1,j,k} - 2T^{n+\frac{1}{2}}_{i,j,k}}{\Delta x} + k \Delta x \Delta z \frac{T^{n}_{i,j+1,k} - T^{n}_{i,j,k}}{\Delta x} \\ & + k \Delta x \frac{\Delta y}{2} \frac{T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}}{\Delta z} \end{aligned} \tag{17}$$

The radiation term, assuming extensive surroundings, will be

$$\dot{E}_{RAD} = \sigma \varepsilon \Delta x \Delta z \left(T^{4}_{\infty} - T^{n^4}_{i,j,k} \right) \tag{18}$$

For applications involving an orbiting spacecraft, additional input flux terms need to be considered. These include the solar flux, the infrared emission from the earth, and the reflected component of the

solar flux from the earth's atmosphere, the albedo. The solar flux term will be

$$\dot{E}_{SOLAR} = \Delta x \Delta z \alpha C_{SA} \phi_S \tag{19}$$

The reflected solar flux (albedo) term will be

$$\dot{E}_{ALBEDO} = \Delta x \Delta z F_A \alpha a \phi_S \tag{20}$$

The earth flux term will be

$$\dot{E}_{EARTH} = \Delta x \Delta z F_E \varepsilon \phi_E \tag{21}$$

The imposed heat flux term will be

$$\dot{E}_{FLUX} = \Delta x \Delta z q'' \tag{22}$$

Substituting equations (15) through (22) into equation 2 yields

$$\begin{aligned}
&k\frac{\Delta y}{2}\Delta z\frac{T^{n+\frac{1}{2}}_{i-1,j,k} + T^{n+\frac{1}{2}}_{i+1,j,k} - 2T^{n+\frac{1}{2}}_{i,j,k}}{\Delta x} + k\Delta x\Delta z\frac{T^{n}_{i,j+1,k} - T^{n}_{i,j,k}}{\Delta y} \\
&+ k\Delta x\frac{\Delta y}{2}\frac{T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}}{\Delta z} + \sigma\varepsilon\Delta x\Delta z\left(T^{4}_{\infty} - T^{n^4}_{i,j,k}\right) \\
&+ \Delta x\Delta z\alpha C_{SA}\phi_S + \Delta x\Delta z F_A\,\alpha a\phi_S + \Delta x\Delta z F_E\varepsilon\phi_E + \Delta x\Delta z q'' \\
&+ h\Delta x\Delta z\left(T_{\infty} - T^{n}_{i,j,k}\right) = \rho C\Delta x\frac{\Delta y}{2}\Delta z\frac{T^{n+\frac{1}{2}}_{i,j,k} - T^{n}_{i,j,k}}{\frac{\Delta t}{2}}
\end{aligned} \tag{23}$$

The temperatures at time level $n+\frac{1}{2}$ are next expressed in terms of temperatures at the previous time level, n. To accomplish this, terms in equation 23 are rearranged.

Multiplying each term in equation 23 by $\frac{\Delta x}{k\Delta y\Delta z}$ yields

$$\frac{1}{2}\left(T^{n+\frac{1}{2}}_{i-1,j,k}+T^{n+\frac{1}{2}}_{i+1,j,k}-2T^{n+\frac{1}{2}}_{i,j,k}\right)+\left(\frac{\Delta x}{\Delta y}\right)^2\left(T^{n}_{i,j+1,k}-T^{n}_{i,j,k}\right)$$

$$+\frac{1}{2}\left(\frac{\Delta x}{\Delta z}\right)^2\left(T^{n}_{i,j,k-1}+T^{n}_{i,j,k+1}-2T^{n}_{i,j,k}\right)+\frac{\sigma\varepsilon\Delta x^2}{k\Delta y}\left(T^4_{\infty}-T^{n^4}_{i,j,k}\right)$$

$$+\frac{\alpha C_{SA}\Delta x^2}{k\Delta y}\phi_S+\frac{\alpha F_A a\Delta x^2}{k\Delta y}\phi_S+\frac{F_E\varepsilon\Delta x^2}{k\Delta y}\phi_E+\frac{\Delta x^2}{k\Delta y}q'' \tag{24}$$

$$+\frac{h\Delta x^2}{k\Delta y}q''+\frac{h\Delta x^2}{k\Delta y}\left(T_{\infty}-T^{n}_{i,j,k}\right)=\frac{2\rho C\Delta x^2}{2k\Delta t}\left(T^{n+\frac{1}{2}}_{i,j,k}-T^{n}_{i,j,k}\right)$$

Equation 24 is simplified using Biot and Fourier numbers, which are written as $\frac{h}{k}\Delta x$ and $\frac{k\Delta t}{2\rho C\Delta x^2}$, respectively. Equation 24 now becomes

$$\frac{1}{2}\left(T^{n+\frac{1}{2}}_{i-1,j,k}+T^{n+\frac{1}{2}}_{i+1,j,k}-2T^{n+\frac{1}{2}}_{i,j,k}\right)+R_1\left(T^{n}_{i,j+1,k}-T^{n}_{i,j,k}\right)$$

$$+\frac{1}{2}R_2\left(T^{n}_{i,j,k-1}+T^{n}_{i,j,k+1}-2T^{n}_{i,j,k}\right)+\frac{\sigma\varepsilon}{k}R_1\Delta y\left(T^4_{\infty}-T^{n^4}_{i,j,k}\right)$$

$$+\frac{\alpha C_{SA}R_1\Delta y}{k}\phi_S+\frac{\alpha F_A aR_1\Delta y}{k}\phi_S+\frac{F_E\varepsilon R_1\Delta y}{k}\phi_E \tag{25}$$

$$+\frac{R_1\Delta y}{k}q''+B_i R_1^{\frac{1}{2}}\left(T_{\infty}-T^{n}_{i,j,k}\right)=\frac{1}{2Fo}\left(T^{n+\frac{1}{2}}_{i,j,k}-T^{n}_{i,j,k}\right)$$

Rearranging terms, the temperatures at the first half-time step level in the x direction are

$$
\begin{aligned}
&-\mathrm{Fo}T_{i-1,j,k}^{n+\frac{1}{2}} + (1+2\mathrm{Fo})T_{i,j,k}^{n+\frac{1}{2}} - \mathrm{Fo}T_{i,j,k}^{n+\frac{1}{2}} = T_{i,j+1,k}^{n} \\
&+2\mathrm{Fo}R_1\left(T_{i,j+1,k}^{n} - T_{i,j,k}^{n}\right) + \mathrm{Fo}R_2\left(T_{i,j,k-1}^{n} + T_{i,j,k+1}^{n} - 2T_{i,j,k}^{n}\right) \\
&+2\mathrm{Fo}\,R_1\frac{\alpha C_{SA}}{k}\phi_S + 2\mathrm{Fo}\,R_1\Delta y\frac{\alpha F_A a}{k}\phi_S + 2\mathrm{Fo}\,R_1\Delta y\frac{F_E\varepsilon}{k}\phi_E \qquad (26) \\
&+2\mathrm{Fo}\,R_1\Delta y + \frac{q''}{k} + 2\mathrm{FoBi}\;R_1^{\frac{1}{2}}\left(T_\infty - T_{i,j,k}^{n}\right)
\end{aligned}
$$

The next step is to express the temperatures in the y-direction at time level $n+\frac{1}{2}$ using x-direction temperatures determined at time level $n+\frac{1}{2}$ from equation 26 and z-direction temperatures at time level n. For greater clarity, the x-direction temperatures at time level $n+\frac{1}{2}$, $T^{n+\frac{1}{2}}$, are rewritten as T*. Equation 26 now becomes

$$
\begin{aligned}
&-\mathrm{Fo}T_{i-1,j,k}^{*} + (1+2\mathrm{Fo})T_{i,j,k}^{*} - \mathrm{Fo}T_{i+1,j,k}^{*} = T_{i,j+1,k}^{n} \\
&+2\mathrm{Fo}R_1\left(T_{i,j+1,k}^{n} - T_{i,j,k}^{n}\right) + \mathrm{Fo}R_2\left(T_{i,j,k-1}^{n} + T_{i,j,k+1}^{n} - 2T_{i,j,k}^{n}\right) \\
&+2\mathrm{Fo}\,R_1\Delta y\frac{\alpha\varepsilon}{k}\left(T_\infty^4 - T_{i,j,k}^{n^4}\right) + 2\mathrm{Fo}\,R_1\Delta y\frac{\alpha C_{SA}}{k}\phi_S + 2\mathrm{Fo}\,R_1\Delta y\frac{\alpha F_A a}{k}\phi_S \qquad (27) \\
&+2\mathrm{Fo}\,R_1\Delta y\frac{F_E\varepsilon}{k}\phi_E + 2\mathrm{Fo}\,R_1\Delta y\frac{q''}{k} + 2\mathrm{FoBi}\;R_1^{\frac{1}{2}}\left(T_\infty - T_{i,j,k}^{n}\right)
\end{aligned}
$$

The derivation for the y-direction nodal temperature equation is similar to the x-direction derivation. The y-direction equation for the same node is

$$
\begin{aligned}
&(1+2FoR_1)T_{i,j,k}^{n+\frac{1}{2}} - 2FoR_1 T_{i,j+1,k}^{n+\frac{1}{2}} = T_{i,j,k}^{n} \\
&+ Fo\left(T_{i-1,j,k}^{*} + T_{i+1,j,k}^{*} - 2T_{i,j,k}^{*}\right) + FoR_2\left(T_{i,j,k-1}^{n} + T_{i,j,k+1}^{n} - 2T_{i,j,k}^{n}\right) \\
&+ 2Fo\,R_1\Delta y\frac{\sigma\varepsilon}{k}\left(T_{\infty}^{4} - T_{i,j,k}^{n^4}\right) + 2Fo\,R_1\Delta y\frac{\alpha C_{SA}}{k}\phi_S \\
&+ 2Fo\,R_1\Delta y\frac{\alpha F_A a}{k}\phi_S + 2Fo\,R_1\Delta y\frac{F_E\varepsilon}{k}\phi_E \qquad (28) \\
&+ 2Fo\,R_1\Delta y\frac{q''}{k} + 2FoBi\;R_1^{\frac{1}{2}}\left(T_{\infty} - T_{i,j,k}^{n}\right)
\end{aligned}
$$

The y-direction temperatures at time level $n+\frac{1}{2}$, $T^{n+\frac{1}{2}}$, are rewritten as T**. Equation 28 now becomes

$$
\begin{aligned}
&(1+2FoR_1)T_{i,j,k}^{**} - 2FoR_1\,T_{i,j+1,k}^{**} = T_{i,j,k}^{n} + Fo\left(T_{i-1,j,k}^{*} + T_{i+1,j,k}^{*} - 2T_{i,j,k}^{*}\right) \\
&+ FoR_2\left(T_{i,j,k-1}^{n} + T_{i,j,k+1}^{n} - 2T_{i,j,k}^{n}\right) + 2Fo\,R_1\Delta y\frac{\sigma\varepsilon}{k}\left(T_{\infty}^{4} - T_{i,j,k}^{n^4}\right) \\
&+ 2Fo\,R_1\Delta y\frac{\alpha C_{SA}}{k}\phi_S + 2Fo\,R_1\Delta y\frac{\alpha F_A a}{k}\phi_S + 2Fo\,R_1\Delta y\frac{F_E\varepsilon}{k}\phi_E \qquad (29) \\
&+ 2Fo\,R_1\Delta y\frac{q''}{k} + 2FoBi\;R_1^{\frac{1}{2}}\left(T_{\infty} - T_{i,j,k}^{n}\right)
\end{aligned}
$$

The expression for the z-direction temperatures at time level n+1 is dependent on the x-direction and y-direction temperatures at time level $n+\frac{1}{2}$, T* and T**, respectively. The z-direction nodal temperature equation is

$$
\begin{aligned}
&-\mathrm{FoR}_2 T^{n+1}_{i,j,k-1} + (1+2\mathrm{FoR}_2)T^{n+1}_{i,j,k} - \mathrm{FoR}_2 T^{n+1}_{i,j,k} = \\
&T^{**}_{i,j,k} + \mathrm{Fo}\left(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k}\right) + 2\mathrm{FoR}_1\left(T^{**}_{i,j+1,k} - T^{**}_{i,j,k}\right) \\
&+ 2\,\mathrm{Fo}\,R_1\Delta y\frac{\sigma\varepsilon}{k}\left(T^4_{\infty} - T^{**}_{i,j,k}\right) + 2\,\mathrm{Fo}\,R_1\Delta y\frac{\alpha C_{SA}}{k}\phi_S \\
&+ 2\,\mathrm{Fo}\,R_1\Delta y\frac{\alpha F_A a}{k}\phi_S + 2\,\mathrm{Fo}\,R_1\Delta y\frac{F_E\varepsilon}{k}\phi_E \qquad (30) \\
&+ 2\,\mathrm{Fo}\,R_1\Delta y\frac{q''}{k} + 2\mathrm{FoBiR}_1^{\frac{1}{2}}\left(T_{\infty} - T^{**}_{i,j,k}\right)
\end{aligned}
$$

In a similar manner, the temperature equations for each node are derived. We note that during traverse along each coordinate direction in equations 26, 28, and 30, a tridiagonal matrix results for the solution of temperature.

2. Solution of the Nodal Temperature Equations

The x-direction nodal temperature equations for the rectangular block are of the form

$$
A_{1_{i,j,k}} T^{*}_{i-1,j,k} + B_{1_{i,j,k}} T^{*}_{i,j,k} + C_{1_{i,j,k}} T^{*}_{i+1,j,k} = D_{1_{i,j,k}} \qquad (31)
$$

where the coefficients $A_{1_{i,j,k}}, \ldots, D_{1_{i,j,k}}$ are known from the previous time step.

Each row of T* equations parallel to the x-axis is a tridiagonal matrix of coefficients $A_{1_{i,j,k}}$, $B_{1_{i,j,k}}$, and $C_{1_{i,j,k}}$. It was solved using the standard Tridiagonal Matrix Algorithm (TDMA) [Ref. 5]. After each row of T* equations is solved, the y-direction equations will be solved [Ref. 3]. These are of the type

$$A_{2_{i,j,k}} T^{**}_{i,j-1,k} + B_{2_{i,j,k}} T^{**}_{i,j,k} + C_{2_{i,j,k}} T^{**}_{i,j+1,k} = D_{2_{i,j,k}} \tag{32}$$

As in the x-direction case, each row of T^{**} equations parallel to the y-axis is solved using a tridiagonal matrix solver. The z-direction equations are of the form

$$A_{3_{i,j,k}} T^{n+1}_{i,j,k-1} + B_{3_{i,j,k}} T^{n+1}_{i,j,k} + C_{3_{i,j,k}} T^{n+1}_{i,j,k+1} = D_{3_{i,j,k}} \tag{33}$$

The solution of the T^{n+1} equations is accomplished in a similar manner as used for both the x- and y-directions [Ref. 3].

3. Numerical Calculations

A computer program (program (BRIAN) was written to evaluate the transient temperature distribution within the block using the Brian technique outlined in the previous sections. Figure 3 is a flow chart description of the program. A complete listing of program BRIAN is provided in Appendix D.

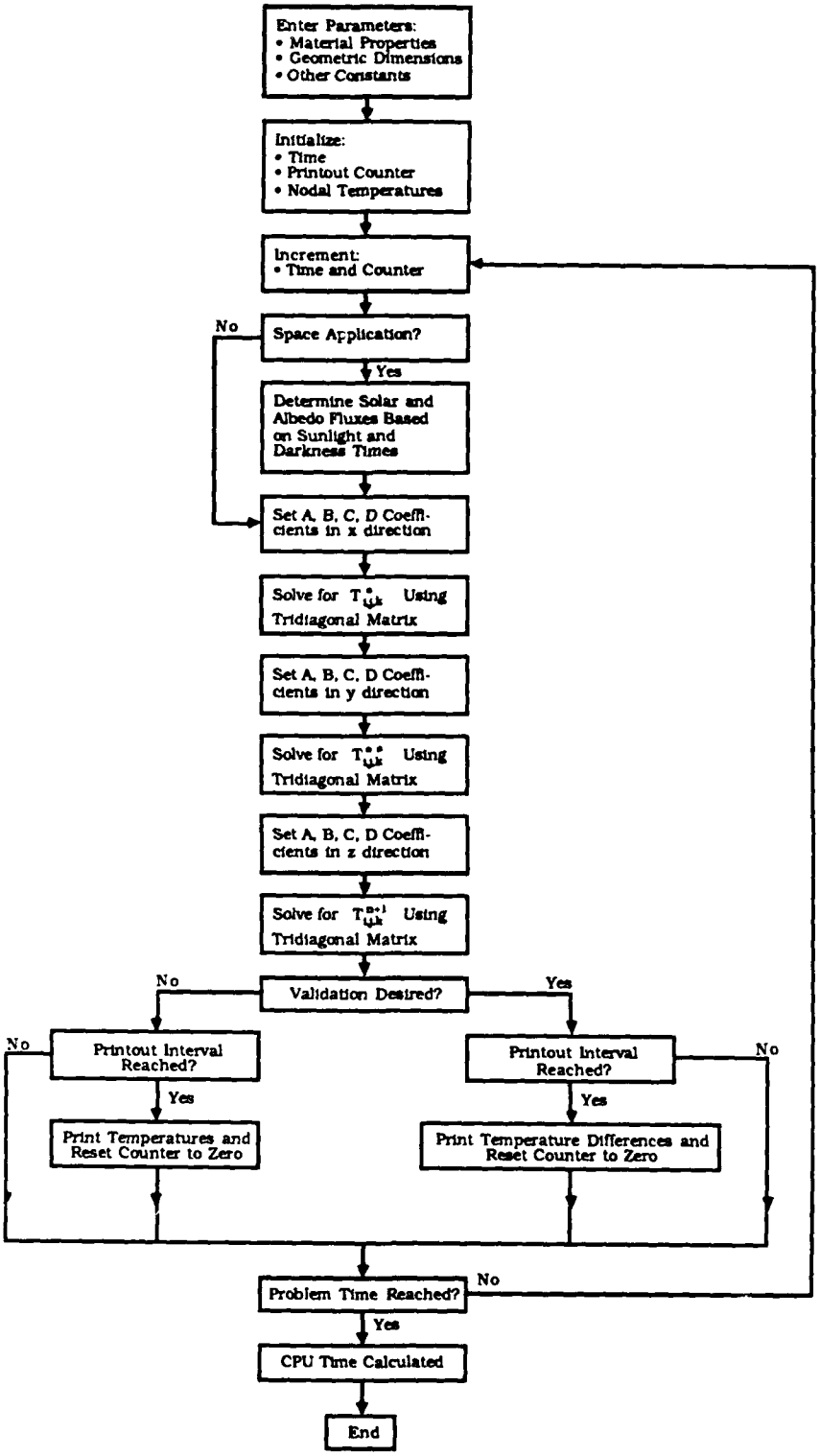

Figure 3. **Flow Chart for Program BRIAN**

III. VALIDATION

A. SELECTED GEOMETRY AND CONDITIONS

Both the Brian's technique and explicit technique programs required validation to ensure accuracy in the temperature solutions. Identical geometric and boundary conditions were used for these techniques in the validations. The left face had a constant surface heat flux with all other surfaces treated as adiabatic. Figure 4 shows the geometric and boundary conditions for the validation model. Table 1 lists the parameters used in the validation runs.

The Brian's technique and explicit technique programs were written to allow the user to conduct a validation test. The test involved taking the temperature differences at identical nodes in each program and then comparing each node's temperature difference on a graph.

The finite difference solutions for the rectangular block were also compared with an analytical solution for a semi-infinite solid exposed to a uniform flux at its surface. The closed-form solution would be a reasonable approximation to the actual solution for small times when interior nodes are not influenced by the lateral boundary surfaces of the rectangular block [Ref. 2].

Incropera and DeWitt [Ref. 2] tabulated the closed-form solution for the case of constant surface heat flux in terms of complementary error functions. These were evaluated in the present study based on

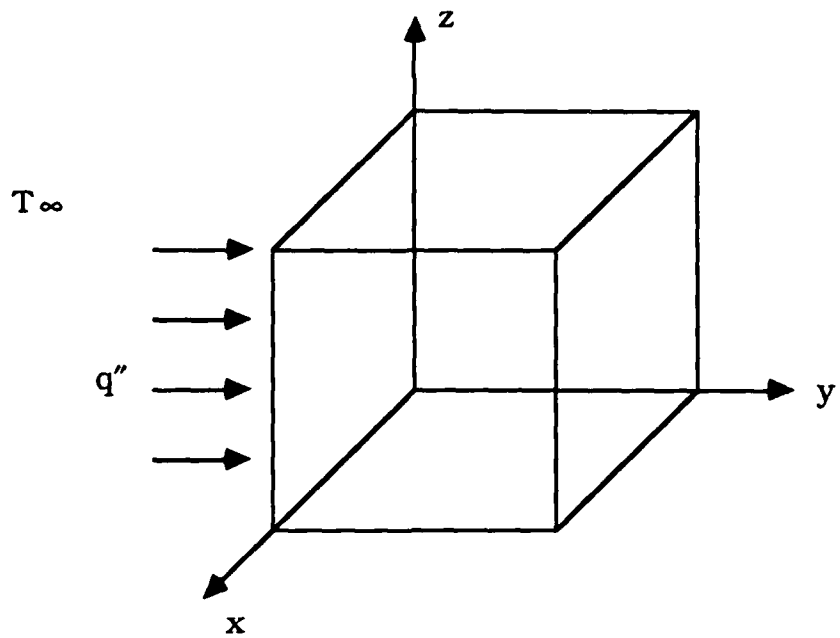

Figure 4. **Rectangular Model Showing Boundary Conditions Used in Validation Runs**

TABLE 1

PARAMETERS USED IN VALIDATION RUNS

Left face flux	1,500 W/m^2
Other faces	0 W/m^2 (adiabatic)
Material density	2,770 kg/m^3
Thermal conductivity	177 W/m-K
Specific Heat	875 J/kg-K
Initial Temperature	275K
Ambient Temperature	295 K
Distances from face	1.0, 0.4, 0.0 m

the polynomial approximations provided by Abramowitz and Stegun [Ref. 5].

B. NUMERICAL COMPUTATIONS

A computer program was written to evaluate the closed-form solution for a semi-infinite solid with a constant surface heat flux. The fifth-order polynomial approximations to the complementary error function have an error less than 1.5×10^{-7} [Ref. 5].

Figure 5 is a flow-chart description of the closed-form solution program VALID. The parameters entered are identical to those used in programs EXPLICIT and BRIAN. A complete listing of program VALID is provided in Appendix E. Table 1 lists the parameters used for the validation runs.

The validation period for programs VALID, EXPLICIT, and BRIAN was 3,600 seconds (one hour). The distance inside the semi-infinite solid was based on pre-selected nodes in the rectangular model. Three distances were used in the semi-infinite solid, corresponding to three selected nodes in the rectangular block.

In the first validation run, each coordinate direction had 11 nodes. This corresponded to an explicit technique time increment of 17.117 seconds, which was also selected as the time increment for programs BRIAN and VALID. In the second validation run, the number of nodes was increased to 21 in each coordinate direction. The time increment used in this run was 4.279 seconds. A third validation run was conducted using 11 nodes in each coordinate direction. The explicit technique and program VALID used a time increment of 17.117 seconds. The time increment for program BRIAN was

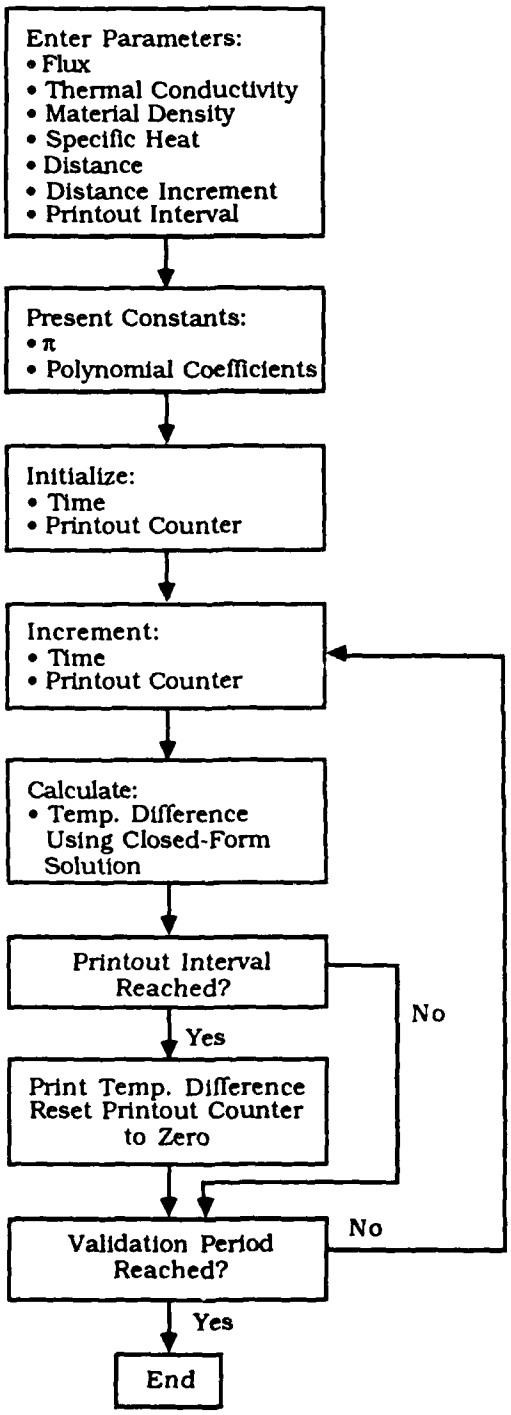

Figure 5. **Flow Chart for Program VALID**

increased to 120 seconds. The increased time increment was used to determine whether the program BRIAN temperature differences would differ significantly from those previously obtained with the 17.117-second time increment.

C. VALIDATION COMPUTATIONS

Graphical representations of the temperature differences experienced over the validation period are presented in Figures 6 through 14.

At a distance of one meter into the block (Figures 6 and 7), the temperature differences calculated in EXPLICIT and BRIAN diverge significantly from VALID after a period of 616 seconds. This is to be expected as the effects of the constant surface heat flux will provide higher temperatures in the rectangular block than in the semi-infinite solid. The rectangular block can thus be modeled as a semi-infinite solid for only about 10 minutes at a distance of one meter.

At a distance of 0.4 meters into the solid and on the surface (Figures 8 through 11), both EXPLICIT AND BRIAN temperature differences were in close agreement with the closed-form temperature differences for almost the entire validation period of one hour.

Increasing the number of nodes and, consequently, decreasing the time increment had virtually no effect on temperature differences, as can be seen in Figures 6 through 11.

Increasing the program BRIAN time increment to 120 seconds did not degrade the accuracy of the solution. In the three cases

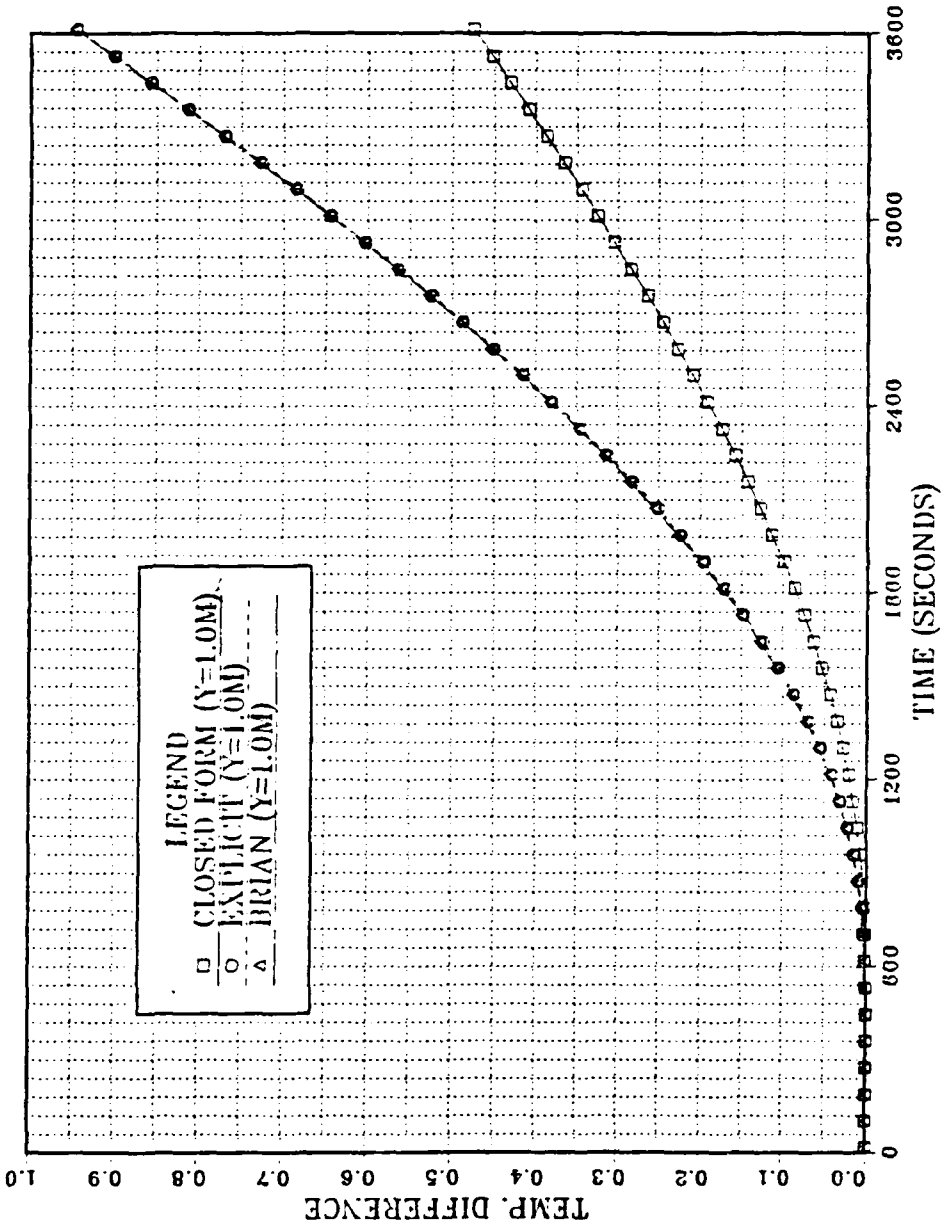

Figure 6. **Temperature Difference Versus Time for Time Increment of 17.117 Seconds with Distance of 1 Meter Into Block**

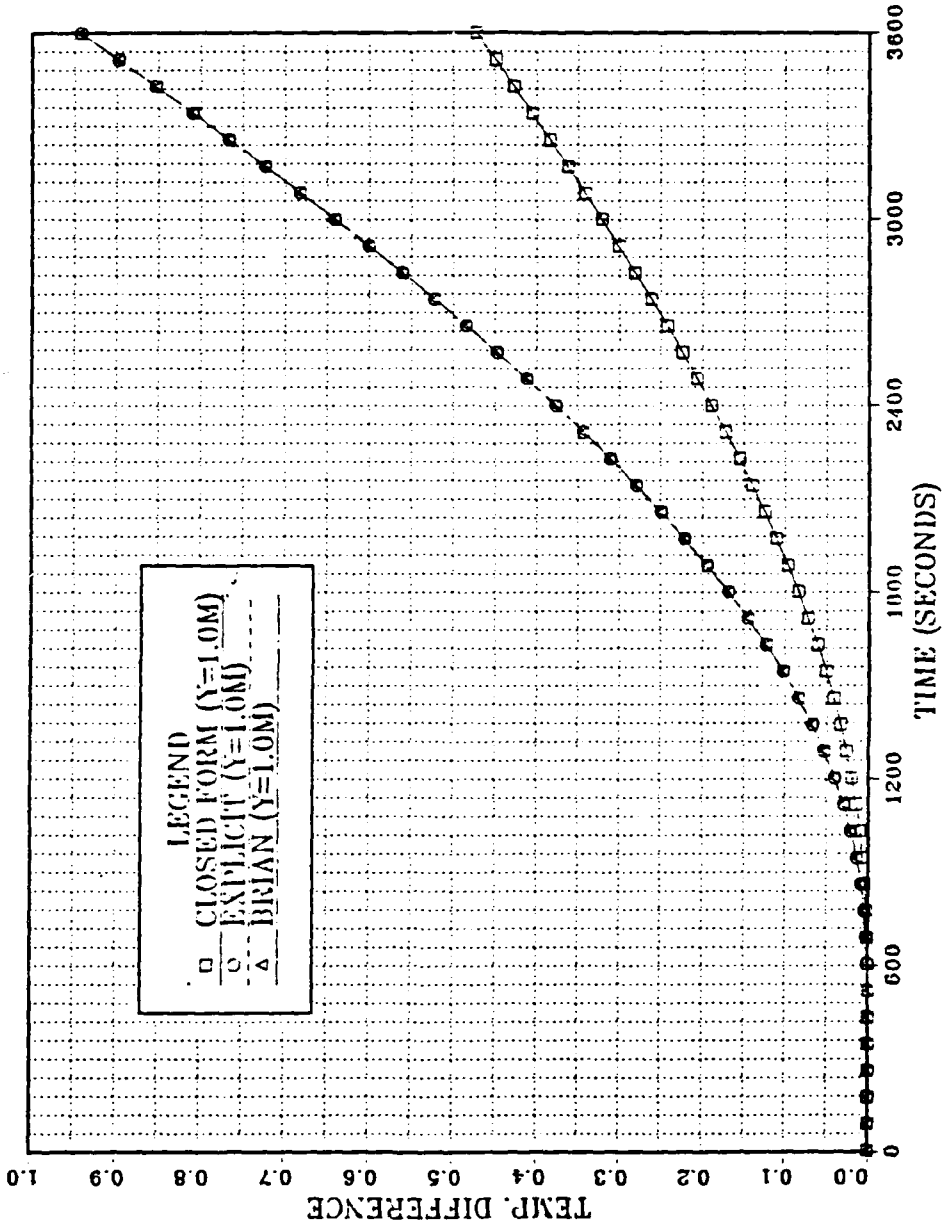

Figure 7. Temperature Difference Versus Time for Time Increment of 4.279 Seconds with Distance of 1 Meter Into Block

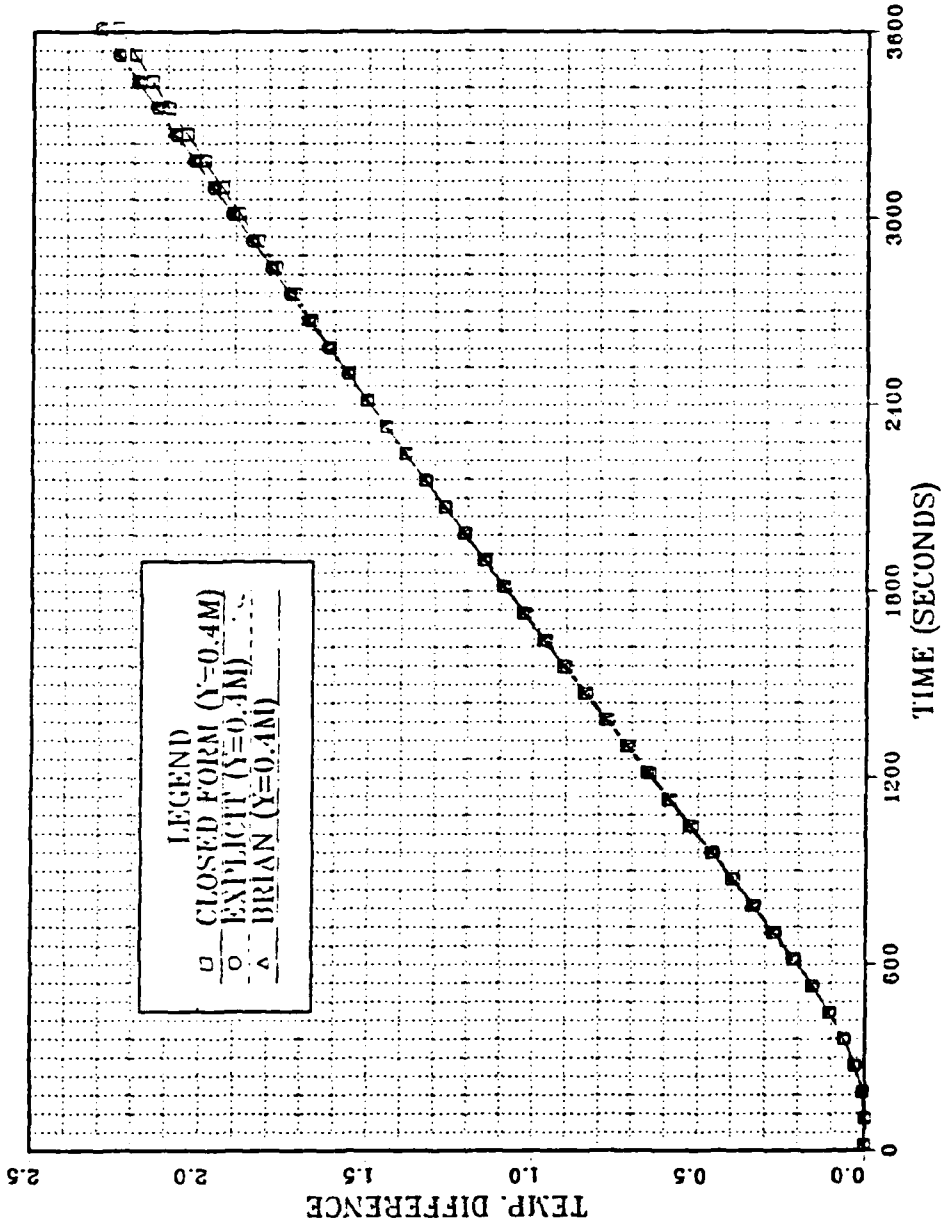

Figure 8. **Temperature Difference Versus Time for Time Increment of 17.117 Seconds with Distance of 0.4 Meters Into Block**

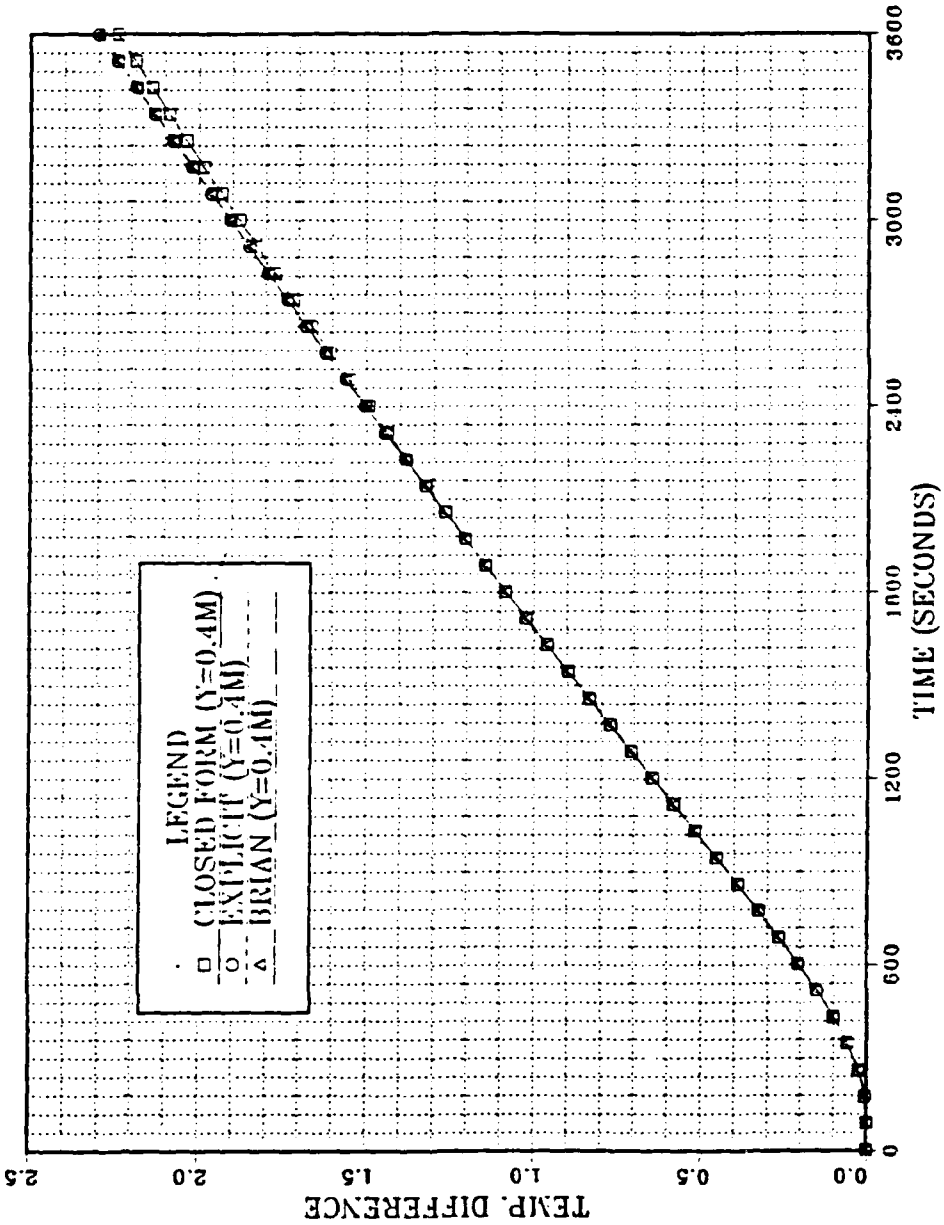

Figure 9. **Temperature Difference Versus Time for Time Increment of 4.279 Seconds with Distance of 0.4 Meters Into Block**

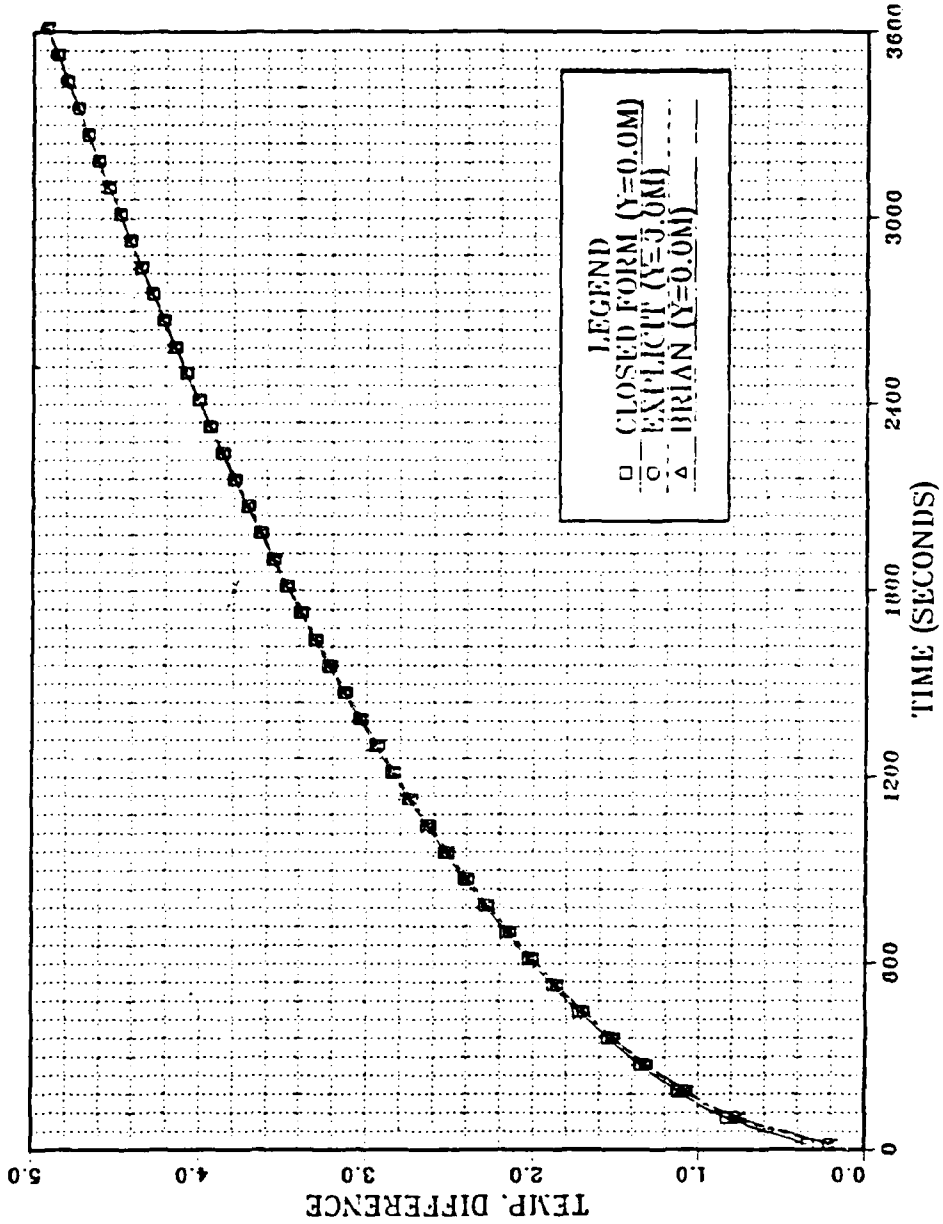

Figure 10. **Temperature Difference Versus Time for Time Increment of 17.117 Seconds on Left Surface of Block**

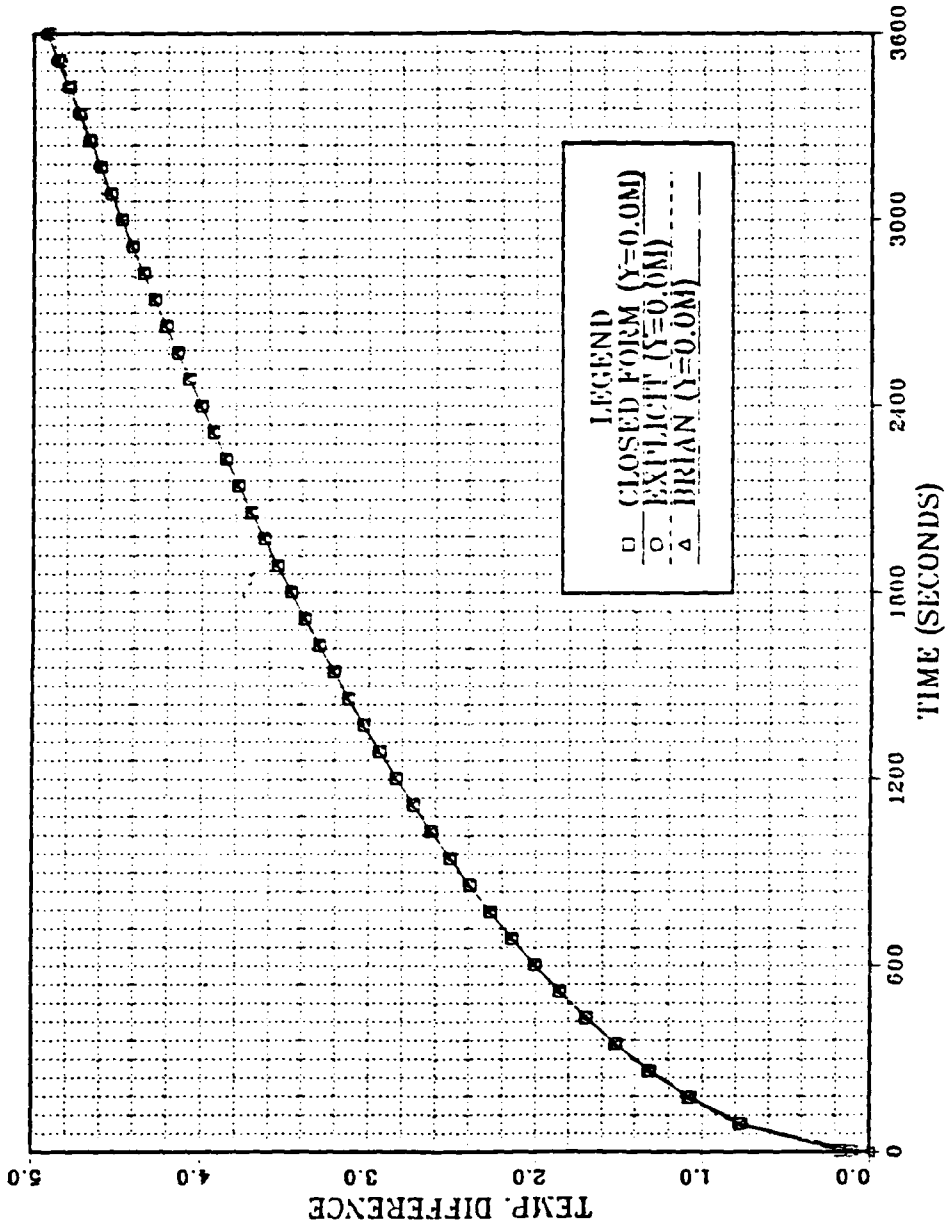

Figure 11. **Temperature Difference Versus Time for Time Increment of 4.279 Seconds on Left Surface of Block**

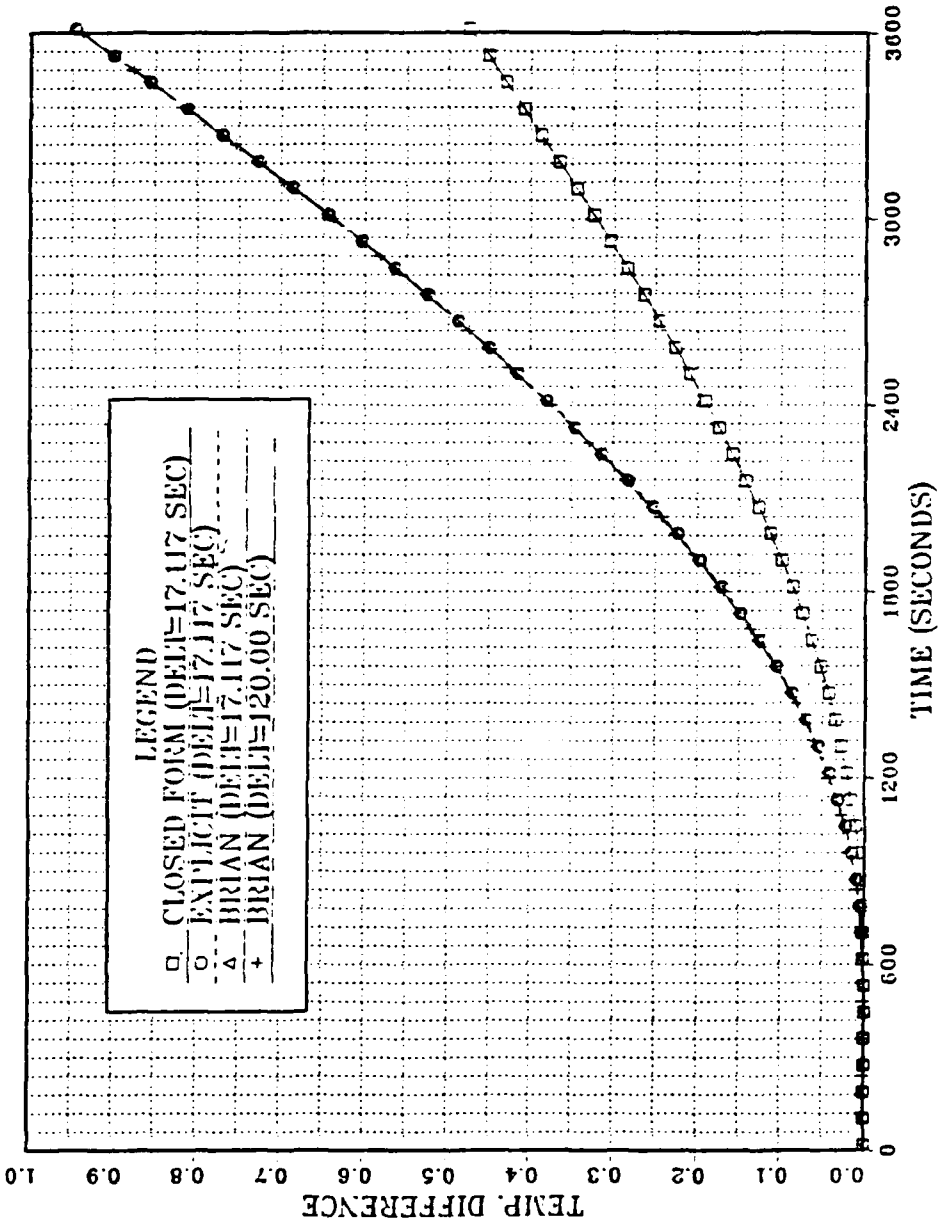

Figure 12. **Temperature Difference Versus Time for Time Increments of 17.117 and 120 Seconds with Distance of 1 Meter Into Block**

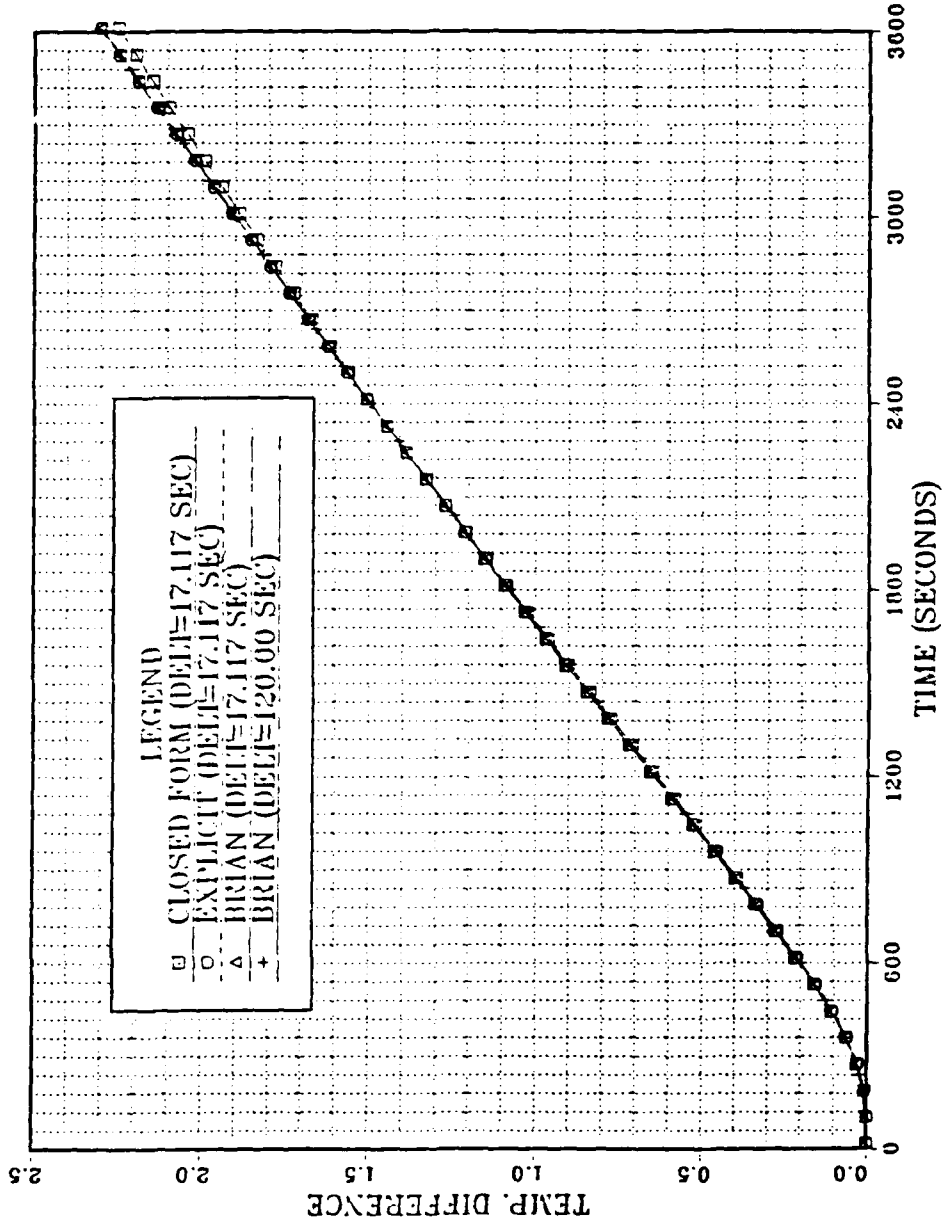

Figure 13. **Temperature Difference Versus Time for Time Increments of 17.117 and 120 Seconds with Distance of 0.4 Meters Into Block**

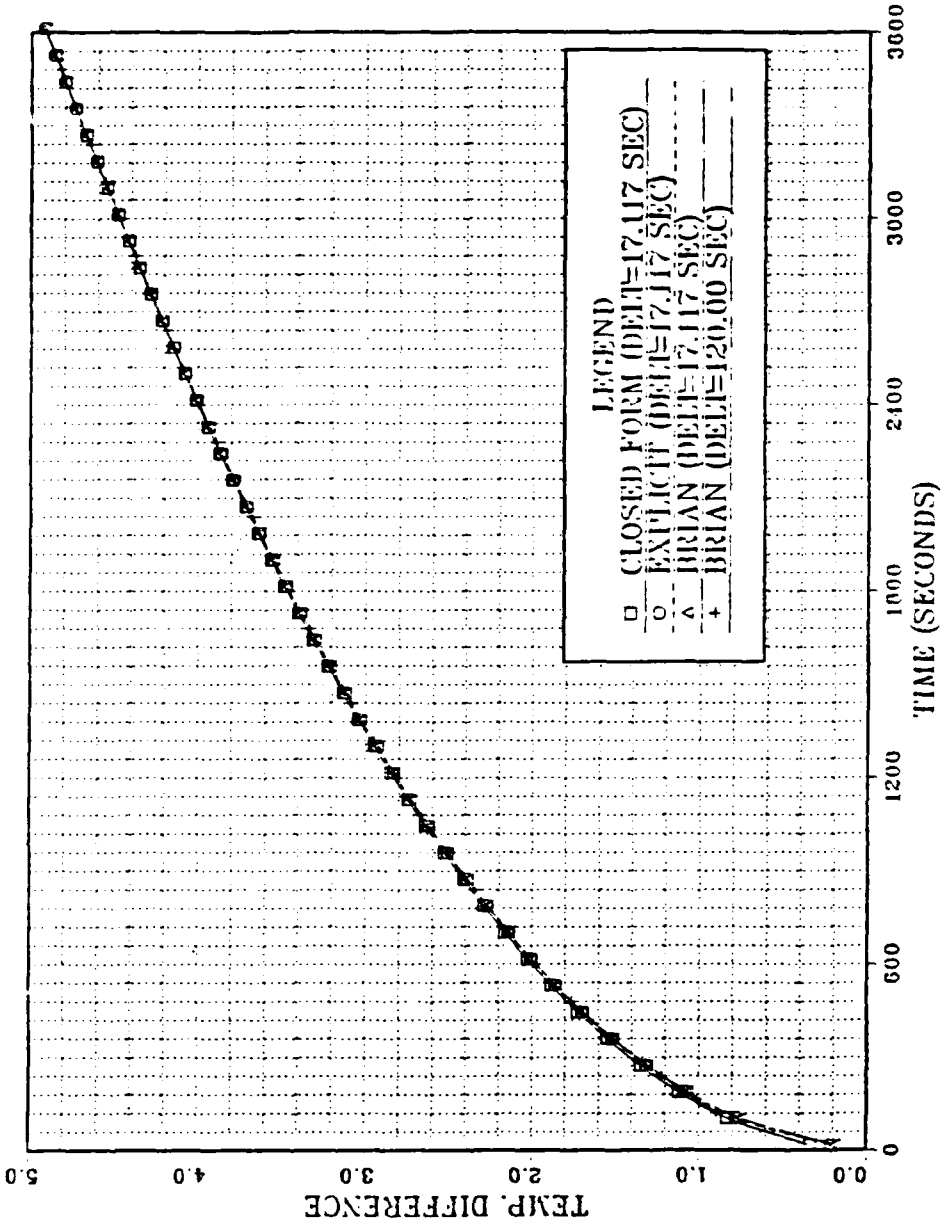

Figure 14. **Temperature Difference Versus Time for Time Increments of 17.117 and 120 Seconds on Left Surface of Block**

involving distances of 1.0 and 0.4 meters into the surface and on the surface, the temperature differences did not vary by more than 0.045 percent from those obtained using a 17.117-second time increment. Figures 12 through 14 show the 120-second time increment plot of temperature difference versus time being in close agreement to the corresponding 17.117-second time increment plots.

Using a time increment of 17.117 seconds, the maximum error was found to be within 0.019 percent for an 11 x 11 x 11 grid. The maximum error was calculated using

$$\%\,\mathrm{ERROR} = \frac{T_{\mathrm{COMPUTED}} - T_{\mathrm{EXACT}}}{T_{\mathrm{EXACT}}} \times 100\% \tag{34}$$

where T_{COMPUTED} is the Brian's technique computed temperature and T_{EXACT} is the closed-form temperature. Sampling points were taken at 600-second intervals.

In a similar manner, the maximum error was calculated for a time increment of 4.279 seconds. The maximum error was found to be 0.021 percent for a 21 x 21 x 21 grid.

A time increment of 120 seconds was above the maximum allowable time step from stability considerations for the explicit technique. Brian's technique thus allowed time steps above the limitation of the explicit technique with very little compromise in solution accuracy.

Based on the validation test results, program BRIAN can be considered correct and validated for further use.

IV. APPLICATIONS

A. GENERAL

The Brian technique was used to solve for the temperature distribution in an application example. The example involves the transient thermal response of a component on board an orbiting satellite.

B. ORBITING SATELLITE COMPONENT

1. Geometry and Initial Conditions

The transient thermal response of an orbiting spacecraft component will be investigated. The spacecraft component is in a non-geosynchronous circular orbit at an altitude of 1,609.364 kilometers (1,000 miles). The equation for the orbital period is listed in Agrawal [Ref. 7] as

$$P = 2\pi\sqrt{\frac{d^3}{\mu}} \tag{35}$$

where μ is the gravitational constant of the earth and d is the sum of the satellite altitude and radius of the earth. Eisele and Nichols [Ref. 8] list the values of μ and earth radius as 398,603.2 km^2/s^2 and 6,378.165 kilometers, respectively.

The period of revolution in this case is 7,104.4 seconds or 118.40 minutes. For a satellite in a circular orbit at an altitude of 100 miles, Stevenson and Grafton [Ref. 9] listed the sun exposure time as 72 percent of the orbital time. During one revolution around the earth,

the satellite will have 85.25 minutes of sun exposure and spend 33.15 minutes in the earth's shadow. For the example application, the satellite is assumed to start the orbital period at the instant it becomes exposed to the sun.

Single spin stabilization is assumed, thus allowing the spacecraft component to spin about a principal moment of inertia axis [Ref. 7]. The spacecraft component spins about the z-axis, as depicted in Figure 15. With single spin stabilization, it is assumed that the component analyzed here will have all four sides continuously exposed to the sun with the top and bottom surfaces not receiving solar flux. The fraction of solar flux that is incident on the satellite surface is usually defined as the solar aspect coefficient [Ref. 7]. Thus, for this example, the solar aspect coefficients are assumed to be 1.0 for the side surfaces and 0.0 for the top and bottom surfaces.

Stevenson and Grafton [Ref. 9] compiled the geometric factors for both earth radiation and reflected solar radiation (albedo) on a flat plate. The geometric factor for earth radiation depends on the satellite's altitude and the altitude angle between the normal to the rectangular component and the vertical to the satellite from the earth [Ref. 9]. Figure 16 depicts the geometry involved. The geometric factor for a rectangular plate at an altitude of 1,000 miles (1,609.364 kilometers) and altitude angle of 60 degrees is listed as 0.3174 [Ref. 9].

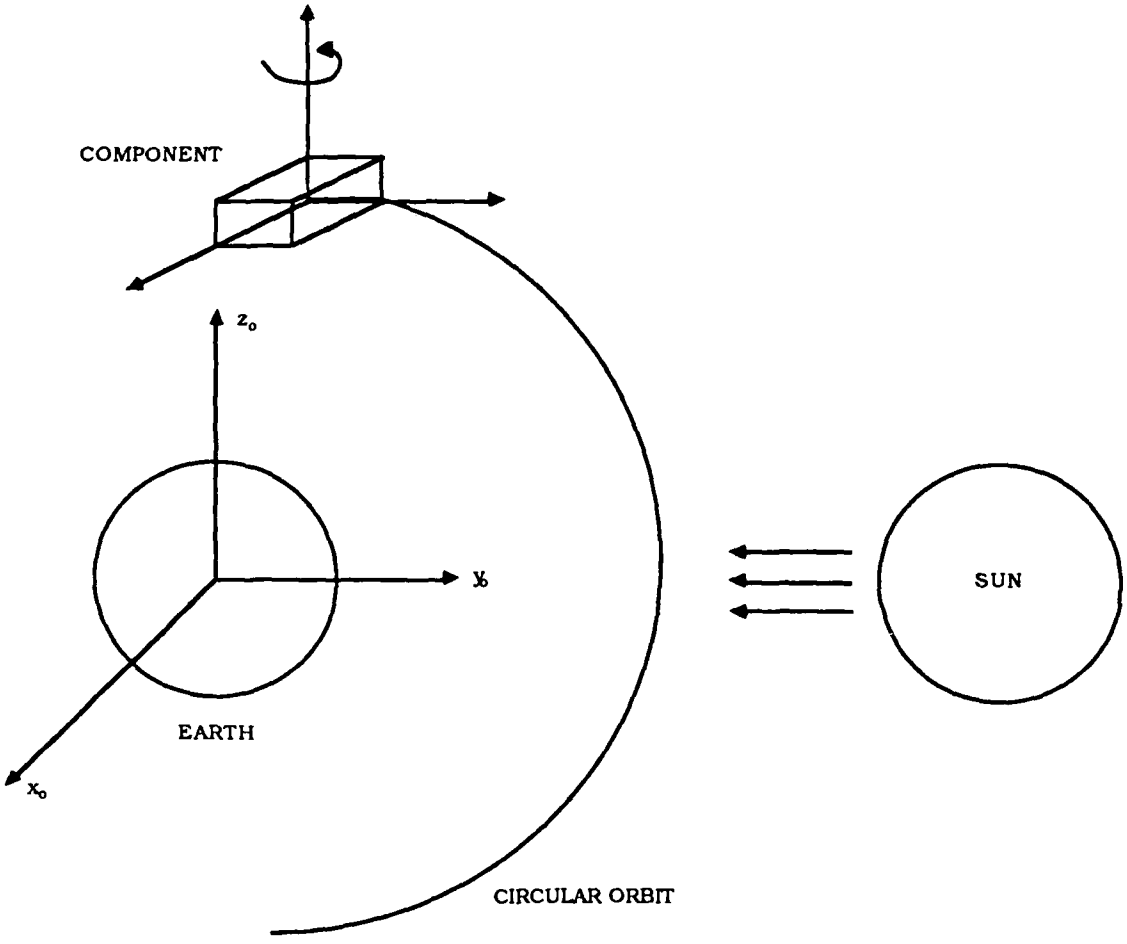

Figure 15. **Component on a Satellite in Circular Orbit**

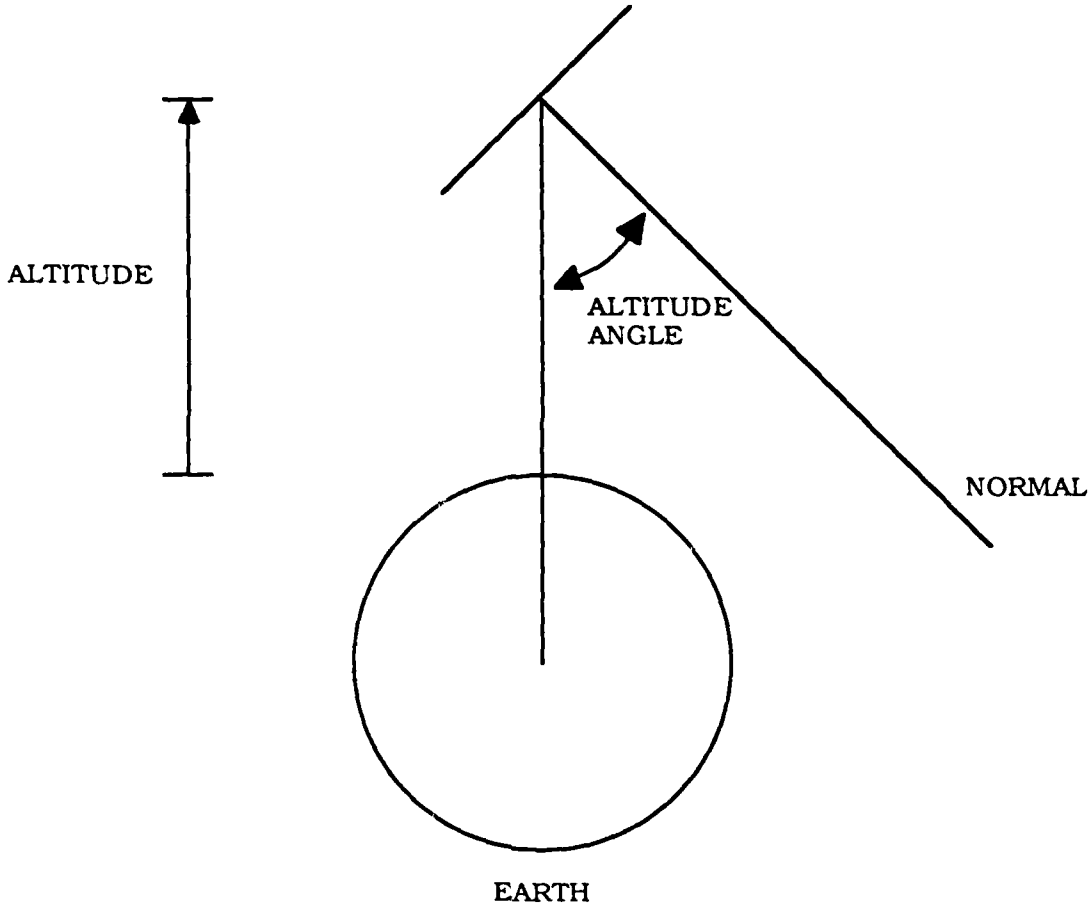

Figure 16. **Geometry Used in Determining Geometric Factor for Earth Thermal Radiation**

The geometric factor for reflected solar radiation is dependent on the satellite's altitude, the altitude angle, the sun angle, and the angle of axis rotation [Ref. 7]. Figure 17 depicts the geometry involved. The geometric factor is 0.0456 for a satellite at 1,000 miles altitude, altitude angle of 60 degrees, sun angle of 90 degrees, and rotation angle of 0 degrees [Ref. 9].

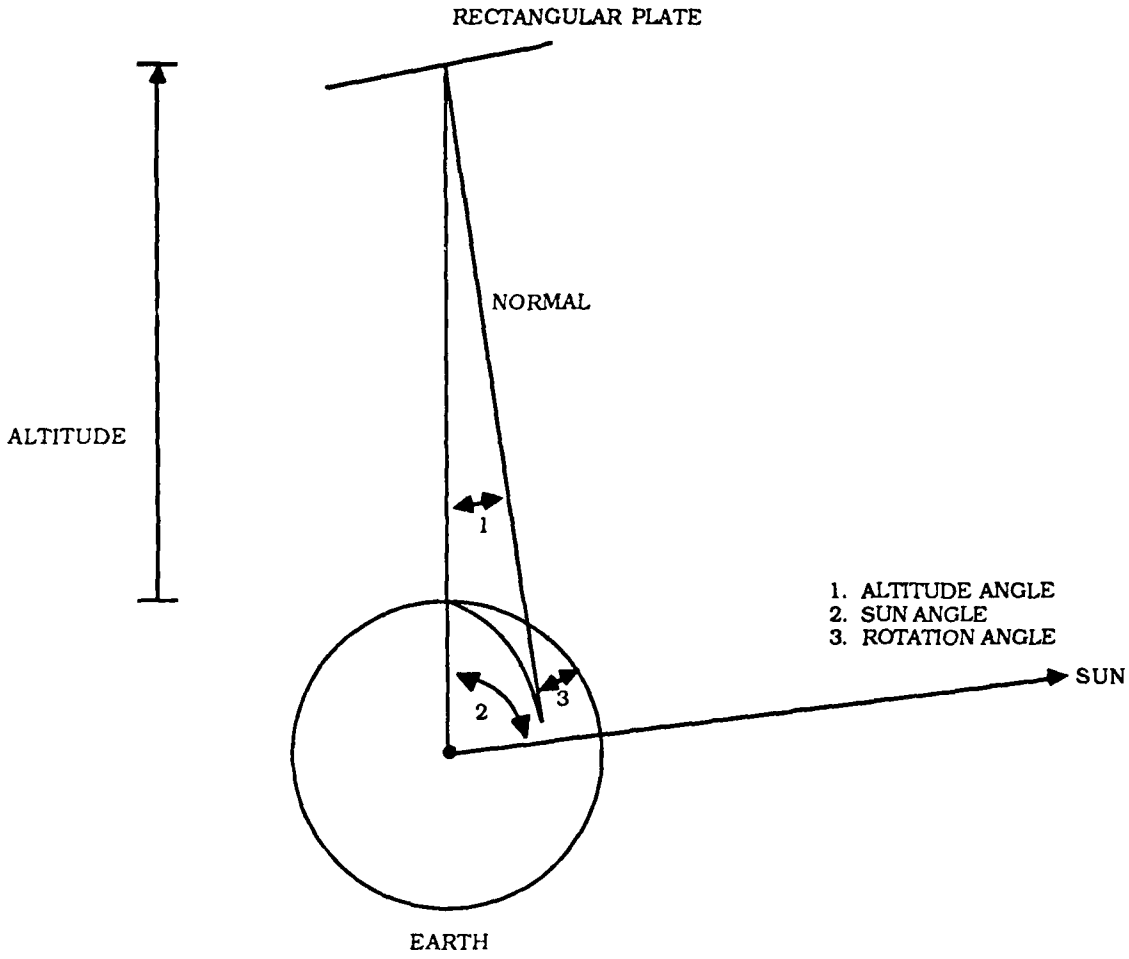

Figure 17. **Geometry Used in Determining Geometric Factor for Solar Reflected Radiation**

TABLE 2

PARAMETERS USED IN ORBITING SATELLITE APPLICATION

Dimensions	**0.15 m x 0.10 m x 0.015 m**
Material	aluminum alloy 2024-T6
• density	2,770 kg/m^3
• thermal conductivity	186 W/m-K
• specific heat	1,000 J/kg-K
Exterior Coating	black paint
• solar absorptivity	0.97
• emissivity	0.92
Imposed Flux	
• bottom surface	50,000 W/m^2
Orbital Data	
• altitude	1,609.364 km (1,000 mi)
• period of rotation	118.40 min
• exposure to sun	85.25 min
• exposure to darkness	33.15 min
Solar Aspect Coefficients	
• side surfaces	1.0
• top and bottom surfaces	0.0
External Flux	
• solar flux	1,353 W/m^2
• earth radiation	237 W/m^2
• albedo coefficient	0.3
Geometric Factors	
• earth radiation	0.3174
• albedo flux	0.0456
Initial Data	
• heat transfer coefficients	0.0 W/m^2-K
• ambient temperature	0 K
• initial nodal temperature	500 K

Agrawal [Ref. 7] listed the mean annual solar flux, mean annual earth radiation, and mean annual value of albedo coefficient as 1,353 W/m^2, 237 W/m^2, and 0.30, respectively. These value are used in the example.

The material coating on the component's exterior determines the solar absorptivity and emissivity. The component is assumed to be coated with black paint, thus providing a solar absorptivity of 0.97 and an emissivity of 0.92 [Ref. 10].

The component is rectangular with dimensions of 0.25 m x 0.10 m x 0.015 m and is made of aluminum alloy 2024-T6. Incropeia and DeWitt [Ref. 2] list the values for density, thermal conductivity, and specific heat.

The component is mounted on the exterior of the satellite. The bottom surface is attached to the satellite with all other surfaces exposed to the space environment. The flux on the bottom surface is assumed to be 50,000 W/m^2.

Table 2 lists the properties and other parameters used in the example.

2. Numerical Calculations

A time increment of 120 seconds (2 minutes) was used in program BRIAN. The program covered a period of 120,000 seconds (33.33 hours), during which the spacecraft component conducted 16.9 orbits. The temperatures of top surface node were plotted versus time, as shown in Figures 18 and 19. The temperature versus time

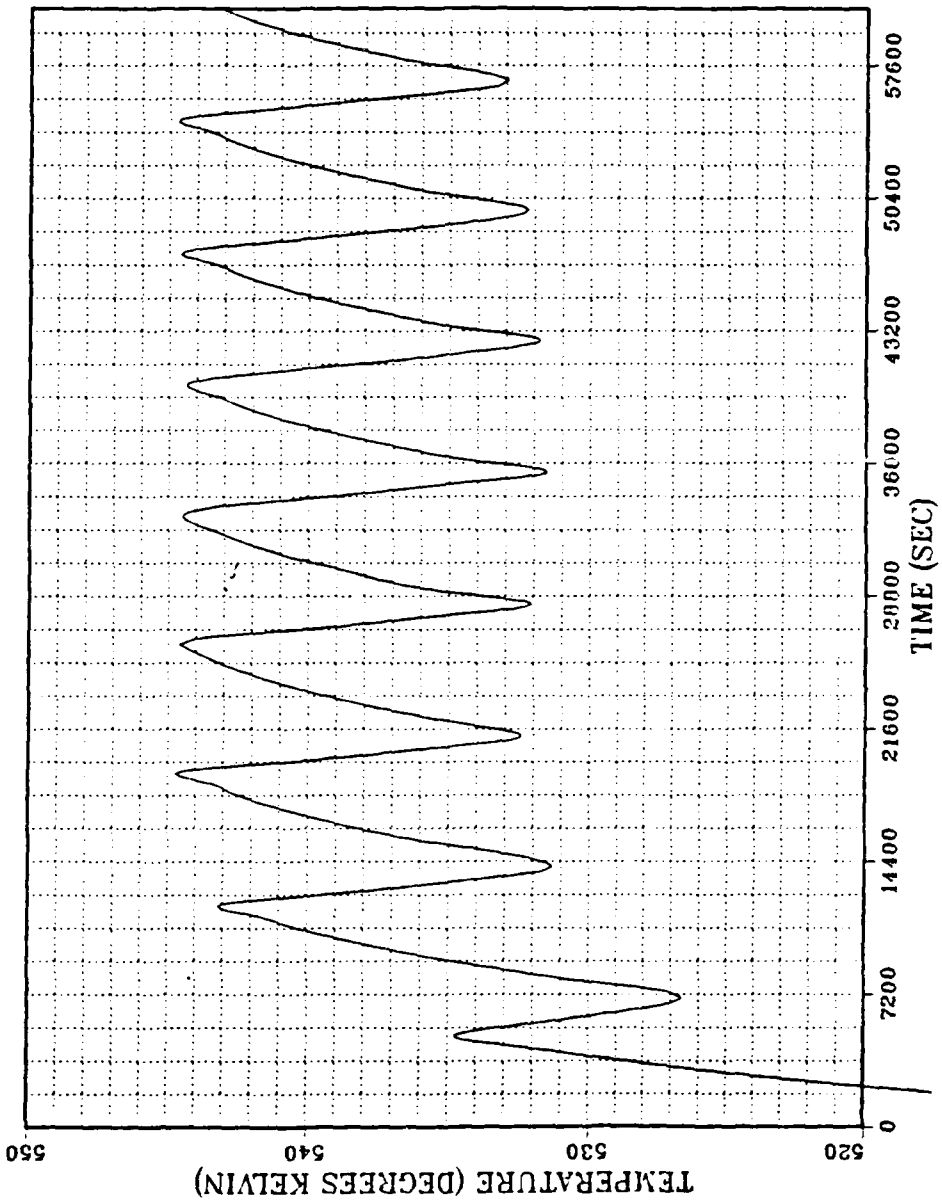

Figure 18. **Temperature Variation of the Top Surface Center Node**

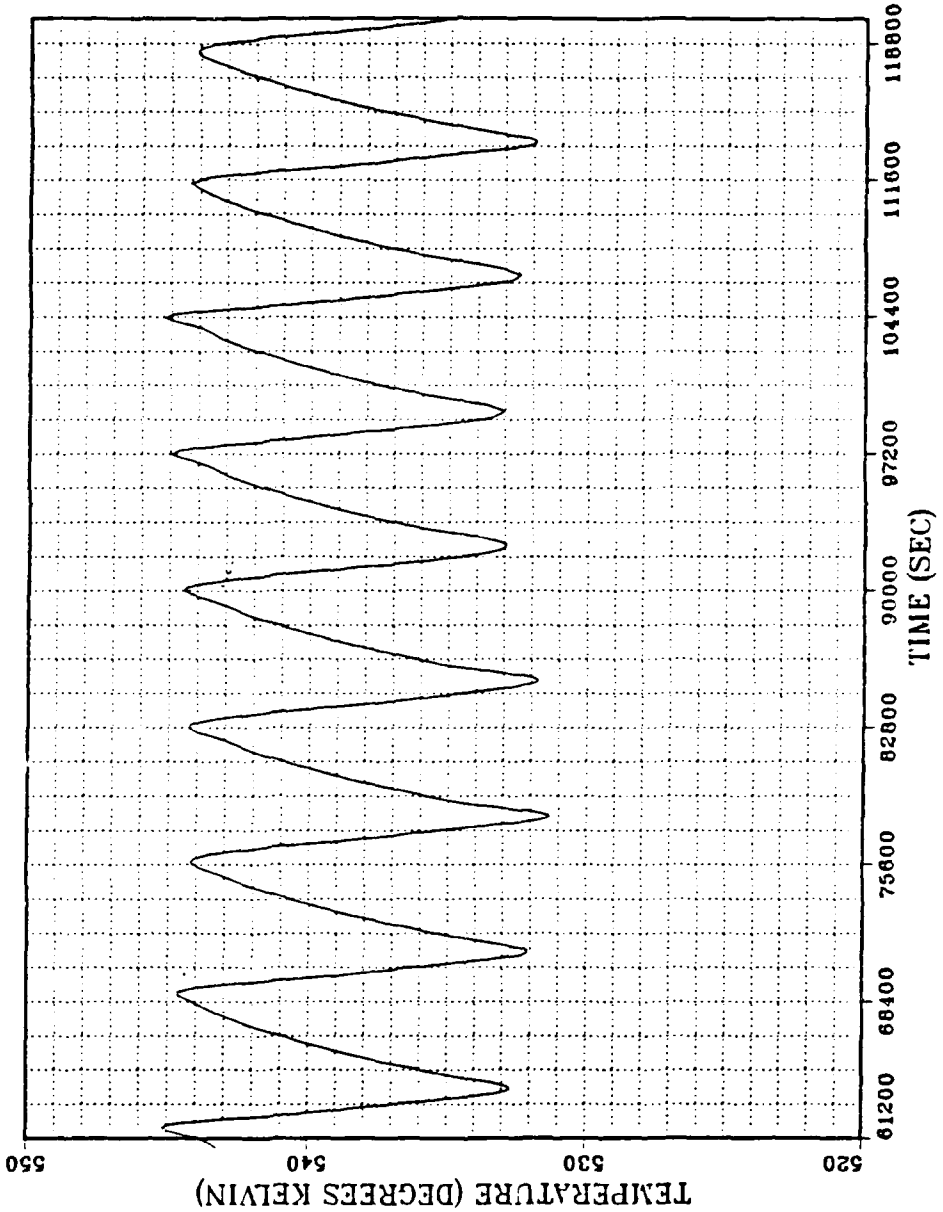

Figure 19. **Temperature Variation of the Top Surface Center Node**

plots in both figures demonstrate an imperfect sinusoid. Steady periodic response occurs following orbit. The peak temperatures during the third orbit vary from the second orbit peak temperatures by 0.27 percent. The largest temperature difference in successive peak temperatures is 0.27 percent and occurs between the second and third orbits.

The oscillations in temperature distribution may be attributed to overlaps in the sunlight and eclipse periods. The sunlight exposure time for the orbit is 85.25 minutes. A time increment of 120 seconds (2 minutes) would allow solar flux to be seen by the satellite component for an additional 1.25 minutes on the first orbit. Thus, 1.25 minutes of the first eclipse period would be missed at the beginning of the eclipse period. At the end of the eclipse period, the component is in darkness an extra 1.60 minutes. These overlaps occur throughout the running of the program with the maximum overlap being 2 minutes.

V. RESULTS

Brian's technique was validated for four of the six rectangular surfaces. For cases involving flux or convective boundary conditions on the front and back surfaces of the model, temperature distributions did not follow the pattern obtained in previous validation cases. This anomaly requires further investigation.

VI. CONCLUSIONS

Brian's technique was validated using the explicit technique and the closed form solution for a semi-infinite solid. For the same time increment, the explicit technique uses six times less CPU time. However, as the time increment was increased, the CPU time used in Brian's technique became less than that used in the explicit technique for time steps larger than 110 seconds. Brian's technique used time increments that would lead to instability if used in the explicit technique.

VII. RECOMMENDATIONS

While developing the computer code for Brian's technique, material properties were assumed to be independent of temperature. A more precise code would make thermal conductivity a function of temperature.

Another improvement for the computer code when used for spacecraft thermal analysis computations involves the application of orbital mechanics. This would permit the use of non-circular orbits in the code.

This study was restricted to the use of Cartesian coordinates in Brian's technique. Further study is required in applying Brian's technique in cylindrical and spherical coordinates.

APPENDIX A

EXPLICIT TECHNIQUE NODE EQUATIONS

A. INTERIOR NODE EQUATION

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + Fo\left(T^{n}_{i-1,j,k} + T^{n}_{i+1,j,k} - 2T^{n}_{i,j,k}\right)$$

$$+ R_1 Fo\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ R_2 Fo\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right)$$

B. SURFACE NODE EQUATIONS

1. Left Face

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + Fo\left(T^{n}_{i-1,j,k} + T^{n}_{i+1,j,k} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ FoR_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right)$$

$$+ 2Foq''R_1^{\frac{1}{2}}\left(T_{\infty} - T^{n}_{i,j,k}\right)$$

2. Right Face

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + Fo\left(T^{n}_{i-1,j,k} + T^{n}_{i+1,j,k} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ FoR_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right)$$

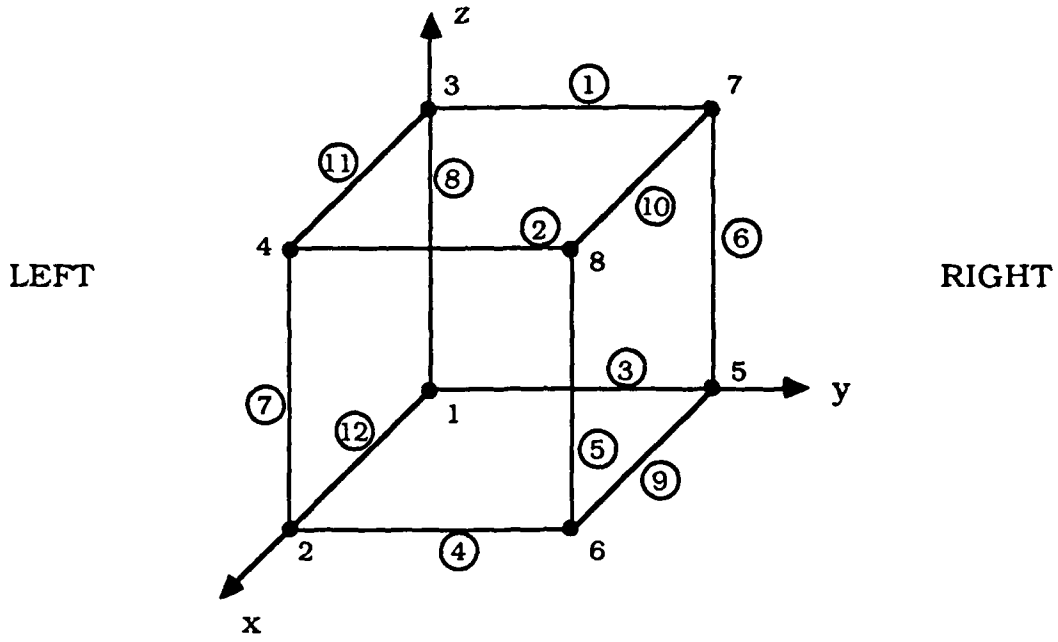

Figure 29. **Geometry of Rectangular Model**

3. Bottom Face

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + Fo\left(T^{n}_{i-1,j,k} + T^{n}_{i+1,j,k} - 2T^{n}_{i,j,k}\right)$$

$$+ FoR_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

4. Top Face

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + Fo\left(T^{n}_{i-1,j,k} + T^{n}_{i+1,j,k} - 2T^{n}_{i,j,k}\right)$$

$$+ FoR_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right)$$

5. Front Face

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i-1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ FoR_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ FoR_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right)$$

6. Back Face

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i+1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ FoR_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ FoR_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right)$$

C. PERIMETER NODE EQUATIONS

1. Perimeter 1

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i+1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ FoR_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right)$$

2. Perimeter 2

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i-1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ FoR_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right)$$

3. Perimeter 3

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i+1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ FoR_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

4. Perimeter 4

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i-1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ FoR_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

5. Perimeter 5

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i-1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ FoR_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right)$$

6. Perimeter 6

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i+1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ FoR_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right)$$

7. Perimeter 7

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i-1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ FoR_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoBiR_1^{\frac{1}{2}}\left(T_\infty - T^{n}_{i,j,k}\right) + 2Foq''R_1\frac{\Delta y}{k}$$

8. Perimeter 8

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i+1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ FoR_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoBiR_1^{\frac{1}{2}}\left(T_\infty - T^{n}_{i,j,k}\right) + 2Foq''R_1\frac{\Delta y}{k}$$

9. Perimeter 9

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + Fo\left(T^{n}_{i-1,j,k} + T^{n}_{i+1,j,k} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

10. Perimeter 10

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + Fo\left(T^{n}_{i-1,j,k} + T^{n}_{i+1,j,k} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right)$$

11. Perimeter 11

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + Fo\left(T^{n}_{i-1,j,k} + T^{n}_{i+1,j,k} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right)$$

$$+ 2FoBiR_1^{\frac{1}{2}}\left(T_\infty - T^{n}_{i,j,k}\right) + 2Foq''R_1\frac{\Delta y}{k}$$

12. Perimeter 12

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + Fo\left(T^{n}_{i-1,j,k} + T^{n}_{i+1,j,k} - 2T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

$$+ 2FoBiR_1^{\frac{1}{2}}\left(T_\infty - T^{n}_{i,j,k}\right) + 2Foq''R_1\frac{\Delta y}{k}$$

D. CORNER NODE EQUATIONS

1. Corner 1

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i+1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

$$+ 2Foq''R_1\frac{\Delta y}{k}$$

2. Corner 2

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i-1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

$$+ 2Foq''R_1\frac{\Delta y}{k}$$

3. Corner 3

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i+1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right)$$

$$+ 2Foq''R_1\frac{\Delta y}{k}$$

4. Corner 4

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i-1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right)$$

$$+ 2Foq''R_1\frac{\Delta y}{k}$$

5. Corner 5

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i+1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

$$+ 2FoBiR_1^{\frac{1}{2}}\left(T_\infty - T^{n}_{i,j,k}\right)$$

6. Corner 6

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i-1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

$$+ 2FoBiR_1^{\frac{1}{2}}\left(T_\infty - T^{n}_{i,j,k}\right)$$

7. Corner 7

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i+1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right)$$

$$+ 2FoBiR_1^{\frac{1}{2}}\left(T_\infty - T^{n}_{i,j,k}\right)$$

8. Corner 8

$$T^{n+1}_{i,j,k} = T^{n}_{i,j,k} + 2Fo\left(T^{n}_{i-1,j,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_1\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ 2FoR_2\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right)$$

$$+ 2FoBiR_1^{\frac{1}{2}}\left(T_\infty - T^{n}_{i,j,k}\right)$$

APPENDIX B

BRIAN'S TECHNIQUE NODE EQUATIONS

A. X-DIRECTION EQUATIONS

1. Corner 1

$$(1+2Fo)T^{*}_{i,j,k} - 2FoT^{*}_{i,j,k} = T^{n}_{i,j,k} + U_3\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ U_4\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right) + V_1\left(T_{\infty 1} - T^{n}_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^{n}_{i,j,k}\right)$$

$$+ V_6\left(T_{\infty 6} - T^{n}_{i,j,k}\right) + W_1 + W_3 + W_6 + X_1\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 1}\right)$$

$$+ X_3\left(T^{4}_{i,j,k} - T^{4}_{\infty 3}\right) + X_6\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 6}\right) + Y_1 E_{G_{i,j,k}}$$

2. Perimeter 12

$$-FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} - FoT_{i+1,j,k} =$$

$$T^{n}_{i,j,k} + U_3\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right) + U_4\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

$$+ V_1\left(T_{\infty 1} - T^{n}_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^{n}_{i,j,k}\right) + W_1 + W_3$$

$$+ X_1\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 1}\right) + X_3\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 3}\right) + Y_1 E_{G_{i,j,k}}$$

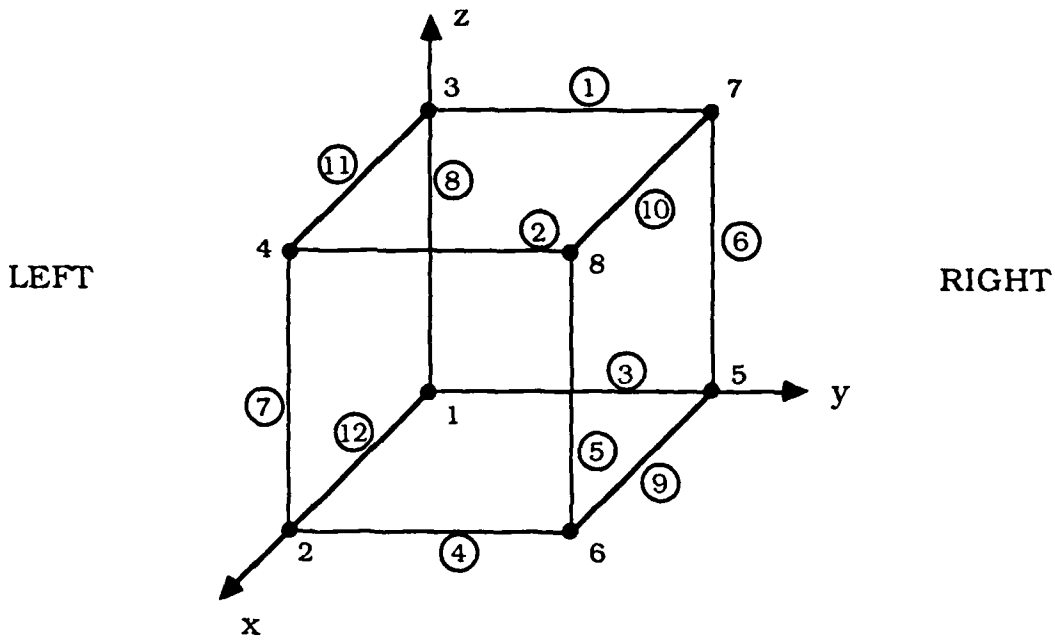

Figure 30. **Geometry of Rectangular Model**

3. Corner 2

$$-2FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} = T^{n}_{i,j,k} + U_3\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ U_4\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right) + V_1\left(T_{\infty 1} - T^{n}_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^{n}_{i,j,k}\right)$$

$$+ F_5\left(T_{\infty 5} - T^{n}_{i,j,k}\right) + W_1 + W_3 + W_5 + X_1\left(T^{n\,4}_{i,j,k} - T^{4}_{\infty 1}\right)$$

$$+ X_3\left(T^{n\,4}_{i,j,k} - T^{4}_{\infty 3}\right) + X_5\left(T^{n\,4}_{i,j,k} - T^{4}_{\infty 5}\right) + Y_1 E_{G\,i,j,k}$$

4. Perimeter 8

$$(1+2Fo)T^*_{i,j,k} - 2FoT^*_{i+1,j,k} = T^n_{i,j,k} + U_2\left(T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}\right)$$

$$+ U_3\left(T^n_{i,j+1,k} - T^n_{i,j,k}\right) + V_1\left(T_{\infty_1} - T^n_{i,j,k}\right) + V_6\left(T_{\infty_6} - T^n_{i,j,k}\right)$$

$$+ W_1 + W_6 + X_1\left(T^{n^4}_{i,j,k} - T^4_{\infty_1}\right) + X_6\left(T^{n^4}_{i,j,k} - T^4_{\infty_6}\right) + Y_1 E_{G_{i,j,k}}$$

5. Left Face

$$-FoT^*_{i-1,j,k} + (1+2Fo)T^*_{i,j,k} - FoT^*_{i+1,j,k} = T^n_{i,j,k}$$

$$+ U_2\left(T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}\right) + U_3\left(T^n_{i,j+1,k} - T^n_{i,j,k}\right)$$

$$+ V_1\left(T_{\infty_1} - T^n_{i,j,k}\right) + W_1 + X_1\left(T^{n^4}_{i,j,k} - T^4_{\infty_1}\right) + Y_1 E_{G_{i,j,k}}$$

6. Perimeter 7

$$-2FoT^*_{i-1,j,k} + (1+2Fo)T^*_{i,j,k} = T^n_{i,j,k}$$

$$+ U_2\left(T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}\right) + U_3\left(T^n_{i,j+1,k} - T^n_{i,j,k}\right)$$

$$+ V_1\left(T_{\infty_1} - T^n_{i,j,k}\right) + V_5\left(T_{\infty_5} - T^n_{i,j,k}\right) + W_1 + W_5$$

$$+ X_1\left(T^{n^4}_{i,j,k} - T^4_{\infty_1}\right) + X_5\left(T^{n^4}_{i,j,k} - T^4_{\infty_5}\right) + Y_1 E_{G_{i,j,k}}$$

7. Corner 3

$$(1+2Fo)T^{*}_{i,j,k} - 2FoT^{*}_{i+1,j,k} = T^{n}_{i,j,k} + U_3\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ U_4\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right) + V_1\left(T_{\infty_1} - T^{n}_{i,j,k}\right)$$

$$+ V_4\left(T_{\infty_4} - T^{n}_{i,j,k}\right) + V_6\left(T_{\infty_6} - T^{n}_{i,j,k}\right) + W_1 + W_4 + W_6$$

$$+ X_1\left(T^{n^4}_{i,j,k} - T^{4}_{\infty_1}\right) + X_4\left(T^{n^4}_{i,j,k} - T^{4}_{\infty_4}\right)$$

$$+ X_6\left(T^{n^4}_{i,j,k} - T^{4}_{\infty_6}\right) + Y_1 E_{G_{i,j,k}}$$

8. Perimeter 11

$$-FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} - FoT^{*}_{i+1,j,k} = T^{n}_{i,j,k}$$

$$+ U_3\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right) + U_4\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right) + V_1\left(T_{\infty_1} - T^{n}_{i,j,k}\right)$$

$$+ V_4\left(T_{\infty_4} - T^{n}_{i,j,k}\right) + W_1 + W_4 + X_1\left(T^{n^4}_{i,j,k} - T^{4}_{\infty_1}\right)$$

$$+ X_4\left(T^{n^4}_{i,j,k} - T^{4}_{\infty_4}\right) + Y_1 E_{G_{i,j,k}}$$

9. Corner 4

$$-2FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} = T^{n}_{i,j,k} + U_3\left(T^{n}_{i,j+1,k} - T^{n}_{i,j,k}\right)$$

$$+ U_4\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right) + V_1\left(T_{\infty_1} - T^{n}_{i,j,k}\right) + V_4\left(T_{\infty_4} - T^{n}_{i,j,k}\right)$$

$$+ V_5\left(T_{\infty_5} - T^{n}_{i,j,k}\right) + W_1 + W_4 + W_5 + X_1\left(T^{n^4}_{i,j,k} - T^{4}_{\infty_1}\right)$$

$$+ X_4\left(T^{n^4}_{i,j,k} - T^{4}_{\infty_4}\right) + X_5\left(T^{n^4}_{i,j,k} - T^{4}_{\infty_5}\right) + Y_1 E_{G_{i,j,k}}$$

10. Perimeter 3

$$(1+2Fo)T^{*}_{i,j,k} - 2FoT^{*}_{i+1,j,k} = T^{n}_{i,j,k} + U_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ U_4\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^{n}_{i,j,k}\right) + V_6\left(T_{\infty 6} - T^{n}_{i,j,k}\right)$$

$$+ W_3 + W_6 + X_3\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 3}\right) + X_6\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 6}\right) + Y_1 E_{G_{i,j,k}}$$

11. Bottom Face

$$-FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} - FoT^{*}_{i+1,j,k} = T^{n}_{i,j,k} +$$

$$U_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right) + U_4\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

$$+ V_3\left(T_{\infty 3} - T^{n}_{i,j,k}\right) + W_3 + X_3\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 3}\right) + Y_1 E_{G_{i,j,k}}$$

12. Perimeter 4

$$-2FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} = T^{n}_{i,j,k} +$$

$$U_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right) + U_4\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

$$+ V_3\left(T_{\infty 3} - T^{n}_{i,j,k}\right) + V_5\left(T_{\infty 5} - T^{n}_{i,j,k}\right) + W_3 + W_5$$

$$+ X_3\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 3}\right) + X_5\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 5}\right) + Y_1 E_{G_{i,j,k}}$$

13. Back Face

$$(1+2Fo)T^{*}_{i,j,k} - 2FoT^{*}_{i+1,j,k} = T^{n}_{i,j,k} + U_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ U_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right) + V_6\left(T_{\infty 6} - T^{n}_{i,j,k}\right) + W_6$$

$$+ X_6\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 6}\right) + Y_1 E_{G\,i,j,k}$$

14. Interior

$$-FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} - FoT^{*}_{i+1,j,k} = T^{n}_{i,j,k}$$

$$+ U_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ U_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right) + Y_1 E_{G\,i,j,k}$$

15. Front Face

$$-2FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} = T^{n}_{i,j,k} + U_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ U_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right) + V_5\left(T_{\infty 5} - T^{n}_{i,j,k}\right)$$

$$+ W_5 + X_5\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 5}\right) + Y_1 E_{G\,i,j,k}$$

16. Perimeter 1

$$(1+2Fo)T^{*}_{i,j,k} - 2FoT^{*}_{i+1,j,k} = T^{n}_{i,j,k} + U_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ U_4\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right) + V_4\left(T_{\infty 4} - T^{n}_{i,j,k}\right) + V_6\left(T_{\infty 6} - T^{n}_{i,j,k}\right)$$

$$+ W_4 + W_6 + X_4\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 4}\right) + X_6\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 6}\right) + Y_1 E_{G\,i,j,k}$$

17. Top Face

$$-FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} - FoT^{*}_{i+1,j,k} = T^{n}_{i,j,k}$$

$$+ U_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right) + U_4\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right)$$

$$+ V_4\left(T_{\infty 4} - T^{n}_{i,j,k}\right) + W_4 + X_4\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 4}\right) + Y_1 E_{G\,i,j,k}$$

18. Perimeter 2

$$-2FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} = T^{n}_{i,j,k} + U_1\left(T^{n}_{i,j-1,k} + T^{n}_{i,j+1,k} - 2T^{n}_{i,j,k}\right)$$

$$+ U_4\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right) + V_4\left(T_{\infty 4} - T^{n}_{i,j,k}\right) + V_5\left(T_{\infty 5} - T^{n}_{i,j,k}\right)$$

$$+ W_4 + W_5 + X_4\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 4}\right) + X_5\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 5}\right) + Y_1 E_{G\,i,j,k}$$

19. Corner 5

$$(1+2Fo)T^{*}_{i,j,k} - 2FoT^{*}_{i+1,j,k} = T^{n}_{i,j,k} + U_3\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ U_4\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^{n}_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^{n}_{i,j,k}\right)$$

$$+ V6\left(T_{\infty 6} - T^{n}_{i,j,k}\right) + W_2 + W_3 + W_6 + X_2\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 2}\right)$$

$$+ X_3\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 3}\right) + X_6\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 6}\right) + Y_1 E_{G\,i,j,k}$$

20. Perimeter 9

$$-FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} - FoT^{*}_{i+1,j,k} = T^{n}_{i,j,k} + U_3\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ U_4\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^{n}_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^{n}_{i,j,k}\right)$$

$$+ W_2 + W_3 + X_2\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 2}\right) + X_3\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 3}\right) + Y_1 E_{G\,i,j,k}$$

21. Corner 6

$$-2FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} = T^{n}_{i,j,k} + U_3\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ U_4\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^{n}_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^{n}_{i,j,k}\right)$$

$$+ V_5\left(T_{\infty 5} - T^{n}_{i,j,k}\right) + W_2 + W_3 + W_5 + X_2\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 2}\right)$$

$$+ X_3\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 3}\right) + X_5\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 5}\right) + Y_1 E_{G\,i,j,k}$$

22. Perimeter 6

$$(1+2Fo)T^{*}_{i,j,k} - 2FoT^{*}_{i+1,j,k} = T^{n}_{i,j,k} + U_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right)$$

$$+ U_3\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^{n}_{i,j,k}\right) + V_6\left(T_{\infty 6} - T^{n}_{i,j,k}\right)$$

$$+ W_2 + W_6 + X_2\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 2}\right) + X_6\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 6}\right) + Y_1 E_{G\,i,j,k}$$

23. Right Face

$$-FoT^*_{i-1,j,k} + (1+2Fo)T^*_{i,j,k} - FoT^*_{i+1,j,k} = T^n_{i,j,k}$$

$$+ U_2\left(T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}\right) + U_3\left(T^n_{i,j-1,k} - T^n_{i,j,k}\right)$$

$$+ V_2\left(T_{\infty 2} - T^n_{i,j,k}\right) + W_2 + X_2\left(T^{n^4}_{i,j,k} - T^4_{\infty 2}\right) + Y_1 E_{G\,i,j,k}$$

24. Perimeter 5

$$-2FoT^*_{i-1,j,k} + (1+2Fo)T^*_{i,j,k} = T^n_{i,j,k} + U_2\left(T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}\right)$$

$$+ U_3\left(T^n_{i,j-1,k} - T^n_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^n_{i,j,k}\right) + V_5\left(T_{\infty 5} - T^n_{i,j,k}\right)$$

$$+ W_2 + W_5 + X_2\left(T^{n^4}_{i,j,k} - T^4_{\infty 2}\right) + X_5\left(T^{n^4}_{i,j,k} - T^4_{\infty 5}\right) + Y_1 E_{G\,i,j,k}$$

25. Corner 7

$$(1+2Fo)T^*_{i,j,k} - 2FoT^*_{i+1,j,k} = T^n_{i,j,k} + U_3\left(T^n_{i,j-1,k} - T^n_{i,j,k}\right)$$

$$+ U_4\left(T^n_{i,j,k-1} - T^n_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^n_{i,j,k}\right) + V_4\left(T_{\infty 4} - T^n_{i,j,k}\right)$$

$$+ V_6\left(T_{\infty 6} - T^n_{i,j,k}\right) + W_2 + W_4 + W_6 + X_2\left(T^{n^4}_{i,j,k} - T^4_{\infty 2}\right)$$

$$+ X_4\left(T^{n^4}_{i,j,k} - T^4_{\infty 4}\right) + X_6\left(T^{n^4}_{i,j,k} - T^4_{\infty 6}\right) + Y_1 E_{G\,i,j,k}$$

26. Perimeter 10

$$-FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} - FoT^{*}_{i+1,j,k} = T^{n}_{i,j,k}$$

$$+ U_3\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right) + U_4\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right)$$

$$+ V_2\left(T_{\infty 2} - T^{n}_{i,j,k}\right) + V_4\left(T_{\infty 4} - T^{n}_{i,j,k}\right) + W_2 + W_4$$

$$+ X_2\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 2}\right) + X_4\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 4}\right) + Y_1 E_{G\,i,j,k}$$

27. Corner 8

$$-2FoT^{*}_{i-1,j,k} + (1+2Fo)T^{*}_{i,j,k} = T^{n}_{i,j,k} + U_3\left(T^{n}_{i,j-1,k} - T^{n}_{i,j,k}\right)$$

$$+ U_4\left(T^{n}_{i,j,k-1} - T^{n}_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^{n}_{i,j,k}\right) + V_4\left(T_{\infty 4} - T^{n}_{i,j,k}\right)$$

$$+ V_5\left(T_{\infty 5} - T^{n}_{i,j,k}\right) + W_2 + W_4 + W_5 + X_2\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 2}\right)$$

$$+ X_4\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 4}\right) + X_5\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 5}\right) + Y_1 E_{G\,i,j,k}$$

B. Y-DIRECTION EQUATIONS

1. Corner 1

$$(1+2FoR_1)T^{**}_{i,j,k} - 2FoR_1 T^{**}_{i,j+1,k} = T^{n}_{i,j,k} + U_4\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

$$+ U_5\left(T^{*}_{i+1,j,k} - T^{*}_{i,j,k}\right) + V_1\left(T_{\infty 1} - T^{n}_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^{n}_{i,j,k}\right)$$

$$+ V_6\left(T_{\infty 6} - T^{n}_{i,j,k}\right) + W_1 + W_3 + W_6 + X_1\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 1}\right)$$

$$+ X_3\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 3}\right) + X_6\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 6}\right) + Y_1 E_{G\,i,j,k}$$

2. Perimeter 3

$$-FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} - FoR_1 T^{**}_{i,j+1,k} =$$

$$T^n_{i,j,k} + U_4\left(T^n_{i,j,k+1} - T^n_{i,j,k}\right) + U_5\left(T^*_{i+1,j,k} - T^*_{i,j,k}\right)$$

$$+ V_3\left(T_{\infty 3} - T^n_{i,j,k}\right) + V_6\left(T_{\infty 6} - T^n_{i,j,k}\right) + W_3 + W_6$$

$$+ X_3\left(T^{n^4}_{i,j,k} - T^4_{\infty 3}\right) + X_6\left(T^{n^4}_{i,j,k} - T^4_{\infty 6}\right) + Y_1 E_{G_{i,j,k}}$$

3. Corner 5

$$-2FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} = T^n_{i,j,k} + U_4\left(T^n_{i,j,k+1} - T^n_{i,j,k}\right)$$

$$+ U_5\left(T^*_{i+1,j,k} - T^*_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^n_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^n_{i,j,k}\right)$$

$$+ V_6\left(T_{\infty 6} - T^n_{i,j,k}\right) + W_2 + W_3 + W_6 + X_2\left(T^{n^4}_{i,j,k} - T^4_{\infty 2}\right)$$

$$+ X_3\left(T^{n^4}_{i,j,k} - T^4_{\infty 3}\right) + X_6\left(T^{n^4}_{i,j,k} - T^4_{\infty 6}\right) + Y_1 E_{G_{i,j,k}}$$

4. Perimeter 8

$$(1+2FoR_1)T^*_{i,j,k} - 2FoR_1 T^*_{i,j+1,k} = T^n_{i,j,k} + U_2\left(T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}\right)$$

$$+ U_5\left(T^*_{i+1,j,k} - T^*_{i,j,k}\right) + V_1\left(T_{\infty 1} - T^n_{i,j,k}\right) + V_6\left(T_{\infty 6} - T^n_{i,j,k}\right)$$

$$+ W_1 + W_6 + X_1\left(T^{n^4}_{i,j,k} - T^4_{\infty 1}\right) + X_6\left(T^{n^4}_{i,j,k} - T^4_{\infty 6}\right) + Y_1 E_{G_{i,j,k}}$$

5. Back Face

$$-FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} - FoR_1 T^{**}_{i,j+1,k} = T^n_{i,j,k}$$

$$+ U_2\left(T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}\right) + U_5\left(T^*_{i+1,j,k} - T^*_{i,j,k}\right) +$$

$$+ V_6\left(T_{\infty 6} - T^n_{i,j,k}\right) + W_6 + X_6\left(T^{n^4}_{i,j,k} - T^4_{\infty 6}\right) + Y_1 E_{G\,i,j,k}$$

6. Perimeter 6

$$-2FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} = T^n_{i,j,k} + U_2\left(T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}\right)$$

$$+ U_5\left(T^*_{i+1,j,k} - T^*_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^n_{i,j,k}\right) + V_6\left(T_{\infty 6} - T^n_{i,j,k}\right)$$

$$+ W_2 + W_6 + X_2\left(T^{n^4}_{i,j,k} - T^4_{\infty 2}\right) + X_6\left(T^{n^4}_{i,j,k} - T^4_{\infty 6}\right) + Y_1 E_{G\,i,j,k}$$

7. Corner 3

$$(1+2FoR_1)T^*_{i,j,k} - 2FoR_1 T^*_{i,j+1,k} = T^n_{i,j,k} + U_4\left(T^n_{i,j,k-1} - T^n_{i,j,k}\right)$$

$$+ U_5\left(T^*_{i+1,j,k} - T^*_{i,j,k}\right) + V_1\left(T_{\infty 1} - T^n_{i,j,k}\right) + V_4\left(T_{\infty 4} - T^n_{i,j,k}\right)$$

$$+ V_6\left(T_{\infty 6} - T^n_{i,j,k}\right) + W_1 + W_4 + W_6 + X_1\left(T^{n^4}_{i,j,k} - T^4_{\infty 1}\right)$$

$$+ X_4\left(T^{n^4}_{i,j,k} - T^4_{\infty 4}\right) + X_6\left(T^{n^4}_{i,j,k} - T^4_{\infty 6}\right) + Y_1 E_{G\,i,j,k}$$

8. Perimeter 1

$$-FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} - FoR_1 T^{**}_{i,j+1,k} = T^n_{i,j,k}$$

$$+ U_4\left(T^n_{i,j,k-1} - T^n_{i,j,k}\right) + U_5\left(T^*_{i+1,j,k} - T^*_{i,j,k}\right) + V_4\left(T_{\infty_4} - T^n_{i,j,k}\right)$$

$$+ V_6\left(T_{\infty_6} - T^n_{i,j,k}\right) + W_4 + W_6 + X_4\left(T^{n^4}_{i,j,k} - T^4_{\infty_4}\right)$$

$$+ X_6\left(T^{n^4}_{i,j,k} - T^4_{\infty_6}\right) + Y_1 E_{G_{i,j,k}}$$

9. Corner 7

$$-2FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} = T^n_{i,j,k} + U_4\left(T^n_{i,j,k-1} - T^n_{i,j,k}\right)$$

$$+ U_5\left(T^*_{i+1,j,k} - T^*_{i,j,k}\right) + V_2\left(T_{\infty_2} - T^n_{i,j,k}\right) + V_4\left(T_{\infty_4} - T^n_{i,j,k}\right)$$

$$+ V_6\left(T_{\infty_6} - T^n_{i,j,k}\right) + W_2 + W_4 + W_6 + X_2\left(T^{n^4}_{i,j,k} - T^4_{\infty_2}\right)$$

$$+ X_4\left(T^{n^4}_{i,j,k} - T^4_{\infty_4}\right) + X_6\left(T^{n^4}_{i,j,k} - T^4_{\infty_6}\right) + Y_1 E_{G_{i,j,k}}$$

10. Perimeter 12

$$(1+2FoR_1)T^*_{i,j,k} - 2FoR_1 T^*_{i,j+1,k} = T^n_{i,j,k} + U_4\left(T^n_{i,j,k+1} - T^n_{i,j,k}\right)$$

$$+ Fo\left(T^*_{i-1,j,k} + T^*_{i+1,j,k} - 2T^*_{i,j,k}\right) + V_1\left(T_{\infty_1} - T^n_{i,j,k}\right) + V_3\left(T_{\infty_3} - T^n_{i,j,k}\right)$$

$$+ W_1 + W_3 + X_1\left(T^{n^4}_{i,j,k} - T^4_{\infty_1}\right) + X_3\left(T^{n^4}_{i,j,k} - T^4_{\infty_3}\right) + Y_1 E_{G_{i,j,k}}$$

11. Bottom Face

$$-FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} - FoR_1 T^{*}_{i,j+1,k} = T^{n}_{i,j,k}$$

$$+ U_4\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right) + Fo\left(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k}\right)$$

$$+ V_3\left(T_{\infty 3} - T^{n}_{i,j,k}\right) + W_3 + X_3\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 3}\right) + Y_1 E_{G\,i,j,k}$$

12. Perimeter 9

$$-2FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} = T^{n}_{i,j,k} + U_4\left(T^{n}_{i,j,k+1} - T^{n}_{i,j,k}\right)$$

$$+ Fo\left(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^{n}_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^{n}_{i,j,k}\right)$$

$$+ W_2 + W_3 + X_2\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 2}\right) + X_3\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 3}\right) + Y_1 E_{G\,i,j,k}$$

13. Left Face

$$(1+2FoR_1)T^{**}_{i,j,k} - 2FoR_1 T^{**}_{i,j+1,k} = T^{n}_{i,j,k} + U_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right)$$

$$+ Fo\left(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k}\right) + V_1\left(T_{\infty 1} - T^{n}_{i,j,k}\right)$$

$$+ W_1 + X_1\left(T^{n^4}_{i,j,k} - T^{4}_{\infty 1}\right) + Y_1 E_{G\,i,j,k}$$

14. Interior

$$-FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} - FoR_1 T^{*}_{i,j+1,k} = T^{n}_{i,j,k}$$

$$+ U_2\left(T^{n}_{i,j,k-1} + T^{n}_{i,j,k+1} - 2T^{n}_{i,j,k}\right) + Fo\left(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k}\right)$$

$$+ Y_1 E_{G\,i,j,k}$$

15. Right Face

$$-2FoR_1T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} = T^n_{i,j,k} + U_2\left(T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}\right)$$

$$+ Fo\left(T^*_{i-1,j,k} + T^*_{i+1,j,k} - 2T^*_{i,j,k}\right) + V_2\left(T_{\infty_2} - T^n_{i,j,k}\right)$$

$$+ W_2 + X_2\left(T^{n^4}_{i,j,k} - T^4_{\infty_2}\right) + Y_1 E_{G_{i,j,k}}$$

16. Perimeter 11

$$(1+2FoR_1)T^{**}_{i,j,k} - 2FoR_1T^{**}_{i,j+1,k} = T^n_{i,j,k} + U_4\left(T^n_{i,j,k-1} - T^n_{i,j,k}\right)$$

$$+ Fo\left(T^*_{i-1,j,k} + T^*_{i+1,j,k} - 2T^*_{i,j,k}\right) + V_1\left(T_{\infty_1} - T^n_{i,j,k}\right)$$

$$+ V_4\left(T_{\infty_4} - T^n_{i,j,k}\right) + W_1 + W_4 + X_1\left(T^{n^4}_{i,j,k} - T^4_{\infty_1}\right)$$

$$+ X_4\left(T^{n^4}_{i,j,k} - T^4_{\infty_4}\right) + Y_1 E_{G_{i,j,k}}$$

17. Top Face

$$-FoR_1T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} - FoR_1T^*_{i,j+1,k} = T^n_{i,j,k}$$

$$+ U_4\left(T^n_{i,j,k-1} - T^n_{i,j,k}\right) + Fo\left(T^*_{i-1,j,k} + T^*_{i+1,j,k} - 2T^*_{i,j,k}\right)$$

$$+ V_4\left(T_{\infty_4} - T^n_{i,j,k}\right) + W_4 + X_4\left(T^{n^4}_{i,j,k} - T^4_{\infty_4}\right) + Y_1 E_{G_{i,j,k}}$$

18. Perimeter 10

$$-2FoR_1T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} = T^n_{i,j,k} + U_4\left(T^n_{i,j,k-1} - T^n_{i,j,k}\right)$$

$$+ Fo\left(T^*_{i-1,j,k} + T^*_{i+1,j,k} - 2T^*_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^n_{i,j,k}\right)$$

$$+ V_4\left(T_{\infty 4} - T^n_{i,j,k}\right) + W_2 + W_4 + X_2\left(T^{n^4}_{i,j,k} - T^4_{\infty 2}\right)$$

$$+ X_4\left(T^{n^4}_{i,j,k} - T^4_{\infty 4}\right) + Y_1 E_{G\,i,j,k}$$

19. Corner 2

$$(1+2FoR_1)T^{**}_{i,j,k} - 2FoR_1T^{**}_{i,j+1,k} = T^n_{i,j,k} + U_4\left(T^n_{i,j,k+1} - T^n_{i,j,k}\right)$$

$$+ U_5\left(T^*_{i-1,j,k} - T^*_{i,j,k}\right) + V_1\left(T_{\infty 1} - T^n_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^n_{i,j,k}\right)$$

$$+ V_5\left(T_{\infty 5} - T^n_{i,j,k}\right) + W_1 + W_3 + W_6 + X_1\left(T^{n^4}_{i,j,k} - T^4_{\infty 1}\right)$$

$$+ X_3\left(T^{n^4}_{i,j,k} - T^4_{\infty 3}\right) + X_5\left(T^{n^4}_{i,j,k} - T^4_{\infty 5}\right) + Y_1 E_{G\,i,j,k}$$

20. Perimeter 4

$$-FoR_1T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} - FoR_1T^{**}_{i,j+1,k} = T^n_{i,j,k}$$

$$+ U_4\left(T^n_{i,j,k+1} - T^n_{i,j,k}\right) + U_5\left(T^*_{i-1,j,k} - T^*_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^n_{i,j,k}\right)$$

$$+ V_5\left(T_{\infty 5} - T^n_{i,j,k}\right) + W_3 + W_5 + X_3\left(T^{n^4}_{i,j,k} - T^4_{\infty 3}\right)$$

$$+ X_5\left(T^{n^4}_{i,j,k} - T^4_{\infty 5}\right) + Y_1 E_{G\,i,j,k}$$

21. Corner 6

$$-2FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} = T^n_{i,j,k} + U_4\left(T^n_{i,j,k+1} - T^n_{i,j,k}\right)$$

$$+ U_5\left(T^*_{i-1,j,k} - T^*_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^n_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^n_{i,j,k}\right)$$

$$+ V_5\left(T_{\infty 5} - T^n_{i,j,k}\right) + W_2 + W_3 + W_5 + X_2\left(T^{n^4}_{i,j,k} - T^4_{\infty 2}\right)$$

$$+ X_3\left(T^{n^4}_{i,j,k} - T^4_{\infty 3}\right) + X_5\left(T^{n^4}_{i,j,k} - T^4_{\infty 5}\right) + Y_1 E_{G\,i,j,k}$$

22. Perimeter 7

$$(1+2FoR_1)T^{**}_{i,j,k} - 2FoR_1 T^{**}_{i,j+1,k} = T^n_{i,j,k} + U_2\left(T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}\right)$$

$$+ U_5\left(T^*_{i-1,j,k} - T^*_{i,j,k}\right) + V_1\left(T_{\infty 1} - T^n_{i,j,k}\right) + V_5\left(T_{\infty 5} - T^n_{i,j,k}\right) + W_1 + W_5$$

$$+ X_1\left(T^{n^4}_{i,j,k} - T^4_{\infty 1}\right) + X_5\left(T^{n^4}_{i,j,k} - T^4_{\infty 5}\right) + Y_1 E_{G\,i,j,k}$$

23. Front Face

$$-FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} - FoR_1 T^{**}_{i,j+1,k} = T^n_{i,j,k}$$

$$+ U_2\left(T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}\right) + U_5\left(T^*_{i-1,j,k} - T^*_{i,j,k}\right)$$

$$+ V_5\left(T_{\infty 5} - T^n_{i,j,k}\right) + W_5 + X_5\left(T^{n^4}_{i,j,k} - T^4_{\infty 5}\right) + Y_1 E_{G\,i,j,k}$$

24. Perimeter 5

$$-2FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} = T^n_{i,j,k} + U_2\left(T^n_{i,j,k-1} + T^n_{i,j,k+1} - 2T^n_{i,j,k}\right)$$

$$+ U_5\left(T^*_{i-1,j,k} - T^*_{i,j,k}\right) + V_2\left(T_{\infty_2} - T^n_{i,j,k}\right) + V_5\left(T_{\infty_5} - T^n_{i,j,k}\right) + W_2 + W_5$$

$$+ X_2\left(T^{n^4}_{i,j,k} - T^4_{\infty_2}\right) + X_5\left(T^{n^4}_{i,j,k} - T^4_{\infty_5}\right) + Y_1 E_{G_{i,j,k}}$$

25. Corner 4

$$(1+2FoR_1)T^{**}_{i,j,k} - 2FoR_1 T^{**}_{i,j+1,k} = T^n_{i,j,k} + U_4\left(T^n_{i,j,k-1} - T^n_{i,j,k}\right)$$

$$+ U_5\left(T^*_{i-1,j,k} - T^*_{i,j,k}\right) + V_1\left(T_{\infty_1} - T^n_{i,j,k}\right) + V_4\left(T_{\infty_4} - T^n_{i,j,k}\right)$$

$$+ V_5\left(T_{\infty_5} - T^n_{i,j,k}\right) + W_1 + W_4 + W_5 + X_1\left(T^{n^4}_{i,j,k} - T^4_{\infty_1}\right)$$

$$+ X_4\left(T^{n^4}_{i,j,k} - T^4_{\infty_4}\right) + X_5\left(T^{n^4}_{i,j,k} - T^4_{\infty_5}\right) + Y_1 E_{G_{i,j,k}}$$

26. Perimeter 2

$$-FoR_1 T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} - FoR_1 T^{**}_{i,j+1,k} = T^n_{i,j,k}$$

$$+ U_4\left(T^n_{i,j,k-1} - T^n_{i,j,k}\right) + U_5\left(T^*_{i-1,j,k} - T^*_{i,j,k}\right) + V_4\left(T_{\infty_4} - T^n_{i,j,k}\right)$$

$$+ V_5\left(T_{\infty_5} - T^n_{i,j,k}\right) + W_4 + W_5 + X_4\left(T^{n^4}_{i,j,k} - T^4_{\infty_4}\right)$$

$$+ X_5\left(T^{n^4}_{i,j,k} - T^4_{\infty_5}\right) + Y_1 E_{G_{i,j,k}}$$

27. Corner 8

$$-2FoR_1T^{**}_{i,j-1,k} + (1+2FoR_1)T^{**}_{i,j,k} = T^n_{i,j,k} + U_4\left(T^n_{i,j,k-1} - T^n_{i,j,k}\right)$$

$$+ U_5\left(T^*_{i-1,j,k} - T^*_{i,j,k}\right) + V_2\left(T_{\infty_2} - T^n_{i,j,k}\right) + V_4\left(T_{\infty_4} - T^n_{i,j,k}\right)$$

$$+ V_5\left(T_{\infty_5} - T^n_{i,j,k}\right) + W_2 + W_4 + W_5 + X_2\left(T^{n^4}_{i,j,k} - T^4_{\infty_2}\right)$$

$$+ X_4\left(T^{n^4}_{i,j,k} - T^4_{\infty_4}\right) + X_5\left(T^{n^4}_{i,j,k} - T^4_{\infty_5}\right) + Y_1E_{G_{i,j,k}}$$

C. Z-DIRECTION EQUATIONS

1. Corner 1

$$(1+2FoR_2)T^{n+1}_{i,j,k} - 2FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k} + U_3\left(T^{**}_{i,j+1,k} - T^{**}_{i,j,k}\right)$$

$$+ U_5\left(T^*_{i+1,j,k} - T^*_{i,j,k}\right) + V_1\left(T_\infty - T^{**}_{i,j,k}\right) + V_3\left(T_{\infty_3} - T^{**}_{i,j,k}\right)$$

$$+ V_6\left(T_{\infty_6} - T^{**}_{i,j,k}\right) + W_1 + W_3 + W_6 + X_1\left(T^{**^4}_{i,j,k} - T^4_{\infty_1}\right)$$

$$+ X_3\left(T^{**^4}_{i,j,k} - T^4_{\infty_3}\right) + X_6\left(T^{**^4}_{i,j,k} - T^4_{\infty_6}\right) + Y_1E_{G_{i,j,k}}$$

2. Perimeter 8

$$-FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} - FoR_2T^{n+1}_{i,j,k} = T^{**}_{i,j,k}$$

$$+ U_3\left(T^{**}_{i,j+1,k} - T^{**}_{i,j,k}\right) + U_5\left(T^*_{i+1,j,k} - T^*_{i,j,k}\right) + V_1\left(T_{\infty_1} - T^{**}_{i,j,k}\right)$$

$$+ V_6\left(T_{\infty_6} - T^{**}_{i,j,k}\right) + W_1 + W_6 + X_1\left(T^{**^4}_{i,j,k} - T^4_{\infty_1}\right)$$

$$+ X_6\left(T^{**^4}_{i,j,k} - T^4_{\infty_6}\right) + Y_1E_{G_{i,j,k}}$$

3. Corner 3

$$-2FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} = T^{**}_{i,j,k} + U_3\left(T^{**}_{i,j+1,k} - T^{**}_{i,j,k}\right)$$

$$+ U_5\left(T^{*}_{i+1,j,k} - T^{*}_{i,j,k}\right) + V_1\left(T_{\infty_1} - T^{**}_{i,j,k}\right) + V_4\left(T_{\infty_4} - T^{**}_{i,j,k}\right)$$

$$+ V_6\left(T_{\infty_6} - T^{**}_{i,j,k}\right) + W_1 + W_4 + W_6 + X_1\left(T^{**4}_{i,j,k} - T^{4}_{\infty_1}\right)$$

$$+ X_4\left(T^{**4}_{i,j,k} - T^{4}_{\infty_4}\right) + X_6\left(T^{**4}_{i,j,k} - T^{4}_{\infty_6}\right) + Y_1E_{G_{i,j,k}}$$

4. Perimeter 12

$$(1+2FoR_2)T^{n+1}_{i,j,k} - 2FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k} + U_3\left(T^{**}_{i,j+1,k} - T^{**}_{i,j,k}\right)$$

$$+ Fo\left(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k}\right) + V_1\left(T_{\infty} - T^{**}_{i,j,k}\right) + V_3\left(T_{\infty_3} - T^{**}_{i,j,k}\right)$$

$$+ W_1 + W_3 + X_1\left(T^{**4}_{i,j,k} - T^{4}_{\infty_1}\right) + X_3\left(T^{**4}_{i,j,k} - T^{4}_{\infty_3}\right) + Y_1E_{G_{i,j,k}}$$

5. Left Face

$$-2FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} = T^{**}_{i,j,k} + U_3\left(T^{**}_{i,j+1,k} - T^{**}_{i,j,k}\right)$$

$$+ Fo\left(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k}\right) + V_1\left(T_{\infty_1} - T^{**}_{i,j,k}\right) + V_4\left(T_{\infty_4} - T^{**}_{i,j,k}\right)$$

$$+ W_1 + W_4 + X_1\left(T^{**4}_{i,j,k} - T^{4}_{\infty_1}\right) + X_4\left(T^{**4}_{i,j,k} - T^{4}_{\infty_4}\right) + Y_1E_{G_{i,j,k}}$$

6. Perimeter 11

$$-2FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} = T^{**}_{i,j,k} + U_3(T^{**}_{i,j+1,k} - T^{**}_{i,j,k})$$

$$+ Fo(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k}) + V_1(T_{\infty_1} - T^{**}_{i,j,k}) + V_4(T_{\infty_4} - T^{**}_{i,j,k})$$

$$+ W_1 + W_4 + X_1(T^{**4}_{i,j,k} - T^{4}_{\infty_1}) + X_4(T^{**4}_{i,j,k} - T^{4}_{\infty_4}) + Y_1 E_{G_{i,j,k}}$$

7. Corner 2

$$(1+2FoR_2)T^{n+1}_{i,j,k} - FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k} + U_3(T^{**}_{i,j+1,k} - T^{**}_{i,j,k})$$

$$+ U_5(T^{*}_{i-1,j,k} - T^{*}_{i,j,k}) + V_1(T_{\infty_1} - T^{**}_{i,j,k}) + V_3(T_{\infty_3} - T^{**}_{i,j,k})$$

$$+ V_5(T_{\infty_5} - T^{**}_{i,j,k}) + W_1 + W_3 + W_5 + X_1(T^{**4}_{i,j,k} - T^{4}_{\infty_1})$$

$$+ X_3(T^{**4}_{i,j,k} - T^{4}_{\infty_3}) + X_5(T^{**4}_{i,j,k} - T^{4}_{\infty_5}) + Y_1 E_{G_{i,j,k}}$$

8. Perimeter 7

$$-FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} - FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_3(T^{**}_{i,j+1,k} - T^{**}_{i,j,k}) + U_5(T^{*}_{i-1,j,k} - T^{*}_{i,j,k})$$

$$+ V_1(T_{\infty_1} - T^{**}_{i,j,k}) + V_5(T_{\infty_5} - T^{**}_{i,j,k}) + W_1 + W_5$$

$$+ X_1(T^{**4}_{i,j,k} - T^{4}_{\infty_1}) + X_5(T^{**4}_{i,j,k} - T^{4}_{\infty_5}) + Y_1 E_{G_{i,j,k}}$$

9. Corner 4

$$-2FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} = T^{**}_{i,j,k} + U_3(T^{**}_{i,j+1,k} - T^{**}_{i,j,k})$$

$$+ U_5(T^{*}_{i-1,j,k} - T^{*}_{i,j,k}) + V_1(T_{\infty_1} - T^{**}_{i,j,k}) + V_4(T_{\infty_4} - T^{**}_{i,j,k})$$

$$+ V_5(T_{\infty_5} - T^{**}_{i,j,k}) + W_1 + W_4 + W_5 + X_1(T^{**4}_{i,j,k} - T^{4}_{\infty_1})$$

$$+ X_4(T^{**4}_{i,j,k} - T^{4}_{\infty_4}) + X_5(T^{**4}_{i,j,k} - T^{4}_{\infty_5}) + Y_1E_{G_{i,j,k}}$$

10. Perimeter 3

$$(1+2FoR_2)T^{n+1}_{i,j,k} - 2FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_1(T^{**}_{i,j-1,k} + T^{**}_{i,j+1,k} - 2T^{**}_{i,j,k}) + U_5(T^{*}_{i+1,j,k} - T^{*}_{i,j,k})$$

$$+ V_3(T_{\infty_3} - T^{**}_{i,j,k}) + V_6(T_{\infty_6} - T^{**}_{i,j,k}) + W_3 + W_6$$

$$+ X_3(T^{**4}_{i,j,k} - T^{4}_{\infty_3}) + X_6(T^{**4}_{i,j,k} - T^{4}_{\infty_6}) + Y_1E_{G_{i,j,k}}$$

11. Back Face

$$-FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} - FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_1(T^{**}_{i,j-1,k} + T^{**}_{i,j+1,k} - 2T^{**}_{i,j,k}) + U_5(T^{*}_{i+1,j,k} - T^{*}_{i,j,k})$$

$$+ V_6(T_{\infty_6} - T^{**}_{i,j,k}) + W_6 + X_6(T^{**4}_{i,j,k} - T^{4}_{\infty_6}) + Y_1E_{G_{i,j,k}}$$

12. Perimeter 1

$$-2FoR_2T^{n+1}_{i,j,k-1} + (1 + 2FoR_2)T^{n+1}_{i,j,k} = T^{**}_{i,j,k}$$

$$+ U_1(T^{**}_{i,j-1,k} + T^{**}_{i,j+1,k} - 2T^{**}_{i,j,k}) + U_5(T^{*}_{i+1,j,k} - T^{*}_{i,j,k})$$

$$+ V_4(T_{\infty 4} - T^{**}_{i,j,k}) + V_6(T_{\infty 6} - T^{**}_{i,j,k}) + W_4 + W_6$$

$$+ X_4\left(T^{**4}_{i,j,k} - T^{4}_{\infty 4}\right) + X_6\left(T^{**4}_{i,j,k} - T^{4}_{\infty 6}\right) + Y_1 E_{G_{i,j,k}}$$

13. Bottom Face

$$(1 + 2FoR_2)T^{n+1}_{i,j,k} - 2FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_1(T^{**}_{i,j-1,k} + T^{**}_{i,j+1,k} - 2T^{**}_{i,j,k})$$

$$+ Fo(T^{**}_{i-1,j,k} + T^{**}_{i+1,j,k} - 2T^{**}_{i,j,k}) + V_3(T_{\infty 3} - T^{**}_{i,j,k})$$

$$+ W_3 + X_3\left(T^{**4}_{i,j,k} - T^{4}_{\infty 3}\right) + Y_1 E_{G_{i,j,k}}$$

14. Interior

$$-FoR_2T^{n+1}_{i,j,k-1} + (1 + 2FoR_2)T^{n+1}_{i,j,k} - FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_1(T^{**}_{i,j-1,k} + T^{**}_{i,j+1,k} - 2T^{**}_{i,j,k})$$

$$+ Fo(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k}) + V_4(T_{\infty 4} - T^{**}_{i,j,k})$$

$$+ W_4 + X_4\left(T^{**4}_{i,j,k} - T^{4}_{\infty 4}\right) + Y_1 E_{G_{i,j,k}}$$

15. Top Face

$$-2FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} = T^{**}_{i,j,k}$$

$$+ U_1(T^{**}_{i,j-1,k} + T^{**}_{i,j+1,k} - 2T^{**}_{i,j,k}) + Fo(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k})$$

$$+ V_4(T_{\infty 4} - T^{**}_{i,j,k}) + W_4 + X_4\left(T^{**4}_{i,j,k} - T^{4}_{\infty 4}\right) + Y_1 E_{G_{i,j,k}}$$

16. Perimeter 4

$$(1+2FoR_2)T^{n+1}_{i,j,k} - 2FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_1(T^{**}_{i,j-1,k} + T^{**}_{i,j+1,k} - 2T^{**}_{i,j,k}) + U_5(T^{*}_{i-1,j,k} - T^{*}_{i,j,k})$$

$$+ V_3(T_{\infty 3} - T^{**}_{i,j,k}) + V_5(T_{\infty 5} - T^{**}_{i,j,k}) + W_3 + W_5$$

$$+ X_3\left(T^{**4}_{i,j,k} - T^{4}_{\infty 5}\right) + Y_1 E_{G_{i,j,k}}$$

17. Front Face

$$-FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} - FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_1(T^{**}_{i,j+1,k} + T^{**}_{i,j-1,k} - 2T^{**}_{i,j,k})$$

$$+ U_5(T^{*}_{i-1,j,k} - T^{*}_{i,j,k}) + V_5(T_{\infty 5} - T^{**}_{i,j,k})$$

$$+ W_5 + X_5\left(T^{**4}_{i,j,k} - T^{4}_{\infty 5}\right) + Y_1 E_{G_{i,j,k}}$$

18. Perimeter 2

$$-2FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} = T^{**}_{i,j,k}$$

$$+ U_1\left(T^{**}_{i,j-1,k} + T^{**}_{i,j+1,k} - 2T^{**}_{i,j,k}\right) + U_5\left(T^{*}_{i-1,j,k} - T^{*}_{i,j,k}\right)$$

$$+ V_4\left(T_{\infty 4} - T^{**}_{i,j,k}\right) + V_5\left(T_{\infty 5} - T^{**}_{i,j,k}\right) + W_4 + W_5$$

$$+ X_4\left(T^{**4}_{i,j,k} - T^{4}_{\infty 4}\right) + X_5\left(T^{**4}_{i,j,k} - T^{4}_{\infty 5}\right) + Y_1 E_{G_{i,j,k}}$$

19. Corner 5

$$(1+2FoR_2)T^{n+1}_{i,j,k} - 2FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_3\left(T^{**}_{i,j-1,k} - T^{**}_{i,j,k}\right) + U_5\left(T^{*}_{i+1,j,k} - T^{*}_{i,j,k}\right)$$

$$+ V_2\left(T_{\infty 2} - T^{**}_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^{**}_{i,j,k}\right) + V_6\left(T_{\infty 6} - T^{**}_{i,j,k}\right)$$

$$+ W_2 + W_3 + W_6 + X_2\left(T^{**4}_{i,j,k} - T^{4}_{\infty 2}\right)$$

$$+ X_3\left(T^{**4}_{i,j,k} - T^{4}_{\infty 3}\right) + X_6\left(T^{**4}_{i,j,k} - T^{4}_{\infty 6}\right) + Y_1 E_{G_{i,j,k}}$$

20. Perimeter 6

$$-FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} - FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_3\left(T^{**}_{i,j-1,k} - T^{**}_{i,j,k}\right) + U_5\left(T^{*}_{i+1,j,k} - T^{**}_{i,j,k}\right)$$

$$+ V_2\left(T_{\infty 2} - T^{**}_{i,j,k}\right) + V_6\left(T_{\infty 6} - T^{**}_{i,j,k}\right) + W_2 + W_6$$

$$+ X_2\left(T^{**4}_{i,j,k} - T^{4}_{\infty 2}\right) + X_6\left(T^{**4}_{i,j,k} - T^{4}_{\infty 6}\right) + Y_1 E_{G_{i,j,k}}$$

21. Corner 7

$$-2FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} = T^{**}_{i,j,k} + U_3\left(T^{**}_{i,j-1,k} - T^{**}_{i,j,k}\right)$$

$$+ U_5\left(T^{*}_{i+1,j,k} - T^{*}_{i,j,k}\right) + V_2\left(T_{-2} - T^{**}_{i,j,k}\right) + V_4\left(T_{-4} - T^{**}_{i,j,k}\right)$$

$$+ V_6\left(T_{-6} - T^{**}_{i,j,k}\right) + W_2 + W_4 + W_6 + X_2\left(T^{**4}_{i,j,k} - T^{4}_{-2}\right)$$

$$+ X_4\left(T^{**4}_{i,j,k} - T^{4}_{-4}\right) + X_6\left(T^{**4}_{i,j,k} - T^{4}_{-6}\right) + Y_1E_{G_{i,j,k}}$$

22. Perimeter 9

$$(1+2FoR_2)T^{n+1}_{i,j,k} - 2FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_3\left(T^{**}_{i,j-1,k} - T^{**}_{i,j,k}\right) + Fo\left(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k}\right)$$

$$+ V_2\left(T_{-2} - T^{**}_{i,j,k}\right) + V_3\left(T_{-3} - T^{**}_{i,j,k}\right) + W_2 + W_3$$

$$+ X_2\left(T^{**4}_{i,j,k} - T^{4}_{-2}\right) + X_3\left(T^{**4}_{i,j,k} - T^{4}_{-3}\right) + Y_1E_{G_{i,j,k}}$$

23. Right Face

$$-FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} - FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_3\left(T^{**}_{i,j-1,k} - T^{**}_{i,j,k}\right) + Fo\left(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k}\right)$$

$$+ V_2\left(T_{-2} - T^{**}_{i,j,k}\right) + W_2 + X_2\left(T^{**4}_{i,j,k} - T^{4}_{-2}\right) + Y_1E_{G_{i,j,k}}$$

24. Perimeter 10

$$-2FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} = T^{**}_{i,j,k} + U_3\left(T^{**}_{i,j-1,k} - T^{**}_{i,j,k}\right)$$

$$+ Fo\left(T^{*}_{i-1,j,k} + T^{*}_{i+1,j,k} - 2T^{*}_{i,j,k}\right) + V_2\left(T_{\infty 2} - T^{**}_{i,j,k}\right)$$

$$+ V_4\left(T_{\infty 4} - T^{**}_{i,j,k}\right) + W_2 + W_4 + X_2\left(T^{**4}_{i,j,k} - T^{4}_{\infty 2}\right)$$

$$+ X_4\left(T^{**4}_{i,j,k} - T^{4}_{\infty 4}\right) + Y_1 E_{G_{i,j,k}}$$

25. Corner 6

$$(1+2FoR_2)T^{n+1}_{i,j,k} - 2FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_3\left(T^{**}_{i,j-1,k} - T^{**}_{i,j,k}\right) + U_5\left(T^{*}_{i-1,j,k} - T^{*}_{i,j,k}\right)$$

$$+ V_2\left(T_{\infty 2} - T^{**}_{i,j,k}\right) + V_3\left(T_{\infty 3} - T^{**}_{i,j,k}\right) + V_5\left(T_{\infty 5} - T^{**}_{i,j,k}\right)$$

$$+ W_2 + W_3 + W_5 + X_2\left(T^{**4}_{i,j,k} - T^{4}_{\infty 2}\right) + X_3\left(T^{**4}_{i,j,k} - T^{4}_{\infty 3}\right)$$

$$+ X_5\left(T^{**4}_{i,j,k} - T^{4}_{\infty 5}\right) + Y_1 E_{G_{i,j,k}}$$

26. Perimeter 5

$$-FoR_2T^{n+1}_{i,j,k-1} + (1+2FoR_2)T^{n+1}_{i,j,k} - FoR_2T^{n+1}_{i,j,k+1} = T^{**}_{i,j,k}$$

$$+ U_3\left(T^{**}_{i,j-1,k} - T^{**}_{i,j,k}\right) + U_5\left(T^{*}_{i-1,j,k} - T^{**}_{i,j,k}\right)$$

$$+ V_2\left(T_{\infty 2} - T^{**}_{i,j,k}\right) + V_5\left(T_{\infty 5} - T^{**}_{i,j,k}\right) + W_2 + W_5$$

$$+ X_2\left(T^{**4}_{i,j,k} - T^{4}_{\infty 2}\right) X_5\left(T^{**4}_{i,j,k} - T^{4}_{\infty 5}\right) + Y_1 E_{G_{i,j,k}}$$

27. Corner 8

$$-2FoR_2T^{n+1}_{i,j,k-1} + (1 + 2FoR_2)T^{n+1}_{i,j,k} = T^{**}_{i,j,k} + U_3\left(T^{**}_{i,j-1,k} - T^{**}_{i,j,k}\right)$$

$$+ U_5\left(T^{*}_{i-1,j,k} - T^{*}_{i,j,k}\right) + V_2\left(T_{-2} - T^{**}_{i,j,k}\right) + V_4\left(T_{-4} - T^{**}_{i,j,k}\right)$$

$$+ V_5\left(T_{-5} - T^{**}_{i,j,k}\right) + W_2 + W_4 + W_5 + X_2\left(T^{**4}_{i,j,k} - T^{4}_{-2}\right)$$

$$+ X_4\left(T^{**4}_{i,j,k} - T^{4}_{-4}\right) + X_5\left(T^{**4}_{i,j,k} - T^{4}_{-5}\right) + Y_1 E_{G_{i,j,k}}$$

APPENDIX C

PROGRAM EXPLICIT

```
*************************************************************************
*
*   PROGRAM EXPLICIT
*
*************************************************************************
*************************************************************************
*
*   EXPLICIT METHOD FOR TRANSIENT HEAT CONDUCTION IN THREE
*   DIMENSIONAL CARTESIAN COORDINATES
*
*************************************************************************
*************************************************************************
*
*   DESCRIPTION OF VARIABLES USED IN PROGRAM EXPLICIT
*
*   NOTE: SI UNITS ARE USED IN THIS PROGRAM UNLESS
*         OTHERWISE NOTED
*
*  1. USER SPECIFIED VARIABLES
*
*   A. GEOMETRIC DIMENSIONS:
*
*     DELX-DISTANCE INCREMENT IN THE X DIRECTION
*     DELY-DISTANCE INCREMENT IN THE Y DIRECTION
*     DELZ-DISTANCE INCREMENT IN THE Z DIRECTION
*
*     I----X COORDINATE OF NODE
*     J----Y COORDINATE OF NODE
*     K----Z COORDINATE OF NODE
*
```

```
*     LX---LENGTH OF MODEL IN THE X DIRECTION
*     LY---LENGTH OF MODEL IN THE Y DIRECTION
*     LZ---LENGTH OF MODEL IN THE Z DIRECTION
*
*     M----NUMBER OF NODES IN THE X DIRECTION
*     N----NUMBER OF NODES IN THE Y DIRECTION
*     P----NUMBER OF NODES IN THE Z DIRECTION
*
*
*   B. MATERIAL PROPERTIES:
*
*     CP---SPECIFIC HEAT
*     K1---THERMAL CONDUCTIVITY
*     RHO--DENSITY
*
*
*   C. INITIAL DATA:
*
*     FLUX--SURFACE FLUX
*     FREQ--NUMBER OF TIME STEPS BETWEEN SUCCESSIVE
*           TEMPERATURE PRINTOUTS
*     H-----HEAT TRANSFER COEFFICIENT
*     TAMB--AMBIENT TEMPERATURE
*     TINIT-INITIAL TEMPERATURE OF ALL NODES
*     VAL---LOGIC NUMBER TO DETERMINE IF VALIDATION
*           TEST IS TO BE RUN
*
*
* 2. NON-USER VARIABLES
*
*   A. GENERAL:
*
*     BIO----BIOT NUMBER
*     COUNT--COUNTER FOR DETERMINING THE NUMBER OF ELAPSED
```

```
*          TIME STEPS
*     DELT---TIME INCREMENT
*     FO-----FOURIER NUMBER
*     TIME---DURATION OF CALCULATIONS
*     TNEW---MATRIX OF TEMPERATURES AFTER TIME STEP DELT
*
*
*   B. EXTERNAL PROGRAMS:
*
*     CTIME-----USED IN EXTERNAL IMSL ROUTINE FOR DETERMINING
*           AMOUNT OF CPU TIME USED
*     NOUT------USED IN EXTERNAL IMSL ROUTINE FOR DETERMINING
*           AMOUNT OF CPU TIME USED
*     TIMEO-----CPU TIME AT BEGINNING OF CALCULATIONS
*     TIME1-----CPU TIME AT END OF CALCULATIONS
*     UMACH-----USED IN EXTERNAL IMSL ROUTINE FOR DETERMINING
*           AMOUNT OF CPU TIME USED
*
*************************************************************************
*************************************************************************
*
      DIMENSION T(25,25,25), TNEW(25,25,25)
*
      REAL TAMB, H, RHO, ALFA, FO, FLUX, R, R1, R2, K1
      REAL DELX, DELY, DELZ, DELT, TIME, TINIT, TNEW, BIO
      REAL TDIFE1, TDIFE2, TDIFE3
      REAL LX, LY, LZ
      REAL CTIME, TIME0, TIME1
*
      INTEGER I, J, K, M, N, P, COUNT, FREQ, VAL, NOUT
*
      EXTERNAL CTIME, UMACH
*
      COMMON TNEW, TIME, DELT, M, N, P, TDIFE1, TDIFE2, TDIFE3, TINIT
```

```
      COMMON COUNT, FREQ
*
*************************************************************************
*************************************************************************
*
*     INITIALIZING CPU TIME
*
      TIMEO = CTIME()
*
*************************************************************************
*************************************************************************
*
*     VARIOUS PARAMETERS TO BE USED THROUGHOUT THE PROGRAM
*     WILL NOW BE READ INTO THE PROGRAM.
*
      WRITE(*,*) 'ENTER TAMB, THE AMBIENT TEMPERATURE.'
      READ(*,*) TAMB
      WRITE(*,*) 'ENTER TINIT, THE INITIAL TEMPERATURE.'
      READ(*,*) TINIT
      WRITE(*,*) 'ENTER H, THE HEAT TRANSFER COEFFICIENT.'
      READ(*,*) H
      WRITE(*,*) 'ENTER K1, THE THERMAL CONDUCTIVITY.'
      READ(*,*) K1
      WRITE(*,*) 'ENTER CP, THE SPECIFIC HEAT OF THE MATERIAL.'
      READ(*,*) CP
      WRITE(*,*) 'ENTER RHO, THE MATERIAL DENSITY.'
      READ(*,*) RHO
      WRITE(*,*) 'ENTER FLUX.'
      READ(*,*) FLUX
      WRITE(*,*) 'ENTER LX, THE LENGTH IN THE X DIRECTION.'
      READ(*,*) LX
      WRITE(*,*) 'ENTER LY, THE LENGTH IN THE Y DIRECTION.'
      READ(*,*) LY
      WRITE(*,*) 'ENTER LZ, THE LENGTH IN THE Z DIRECTION.'
```

```
      READ(*,*) LZ
      WRITE(*,*) 'ENTER M, THE NUMBER OF NODES IN THE X DIRECTION.'
      READ(*,*) M
      WRITE(*,*) 'ENTER N, THE NUMBER OF NODES IN THE Y DIRECTION.'
      READ(*,*) N
      WRITE(*,*) 'ENTER P, THE NUMBER OF NODES IN THE Z DIRECTION.'
      READ(*,*) P
      WRITE(*,*) 'ENTER FREQ, THE NUMBER OF TIME STEPS BETWEEN'
      WRITE(*,*) 'SUCCESSIVE PRINTINGS.'
      READ(*,*) FREQ
      WRITE(*,*) 'ENTER VAL: IF YOU WANT THE VALIDATION SUBROUTINE'
      WRITE(*,*) 'CALLED, USE 1; IF THE VALIDATION SUBROUTINE IS NOT'
      WRITE(*,*) 'DESIRED, USE 0.'
      READ(*,*) VAL
*
*************************************************************************
*************************************************************************
*
*     CALCULATIONS OF CONSTANTS TO BE USED THROUGHOUT THE
*     PROGRAM
*
      COUNT = 0
      TIME = 0.0
*
      DELX = LX (M-1)
      DELY = LY (N-1)
      DELZ = LZ (P-1)
*
      R = DELX (DELY*DELZ)
      R1 = (DELX**2) (DELY**2)
      R2 = (DELX**2) (DELZ**2)
*
      ALFA = K1 (RHO*CP)
      BIO = H K1*DELX
```

```
      FO = 0.5 (1 + R1 + R2 +  BIO*SQRT(R2))
      DELT = FO*(DELX**2) ALFA
*
*     NOTE: THE FOLLOWING EXPRESSION FOR DELT WAS USED FOR
*           VALIDATING THE EXPLICIT TECHNIQUE WITH THE CLOSED-FORM
*           SOLUTION.
*
*     DELT = 0.125*(DELY**2) ALFA
*
*************************************************************************
*************************************************************************
*
*     INITIALIZE ALL TEMPERATURES TO TINIT, THE INITIAL
*     TEMPERATURE.
*
      DO 10 I = 1, M
        DO 15 J = 1, N
          DO 20 K = 1, P
*
      TNEW(I,J,K) = TINIT
*
 20       CONTINUE
 15     CONTINUE
 10   CONTINUE
*
*
*************************************************************************
*************************************************************************
*
*     THE OLD TEMPERATURES WILL NOW BE SAVED FOR FUTURE USE
*     IN THE TEMPERATURE EQUATIONS.
*
      DO 30 I = 1, M
        DO 35 J = 1, N
```

```
      DO 40 K = 1, P
*
   T(I,J,K) = TNEW(I,J,K)
*
 40     CONTINUE
 35   CONTINUE
 30 CONTINUE
*
*************************************************************************
*************************************************************************
*
*   INCREMENTING TIME AND COUNTER
*
 1  COUNT = COUNT + 1
    TIME = TIME + DELT
*
*************************************************************************
*************************************************************************
*
*   NODE EQUATIONS FOR THE THREE DIMENSIONAL BLOCK ARE
*   NOW PRESENTED.
*
*************************************************************************
*************************************************************************
*
*   INTERIOR NODES
*
    DO 50 I = 2, M-1
      DO 55 J = 2, N-1
        DO 60 K = 2, P-1
*
    TNEW(I,J,K) = T(I,J,K) + R1*FO*( T(I,J-1,K) + T(I,J+1,K) - 2*
   &  T(I,J,K) ) + R2*FO*( T(I,J,K+1) + T(I,J,K-1) - 2*T(I,J,K) )
```

```
     &   + FO*( T(I+1,J,K) + T(I-1,J,K) - 2*T(I,J,K) )
*
 60       CONTINUE
 55     CONTINUE
 50  CONTINUE
*
*
*************************************************************************
*************************************************************************
*
*    CORNER NODES
*
* CORNER NR. 1
*
      I = 1
      J = 1
      K = 1
*
      TNEW(I,J,K)= T(I,J,K) + 2*R2*FO*( T(I,J,K+1) - T(I,J,K) ) +
     & 2*R1*FO*( T(I,J+1,K) - T(I,J,K) ) + 2*F0*( T(I+1,J,K) -
     & T(I,J,K) ) + 2*FO*R1*DELY*FLUX/K1
*
*
* CORNER NR. 2
*
      I = M
      J = 1
      K = 1
*
      TNEW(I,J,K)= T(I,J,K) + 2*R2*FO*( T(I,J,K+1) - T(I,J,K) ) +
     &  2*R1*FO*( T(I,J+1,K) - T(I,J,K) ) + 2*FO*( T(I-1,J,K) -
     &  T(I,J,K) ) + 2*FO*R1*DELY*FLUX/K1
*
*
```

```
* CORNER NR. 3
*
      I = 1
      J = 1
      K = P
*
      TNEW(I,J,K) = T(I,J,K) + 2*FO*( T(I+1,J,K) - T(I,J,K) ) +
     & 2*R1*FO*( T(I,J+1,K) - T(I,J,K) ) + 2*R2*FO*( T(I,J,K-1) -
     & T(I,J,K) ) + 2*FO*R1*DELY*FLUX K1
*
*
* CORNER NR. 4
*
      I = M
      J = 1
      K = P
*
      TNEW(I,J,K) = T(I,J,K) + 2*FO*( T(I-1,J,K) - T(I,J,K) ) +
     & 2*R1*FO*( T(I,J+1,K) - T(I,J,K) ) + 2*R2*FO*( T(I,J,K-1) -
     & T(I,J,K) ) + 2*FO*R1*DELY*FLUX K1
*
*
* CORNER NR. 5
*
      I = 1
      J = N
      K = 1
*
      TNEW(1,N,1) = T(1,N,1) + 2*FO*( T(2  ,N,1) - T(1,N,1) ) +
     & 2*R1*FO*( T(1,N-1,1) - T(1,N,1) ) + 2*R2*FO*( T(1,N,  2) -
     & T(1,N,1) ) + 2*FO*BIO*(R1**0.5)*( TAMB - T(1,N,1) )
*
*
```

```
* CORNER NR. 6
*
      I = M
      J = N
      K = 1
*
      TNEW(I,J,K) = T(I,J,K) + 2*FO*( T(I-1,J,K) - T(I,J,K) ) +
     &  2*R1*FO*( T(I,J-1,K) - T(I,J,K) ) + 2*R2*FO*( T(I,J,K+1) -
     &  T(I,J,K) ) + 2*FO*BIO*(R1**0.5)*( TAMB - T(I,J,K) )
*
*
* CORNER NR. 7
*
      I = 1
      J = N
      K = P
*
      TNEW(I,J,K) = T(I,J,K) + 2*FO*( T(I+1,J,K) - T(I,J,K) ) +
     &  2*R1*FO*( T(I,J-1,K) - T(I,J,K) ) + 2*R2*FO*( T(I,J,K-1) -
     &  T(I,J,K) ) + 2*FO*BIO*(R1**0.5)*( TAMB - T(I,J,K) )
*
*
* CORNER NR. 8
*
      I = M
      J = N
      K = P
*
      TNEW(I,J,K) = T(I,J,K) + 2*FO*( T(I-1,J,K) - T(I,J,K) ) +
     &  2*R1*FO*( T(I,J-1,K) - T(I,J,K) ) + 2*R2*FO*( T(I,J,K-1) -
     &  T(I,J,K) ) + 2*FO*BIO*(R1**0.5)*( TAMB - T(I,J,K) )
*
*
************************************************************************
```

```
*****************************************************************************
*
*    ADIABATIC SURFACES
*
* FRONT YZ SURFACE
*
      DO 70 J = 2, N-1
        DO 75 K = 2, P-1
          I = M
*
      TNEW(I,J,K) = T(I,J,K) + 2*FO*( T(I-1,J,K) - T(I,J,K) ) +
     & R1*FO*( T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K) ) +
     & R2*FO*( T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K) )
*
 75   CONTINUE
 70 CONTINUE
*
*
* BACK YZ ADIABATIC SURFACE
*
      DO 80 J = 2, N-1
        DO 85 K = 2, P-1
          I = 1
*
      TNEW(I,J,K) = T(I,J,K) + 2*FO*( T(I+1,J,K) - T(I,J,K) ) +
     & R1*FO*( T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K) ) +
     & R2*FO*( T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K) )
*
 85   CONTINUE
 80 CONTINUE
*
*
*****************************************************************************
*****************************************************************************
```

```
*
* BOTTOM XY ADIABATIC SURFACE
*
      DO 90 I = 2, M-1
        DO 95 J = 2, N-1
          K = 1
*
      TNEW(I,J,K) = T(I,J,K) + FO*( T(I+1,J,K) + T(I-1,J,K) -
     & 2*T(I,J,K) ) + R1*FO*( T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K) ) +
     &  2*R2*FO*( T(I,J,K+1) - T(I,J,K) )
*
 95     CONTINUE
 90   CONTINUE
*
*
* TOP XY ADIABATIC SURFACE
*
      DO 100 I = 2, M-1
        DO 105 J = 2, N-1
          K = P
*
      TNEW(I,J,K) = T(I,J,K) + FO*( T(I+1,J,K) + T(I-1,J,K) -
     & 2*T(I,J,K) ) + R1*FO*( T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K) ) +
     & 2*R2*FO*( T(I,J,K-1) - T(I,J,K) )
*
 105    CONTINUE
 100    CONTINUE
*
*
*************************************************************************
*************************************************************************
*
*    RIGHT XZ CONVECTIVE SURFACE
*
```

```
      DO 110 I = 2, M-1
        DO 115 K = 2, P-1
          J = N
*
      TNEW(I,J,K) = T(I,J,K) + FO*( T(I+1,J,K) + T(I-1,J,K) -
     & 2*T(I,J,K) ) + 2*R1*FO*( T(I,J-1,K) - T(I,J,K) ) + R2*FO*
     &  ( T(I,J,K+1) + T(I,J,K-1) - 2*T(I,J,K) ) +
     &  2*FO*BIO*(R1**0.5)*( TAMB - T(I,J,K) )
*
 115    CONTINUE
 110  CONTINUE
*
*
*************************************************************************
*************************************************************************
*
*    LEFT XZ FLUX SURFACE
*
      DO 120 I = 2, M-1
        DO 125 K = 2, P-1
          J = 1
*
      TNEW(I,J,K) = T(I,J,K) + FO*( T(I-1,J,K) + T(I+1,J,K) -
     & 2*T(I,J,K) ) + 2*R1*FO*( T(I,J+1,K) - T(I,J,K) )  +
     &  R2*FO*( T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K) ) +
     &  2*FO*R1*DELY*FLUX K1
*
 125    CONTINUE
 120  CONTINUE
*
*
*************************************************************************
*************************************************************************
*
```

```
*    OUTER PERIMETER SURFACES
*
* PERIMETER NR. 1
*
      DO 130 J = 2, N-1
        I = 1
          K = P
*
      TNEW(I,J,K) = T(I,J,K) + R1*FO*( T(I,J-1,K) + T(I,J+1,K) -
     & 2*T(I,J,K) ) + 2*FO*( T(I+1,J,K) - T(I,J,K) ) +
     & 2*R2*FO*( T(I,J,K-1) - T(I,J,K) )
*
  130 CONTINUE
*
*
* PERIMETER NR. 2
*
      DO 135 J = 2, N-1
        I = M
          K = P
*
      TNEW(I,J,K) = T(I,J,K) + R1*FO*( T(I,J-1,K) + T(I,J+1,K) -
     & 2*T(I,J,K) ) + 2*FO*( T(I-1,J,K) - T(I,J,K) ) +
     & 2*R2*FO*( T(I,J,K-1) - T(I,J,K) )
*
  135 CONTINUE
*
*
* PERIMETER NR. 3
*
      DO 140 J = 2, N-1
        I = 1
          K = 1
*
```

```
      TNEW(I,J,K) = T(I,J,K) + R1*FO*( T(I,J-1,K) + T(I,J+1,K) -
     &  2*T(I,J,K) ) + 2*FO*( T(I+1,J,K) - T(I,J,K) ) +
     &  2*R2*FO*( T(I,J,K+1) - T(I,J,K) )
*
  140 CONTINUE
*
*
* PERIMETER NR. 4
*
      DO 145 J=2, N-1
        I=M
          K=1
*
      TNEW(I,J,K) = T(I,J,K) + R1*FO*( T(I,J-1,K) + T(I,J+1,K) -
     &  2*T(I,J,K) ) + 2*FO*( T(I-1,J,K) - T(I,J,K) ) +
     &  2*R2*FO*( T(I,J,K+1) - T(I,J,K) )
*
  145 CONTINUE
*
*
* PERIMETER NR. 5
*
      DO 150 K=2, P-1
        I=M
          J=N
*
      TNEW(I,J,K) = T(I,J,K) + R2*FO*( T(I,J,K+1) + T(I,J,K-1) -
     & 2*T(I,J,K) ) + 2*R1*FO*( T(I,J-1,K) - T(I,J,K) ) +
     & 2*FO*( T(I-1,J,K) - T(I,J,K) ) + 2*FO*BIO*(R1**0.5)*( TAMB -
     & T(I,J,K) )
*
  150 CONTINUE
*
*
```

```
* PERIMETER NR. 6
*
      DO 155 K = 2, P-1
        I = 1
          J = N
*
      TNEW(I,J,K) = T(I,J,K) + R2*FO*( T(I,J,K + 1) + T(I,J,K-1) -
     &  2*T(I,J,K) ) + 2*R1*FO*( T(I,J-1,K) - T(I,J,K) ) +
     &  2*FO*( T(I + 1,J,K) - T(I,J,K) ) + 2*FO*BIO*(R1**0.5)*( TAMB
     &  - T(I,J,K) )
*
  155 CONTINUE
*
*
* PERIMETER NR. 7
*
      DO 160 K = 2, P-1
        I = M
          J = 1
*
      TNEW(I,J,K) = T(I,J,K) + FO*R2*( T(I,J,K + 1) + T(I,J,K-1) -
     &  2*T(I,J,K) ) + 2*FO*( T(I-1,J,K) - T(I,J,K) ) +
     &  2*FO*R1*( T(I,J + 1,K) - T(I,J,K) ) + 2*FO*R1*DELY*FLUX(K)
*
  160 CONTINUE
*
*
* PERIMETER NR. 8
*
      DO 165 K = 2, P-1
        I = 1
          J = 1
*
      TNEW(I,J,K) = T(I,J,K) + FO*R2*( T(I,J,K + 1) + T(I,J,K-1) -
```

```
     & 2*T(I,J,K) ) + 2*FO*( T(I+1,J,K) - T(I,J,K) ) +
     & 2*FO*R1*( T(I,J+1,K) - T(I,J,K) ) + 2*FO*R1*DELY*FLUX/K1
*
  165 CONTINUE
*
*
* PERIMETER NR. 9
*
      DO 170 I=2, M-1
        J=N
          K=1
*
      TNEW(I,J,K) = T(I,J,K) + 2*FO*R1*( T(I,J-1,K) - T(I,J,K) ) +
     & 2*FO*R2*( T(I,J,K-1) - T(I,J,K) ) + FO*( T(I+1,J,K) +
     & T(I-1,J,K) - 2*T(I,J,K) ) + 2*FO*BIO*(R1**0.5)*( TAMB -
     & T(I,J,K) )
*
  170 CONTINUE
*
*
* PERIMETER NR. 10
*
      DO 175 I=2, M-1
        J=N
          K=P
*
      TNEW(I,J,K) = T(I,J,K) + 2*FO*R1*( T(I,J-1,K) - T(I,J,K) ) +
     & 2*FO*R2*( T(I,J,K-1) - T(I,J,K) ) + FO*( T(I+1,J,K) +
     & T(I-1,J,K) - 2*T(I,J,K) ) + 2*FO*BIO*(R1**0.5)*( TAMB -
     & T(I,J,K) )
*
  175 CONTINUE
*
*
```

```
* PERIMETER NR. 11
*
      DO 180 I = 2, M-1
        J = 1
          K = P
*
      TNEW(I,J,K) = T(I,J,K) + FO*( T(I+1,J,K) + T(I-1,J,K) -
     & 2*T(I,J,K) ) + 2*FO*R1*( T(I,J+1,K) - T(I,J,K) ) +
     & 2*FO*R2*( T(I,J,K-1) - T(I,J,K) ) + 2*FO*R1*DELY*FLUX(K)
*
  180 CONTINUE
*
*
* PERIMETER NR. 12
*
      DO 185 I = 2, M-1
        J = 1
          K = 1
*
      TNEW(I,J,K) = T(I,J,K) + FO*( T(I+1,J,K) + T(I-1,J,K) -
     & 2*T(I,J,K) ) + 2*FO*R1*( T(I,J+1,K) - T(I,J,K) ) +
     & 2*FO*R2*( T(I,J,K+1) - T(I,J,K) ) + 2*FO*R1*DELY*FLUX(K)
*
  185 CONTINUE
*
*
*************************************************************************
*************************************************************************
*
*     UPDATING THE OLD NODE TEMPERATURES TO THE NEW
*     NODE TEMPERATURES.
*
      DO 200 I = 1, M
        DO 205 J = 1, N
```

```
      DO 210 K = 1, P
*
   T(I,J,K) = TNEW(I,J,K)
*
 210     CONTINUE
 205   CONTINUE
 200 CONTINUE
*
*************************************************************************
*************************************************************************
*
*   THIS PART OF PROGRAM DETERMINES IF SUBROUTINE VALID OR
*   SUBROUTINE VALID IS TO BE CALLED.  IF THE USER DESIRES TO
*   VALIDATE THE PROGRAM IN CONJUNCTION WITH PROGRAM BRIAN
*   AND PROGRAM VALID, THEN THE SUBROUTINE OUTPUT WILL BE
*   BYPASSED.
*
   IF(VAL .NE. 1) GO TO 250
   IF(TIME .EQ. DELT) CALL VALID
   IF(COUNT .NE. FREQ) GO TO 300
   CALL VALID
   GO TO 300
*
 250 IF(TIME .EQ. DELT) CALL OUTPUT
   IF(COUNT .NE. FREQ) GO TO 300
   CALL OUTPUT
*
 300 IF (TIME .LE. 3600) GO TO 1
*
*   NOTE: THE TIME PERIOD WAS ARBITRARILY TAKEN
*         AS 3600 SECONDS (1 HOUR).
*
   PRINT*
```

```
      PRINT*
*
*     CALCULATING AND PRINTING CPU TIME
*
      TIME1 = CTIME()
      CALL UMACH(2,NOUT)
      WRITE(NOUT,*) 'CPU TIME (SECONDS) = ', TIME1-TIME0
*
      STOP
      END
*
*************************************************************************
*************************************************************************
*
*     THIS SUBROUTINE WILL BE USED FOR PRINTING THE NODE
*     TEMPERATURES.
*
      SUBROUTINE OUTPUT
*
      DIMENSION T(25,25,25), TNEW(25,25,25)
*
      REAL TAMB, H, RHO, ALFA, FO, FLUX, R, R1, R2, K1
      REAL DELX, DELY, DELZ, DELT, TIME, TINIT, TNEW, BIO
      REAL TDIFE1, TDIFE2, TDIFE3
      REAL LX, LY, LZ
      REAL CTIME, TIME0, TIME1
*
      INTEGER I, J, K, M, N, P, COUNT, FREQ, VAL, NOUT
*
      COMMON TNEW, TIME, DELT, M, N, P, TDIFE1, TDIFE2, TDIFE3, TINIT
      COMMON COUNT, FREQ
*
      WRITE(*,*) 'TIME = ',TIME
```

```
      PRINT*
*
 350  DO 370 K = 1, P
*
      WRITE(*,*) 'TEMPERATURE DISTRIBUTION ON PLANE'
      WRITE(6,360) 'K = ',K
 360  FORMAT(A3,I2)
      PRINT*
      PRINT 380, ((TNEW(I,J,K), J = 1,N), I = 1,M)
 380  FORMAT(1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,
     & 1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1/)
*
      PRINT*
      PRINT*
*
 370  CONTINUE
*
*     REINITIALIZING THE COUNTER TO ZERO
*
      COUNT = 0
*
      RETURN
      END
*
*************************************************************************
*************************************************************************
*
      SUBROUTINE VALID
*
      DIMENSION T(25,25,25), TNEW(25,25,25)
*
      REAL TAMB, H, RHO, ALFA, FO, FLUX, R, R1, R2, K1
      REAL DELX, DELY, DELZ, DELT, TIME, TINIT, TNEW, BIO
      REAL TDIFE1, TDIFE2, TDIFE3
```

```
      REAL LX, LY, LZ
      REAL CTIME, TIME0, TIME1
*
      INTEGER I, J, K, M, N, P, COUNT, FREQ, VAL, NOUT
*
      COMMON TNEW, TIME, DELT, M, N, P, TDIFE1, TDIFE2, TDIFE3, TINIT
      COMMON COUNT, FREQ
*
      TDIFE1 = TNEW( 5,11, 5)- TINIT
      TDIFE2 = TNEW( 5, 5, 5)- TINIT
      TDIFE3 = TNEW( 5, 1, 5)- TINIT
*
      IF(TIME .NE. DELT) GO TO 450
      PRINT*
      PRINT*
      WRITE(*,*) 'TIME(SEC)   TDIFE1    TDIFE2    TDIFE3'
      WRITE(*,*) '---------   ------    ------    ------'
*
 450  PRINT 460, TIME, TDIFE1, TDIFE2, TDIFE3
 460  FORMAT(1X,F8.3,3X,F7.3,3X,F7.3,3X,F7.3)
*
*     REINITIALIZING THE COUNTER TO ZERO
*
      COUNT = 0
*
      RETURN
      END
*
************************************************************************
************************************************************************
```

APPENDIX D

PROGRAM BRIAN

```
*****************************************************************
*
*   PROGRAM BRIAN
*
*****************************************************************
*****************************************************************
*
*   BRIAN'S METHOD FOR TRANSIENT HEAT CONDUCTION IN THREE
*   DIMENSIONAL CARTESIAN COORDINATES
*
*****************************************************************
*****************************************************************
*
*   DESCRIPTION OF VARIABLES USED IN PROGRAM BRIAN
*
*   NOTE: SI UNITS ARE USED IN THIS PROGRAM UNLESS
*         OTHERWISE SPECIFIED
*
*  I. USER SPECIFIED VARIABLES
*
*   A. GEOMETRIC DIMENSIONS:
*
*     DELX-DISTANCE INCREMENT IN THE X DIRECTION
*     DELY-DISTANCE INCREMENT IN THE Y DIRECTION
*     DELZ-DISTANCE INCREMENT IN THE Z DIRECTION
*
*     I----X COORDINATE OF NODE
*     J----Y COORDINATE OF NODE
*     K----Z COORDINATE OF NODE
*
```

```
*     LX---LENGTH OF MODEL IN X DIRECTION
*     LY---LENGTH OF MODEL IN Y DIRECTION
*     LZ---LENGTH OF MODEL IN Z DIRECTION
*
*     M----NUMBER OF NODES IN THE X DIRECTION
*     N----NUMBER OF NODES IN THE Y DIRECTION
*     P----NUMBER OF NODES IN THE Z DIRECTION
*
*
*   B. MATERIAL PROPERTIES:
*
*     CP---SPECIFIC HEAT
*     K1---THERMAL CONDUCTIVITY
*     RHO--DENSITY
*
*
*   C. INITIAL DATA:
*     SUBSCIPTS HAVE THE FOLLOWING MEANING:
*           1-LEFT FACE
*           2-RIGHT FACE
*           3-BOTTOM FACE
*           4-TOP FACE
*           5-FRONT FACE
*           6-BACK FACE
*
*     BIO1,2,3,4,5,6-----BIOT NUMBER
*     DELT-------------TIME INCREMENT
*     EPS1,2,3,4,5,6-----EMISSIVITY
*     FLUX1,2,3,4,5,6----SURFACE FLUX
*     FREQ---------------NUMBER OF TIME STEPS BETWEEN SUCCESSIVE
*                   TEMPERATURE PRINTOUTS
*     GEN----------------ENERGY GENERATION TERM
*     H1,2,3,4,5,6-------HEAT TRANSFER COEFFICIENT
*     SIG1,2,3,4,5,6-----STEFAN--BOLTZMANN CONSTANT (USE 0 IF NO
```

```
*                 RADIATION HEAT TRANSFER; USE 5.67E-8 IF
*                 RADIATION HEAT TRANSFER PRESENT)
*     TAMB1,2,3,4,5,6----AMBIENT TEMPERATURE
*     TINIT-------------INITIAL TEMPERATURE OF ALL NODES
*     VAL---------------LOGIC NUMBER TO DETERMINE IF VALIDATION
*                 TEST IS TO BE RUN
*
*
*
*
*   D. ADDITIONAL VARIABLES FOR SATELLITE APPLICATIONS:
*
*     ABS1,2,3,4,5,6-----ABSORPTIVITY
*     ALBCO-------------ALBEDO COEFFICIENT
*     ALT---------------ALTITUDE OF SATELLITE IN KILOMETERS
*     EARTH-------------EARTH FLUX (NOTE: THE ABSORPTIVITY FOR
*                 EARTH FLUX HAS THE SAME VALUE AS THE
*                 EMISSIVITY FOR RADIATION HEAT TRANSFER
*                 SINCE BOTH ARE IN THE INFRA RED REGION)
*     ECLIPS------------PERIOD OF TIME IN MINUTES THE SATELLITE
*                 IS IN THE SHADOW OF THE EARTH
*     FA----------------GEOMETRIC FACTOR FOR ALBEDO FLUX
*     FE----------------GEOMETRIC FACTOR FOR EARTH FLUX
*     SAC1,2,3,4,5,6-----SOLAR ASPECT COEFFICIENTS
*     SOLAR-------------SOLAR FLUX
*
*
*  2. NON-USER VARIABLES
*
*   A. GENERAL:
*
*     A,B,C,D----------COEFFICIENT VECTORS FOR SOLVING THE
*                 TEMPERATURE EQUATIONS IN SUBROUTINE TRIDAG
*     BETA------------VECTOR OF INTERMEDIATE COEFFICIENTS SOLVED
```

```
*               SUBROUTINE TRIDAG
*     BIOT1,2,3,4,5,6--BIOT NUMBER
*     COUNT-----------COUNTER FOR DETERMINING THE NUMBER OF
*               ELAPSED TIME STEPS
*     FO--------------FOURIER NUMBER
*     GAMMA-----------VECTOR OF INTERMEDIATE COEFFICIENTS
*               SOLVE IN SUBROUTINE TRIDAG
*     IF,L------------NUMBERS CORRESPONDING TO THE FIRST AND LAST
*               EQUATIONS TO BE SOLVED IN SUBROUTINE TRIDAG
*     QDOT------------MATRIX OF NODES DEALING WITH ENERGY
*               GENERATION
*     T---------------MATRIX OF TEMPERATURES AFTER HALF-TIME STEP
*               IN THE Z DIRECTION
*     TEMP------------TEMPORARY STORAGE VECTOR FOR TEMPERATURES
*               CALCULATED IN SUBROUTINE TRIDAG
*     TIME------------DURATION OF CALCULATIONS
*     TSTAR1----------MATRIX OF TEMPERATURES AFTER HALF-TIME STEP
*               IN THE X DIRECTION
*     TSTAR2----------MATRIX OF TEMPERATURES AFTER HALF-TIME STEP
*               IN THE Y DIRECTION
*
*
*   B. ADDITIONAL VARIABLES FOR SATELLITE APPLICATIONS:
*
*     ALBED1,2,3,4,5,6---ALBEDO FLUX RECEIVED ON A SURFACE
*     DIST--------------SUM OF EARTH RADIUS AND SATELLITE ALTITUDE
*     EARTH1,2,3,4,5,6---EARTH FLUX RECEIVED ON A SURFACE
*     MU----------------GRAVITATIONAL CONSTANT
*     PERIOD------------THE TIME FOR ONE SATELLITE ORBIT
*     RE----------------RADIUS OF THE EARTH
*     SOLAR1,2,3,4,5,6---SOLAR FLUX RECEIVED ON A SURFACE
*     SUN---------------THE TIME IN MINUTES THE SATELLITE RECEIVES
*                 SUNLIGHT DURING ONE ORBIT
*
```

```
*
*   C. VARIABLES FROM EXTERNAL PROGRAMS:
*
*     CTIME-----USED IN EXTERNAL IMSL ROUTINE FOR DETERMINING
*           AMOUNT OF CPU TIME USED
*     NOUT------USED IN EXTERNAL IMSL ROUTINE FOR DETERMINING
*           AMOUNT OF CPU TIME USED
*     TIMEO-----CPU TIME AT BEGINNING OF CALCULATIONS
*     TIME1-----CPU TIME AT END OF CALCULATIONS
*     UMACH-----USED IN EXTERNAL IMSL ROUTINE FOR DETERMINING
*           AMOUNT OF CPU TIME USED
*
*
************************************************************************
************************************************************************
*
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT(30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
```

```
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      EXTERNAL CTIME, UMACH
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
```

```
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
*************************************************************************
*************************************************************************
*
*     INITIALIZING CPU TIME
*
      TIMEO = CTIME()
*
*************************************************************************
*************************************************************************
*
*     VARIOUS PARAMETERS TO BE USED THROUGHOUT THE PROGRAM
*     WILL NOW BE READ INTO THE PROGRAM.
*
      WRITE(*,*) 'ENTER TINIT, THE INITIAL TEMPERATURE.'
      READ(*,*) TINIT
      WRITE(*,*) 'ENTER K1, THE THERMAL CONDUCTIVITY.'
      READ(*,*) K1
      WRITE(*,*) 'ENTER CP, THE SPECIFIC HEAT OF THE MATERIAL.'
      READ(*,*) CP
      WRITE(*,*) 'ENTER RHO, THE MATERIAL DENSITY.'
      READ(*,*) RHO
      WRITE(*,*) 'ENTER GEN, THE INITIAL GENERATION TERM.'
      READ(*,*) GEN
      WRITE(*,*) 'ENTER LX, THE LENGTH IN THE X DIRECTION.'
```

```
READ(*,*) LX
WRITE(*,*) 'ENTER LY, THE LENGTH IN THE Y DIRECTION.'
READ(*,*) LY
WRITE(*,*) 'ENTER LZ, THE LENGTH IN THE Z DIRECTION.'
READ(*,*) LZ
WRITE(*,*) 'ENTER M, THE NUMBER OF NODES IN THE X DIRECTION.'
READ(*,*) M
WRITE(*,*) 'ENTER N, THE NUMBER OF NODES IN THE Y DIRECTION.'
READ(*,*) N
WRITE(*,*) 'ENTER P, THE NUMBER OF NODES IN THE Z DIRECTION.'
READ(*,*) P
WRITE(*,*) 'ENTER DELT, THE TIME INCREMENT IN SECONDS.'
READ(*,*) DELT
WRITE(*,*) 'ENTER FREQ, THE NUMBER OF TIME STEPS BETWEEN'
WRITE(*,*) 'SUCCESSIVE PRINTOUTS.'
READ(*,*) FREQ
WRITE(*,*) 'ENTER ALT, THE SATELLITE ALTITIUDE IN KILOMETERS.'
WRITE(*,*) 'NOTE: FOR AN EARTH-BASED MODEL, ENTER 0 FOR ALT.'
READ(*,*) ALT
WRITE(*,*) 'ENTER ECLIPS, THE TIME IN MINUTES THE SATELLITE WILL'
WRITE(*,*) 'BE IN THE SHADOW OF THE EARTH DURING AN ORBIT.'
WRITE(*,*) 'NOTE: FOR EARTH-BASED MODELS, ENTER 0 FOR ECLIPS.'
READ(*,*) ECLIPS
WRITE(*,*) 'ENTER FE, THE GEOMETRIC FACTOR FOR EARTH FLUX.'
WRITE(*,*) 'NOTE: FOR GEOSYNCHRONOUS SATELLITES AND'
WRITE(*,*) 'EARTH-BASED MODELS, ENTER 0 FOR FE.'
READ(*,*) FE
WRITE(*,*) 'ENTER FA, THE GEOMETRIC FACTOR FOR ALBEDO FLUX.'
WRITE(*,*) 'NOTE: FOR GEOSYNCHRONOUS SATELLITES AND'
WRITE(*,*) 'EARTH-BASED MODELS, ENTER 0 FOR FA.'
READ(*,*) FA
WRITE(*,*) 'ENTER SOLAR, THE SOLAR FLUX.'
WRITE(*,*) 'NOTE: AVERAGE ANNUAL VALUE IS 1353.'
WRITE(*,*) 'NOTE: FOR EARTH-BASED MODELS, ENTER 0 FOR SOLAR.'
```

```
      READ(*,*) SOLAR
      WRITE(*,*) 'ENTER EARTH, THE THERMAL RADIATIVE FLUX OF EARTH.'
      WRITE(*,*) 'NOTE: AVERAGE ANNUAL VALUE IS 237.'
      WRITE(*,*) 'NOTE: FOR GEOSYNCHRONOUS SATELLITES AND'
      WRITE(*,*) 'EARTH-BASED MODELS, ENTER 0 FOR EARTH.'
      READ(*,*) EARTH
      WRITE(*,*) 'ENTER ALBCO, THE ALBEDO COEFFICIENT.'
      WRITE(*,*) 'NOTE: AVERAGE ANNUAL VALUE IS 0.3.'
      WRITE(*,*) 'NOTE: FOR GEOSYNCHRONOUS SATELLITES AND'
      WRITE(*,*) 'EARTH-BASED MODELS, ENTER 0 FOR ALBCO.'
      READ(*,*) ALBCO
      WRITE(*,*) 'ENTER VAL: THIS WILL DETERMINE IF YOU WANT THE'
      WRITE(*,*) 'THE VALIDATION SUBROUTINE CALLED. IF YOU WANT'
      WRITE(*,*) 'THE VALIDATION SUBROUTINE CALLED, ENTER 1 FOR'
      WRITE(*,*) 'YES. IF YOUR ANSWER IS NO, ENTER 0.'
      READ(*,*) VAL
*
*     INPUTS FOR LEFT FACE
*
      WRITE(*,*) 'ENTER TAMB1, THE AMBIENT TEMP. ON THE LEFT FACE.'
      READ(*,*) TAMB1
      WRITE(*,*) 'ENTER H1, THE LEFT FACE HEAT TRANSFER COEFFICIENT.'
      READ(*,*) H1
      WRITE(*,*) 'ENTER FLUX1, THE FLUX ON THE LEFT FACE.'
      READ(*,*) FLUX1
      WRITE(*,*) 'ENTER SIG1, THE LEFT FACE STEFAN-BOLTZMANN'
      WRITE(*,*) 'CONSTANT. NOTE: IF RADIATION NOT INVOLVED,
      WRITE(*,*) 'ENTER 0 FOR SIG1.'
      READ(*,*) SIG1
      WRITE(*,*) 'ENTER EPS1, THE LEFT FACE EMISSIVITY.'
      READ(*,*) EPS1
      WRITE(*,*) 'ENTER ABS1, THE LEFT FACE ABSORPTIVITY.'
      WRITE(*,*) 'NOTE: FOR EARHT-BASED MODELS, ENTER 0 FOR ABS1.'
      READ(*,*) ABS1
```

```
      WRITE(*,*) 'ENTER SAC1, THE LEFT FACE SOLAR ASPECT COEFFICIENT.'
      WRITE(*,*) 'NOTE: FOR EARTH-BASED MODELS, ENTER 0 FOR SAC1.'
      READ(*,*) SAC1
*
*     INPUTS FOR RIGHT FACE
*
      WRITE(*,*) 'ENTER TAMB2, THE AMBIENT TEMP. ON THE '
      WRITE(*,*) 'RIGHT FACE.'
      READ(*,*) TAMB2
      WRITE(*,*) 'ENTER H2, THE RIGHT FACE HEAT TRANSFER'
      WRITE(*,*) 'COEFFICIENT.'
      READ(*,*) H2
      WRITE(*,*) 'ENTER FLUX2, THE FLUX ON THE RIGHT FACE.'
      READ(*,*) FLUX2
      WRITE(*,*) 'ENTER SIG2, THE RIGHT FACE STEFAN-BOLTZMANN'
      WRITE(*,*) 'CONSTANT.'
      WRITE(*,*) 'NOTE: IF RADIATION NOT INVOLVED, ENTER 0 FOR SIG1.'
      READ(*,*) SIG2
      WRITE(*,*) 'ENTER EPS2, THE RIGHT FACE EMISSIVITY.'
      READ(*,*) EPS2
      WRITE(*,*) 'ENTER ABS2, THE RIGHT FACE ABSORPTIVITY.'
      WRITE(*,*) 'NOTE: FOR EARTH-BASED MODELS, ENTER 0 FOR ABS2.'
      READ(*,*) ABS2
      WRITE(*,*) 'ENTER SAC2, THE RIGHT FACE SOLAR ASPECT'
      WRITE(*,*) 'COEFFICIENT NOTE: FOR EARTH-BASED MODELS.'
      WRITE(*,*) 'ENTER 0 FOR SAC2.'
      READ(*,*) SAC2
*
*     INPUTS FOR BOTTOM FACE
*
      WRITE(*,*) 'ENTER TAMB3, THE AMBIENT TEMP. ON THE'
      WRITE(*,*) 'BOTTOM FACE.'
      READ(*,*) TAMB3
      WRITE(*,*) 'ENTER H3, THE BOTTOM FACE HEAT TRANSFER'
```

```
      WRITE(*,*) 'COEFFICIENT.'
      READ(*,*) H3
      WRITE(*,*) 'ENTER FLUX3, THE FLUX ON THE BOTTOM FACE.'
      READ(*,*) FLUX3
      WRITE(*,*) 'ENTER SIG3, THE BOTTOM FACE STEFAN-BOLTZMANN'
      WRITE(*,*) 'CONSTANT.'
      WRITE(*,*) 'NOTE: IF RADIATION NOT INVOLVED, ENTER 0 FOR SIG3.'
      READ(*,*) SIG3
      WRITE(*,*) 'ENTER EPS3, THE BOTTOM FACE EMISSIVITY.'
      READ(*,*) EPS3
      WRITE(*,*) 'ENTER ABS3, THE BOTTOM FACE ABSORPTIVITY.'
      WRITE(*,*) 'NOTE: FOR EARTH-BASED MODELS, ENTER 0 FOR ABS3.'
      READ(*,*) ABS3
      WRITE(*,*) 'ENTER SAC3, THE BOTTOM FACE SOLAR ASPECT'
      WRITE(*,*) 'COEFFICIENT. NOTE: FOR EARTH-BASED MODELS,'
      WRITE(*,*) 'ENTER 0 FOR SAC3.'
      READ(*,*) SAC3
*
*     INPUTS FOR THE TOP FACE
*
      WRITE(*,*) 'ENTER TAMB4, THE AMBIENT TEMP. ON THE TOP FACE.'
      READ(*,*) TAMB4
      WRITE(*,*) 'ENTER H4, THE TOP FACE HEAT TRANSFER COEFFICIENT.'
      READ(*,*) H4
      WRITE(*,*) 'ENTER FLUX4, THE FLUX ON THE TOP FACE.'
      READ(*,*) FLUX4
      WRITE(*,*) 'ENTER SIG4, THE TOP FACE STEFAN-BOLTZMANN'
      WRITE(*,*) 'CONSTANT. NOTE: IF RADIATION NOT INVOLVED,'
      WRITE(*,*) 'ENTER 0 FOR SIG4.'
      READ(*,*) SIG4
      WRITE(*,*) 'ENTER EPS4, THE TOP FACE EMISSIVITY.'
      READ(*,*) EPS4
      WRITE(*,*) 'ENTER ABS4, THE TOP FACE ABSORPTIVITY.'
      WRITE(*,*) 'NOTE: FOR EARTH-BASED MODELS, ENTER 0 FOR ABS4.'
```

```
      READ(*,*) ABS4
      WRITE(*,*) 'ENTER SAC4, THE TOP FACE SOLAR ASPECT COEFFICIENT.'
      WRITE(*,*) 'NOTE: FOR EARTH-BASED MODELS, ENTER 0 FOR SAC4.'
      READ(*,*) SAC4
*
*     INPUTS FOR THE FRONT FACE
*
      WRITE(*,*) 'ENTER TAMB5, THE AMBIENT TEMP. ON THE FRONT FACE.'
      READ(*,*) TAMB5
      WRITE(*,*) 'ENTER H5, THE FRONT FACE HEAT TRANSFER'
      WRITE(*,*) 'COEFFICIENT.'
      READ(*,*) H5
      WRITE(*,*) 'ENTER FLUX5, THE FLUX ON THE FRONT FACE.'
      READ(*,*) FLUX5
      WRITE(*,*) 'ENTER SIG5, THE FRONT FACE STEFAN-BOLTZMANN'
      WRITE(*,*) 'CONSTANT.'
      WRITE(*,*) 'NOTE: IF RADIATION NOT INVOLVED, ENTER 0 FOR SIG5.'
      READ(*,*) SIG5
      WRITE(*,*) 'ENTER EPS5, THE FRONT FACE EMISSIVITY.'
      READ(*,*) EPS5
      WRITE(*,*) 'ENTER ABS5, THE FRONT FACE ABSORPTIVITY.'
      WRITE(*,*) 'NOTE: FOR EARTH-BASED MODELS, ENTER 0 FOR ABS5.'
      READ(*,*) ABS5
      WRITE(*,*) 'ENTER SAC5, THE FRONT FACE SOLAR ASPECT'
      WRITE(*,*) 'COEFFICIENT. NOTE: FOR EARTH-BASED MODELS,'
      WRITE(*,*) 'ENTER 0 FOR SAC5.'
      READ(*,*) SAC5
*
*     INPUTS FOR THE BACK FACE
*
      WRITE(*,*) 'ENTER TAMB6, THE AMBIENT TEMP. ON THE BACK FACE.'
      READ(*,*) TAMB6
      WRITE(*,*) 'ENTER H6, THE BACK FACE HEAT TRANSFER COEFFICIENT.'
      READ(*,*) H6
```

```
      WRITE(*,*) 'ENTER FLUX6, THE FLUX ON THE BACK FACE.'
      READ(*,*) FLUX6
      WRITE(*,*) 'ENTER SIG6, THE BACK FACE STEFAN-BOLTZMANN'
      WRITE(*,*) 'CONSTANT.'
      WRITE(*,*) 'NOTE: IF RADIATION NOT INVOLVED, ENTER 0 FOR SIG6.'
      READ(*,*) SIG6
      WRITE(*,*) 'ENTER EPS6, THE BACK FACE EMISSIVITY.'
      READ(*,*) EPS6
      WRITE(*,*) 'ENTER ABS6, THE BACK FACE ABSORPTIVITY.'
      WRITE(*,*) 'NOTE: FOR EARTH-BASED MODELS, ENTER 0 FOR ABS6.'
      READ(*,*) ABS6
      WRITE(*,*) 'ENTER SAC6, THE BACK FACE SOLAR ASPECT COEFFICIENT.'
      WRITE(*,*) 'NOTE: FOR EARTH-BASED MODELS, ENTER 0 FOR SAC6.'
      READ(*,*) SAC6
*
*************************************************************************
*************************************************************************
*
*     CALCULATIONS OF CONSTANTS TO BE USED THROUGHOUT
*     THE PROGRAM
*
      DELX = LX/(M-1)
      DELY = LY/(N-1)
      DELZ = LZ/(P-1)
*
      R = DELX/(DELY*DELZ)
      R1 = (DELX**2)/(DELY**2)
      R2 = (DELX**2)/(DELZ**2)
*
      ALFA = K1/(RHO*CP)
      FO = ALFA*(DELT/2)/(DELX**2)
*
      BIO1 = H1/K1*DELX
      BIO2 = H2/K1*DELX
```

```
      BIO3 = H3 K1*DELX
      BIO4 = H4 K1*DELX
      BIO5 = H5 K1*DELX
      BIO6 = H6 K1*DELX
*
      U1 = R1*FO
      U2 = R2*FO
      U3 = 2*R1*FO
      U4 = 2*R2*FO
      U5 = 2*FO
*
      V1 = 2*FO*BIO1*SQRT(R1)
      V2 = 2*FO*BIO2*SQRT(R1)
      V3 = 2*FO*BIO3*SQRT(R2)
      V4 = 2*FO*BIO4*SQRT(R2)
      V5 = 2*FO*BIO5
      V6 = 2*FO*BIO6
*
      EARTH1 = EPS1*FE*EARTH
      EARTH2 = EPS2*FE*EARTH
      EARTH3 = EPS3*FE*EARTH
      EARTH4 = EPS4*FE*EARTH
      EARTH5 = EPS5*FE*EARTH
      EARTH6 = EPS6*FE*EARTH
*
      Q1 = 2*FO*R1*DELY/K1
      Q2 = 2*FO*R2*DELZ/K1
      Q3 = 2*FO*DELX/K1
*
      P1 = Q1*(FLUX1 + EARTH1)
      P2 = Q1*(FLUX2 + EARTH2)
      P3 = Q2*(FLUX3 + EARTH3)
      P4 = Q2*(FLUX4 + EARTH4)
      P5 = Q3*(FLUX5 + EARTH5)
```

```
      P6 = Q3*(FLUX6 + EARTH6)
*
      X1 = -2*FO*SIG1*EPS1*R1*DELY/K1
      X2 = -2*FO*SIG2*EPS2*R1*DELY/K1
      X3 = -2*FO*SIG3*EPS3*R2*DELZ/K1
      X4 = -2*FO*SIG4*EPS4*R2*DELZ/K1
      X5 = -2*FO*SIG5*EPS5*DELX/K1
      X6 = -2*FO*SIG6*EPS6*DELX/K1
*
      Y1 = FO*(DELX**2)/K1
*
      ALBED1 = ABS1*FA*SOLAR*ALBCO
      ALBED2 = ABS2*FA*SOLAR*ALBCO
      ALBED3 = ABS3*FA*SOLAR*ALBCO
      ALBED4 = ABS4*FA*SOLAR*ALBCO
      ALBED5 = ABS5*FA*SOLAR*ALBCO
      ALBED6 = ABS6*FA*SOLAR*ALBCO
*
      SOLAR1 = SOLAR*SAC1*ABS1
      SOLAR2 = SOLAR*SAC2*ABS2
      SOLAR3 = SOLAR*SAC3*ABS3
      SOLAR4 = SOLAR*SAC4*ABS4
      SOLAR5 = SOLAR*SAC5*ABS5
      SOLAR6 = SOLAR*SAC6*ABS6
*
      W1 = P1 + Q1*(SOLAR1 + ALBED1)
      W2 = P2 + Q1*(SOLAR2 + ALBED2)
      W3 = P3 + Q2*(SOLAR3 + ALBED3)
      W4 = P4 + Q2*(SOLAR4 + ALBED4)
      W5 = P5 + Q3*(SOLAR5 + ALBED5)
      W6 = P6 + Q3*(SOLAR6 + ALBED6)
*
*     FOR EARTH-BASED MODELS, CALCULATIONS FOR PERIOD AND SUN
```

```
*    ARE NOT REQUIRED.
*
     IF(ALT .LE. 0) GO TO 5
*
     PI = ATAN(1.0)*4.0
     MU = 398603.2
     RE = 6378.165
     DIST = RE + ALT
     ECLIPS = ECLIPS*60
     PERIOD = 2*PI*SQRT(DIST**3/MU)
     SUN = PERIOD - ECLIPS
*
*************************************************************************
*************************************************************************
*
*    INITIALIZING THE TIME AND COUNTER
*
   5 TIME = 0.0
     COUNT = 0
*
*************************************************************************
*************************************************************************
*
*    OPENING FILES FOR PRINT STATEMENTS
*
     OPEN(UNIT=1, FILE='DATA1')
     OPEN(UNIT=2, FILE='DATA2')
     OPEN(UNIT=3, FILE='DATA3')
     OPEN(UNIT=4, FILE='DATA4')
     OPEN(UNIT=5, FILE='DATA5')
*
*************************************************************************
*************************************************************************
*
```

```
*    INITIALIZE ALL TEMPERATURES TO TINIT, THE INITIAL
*    TEMPERATURE.
*
     DO 10 I = 1, M
       DO 20 J = 1, N
         DO 30 K = 1, P
*
     T(I,J,K) = TINIT
*
 30      CONTINUE
 20    CONTINUE
 10  CONTINUE
*
*************************************************************************
*************************************************************************
*
*    INITIALIZE ALL GENERATION TERMS TO GEN, THE INITIAL
*    GENERATION.
*
     DO 40 I = 1,M
       DO 50 J = 1,N
         DO 60 K = 1,P
*
     QDOT(I,J,K) = GEN
*
 60      CONTINUE
 50    CONTINUE
 40  CONTINUE
*
*************************************************************************
*************************************************************************
*
*    INCREMENTING THE TIME AND COUNTER
*
```

```
 70 TIME = TIME + DELT
    COUNT = COUNT + 1
*
*************************************************************************
*************************************************************************
*
*   THIS PART OF THE PROGRAM CALCULATES THE SOLAR FLUXES AND
*   ALBEDO FLUXES WHICH WILL CHANGE WITH TIME AS THE SATELLITE
*   REVOLVES AROUND THE EARTH. DURING ECLIPSE PERIODS,
*   THE SOLAR FLUX WILL BE ZERO. FOR EARTH-BASED MODELS,
*   THIS PART OF THE PROGRAM AND THE SUBROUTINES PER1 AND
*   PER2 WILL BE BYPASSED USING AN IF THEN STATEMENT. IF
*   THE USER INPUTS 0 FOR ALT (I.E., EARTH-BASED MODEL)
*   THEN THIS PART OF THE PROGRAM AND THE TWO SUBROUTINES
*   WILL BE BYPASSED.
*
*
    IF(ALT .LE. 0) GO TO 75
*
    IF(TIME .LT. (12*(SUN+ECLIPS))) THEN
      CALL PER1
    ELSE
      CALL PER2
    ENDIF
*
    ALBED1 = ABS1*FA*SOLAR*ALBCO
    ALBED2 = ABS2*FA*SOLAR*ALBCO
    ALBED3 = ABS3*FA*SOLAR*ALBCO
    ALBED4 = ABS4*FA*SOLAR*ALBCO
    ALBED5 = ABS5*FA*SOLAR*ALBCO
    ALBED6 = ABS6*FA*SOLAR*ALBCO
*
    SOLAR1 = SOLAR*SAC1*ABS1
    SOLAR2 = SOLAR*SAC2*ABS2
```

```
      SOLAR3 = SOLAR*SAC3*ABS3
      SOLAR4 = SOLAR*SAC4*ABS4
      SOLAR5 = SOLAR*SAC5*ABS5
      SOLAR6 = SOLAR*SAC6*ABS6
*
      W1 = P1 + Q1*(SOLAR1 + ALBED1)
      W2 = P2 + Q1*(SOLAR2 + ALBED2)
      W3 = P3 + Q2*(SOLAR3 + ALBED3)
      W4 = P4 + Q2*(SOLAR4 + ALBED4)
      W5 = P5 + Q3*(SOLAR5 + ALBED5)
      W6 = P6 + Q3*(SOLAR6 + ALBED6)
*
************************************************************************
************************************************************************
*
*     SETTING COEFFICIENT ARRAYS A, B, AND C IN THE X DIRECTION
*
 75   DO 80 I = 1, M
*
      B(I) = (1 + U5)
*
      IF(I .EQ. 1) THEN
        A(I) = 0.0
        C(I) = -U5
*
      ELSE IF((I .GT. 1) .AND. (I .LT. M)) THEN
        A(I) = -FO
        C(I) = -FO
*
      ELSE
        A(I) = -U5
        C(I) = 0.0
*
```

```
      ENDIF
*
 80   CONTINUE
*
*     SETTING COEFFICIENT ARRAY D IN THE X DIRECTION
*
      DO 90 K = 1, P
        DO 100 J = 1, N
          DO 110 I = 1, M
*
      IF (K.EQ.1) THEN
        CALL XRAY1
      ELSE IF ((K.GT.1) .AND. (K.LT.P)) THEN
        CALL XRAY2
      ELSE
        CALL XRAY3
      ENDIF
*
 110      CONTINUE
*
      IF = 1
      L = M
*
      CALL TRIDAG
*
          DO 120  I = IF,L
*
      TSTAR1(I,J,K) =  TEMP(I)
*
 120      CONTINUE
 100    CONTINUE
  90  CONTINUE
*
*************************************************************************
```

```
*****************************************************************
*
*   SETTING COEFFICIENT ARRAYS A, B, AND C IN THE Y DIRECTION
*
    DO 130 J = 1,N
*
    B(J) = (1 + U3)
*
    IF(J .EQ. 1) THEN
      A(J) = 0.0
      C(J) = -U3
*
    ELSE IF((J .GT. 1) .AND. (J .LT. N)) THEN
      A(J) = -U1
      C(J) = -U1
*
    ELSE
      A(J) = -U3
      C(J) = 0.0
*
    ENDIF
*
 130  CONTINUE
*
*
*   SETTING COEFFICIENT ARRAY D IN THE Y DIRECTION
*
    DO 140 K = 1,P
      DO 150 I = 1,M
        DO 160 J = 1,N
*
    IF(K.EQ.1) THEN
      CALL YANK1
    ELSE IF( ( K.GT.1) .AND. (K.LT.P) ) THEN
```

```
      CALL YANK2
    ELSE
      CALL YANK3
    ENDIF
*
 160      CONTINUE
*
    IF=1
    L=N
*
    CALL TRIDAG
*
    DO 170 J=IF, L
*
    TSTAR2(I,J,K) = TEMP(J)
*
 170      CONTINUE
 150    CONTINUE
 140  CONTINUE
*
*************************************************************************
*************************************************************************
*
*    SETTING COEFFICIENT ARRAYS A, B, AND C IN THE Z DIRECTION
*
    DO 180 K=1, P
*
    B(K) = (1 + U4)
*
    IF(K .EQ. 1) THEN
      A(K) = 0.0
      C(K) = -U4
*
    ELSE IF((K .GT. 1) .AND. (K .LT. P)) THEN
```

```
      A(K) = -U2
      C(K) = -U2
*
    ELSE
      A(K) = -U4
      C(K) = 0.0
*
    ENDIF
*
 180  CONTINUE
*
*
*   SETTING COEFFICIENT ARRAY D IN THE Z DIRECTION
*
    DO 190 J = 1, N
      DO 200 I = 1,M
        DO 210 K = 1,P
*
    IF(J.EQ.1) THEN
      CALL ZULU1
    ELSE IF((J.GT.1) .AND. (J.LT.N)) THEN
      CALL ZULU2
    ELSE
      CALL ZULU3
    ENDIF
*
 210      CONTINUE
*
    IF = 1
    L = P
*
    CALL TRIDAG
*
```

```
      DO 220 K = IF, L
*
     T(I,J,K) = TEMP(K)
*
 220     CONTINUE
 200   CONTINUE
 190 CONTINUE
*
*************************************************************************
*************************************************************************
*
*   THIS PART OF PROGRAM IS USED TO DETERMINE IF THE
*   SUBROUTINE VALID WILL BE CALLED. THIS SUBROUTINE
*   IS USED TO VALIDATE THE PROGRAM BRIAN AND COMPARE
*   TEMPERATURE DIFFERENCES AT SELECTED NODES WITH THE
*   DIFFERENCES FOUND BY THE PROGRAMS EXPLICIT AND VALID.
*
*   IF SUBROUTINE VALID IS USED THE SUBROUTINE OUTPUT WILL BE
*   BYPASSED. IT SHOULD BE NOTED THAT THE SUBROUTINE VALID
*   WILL NOT NORMALLY BE USED.
*
   IF(VAL .NE. 1) GO TO 250
   IF(TIME .EQ. DELT) CALL VALID
   IF(COUNT .NE. FREQ) GO TO 240
   CALL VALID
*
 240 IF(TIME .LE.   3600) GO TO 70
   GO TO 310
*
*
*   THIS PART OF PROGRAM WILL DETERMINE IF THE TIME AND/OR
*   COUNT CRITERIA FOR CALLING THE SUBROUTINE OUTPUT IS MET.
*   THIS SUBROUTINE IS USED FOR VARIOUS PRINTOUTS.
*
```

```
250  IF(TIME .EQ. DELT) CALL OUTPUT
     IF(COUNT .NE. FREQ) GO TO 300
*
     CALL OUTPUT
*
*    NOTE: IF THE SUBROUTINE OUTPUT IS CALLED, THE COUNT WILL BE
*          REINITIALIZED TO ZERO IN THE SUBROUTINE OUTPUT
*
*    NOTE: THE TIME PERIOD SELECTED FOR THIS PROBLEM IS 86400
*          SECONDS OR 1 DAY.  ANOTHER TIME LIMIT CAN BE USED
*          IF THE USER DESIRES A PERIOD DIFFERENT THAN 1 DAY.
*
300  IF(TIME .LE. 86400) GO TO 70
*
*************************************************************************
*
*    CALCULATING AND PRINTING CPU TIME USED DURING THE
*    OPERATING OF THIS PROGRAM
*
310  PRINT*
     PRINT*
*
     TIME1 = CTIME()
     CALL UMACH(2,NOUT)
     WRITE(NOUT,*) 'CPU TIME (SECONDS) = ', TIME1-TIME0
*
     STOP
     END
*
*************************************************************************
*************************************************************************
*
     SUBROUTINE PER1
     DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
```

```
      DIMENSION BETA( 30), GAMMA( 30), QDOT(30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
```

```
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
      IF(TIME .LT. SUN) THEN
        GO TO 10
*
      ELSE IF((TIME .GE. SUN) .AND. (TIME .LT. (SUN+ECLIPS))) THEN
        GO TO 20
*
      ELSE IF((TIME .GE. (SUN+ECLIPS)) .AND.
     &  (TIME .LT. (2*SUN + ECLIPS))) THEN
```

```
      GO TO 10
*
     ELSE IF((TIME .GE. (2*SUN + ECLIPS)) .AND.
    &  (TIME .LT. (2*(SUN+ECLIPS)))) THEN
      GO TO 20
*
     ELSE IF((TIME .GE. (2*(SUN+ECLIPS))) .AND.
    &  (TIME .LT. (3*SUN + 2*ECLIPS))) THEN
      GO TO 10
*
     ELSE IF((TIME .GE. (3*SUN + 2*ECLIPS)) .AND.
    &  (TIME .LT. 3*(SUN+ECLIPS))) THEN
      GO TO 20
*
     ELSE IF((TIME .GE. (3*(SUN+ECLIPS))) .AND.
    &  (TIME .LT. (4*SUN + 3*ECLIPS))) THEN
      GO TO 10
*
     ELSE IF((TIME .GE. (4*SUN + 3*ECLIPS)) .AND.
    &  (TIME .LT. (4*(SUN+ECLIPS)))) THEN
      GO TO 20
*
     ELSE IF((TIME .GE. 4*(SUN+ECLIPS)) .AND.
    &  (TIME .LT. (5*SUN + 4*ECLIPS))) THEN
      GO TO 10
*
     ELSE IF((TIME .GE. (5*SUN +4*ECLIPS)) .AND.
    &  (TIME .LT. (5*(SUN+ECLIPS)))) THEN
      GO TO 20
*
     ELSE IF((TIME .GE. (5*(SUN+ECLIPS))) .AND.
    &  (TIME .LT. (6*SUN + 5*ECLIPS))) THEN
      GO TO 10
*
```

```
      ELSE IF((TIME .GE. (6*SUN + 5*ECLIPS)) .AND.
     &  (TIME .LT. (6*(SUN+ECLIPS)))) THEN
        GO TO 20
*
      ELSE IF((TIME .GE. (6*(SUN+ECLIPS))) .AND.
     &  (TIME .LT. (7*SUN + 6*ECLIPS))) THEN
        GO TO 10
*
      ELSE IF((TIME .GE. (7*SUN + 6*ECLIPS)) .AND.
     &  (TIME .LT. (7*(SUN+ECLIPS)))) THEN
        GO TO 20
*
      ELSE IF((TIME .GE. (7*(SUN+ECLIPS))) .AND.
     &  (TIME .LT. (8*SUN + 7*ECLIPS))) THEN
        GO TO 10
*
      ELSE IF((TIME .GE. (8*SUN + 7*ECLIPS)) .AND.
     &  (TIME .LT. (8*(SUN+ECLIPS)))) THEN
        GO TO 20
*
      ELSE IF((TIME .GE. (8*(SUN+ECLIPS))) .AND.
     &  (TIME .LT. (9*SUN + 8*ECLIPS))) THEN
        GO TO 10
*
      ELSE IF((TIME .GE. (9*SUN + 8*ECLIPS)) .AND.
     &  (TIME .LT. (9*(SUN+ECLIPS)))) THEN
        GO TO 20
*
      ELSE IF((TIME .GE. (9*(SUN+ECLIPS))) .AND.
     &  (TIME .LT. (10*SUN + 9*ECLIPS))) THEN
        GO TO 10
*
      ELSE IF((TIME .GE. (10*SUN + 9*ECLIPS)) .AND.
     &  (TIME .LT. (10*(SUN+ECLIPS)))) THEN
```

```
      GO TO 20
*
    ELSE IF((TIME .GE. (10*(SUN+ECLIPS))) .AND.
   &  (TIME .LT. (11*SUN + 10*ECLIPS))) THEN
      GO TO 10
*
    ELSE IF((TIME .GE. (11*SUN + 10*ECLIPS)) .AND.
   &  (TIME .LT. (11*(SUN+ECLIPS)))) THEN
      GO TO 20
*
    ELSE IF((TIME .GE. (11*(SUN+ECLIPS))) .AND.
   &  (TIME .LT. (12*SUN + 11*ECLIPS))) THEN
      GO TO 10
*
    ELSE
      GO TO 20
*
    ENDIF
*
 10 SOLAR = 1353
    WRITE(6,*) 'SOLAR = ',SOLAR
    GO TO 30
*
 20 SOLAR = 0
    WRITE(6,*) 'SOLAR = ',SOLAR
*
 30 RETURN
    END
*
*************************************************************************
*************************************************************************
*
    SUBROUTINE PER2
    DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
```

```
      DIMENSION BETA( 30), GAMMA( 30), QDOT(30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
```

```
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIME0, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
      IF((TIME .GE. (12*(SUN+ECLIPS))) .AND.
     & (TIME .LT. (13*SUN + 12*ECLIPS))) THEN
        GO TO 10
*
      ELSE IF((TIME .GE. (13*SUN + 12*ECLIPS)) .AND.
     & (TIME .LT. (13*(SUN+ECLIPS)))) THEN
        GO TO 20
*
      ELSE IF((TIME .GE. (13*(SUN+ECLIPS))) .AND.
```

```
     & (TIME .LT. (14*SUN + 13*ECLIPS))) THEN
       GO TO 10
*
      ELSE IF((TIME .GE. (14*SUN + 13*ECLIPS)) .AND.
     & (TIME .LT. (14*(SUN+ECLIPS)))) THEN
       GO TO 20
*
      ELSE IF((TIME .GE. (14*(SUN+ECLIPS))) .AND.
     & (TIME .LT. (15*SUN + 14*ECLIPS))) THEN
       GO TO 10
*
      ELSE IF((TIME .GE. (15*SUN + 14*ECLIPS)) .AND.
     & (TIME .LT. (15*(SUN+ECLIPS)))) THEN
       GO TO 20
*
      ELSE IF((TIME .GE. (15*(SUN+ECLIPS))) .AND.
     & (TIME .LT. (16*SUN + 15*ECLIPS))) THEN
       GO TO 10
*
      ELSE IF((TIME .GE. (16*SUN + 15*ECLIPS)) .AND.
     & (TIME .LT. (16*(SUN+ECLIPS)))) THEN
       GO TO 20
*
      ELSE IF((TIME .GE. (16*(SUN+ECLIPS))) .AND.
     & (TIME .LT. (17*SUN + 16*ECLIPS))) THEN
       GO TO 10
*
      ELSE IF((TIME .GE. (17*SUN + 16*ECLIPS)) .AND.
     & (TIME .LT. (17*(SUN+ECLIPS)))) THEN
       GO TO 20
      ELSE
       GO TO 10
*
```

```
      ENDIF
*
   10 SOLAR = 1353
      WRITE(6,*) 'SOLAR = ',SOLAR
      GO TO 30
*
   20 SOLAR = 0
      WRITE(6,*) 'SOLAR = ',SOLAR
*
   30 RETURN
      END
*
*************************************************************************
*************************************************************************
*
*     SUBROUTINES XRAY1, XRAY2, AND XRAY3 COMPUTE COEFFICIENT
*     ARRAY D IN THE X DIRECTION
*
*     SUBROUTINE XRAY1
*
      SUBROUTINE XRAY1
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT( 30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
```

```
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
```

```
      COMMON COUNT, FREQ
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
      IF((I.EQ.1) .AND. (J.EQ.1)) THEN
        D(I) = T(I,J,K) + U3*(T(I,J+1,K) - T(I,J,K)) +
     &         U4*(T(I,J,K+1) - T(I,J,K)) + V1*(TAMB1 - T(I,J,K)) +
     &         V3*(TAMB3 - T(I,J,K)) + V6*(TAMB6 - T(I,J,K)) + W1 +
     &         W3 + W6 + X1*(T(I,J,K)**4 - TAMB1**4) +
     &         X3*(T(I,J,K)**4 - TAMB3**4) +
     &         X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((I.GT.1) .AND. (I.LT.M)) .AND. (J.EQ.1)) THEN
        D(I) = T(I,J,K) + U3*(T(I,J+1,K) - T(I,J,K)) +
     &         U4*(T(I,J,K+1) - T(I,J,K)) + V1*(TAMB1 - T(I,J,K)) +
     &         V3*(TAMB3 - T(I,J,K)) + W1 + W3 +
     &         X1*(T(I,J,K)**4 - TAMB1**4) +
     &         X3*(T(I,J,K)**4 - TAMB3**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((I.EQ.M) .AND. (J.EQ.1)) THEN
        D(I) = T(I,J,K) + U3*(T(I,J+1,K) - T(I,J,K)) +
     &         U4*(T(I,J,K+1) - T(I,J,K)) + V1*(TAMB1 - T(I,J,K)) +
     &         V3*(TAMB3 - T(I,J,K)) + V5*(TAMB5 - T(I,J,K)) +
     &         W1 + W3 + W5 + X1*(T(I,J,K)**4 - TAMB1**4) +
     &         X3*(T(I,J,K)**4 - TAMB3**4) +
```

```
     &        X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((I.EQ.1) AND. ((J.GT.1) .AND. (J.LT.N))) THEN
        D(I) = T(I,J,K) + U1*(T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K)) +
     &        U4*(T(I,J,K+1) - T(I,J,K)) + V3*(TAMB3 - T(I,J,K)) +
     &        V6*(TAMB6 - T(I,J,K)) + W3 + W6 +
     &        X3*(T(I,J,K)**4 - TAMB3**4) +
     &        X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((I.GT.1) .AND. (I.LT.M)) .AND. ((J.GT.1) .AND.
     & (J.LT.N))) THEN
        D(I) = T(I,J,K) + U1*(T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K)) +
     &        U4*(T(I,J,K+1) - T(I,J,K)) + V3*(TAMB3 - T(I,J,K)) +
     &        W3 + X3*(T(I,J,K)**4 - TAMB3**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((I.EQ.M) .AND. ((J.GT.1) .AND. (J.LT.N))) THEN
        D(I) = T(I,J,K) + U1*(T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K)) +
     &        U4*(T(I,J,K+1) - T(I,J,K)) + V3*(TAMB3 - T(I,J,K)) +
     &        V5*(TAMB5 - T(I,J,K)) + W3 + W5 +
     &        X3*(T(I,J,K)**4 - TAMB3**4) +
     &        X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((I.EQ.1) .AND. (J.EQ.N)) THEN
        D(I) = T(I,J,K) + U3*(T(I,J-1,K) - T(I,J,K)) +
     &        U4*(T(I,J,K+1) - T(I,J,K)) + V2*(TAMB2 - T(I,J,K)) +
     &        V3*(TAMB3 - T(I,J,K)) + V6*(TAMB6 - T(I,J,K)) +
     &        W2 + W3 + W6 + X2*(T(I,J,K)**4 - TAMB2**4) +
     &        X3*(T(I,J,K)**4 - TAMB3**4) +
     &        X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((I.GT.1) .AND. (I.LT.M)) .AND. (J.EQ.N)) THEN
        D(I) = T(I,J,K) + U3*(T(I,J-1,K) - T(I,J,K)) +
     &        U4*(T(I,J,K+1) - T(I,J,K)) + V2*(TAMB2 - T(I,J,K)) +
     &        V3*(TAMB3 - T(I,J,K)) + W2 + W3 +
```

```
     &        X2*(T(I,J,K)**4 - TAMB2**4) +
     &        X3*(T(I,J,K)**4 - TAMB3**4) + Y1*QDOT(I,J,K)
*
      ELSE
        D(I) = T(I,J,K) + U3*(T(I,J-1,K) - T(I,J,K)) +
     &        U4*(T(I,J,K+1) - T(I,J,K)) + V2*(TAMB2 - T(I,J,K)) +
     &        V3*(TAMB3 - T(I,J,K)) + V5*(TAMB5 - T(I,J,K)) +
     &        W2 + W3 + W5 + X2*(T(I,J,K)**4 - TAMB2**4) +
     &        X3*(T(I,J,K)**4 - TAMB3**4) +
     &        X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ENDIF
*
      RETURN
      END
*
*
*     SUBROUTINE XRAY2
*
      SUBROUTINE XRAY2
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT( 30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
```

```
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ
```

```
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
      IF((I.EQ.1) .AND. (J.EQ.1)) THEN
        D(I) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
     &         U3*(T(I,J+1,K) - T(I,J,K)) + V1*(TAMB1 - T(I,J,K)) +
     &         V6*(TAMB6 - T(I,J,K)) + W1 + W6 +
     &         X1*(T(I,J,K)**4 - TAMB1**4) +
     &         X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((I.GT.1) .AND. (I.LT.M)) .AND. (J.EQ.1)) THEN
        D(I) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
     &         U3*(T(I,J+1,K) - T(I,J,K)) + V1*(TAMB1 - T(I,J,K)) +
     &         W1 + X1*(T(I,J,K)**4 - TAMB1**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((I.EQ.M) .AND. (J.EQ.1)) THEN
        D(I) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
     &         U3*(T(I,J+1,K) - T(I,J,K)) + V1*(TAMB1 - T(I,J,K)) +
     &         V5*(TAMB5 - T(I,J,K)) + W1 + W5 +
     &         X1*(T(I,J,K)**4 - TAMB1**4) +
     &         X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((I.EQ.1) .AND. ((J.GT.1) .AND. (J.LT.N))) THEN
        D(I) = T(I,J,K) + U1*(T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K)) +
     &         U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
```

```
&       V6*(TAMB6 - T(I,J,K)) + W6 +
&       X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
 ELSE IF(((I.GT.1) .AND. (I.LT.M)) .AND. ((J.GT.1) .AND.
& (J.LT.N))) THEN
   D(I) = T(I,J,K) + U1*(T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K)) +
&       U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
&       Y1*QDOT(I,J,K)
*
 ELSE IF((I.EQ.M) .AND. ((J.GT.1) .AND. (J.LT.N))) THEN
   D(I) = T(I,J,K) + U1*(T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K)) +
&       U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
&       V5*(TAMB5 - T(I,J,K)) + W5 +
&       X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
 ELSE IF((I.EQ.1) .AND. (J.EQ.N)) THEN
   D(I) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
&       U3*(T(I,J-1,K) - T(I,J,K)) + V2*(TAMB2 - T(I,J,K)) +
&       V6*(TAMB6 - T(I,J,K)) + W2 + W6 +
&       X2*(T(I,J,K)**4 - TAMB2**4) +
&       X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
 ELSE IF(((I.GT.1) .AND. (I.LT.M)) .AND. (J.EQ.N)) THEN
   D(I) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
&       U3*(T(I,J-1,K) - T(I,J,K)) + V2*(TAMB2 - T(I,J,K)) +
&       W2 + X2*(T(I,J,K)**4 - TAMB2**4) + Y1*QDOT(I,J,K)
*
 ELSE
   D(I) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
&       U3*(T(I,J-1,K) - T(I,J,K)) + V2*(TAMB2 - T(I,J,K)) +
&       V5*(TAMB5 - T(I,J,K)) + W2 + W5 +
&       X2*(T(I,J,K)**4 - TAMB2**4) +
&       X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
```

```
      ENDIF
*
      RETURN
      END
*
*
*     SUBROUTINE XRAY3
*
      SUBROUTINE XRAY3
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT( 30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
```

```
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
```

```
      COMMON P1, P2, P3, P4, P5, P6
*
      IF((I.EQ.1) .AND. (J.EQ.1)) THEN
        D(I) = T(I,J,K) + U3*(T(I,J+1,K) - T(I,J,K)) +
     &        U4*(T(I,J,K-1) - T(I,J,K)) + V1*(TAMB1 - T(I,J,K)) +
     &        V4*(TAMB4 - T(I,J,K)) + V6*(TAMB6 - T(I,J,K)) + W1 +
     &        W4 + W6 + X1*(T(I,J,K)**4 - TAMB1**4) +
     &        X4*(T(I,J,K)**4 - TAMB4**4) +
     &        X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((I.GT.1) .AND. (I.LT.M)) .AND. (J.EQ.1)) THEN
        D(I) = T(I,J,K) - U3*(T(I,J+1,K) - T(I,J,K)) +
     &        U4*(T(I,J,K-1) - T(I,J,K)) + V1*(TAMB1 - T(I,J,K)) +
     &        V4*(TAMB4 - T(I,J,K)) + W1 + W4 +
     &        X1*(T(I,J,K)**4 - TAMB1**4) +
     &        X4*(T(I,J,K)**4 - TAMB4**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((I.EQ.M) .AND. (J.EQ.1)) THEN
        D(I) = T(I,J,K) + U3*(T(I,J+1,K) - T(I,J,K)) +
     &        U4*(T(I,J,K-1) - T(I,J,K)) + V1*(TAMB1 - T(I,J,K)) +
     &        V4*(TAMB4 - T(I,J,K)) + V5*(TAMB5 - T(I,J,K)) + W1 +
     &        W4 + W5 + X1*(T(I,J,K)**4 - TAMB1**4) +
     &        X4*(T(I,J,K)**4 - TAMB4**4) +
     &        X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((I.EQ.1) .AND. ((J.GT.1) .AND. (J.LT.N))) THEN
        D(I) = T(I,J,K) + U1*(T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K)) +
     &        U4*(T(I,J,K-1) - T(I,J,K)) + V4*(TAMB4 - T(I,J,K)) +
     &        V6*(TAMB6 - T(I,J,K)) + W4 + W6 +
     &        X4*(T(I,J,K)**4 - TAMB4**4) +
     &        X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((I.GT.1) .AND. (I.LT.M)) .AND. ((J.GT.1) .AND.
     & (J.LT.N))) THEN
```

```
      D(I) = T(I,J,K) + U1*(T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K)) +
     &       U4*(T(I,J,K-1) - T(I,J,K)) + V4*(TAMB4 - T(I,J,K)) +
     &       W4 + X4*(T(I,J,K)**4 - TAMB4**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((I.EQ.M) .AND. ((J.GT.1) .AND. (J.LT.N))) THEN
      D(I) = T(I,J,K) + U1*(T(I,J-1,K) + T(I,J+1,K) - 2*T(I,J,K)) +
     &       U4*(T(I,J,K-1) - T(I,J,K)) + V4*(TAMB4 - T(I,J,K)) +
     &       V5*(TAMB5 - T(I,J,K)) + W4 + W5 +
     &       X4*(T(I,J,K)**4 - TAMB4**4) +
     &       X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((I.EQ.1) .AND. (J.EQ.N)) THEN
      D(I) = T(I,J,K) + U3*(T(I,J-1,K) - T(I,J,K)) +
     &       U4*(T(I,J,K-1) - T(I,J,K)) + V2*(TAMB2 - T(I,J,K)) +
     &       V4*(TAMB4 - T(I,J,K)) + V6*(TAMB6 - T(I,J,K)) + W2 +
     &       W4 + W6 + X2*(T(I,J,K)**4 - TAMB2**4) +
     &       X4*(T(I,J,K)**4 - TAMB4**4) +
     &       X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((I.GT.1) .AND. (I.LT.M)) .AND. (J.EQ.N)) THEN
      D(I) = T(I,J,K) + U3*(T(I,J-1,K) - T(I,J,K)) +
     &       U4*(T(I,J,K-1) - T(I,J,K)) + V2*(TAMB2 - T(I,J,K)) +
     &       V4*(TAMB4 - T(I,J,K)) + W2 + W4 +
     &       X2*(T(I,J,K)**4 - TAMB2**4) +
     &       X4*(T(I,J,K)**4 - TAMB4**4) + Y1*QDOT(I,J,K)
*
      ELSE
      D(I) = T(I,J,K) + U3*(T(I,J-1,K) - T(I,J,K)) +
     &       U4*(T(I,J,K-1) - T(I,J,K)) + V2*(TAMB2 - T(I,J,K)) +
     &       V4*(TAMB4 - T(I,J,K)) + V5*(TAMB5 - T(I,J,K)) + W2 +
     &       W4 + W5 + X2*(T(I,J,K)**4 - TAMB2**4) +
     &       X4*(T(I,J,K)**4 - TAMB4**4) +
     &       X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
```

```
      ENDIF
*
      RETURN
      END
*
*
************************************************************************
************************************************************************
*
*     SUBROUTINES YANK1, YANK2, AND YANK3 COMPUTE COEFFICIENT
*     ARRAY D IN THE Y DIRECTION
*
*     SUBROUTINE YANK1
*
      SUBROUTINE YANK1
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT( 30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
```

```
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
```

```
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
      IF((J.EQ.1) .AND. (I.EQ.1)) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K+1) - T(I,J,K)) +
     &         U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K)) +
     &         V1*(TAMB1 - T(I,J,K)) + V3*(TAMB3 - T(I,J,K)) +
     &         V6*(TAMB6 - T(I,J,K)) + W1 + W3 + W6 +
     &         X1*(T(I,J,K)**4 - TAMB1**4) +
     &         X3*(T(I,J,K)**4 - TAMB3**4) +
     &         X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((J.GT.1) .AND. (J.LT.N)) .AND. (I.EQ.1)) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K+1) - T(I,J,K)) +
     &         U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K)) +
     &         V3*(TAMB3 - T(I,J,K)) + V6*(TAMB6 - T(I,J,K)) +
     &         W3 + W6 + X3*(T(I,J,K)**4 - TAMB3**4) +
     &         X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((J.EQ.N) .AND. (I.EQ.1)) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K+1) - T(I,J,K)) +
     &         U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K)) +
     &         V2*(TAMB2 - T(I,J,K)) + V3*(TAMB3 - T(I,J,K)) +
     &         V6*(TAMB6 - T(I,J,K)) + W2 + W3 + W6 +
     &         X2*(T(I,J,K)**4 - TAMB2**4) +
     &         X3*(T(I,J,K)**4 - TAMB3**4) +
     &         X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((J.EQ.1) .AND. ((I.GT.1) .AND. (I.LT.M))) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K+1) - T(I,J,K)) +
```

```
     &        FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K) - 2*TSTAR1(I,J,K))
     &        + V1*(TAMB1 - T(I,J,K)) + V3*(TAMB3 - T(I,J,K)) + W1 +
     &        W3 + X1*(T(I,J,K)**4 - TAMB1**4) +
     &        X3*(T(I,J,K)**4 - TAMB3**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((J.GT.1) .AND. (J.LT.N)) .AND. ((I.GT.1) .AND.
     & (I.LT.M))) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K+1) - T(I,J,K)) +
     &        FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K) - 2*TSTAR1(I,J,K))
     &        + V3*(TAMB3 - T(I,J,K)) + W3 +
     &        X3*(T(I,J,K)**4 - TAMB3**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((J.EQ.N) .AND. ((I.GT.1) .AND. (I.LT.M))) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K+1) - T(I,J,K)) +
     &        FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K) - 2*TSTAR1(I,J,K))
     &        + V2*(TAMB2 - T(I,J,K)) + V3*(TAMB3 - T(I,J,K)) + W2 +
     &        W3 + X2*(T(I,J,K)**4 - TAMB2**4) +
     &        X3*(T(I,J,K)**4 - TAMB3**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((J.EQ.1) .AND. (I.EQ.M)) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K+1) - T(I,J,K)) +
     &        U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K)) +
     &        V1*(TAMB1 - T(I,J,K)) + V3*(TAMB3 - T(I,J,K)) +
     &        V5*(TAMB5 - T(I,J,K)) + W1 + W3 + W5 +
     &        X1*(T(I,J,K)**4 - TAMB1**4) +
     &        X3*(T(I,J,K)**4 - TAMB3**4) +
     &        X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((J.GT.1) .AND. (J.LT.N)) .AND. (I.EQ.M)) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K+1) - T(I,J,K)) +
     &        U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K)) +
     &        V3*(TAMB3 - T(I,J,K)) + V5*(TAMB5 - T(I,J,K)) + W3 +
     &        W5 + X3*(T(I,J,K)**4 - TAMB3**4) +
```

```
     &        X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE
        D(J) = T(I,J,K) + U4*(T(I,J,K+1) - T(I,J,K)) +
     &        U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K)) +
     &        V2*(TAMB2 - T(I,J,K)) + V3*(TAMB3 - T(I,J,K)) +
     &        V5*(TAMB5 - T(I,J,K)) + W2 + W3 + W5 +
     &        X2*(T(I,J,K)**4 - TAMB2**4) +
     &        X3*(T(I,J,K)**4 - TAMB3**4) +
     &        X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ENDIF
*
      RETURN
      END
*
*
*     SUBROUTINE YANK2
*
      SUBROUTINE YANK2
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT( 30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
```

```
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ
```

```
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
      IF((J.EQ.1) .AND. (I.EQ.1)) THEN
        D(J) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
     &       U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K))
     &       + V1*(TAMB1 - T(I,J,K)) + V6*(TAMB6 - T(I,J,K)) + W1 +
     &       W6 + X1*(T(I,J,K)**4 - TAMB1**4) +
     &       X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((J.GT.1) .AND. (J.LT.N)) .AND. (I.EQ.1)) THEN
        D(J) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
     &       U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K))
     &       + V6*(TAMB6 - T(I,J,K)) + W6 +
     &       X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((J.EQ.N) .AND. (I.EQ.1)) THEN
        D(J) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
     &       U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K))
     &       + V2*(TAMB2 - T(I,J,K)) + V6*(TAMB6 - T(I,J,K)) + W2 +
     &       W6 + X2*(T(I,J,K)**4 - TAMB2**4) +
     &       X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((J.EQ.1) .AND. ((I.GT.1) .AND. (I.LT.M))) THEN
        D(J) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
```

```
&       FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K) - 2*TSTAR1(I,J,K))
&       + V1*(TAMB1 - T(I,J,K)) + W1 +
&       X1*(T(I,J,K)**4 - TAMB1**4) + Y1*QDOT(I,J,K)
*
 ELSE IF(((J.GT.1) .AND. (J.LT.N)) .AND. ((I.GT.1) .AND.
& (I.LT.M))) THEN
  D(J) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
&       FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K) - 2*TSTAR1(I,J,K))
&       + Y1*QDOT(I,J,K)
*
 ELSE IF((J.EQ.N) .AND. ((I.GT.1) .AND. (I.LT.M))) THEN
  D(J) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
&       FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K) - 2*TSTAR1(I,J,K))
&       + V2*(TAMB2 - T(I,J,K)) + W2 +
&       X2*(T(I,J,K)**4 - TAMB2**4) + Y1*QDOT(I,J,K)
*
 ELSE IF((J.EQ.1) .AND. (I.EQ.M)) THEN
  D(J) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
&       U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K))
&       + V1*(TAMB1 - T(I,J,K)) + V5*(TAMB5 - T(I,J,K)) + W1 +
&       W5 + X1*(T(I,J,K)**4 - TAMB1**4) +
&       X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
 ELSE IF(((J.GT.1) .AND. (J.LT.N)) .AND. (I.EQ.M)) THEN
  D(J) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
&       U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K))
&       + V5*(TAMB5 - T(I,J,K)) + W5 +
&       X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
 ELSE
  D(J) = T(I,J,K) + U2*(T(I,J,K-1) + T(I,J,K+1) - 2*T(I,J,K)) +
&       U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K))
&       + V2*(TAMB2 - T(I,J,K)) + V5*(TAMB5 - T(I,J,K)) + W2 +
&       W5 + X2*(T(I,J,K)**4 - TAMB2**4) +
```

```
     &        X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ENDIF
*
      RETURN
      END
*
*
*     SUBROUTINE YANK3
*
      SUBROUTINE YANK3
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT( 30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
```

```
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
```

```
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
      IF((J.EQ.1) .AND. (I.EQ.1)) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K-1) - T(I,J,K)) +
     &        U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K)) +
     &        V1*(TAMB1 - T(I,J,K)) + V4*(TAMB4 - T(I,J,K)) +
     &        V6*(TAMB6 - T(I,J,K)) + W1 + W4 + W6 +
     &        X1*(T(I,J,K)**4 - TAMB1**4) +
     &        X4*(T(I,J,K)**4 - TAMB4**4) +
     &        X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((J.GT.1) .AND. (J.LT.N)) .AND. (I.EQ.1)) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K-1) - T(I,J,K)) +
     &        U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K)) +
     &        V4*(TAMB4 - T(I,J,K)) + V6*(TAMB6 - T(I,J,K)) + W4 +
     &        W6 + X4*(T(I,J,K)**4 - TAMB4**4) +
     &        X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((J.EQ.N) .AND. (I.EQ.1)) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K-1) - T(I,J,K)) +
     &        U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K)) +
     &        V2*(TAMB2 - T(I,J,K)) + V4*(TAMB4 - T(I,J,K)) +
     &        V6*(TAMB6 - T(I,J,K)) + W2 + W4 + W6 +
     &        X2*(T(I,J,K)**4 - TAMB2**4) +
     &        X4*(T(I,J,K)**4 - TAMB4**4) +
     &        X6*(T(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((J.EQ.1) .AND. ((I.GT.1) .AND. (I.LT.M))) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K-1) - T(I,J,K)) +
     &        FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K) - 2*TSTAR1(I,J,K))
     &        + V1*(TAMB1 - T(I,J,K)) + V4*(TAMB4 - T(I,J,K)) + W1 +
     &        W4 + X1*(T(I,J,K)**4 - TAMB1**4) +
```

```
     &        X4*(T(I,J,K)**4 - TAMB4**4) - Y1*QDOT(I,J,K)
*
      ELSE IF(((J.GT.1) .AND. (J.LT.N)) .AND. ((I.GT.1) .AND.
     & (I.LT.M))) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K-1) - T(I,J,K)) +
     &        FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K) - 2*TSTAR1(I,J,K))
     &        + V4*(TAMB4 - T(I,J,K)) + W4 +
     &        X4*(T(I,J,K)**4 - TAMB4**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((J.EQ.N) .AND. ((I.GT.1) .AND. (I.LT.M))) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K-1) - T(I,J,K)) +
     &        FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K) - 2*TSTAR1(I,J,K))
     &        + V2*(TAMB2 - T(I,J,K)) + V4*(TAMB4 - T(I,J,K)) + W2 +
     &        W4 + X2*(T(I,J,K)**4 - TAMB2**4) +
     &        X4*(T(I,J,K)**4 - TAMB4**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((J.EQ.1) .AND. (I.EQ.M)) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K-1) - T(I,J,K)) +
     &        U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K)) +
     &        V1*(TAMB1 - T(I,J,K)) + V4*(TAMB4 - T(I,J,K)) +
     &        V5*(TAMB5 - T(I,J,K)) + W1 + W4 + W5 +
     &        X1*(T(I,J,K)**4 - TAMB1**4) +
     &        X4*(T(I,J,K)**4 - TAMB4**4) +
     &        X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((J.GT.1) .AND. (J.LT.N)) .AND. (I.EQ.M)) THEN
        D(J) = T(I,J,K) + U4*(T(I,J,K-1) - T(I,J,K)) +
     &        U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K)) +
     &        V4*(TAMB4 - T(I,J,K)) + V5*(TAMB5 - T(I,J,K)) + W4 +
     &        W5 + X4*(T(I,J,K)**4 - TAMB4**4) +
     &        X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE
        D(J) = T(I,J,K) + U4*(T(I,J,K-1) - T(I,J,K)) +
```

```
     &        U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K)) +
     &        V2*(TAMB2 - T(I,J,K)) + V4*(TAMB4 - T(I,J,K)) +
     &        V5*(TAMB5 - T(I,J,K)) + W2 + W4 + W5 +
     &        X2*(T(I,J,K)**4 - TAMB2**4) +
     &        X4*(T(I,J,K)**4 - TAMB4**4) +
     &        X5*(T(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ENDIF
*
      RETURN
      END
*
*
*************************************************************************
*************************************************************************
*
*     SUBROUTINES ZULU1, ZULU2, AND ZULU3 COMPUTE COEFFICIENT
*     ARRAY D IN THE Z DIRECTION
*
*     SUBROUTINE ZULU1
*
      SUBROUTINE ZULU1
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT( 30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
```

```
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
```

```
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIME0, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
      IF((K.EQ.1) .AND. (I.EQ.1)) THEN
        D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J+1,K) - TSTAR2(I,J,K)) +
     &       U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K)) +
     &       V1*(TAMB1 - TSTAR2(I,J,K)) + V3*(TAMB3 - TSTAR2(I,J,K))
     &       + V6*(TAMB6 - TSTAR2(I,J,K)) + W1 + W3 + W6 +
     &       X1*(TSTAR2(I,J,K)**4 - TAMB1**4) +
     &       X3*(TSTAR2(I,J,K)**4 - TAMB3**4) +
     &       X6*(TSTAR2(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((K.GT.1) .AND. (K.LT.P)) .AND. (I.EQ.1)) THEN
        D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J+1,K) - TSTAR2(I,J,K)) +
     &       U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K)) +
     &       V1*(TAMB1 - TSTAR2(I,J,K)) + V6*(TAMB6 - TSTAR2(I,J,K))
     &       + W1 + W6 + X1*(TSTAR2(I,J,K)**4 - TAMB1**4) +
     &       X6*(TSTAR2(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((K.EQ.P) .AND. (I.EQ.1)) THEN
        D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J+1,K) - TSTAR2(I,J,K)) +
     &       U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K)) +
     &       V1*(TAMB1 - TSTAR2(I,J,K)) + V4*(TAMB4 - TSTAR2(I,J,K))
```

```
&       + V6*(TAMB6 - TSTAR2(I,J,K)) + W1 + W4 + W6 +
&       X1*(TSTAR2(I,J,K)**4 - TAMB1**4) +
&       X4*(TSTAR2(I,J,K)**4 - TAMB4**4) +
&       X6*(TSTAR2(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
 ELSE IF((K.EQ.1) .AND. ((I.GT.1) .AND. (I.LT.M))) THEN
   D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J+1,K) - TSTAR2(I,J,K)) +
&       FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K) - 2*TSTAR1(I,J,K))
&       + V1*(TAMB1 - TSTAR2(I,J,K)) +
&       V3*(TAMB3 - TSTAR2(I,J,K)) + W1 + W3 +
&       X1*(TSTAR2(I,J,K)**4 - TAMB1**4, +
&       X3*(TSTAR2(I,J,K)**4 - TAMB3**4) + Y1*QDOT(I,J,K)
*
 ELSE IF(((K.GT.1) .AND. (K.LT.P)) .AND. ((I.GT.1) .AND.
& (I.LT.M))) THEN
   D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J+1,K) - TSTAR2(I,J,K)) +
&       FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K) - 2*TSTAR1(I,J,K))
&       + V1*(TAMB1 - TSTAR2(I,J,K)) + W1 +
&       X1*(TSTAR2(I,J,K)**4 - TAMB1**4) + Y1*QDOT(I,J,K)
*
 ELSE IF((K.EQ.P) .AND. ((I.GT.1) .AND. (I.LT.M))) THEN
   D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J+1,K) - TSTAR2(I,J,K)) +
&       FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K) - 2*TSTAR1(I,J,K))
&       + V1*(TAMB1 - TSTAR2(I,J,K)) +
&       V4*(TAMB4 - TSTAR2(I,J,K)) + W1 + W4 +
&       X1*(TSTAR2(I,J,K)**4 - TAMB1**4) +
&       X4*(TSTAR2(I,J,K)**4 - TAMB4**4) + Y1*QDOT(I,J,K)
*
 ELSE IF((K.EQ.1) .AND. (I.EQ.M)) THEN
   D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J+1,K) - TSTAR2(I,J,K)) +
&       U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K)) +
&       V1*(TAMB1 - TSTAR2(I,J,K)) + V3*(TAMB3 - TSTAR2(I,J,K))
&       + V5*(TAMB5 - TSTAR2(I,J,K)) + W1 + W3 + W5 +
&       X1*(TSTAR2(I,J,K)**4 - TAMB1**4) +
```

```
&       X3*(TSTAR2(I,J,K)**4 - TAMB3**4) +
&       X5*(TSTAR2(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((K.GT.1) .AND. (K.LT.P)) .AND. (I.EQ.M)) THEN
        D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J+1,K) - TSTAR2(I,J,K)) +
&       U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K)) +
&       V1*(TAMB1 - TSTAR2(I,J,K)) + V5*(TAMB5 - TSTAR2(I,J,K))
&       + W1 + W5 + X1*(TSTAR2(I,J,K)**4 - TAMB1**4) +
&       X5*(TSTAR2(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE
        D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J+1,K) - TSTAR2(I,J,K)) +
&       U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K)) +
&       V1*(TAMB1 - TSTAR2(I,J,K)) + V4*(TAMB4 - TSTAR2(I,J,K))
&       + V5*(TAMB5 - TSTAR2(I,J,K)) + W1 + W4 + W5 +
&       X1*(TSTAR2(I,J,K)**4 - TAMB1**4) +
&       X4*(TSTAR2(I,J,K)**4 - TAMB4**4) +
&       X5*(TSTAR2(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ENDIF
*
      RETURN
      END
*
*
*     SUBROUTINE ZULU2
*
      SUBROUTINE ZULU2
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT( 30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
```

```
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
```

```
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
      IF((K.EQ.1) .AND. (I.EQ.1)) THEN
        D(K) = TSTAR2(I,J,K) + U1*(TSTAR2(I,J-1,K) + TSTAR2(I,J+1,K) -
     &       2*TSTAR2(I,J,K)) + U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K))
     &       + V3*(TAMB3 - TSTAR2(I,J,K)) +
     &       V6*(TAMB6 - TSTAR2(I,J,K)) + W3 + W6 +
     &       X3*(TSTAR2(I,J,K)**4 - TAMB3**4) +
     &       X6*(TSTAR2(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((K.GT.1) .AND. (K.LT.P)) .AND. (I.EQ.1)) THEN
        D(K) = TSTAR2(I,J,K) + U1*(TSTAR2(I,J-1,K) + TSTAR2(I,J+1,K) -
     &       2*TSTAR2(I,J,K)) + U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K))
     &       + V6*(TAMB6 - TSTAR2(I,J,K)) + W6 +
     &       X6*(TSTAR2(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
```

```
      ELSE IF((K.EQ.P) .AND. (I.EQ.1)) THEN
        D(K) = TSTAR2(I,J,K) + U1*(TSTAR2(I,J-1,K) + TSTAR2(I,J+1,K) -
     &       2*TSTAR2(I,J,K)) + U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K))
     &       + V4*(TAMB4 - TSTAR2(I,J,K)) +
     &       V6*(TAMB6 - TSTAR2(I,J,K)) + W4 + W6 +
     &       X4*(TSTAR2(I,J,K)**4 - TAMB4**4) +
     &       X6*(TSTAR2(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((K.EQ.1) .AND. ((I.GT.1) .AND. (I.LT.M))) THEN
        D(K) = TSTAR2(I,J,K) + U1*(TSTAR2(I,J-1,K) + TSTAR2(I,J+1,K) -
     &       2*TSTAR2(I,J,K)) + FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K)
     &       - 2*TSTAR1(I,J,K)) + V3*(TAMB3 - TSTAR2(I,J,K)) +
     &       W3 + X3*(TSTAR2(I,J,K)**4 - TAMB3**4) +
     &       Y1*QDOT(I,J,K)
*
      ELSE IF(((K.GT.1) .AND. (K.LT.P)) .AND. ((I.GT.1) .AND.
     & (I.LT.M))) THEN
        D(K) = TSTAR2(I,J,K) + U1*(TSTAR2(I,J-1,K) + TSTAR2(I,J+1,K) -
     &       2*TSTAR2(I,J,K)) + FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K)
     &       - 2*TSTAR1(I,J,K)) + Y1*QDOT(I,J,K)
*
      ELSE IF((K.EQ.P) .AND. ((I.GT.1) .AND. (I.LT.M))) THEN
        D(K) = TSTAR2(I,J,K) + U1*(TSTAR2(I,J-1,K) + TSTAR2(I,J+1,K) -
     &       2*TSTAR2(I,J,K)) + FO*(TSTAR1(I-1,J,K) + TSTAR1(I+1,J,K)
     &       - 2*TSTAR1(I,J,K)) + V4*(TAMB4 - TSTAR2(I,J,K)) +
     &       W4 + X4*(TSTAR2(I,J,K)**4 - TAMB4**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((K.EQ.1) .AND. (I.EQ.M)) THEN
        D(K) = TSTAR2(I,J,K) + U1*(TSTAR2(I,J-1,K) + TSTAR2(I,J+1,K) -
     &       2*TSTAR2(I,J,K)) + U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K))
     &       + V3*(TAMB3 - TSTAR2(I,J,K)) +
     &       V5*(TAMB5 - TSTAR2(I,J,K)) + W3 + W5 +
     &       X3*(TSTAR2(I,J,K)**4 - TAMB3**4) +
```

```
     &        X5*(TSTAR2(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((K.GT.1) .AND. (K.LT.P)) .AND. (I.EQ.M)) THEN
        D(K) = TSTAR2(I,J,K) + U1*(TSTAR2(I,J-1,K) + TSTAR2(I,J+1,K) -
     &        2*TSTAR2(I,J,K)) + U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K))
     &        + V5*(TAMB5 - TSTAR2(I,J,K)) + W5 +
     &        X5*(TSTAR2(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE
        D(K) = TSTAR2(I,J,K) + U1*(TSTAR2(I,J-1,K) + TSTAR2(I,J+1,K) -
     &        2*TSTAR2(I,J,K)) + U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K))
     &        + V4*(TAMB4 - TSTAR2(I,J,K)) +
     &        V5*(TAMB5 - TSTAR2(I,J,K)) + W4 + W5 +
     &        X4*(TSTAR2(I,J,K)**4 - TAMB4**4) +
     &        X5*(TSTAR2(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ENDIF
*
      RETURN
      END
*
*
*     SUBROUTINE ZULU3
*
      SUBROUTINE ZULU3
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT( 30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
```

```
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
```

```
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
      IF((K.EQ.1) .AND. (I.EQ.1)) THEN
        D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J-1,K) - TSTAR2(I,J,K)) +
     &        U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K)) +
     &        V2*(TAMB2 - TSTAR2(I,J,K)) + V3*(TAMB3 - TSTAR2(I,J,K))
     &        + V6*(TAMB6 - TSTAR2(I,J,K)) + W2 + W3 + W6 +
     &        X2*(TSTAR2(I,J,K)**4 - TAMB2**4) +
     &        X3*(TSTAR2(I,J,K)**4 - TAMB3**4) +
     &        X6*(TSTAR2(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((K.GT.1) .AND. (K.LT.P)) .AND. (I.EQ.1)) THEN
        D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J-1,K) - TSTAR2(I,J,K)) +
     &        U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K)) +
     &        V2*(TAMB2 - TSTAR2(I,J,K)) + V6*(TAMB6 - TSTAR2(I,J,K))
     &        + W2 + W6 + X2*(TSTAR2(I,J,K)**4 - TAMB2**4) +
     &        X6*(TSTAR2(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((K.EQ.P) .AND. (I.EQ.1)) THEN
```

```
      D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J-1,K) - TSTAR2(I,J,K)) +
     &       U5*(TSTAR1(I+1,J,K) - TSTAR1(I,J,K)) +
     &       V2*(TAMB2 - TSTAR2(I,J,K)) + V4*(TAMB4 - TSTAR2(I,J,K))
     &       + V6*(TAMB6 - TSTAR2(I,J,K)) + W2 + W4 + W6 +
     &       X2*(TSTAR2(I,J,K)**4 - TAMB2**4) +
     &       X4*(TSTAR2(I,J,K)**4 - TAMB4**4) +
     &       X6*(TSTAR2(I,J,K)**4 - TAMB6**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((K.EQ.1) .AND. ((I.GT.1) .AND. (I.LT.M))) THEN
      D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J-1,K) - TSTAR2(I,J,K)) +
     &       FO*(TSTAR1(I+1,J,K) + TSTAR1(I-1,J,K) - 2*TSTAR1(I,J,K))
     &       + V2*(TAMB2 - TSTAR2(I,J,K)) +
     &       V3*(TAMB3 - TSTAR2(I,J,K)) + W2 + W3 +
     &       X2*(TSTAR2(I,J,K)**4 - TAMB2**4) +
     &       X3*(TSTAR2(I,J,K)**4 - TAMB3**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((K.GT.1) .AND. (K.LT.P)) .AND. ((I.GT.1) .AND.
     & (I.LT.M))) THEN
      D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J-1,K) - TSTAR2(I,J,K)) +
     &       FO*(TSTAR1(I+1,J,K) + TSTAR1(I-1,J,K) - 2*TSTAR1(I,J,K))
     &       + V2*(TAMB2 - TSTAR2(I,J,K)) + W2 +
     &       X2*(TSTAR2(I,J,K)**4 - TAMB2**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((K.EQ.P) .AND. ((I.GT.1) .AND. (I.LT.M))) THEN
      D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J-1,K) - TSTAR2(I,J,K)) +
     &       FO*(TSTAR1(I+1,J,K) + TSTAR1(I-1,J,K) - 2*TSTAR1(I,J,K))
     &       + V2*(TAMB2 - TSTAR2(I,J,K)) +
     &       V4*(TAMB4 - TSTAR2(I,J,K)) + W2 + W4 +
     &       X2*(TSTAR2(I,J,K)**4 - TAMB2**4) +
     &       X4*(TSTAR2(I,J,K)**4 - TAMB4**4) + Y1*QDOT(I,J,K)
*
      ELSE IF((K.EQ.1) .AND. (I.EQ.M)) THEN
      D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J-1,K) - TSTAR2(I,J,K)) +
     &       U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K)) +
```

```
     &        V2*(TAMB2 - TSTAR2(I,J,K)) + V3*(TAMB3 - TSTAR2(I,J,K))
     &        + V5*(TAMB5 - TSTAR2(I,J,K)) + W2 + W3 + W5 +
     &        X2*(TSTAR2(I,J,K)**4 - TAMB2**4) +
     &        X3*(TSTAR2(I,J,K)**4 - TAMB3**4) +
     &        X5*(TSTAR2(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE IF(((K.GT.1) .AND. (K.LT.P)) .AND. (I.EQ.M)) THEN
        D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J-1,K) - TSTAR2(I,J,K)) +
     &        U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K)) +
     &        V2*(TAMB2 - TSTAR2(I,J,K)) + V5*(TAMB5 - TSTAR2(I,J,K))
     &        + W2 + W5 + X2*(TSTAR2(I,J,K)**4 - TAMB2**4) +
     &        X5*(TSTAR2(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ELSE
        D(K) = TSTAR2(I,J,K) + U3*(TSTAR2(I,J-1,K) - TSTAR2(I,J,K)) +
     &        U5*(TSTAR1(I-1,J,K) - TSTAR1(I,J,K)) +
     &        V2*(TAMB2 - TSTAR2(I,J,K)) + V4*(TAMB4 - TSTAR2(I,J,K))
     &        + V5*(TAMB5 - TSTAR2(I,J,K)) + W2 + W4 + W5 +
     &        X2*(TSTAR2(I,J,K)**4 - TAMB2**4) +
     &        X4*(TSTAR2(I,J,K)**4 - TAMB4**4) +
     &        X5*(TSTAR2(I,J,K)**4 - TAMB5**4) + Y1*QDOT(I,J,K)
*
      ENDIF
*
      RETURN
      END
*
*
************************************************************************
************************************************************************
*
*     SUBROUTINE TRIDAG IS A SUBROUTINE FOR SOLVING A SYSTEM
*     OF LINEAR SIMULTANEOUS EQUATIONS HAVING A TRIDIAGONAL
*     COEFFICIENT MATRIX. THE EQUATIONS ARE NUMBERED IF THROUGH
```

```
*     L, AND THEIR SUBDIAGONAL, DIAGONAL, AND SUPER DIAGONAL
*     COEFFICIENTS ARE STORED IN THE ARRAYS A, B, AND C. THE
*     COMPUTED SOLUTION VECTOR TEMP(IF)...TEMP(L) IS STORED
*     IN THE ARRAY TEMP.
*
      SUBROUTINE TRIDAG
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT( 30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
```

```
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
*     COMPUTE INTERMEDIATE ARRAYS BETA AND GAMMA
*
```

```
      BETA(IF) = B(IF)
      GAMMA(IF) = D(IF) BETA(IF)
      IFP1 = IF + 1
      DO 400 Q=IFP1, L
      BETA(Q) = B(Q) - A(Q)*C(Q-1)/BETA(Q-1)
      GAMMA(Q) = ( D(Q) - A(Q)*GAMMA(Q-1) )/BETA(Q)
400   CONTINUE
*
*
*     COMPUTE FINAL SOLUTION VECTOR TEMP
*
      TEMP(L) = GAMMA(L)
      LAST = L - IF
      DO 410 G=1, LAST
      Q = L - G
      TEMP(Q) = GAMMA(Q) - C(Q)*TEMP(Q+1) BETA(Q)
410   CONTINUE
      RETURN
      END
*
*
*************************************************************************
*************************************************************************
*
*     THIS SUBROUTINE WILL BE USED FOR PRINTING THE NODE
*     TEMPERATURES
*
      SUBROUTINE OUTPUT
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT( 30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
```

```
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
```

```
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ, ANS1, ANS2, ANS3
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      COMMON P1, P2, P3, P4, P5, P6
*
*     THE USER CAN PRINT OUT THE TEMPERATURES AT SELECTED
*     NODES. IF OTHER NODES ARE DESIRED, THEN THE USER MAY
*     CHANGE THE 'WRITE' AND 'PRINT' STATEMENTS TO THE
*     DESIRED NODES.
*
      IF(TIME .NE. DELT) GO TO 500
*
      PRINT*
      WRITE(1,*) '  TIME(SEC)     ( 6, 1,11)    ( 6, 6, )    ( 1, 6, 6)'
      WRITE(1,*) '  ---------     ----------    ---------    ----------'
*
  500 WRITE (1,501) TIME, T( 6, 1, 11), T( 6, 6, 6), T( 1, 6, 6)
  501 FORMAT(1X,F8.0,4X,F8.3,4X,F8.3,4X,F8.3)
*
```

```
*     THE USER CAN ALSO HAVE ALL NODE TEMPERATURES PRINTED
*     OUT. IF A PRINTOUT OF ALL PLANES AND OR ALL NODES IS
*     NOT REQUIRED, THEN THE DO LOOPS MAY BE CHANGED TO
*     ACCOMODATE THE USER'S NEEDS.
*
      WRITE(6,*) 'DO YOU WANT K-PLANE PRINTOUT? (FOR YES, ENTER'
      WRITE(6,*) 'ANS1 AS 1; FOR NO ENTER ANS1 AS 0.)'
      READ(6,*) ANS1
      IF(ANS1 .NE. 1) GO TO 699
*
      WRITE(2,600) 'TIME(SEC) = ',TIME
600   FORMAT(A12,F12.3)
*
      DO 605 K = 1,P
*
      WRITE(2,*) 'TEMPERATURE DISTRIBUTION ON PLANE :'
      WRITE(2,610) 'K = ',K
      WRITE(2,*) '*************************************'
610   FORMAT(1X,A3,I2)
      WRITE(2,620) ((T(I,J,K), J = 1,N), I = 1,M)
620   FORMAT(1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,
     & 1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1)
*
605   CONTINUE
*
699   WRITE(6,*) 'DO YOU WANT J-PLANE PRINTOUT? (FOR YES, ENTER'
      WRITE(6,*) 'ANS2 AS 1; FOR NO ENTER ANS2 AS 0.)'
      READ(6,*) ANS2
      IF(ANS2 .NE. 1) GO TO 799
*
      WRITE(3,700) 'TIME(SEC) = ',TIME
700   FORMAT(A12,F12.3)
*
```

```
      DO 705 J = 1,N
*
      WRITE(3,*) 'TEMPERATURE DISTRIBUTION ON PLANE :'
      WRITE(3,710) 'J = ',J
      WRITE(3,*) '************************************'
 710  FORMAT(1X,A3,I2)
      WRITE(3,720) ((T(I,J,K), I = 1,M), K = P,1,-1)
 720  FORMAT(1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,
     & 1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1)
*
 705  CONTINUE
*
 799  WRITE(6,*) 'DO YOU WANT I-PLANE PRINTOUT? (FOR YES, ENTER'
      WRITE(6,*) 'ANS3 AS 1; FOR NO ENTER ANS3 AS 0.)'
      READ(6,*) ANS3
      IF(ANS3 .NE. 1) GO TO 899
*
      WRITE(4,800) 'TIME(SEC) = ',TIME
 800  FORMAT(A12,F12.3)
*
      DO 805 I = 1,M
*
      WRITE(4,*) 'TEMPERATURE DISTRIBUTION ON PLANE :'
      WRITE(4,810) 'I = ',I
      WRITE(4,*) '************************************'
 810  FORMAT(1X,A3,I2)
      WRITE(4,820) ((T(I,J,K), J = 1,N), K = P,1,-1)
 820  FORMAT(1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1,
     & 1X,F6.1,1X,F6.1,1X,F6.1,1X,F6.1)
*
 805  CONTINUE
*
*     REINITIALIZING THE COUNTER TO ZERO
*
```

```
899  COUNT = 0
*
      RETURN
      END
*
*************************************************************************
*************************************************************************
*
      SUBROUTINE VALID
      DIMENSION A( 30), B( 30), C( 30), D( 30), TEMP( 30)
      DIMENSION BETA( 30), GAMMA( 30), QDOT(30, 30, 30)
      DIMENSION T( 30, 30, 30), TSTAR1( 30, 30, 30)
      DIMENSION TSTAR2( 30, 30, 30)
*
      REAL TINIT, TSTAR1, TSTAR2, T, TEMP
      REAL CP, K1, RHO, R, R1, R2, TIME, DELT
      REAL LX, LY, LZ, DELX, DELY, DELZ
      REAL A, B, C, D, BETA, GAMMA
      REAL FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      REAL TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      REAL H1, H2, H3, H4, H5, H6
      REAL BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      REAL SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      REAL U1, U2, U3, U4, U5
      REAL V1, V2, V3, V4, V5, V6
      REAL W1, W2, W3, W4, W5, W6
      REAL X1, X2, X3, X4, X5, X6
      REAL EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      REAL QDOT, GEN, Y1
      REAL CTIME, TIMEO, TIME1
      REAL SOLAR, EARTH, ALBCO, RE, DIST, ALT, FE, FA, MU
      REAL SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      REAL EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      REAL ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
```

```
      REAL ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      REAL SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      REAL SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
      REAL P1, P2, P3, P4, P5, P6
*
      INTEGER I, J, K, M, N, P, IF, L, IFP1, LAST, G, Q
      INTEGER COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
*
      COMMON TINIT, TSTAR1, TSTAR2, T, TEMP
      COMMON CP, K1, RHO, R, R1, R2, TIME, DELT
      COMMON LX, LY, LZ, DELX, DELY, DELZ
      COMMON I, J, K, M, N, P, IF, L, IFP1, LAST
      COMMON A, B, C, D, BETA, GAMMA, G, Q
      COMMON FLUX1, FLUX2, FLUX3, FLUX4, FLUX5, FLUX6
      COMMON TAMB1, TAMB2, TAMB3, TAMB4, TAMB5, TAMB6
      COMMON H1, H2, H3, H4, H5, H6
      COMMON BIO1, BIO2, BIO3, BIO4, BIO5, BIO6
      COMMON SIG1, SIG2, SIG3, SIG4, SIG5, SIG6
      COMMON U1, U2, U3, U4, U5
      COMMON V1, V2, V3, V4, V5, V6
      COMMON W1, W2, W3, W4, W5, W6
      COMMON X1, X2, X3, X4, X5, X6
      COMMON COUNT, FREQ, ANS1, ANS2, ANS3, VAL, NOUT
      COMMON EPS1, EPS2, EPS3, EPS4, EPS5, EPS6
      COMMON QDOT, GEN, Y1
      COMMON TIMEO, TIME1
      COMMON SOLAR, EARTH, RE, DIST, ALT, FE, FA, MU
      COMMON SOLAR1, SOLAR2, SOLAR3, SOLAR4, SOLAR5, SOLAR6
      COMMON EARTH1, EARTH2, EARTH3, EARTH4, EARTH5, EARTH6
      COMMON ALBED1, ALBED2, ALBED3, ALBED4, ALBED5, ALBED6
      COMMON ABS1, ABS2, ABS3, ABS4, ABS5, ABS6
      COMMON SAC1, SAC2, SAC3, SAC4, SAC5, SAC6
      COMMON SUN, ECLIPS, Q1, Q2, Q3, PERIOD, PI
```

```
      COMMON P1, P2, P3, P4, P5, P6
*
*     VARIOUS NODES WERE SELECTED TO SAMPLE TEMPERATURE
*     DIFFERENCES. THE USER CAN CHANGE THE SELECTED NODES
*     IF DESIRED. IF THE USER CHANGES THE SELECTED NODES,
*     THE NODES SELECTED IN PROGRAM EXPLICIT AND DISTANCES
*     CHOSEN IN PROGRAM VALID SHOULD BE ADJUSTED ACCORDINGLY.
*
      TDIFB1 = T( 1, 6, 6) - TINIT
      TDIFB2 = T( 6, 6,11) - TINIT
      TDIFB3 = T( 6, 1, 6) - TINIT
*
      IF(TIME .NE. DELT) GO TO 920
*
      PRINT*
      WRITE(5,*) 'TIME(SEC)   TDIFB1    TDIFB2    TDIFB3'
      WRITE(5,*) '---------   ------    ------    ------'
*
920   WRITE (5,930) TIME, TDIFB1, TDIFB2, TDIFB3
930   FORMAT(1X,F8.0,3X,F7.3,3X,F7.3,3X,F7.3)
*
*     REINITIALIZING THE COUNTER TO ZERO
*
      COUNT = 0
*
      RETURN
      END
*
*************************************************************************
*************************************************************************
```

APPENDIX E

PROGRAM VALID

```
      PROGRAM VALID
************************************************************************
************************************************************************
*
*     VALIDATION PROGRAM
*
*     THE PURPOSE OF THIS PROGRAM IS TO VALIDATE THE
*     TEMPERATURES OBTAINED BY THE EXPLICIT AND BRIAN
*     PROGRAMS. CONDITIONS USED IN THE VALIDATION PROCESS
*     ARE A CONSTANT SURFACE HEAT FLUX ON THE LEFT FACE
*     AND ADIABATIC CONDITIONS ON ALL OTHER SURFACES.
*
*
*     THIS PROGRAM PROVIDES THE CLOSED-FORM SOLUTION OF
*     TRANSIENT HEAT CONDUCTION IN A SEMI-INFINITE SOLID.
*     THE CLOSED-FORM SOLUTION FOR CONSTANT SURFACE HEAT
*     FLUX IS FROM "INTRODUCTION TO HEAT TRANSFER" BY
*     INCROPERA AND DEWITT. THE RATIONAL APPROXIMATION FOR
*     THE ERROR FUNCTION IN THE CLOSED-FORM SOLUTION IS FROM
*     "HANDBOOK OF MATHEMATICAL FUNCTIONS" EDITED BY
*     ABRAMOWITZ AND STEGUN.
*
************************************************************************
************************************************************************
*
      REAL A1, A2, A3, A4, A5, B, C, D, P, V1, V, W1, W, PI
      REAL TDIFV, FLUX, ALFA, K, RHO, CP, TIME, DELT, Y, DELY, ERRFC
      INTEGER COUNT, FREQ
*
```

```
*     THE USER WILL READ IN PARAMETERS.
*
      WRITE(*,*) 'ENTER FLUX.'
      READ(*,*) FLUX
      WRITE(*,*) 'ENTER K, THE THERMAL CONDUCTIVITY.'
      READ(*,*) K
      WRITE(*,*) 'ENTER RHO, THE MATERIAL DENSITY.'
      READ(*,*) RHO
      WRITE(*,*) 'ENTER CP, THE SPECIFIC HEAT.'
      READ(*,*) CP
      WRITE(*,*) 'ENTER Y, THE DISTANCE IN THE Y DIRECTION.'
      READ(*,*) Y
      WRITE(*,*) 'ENTER DELY, THE INCREMENT IN THE Y DIRECTION.'
      READ(*,*) DELY
      WRITE(*,*) 'ENTER FREQ, THE NUMBER OF TIME STEPS BETWEEN'
      WRITE(*,*) 'SUCCESSIVE PRINTINGS OF TIME AND TDIFV.'
      READ(*,*) FREQ
*
*     CONSTANTS USED THROUGHOUT THE PROGRAM
*
*     NOTE: VALUES FOR A1, A2, A3, A4, A5, AND P ARE FROM
*           "HANDBOOK OF MATHEMATICAL FUNCTIONS".
*
      A1 = 0.254829592
      A2 = -0.284496736
      A3 = 1.421413741
      A4 = -1.453152027
      A5 = 1.061405429
      P = 0.3275911
      PI = 3.141592654
      TIME = 0.0
      COUNT = 0
*
*     CALCULATION OF CONSTANTS USED THROUGHOUT THE PROGRAM
```

```
*
*   NOTE: EQUATIONS FOR ALFA, B, C, D, AND DELT ARE FROM
*         "INTRODUCTION TO HEAT TRANSFER".  EQUATIONS FOR
*         V1 AND W1 ARE FROM "HANDBOOK OF MATHEMATICAL
*         FUNCTIONS".
*
    ALFA = K/(RHO*CP)
    B = 2*FLUX/K*SQRT(ALFA/PI)
    C = FLUX*Y/K
    D = Y**2/(4*ALFA)
    DELT = 0.125*(DELY**2)/ALFA
    V1 = P*Y/(2*SQRT(ALFA))
    W1 = Y/(2*SQRT(ALFA))
*
*   PRINT STATEMENTS FOR VARIOUS PARAMETERS USED
*   IN PROGRAM
*
    PRINT*
    PRINT*
    WRITE(*,*) '*****************************************************'
    WRITE(*,*) 'FLUX= ',FLUX
    WRITE(*,*) 'K= ',K
    WRITE(*,*) 'RHO= ',RHO
    WRITE(*,*) 'CP= ',CP
    WRITE(*,*) 'Y= ',Y
    WRITE(*,*) 'DELY= ',DELY
    WRITE(*,*) 'DELT= ',DELT
    WRITE(*,*) '*****************************************************'
    PRINT*
    PRINT*
*
*   CALCULATION OF TDIFV
*
*   NOTE: EQUATIONS FOR V, W, AND ERRFC (COMPLEMENTARY
```

```
*       ERROR FUNCTION) ARE FROM "HANDBOOK OF
*       MATHEMATICAL FUNCTIONS". THE EQUATION FOR TDIFV
*       IS FROM "INTRODUCTION TO HEAT TRANSFER".
*
 10 COUNT = COUNT + 1
    TIME = TIME + DELT
    V = 1 (1 + V1/SQRT(TIME))
    W = W1/SQRT(TIME)
    ERRFC = ( (A1*V) + (A2*V**2) + (A3*V**3) + (A4*V**4) +
   &        (A5*V**5) ) * EXP(-W**2)
    TDIFV = B*SQRT(TIME)*EXP(-D/TIME) - C*ERRFC
*
*   PRINTING TIME AND TDIFF
*
    IF(TIME .NE. DELT) GO TO 30
    WRITE(*,*) 'TIME(SEC)   TDIFV'
    WRITE(*,*) '---------   -----'
    PRINT 20, TIME, TDIFV
 20 FORMAT(1X,F8.3,3X,F6.3)
*
 30 IF(COUNT .NE. FREQ) GO TO 100
    PRINT 40, TIME, TDIFV
 40 FORMAT(1X,F8.3,3X,F6.3)
*
*   INITIALIZING THE COUNTER TO ZERO
*
 50 COUNT = 0.0
*
*   NOTE: THE VALIDATION TIME PERIOD WAS ARBITRARILY TAKEN
*       AS 3600 SECONDS (1 HOUR).
*
100 IF(TIME .LE. 3600) GO TO 10
*
    STOP
```

LIST OF REFERENCES

1. Özisik, M. N., *Heat Conduction*, p. 471-521, John Wiley & Sons, 1980.

2. Incropeia, F. P., and DeWitt, D. P., *Introduction to Heat Transfer*, pp. 173-242, 669-672, John Wiley & Sons, 1985.

3. Brian, P. L. T., "A Finite-Difference Method of High-Order Accuracy for the Solution of Three-Dimensional Transient Heat Conduction Problems," *American Institute of Chemical Engineers Journal*, n. 7, pp. 367-370, September 1961.

4. Clover, R. M., and Wassel, A. T., "Three-Dimensional Transient Heat Conduction in a Multilayer Medium," *Journal of Spacecraft and Rockets*, n. 22, pp. 211-214, March-April 1985.

5. Carnahan, B., Luther, H. A., and Wilkes, J. O., *Applied Numerical Methods*, pp. 429-530, John Wiley & Sons, 1969.

6. Abramowitz, M., and Stegun, I. A., eds., *Handbook of Mathematical Functions*, p. 299, Dover Publications, 1972.

7. Agrawal, B. N., *Design of Geosynchronous Spacecraft*, pp. 57-124, 265-292, Prentice-Hall, 1986

8. Naval Research Laboratory Report 7975, *Orbital Mechanics of General-Coverage Satellites*, by J. A. Eisele and S. A. Harper, pp. 1-20, 30 April 1976.

9. Aeronautical Systems Division Report 61-119, *Radiation Heat Transfer Analysis for Space Vehicles*, by J. A. Stevenson and J. C. Grafton, pp. 1-32, 223-240, 297-426, December 1961.

10. National Aeronautics and Space Administration Report SP-8105, *Spacecraft Thermal Control*, by R. Lyle and others, pp. 1-45, May 1973.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Professor A. J. Healey, Code Hy Department of Mechanical Engineering Naval Postgraduate School Monterey, CA 93943-5004	2
4.	Professor Y. Joshi, Code 69Ji Department of Mechanical Engineering Naval Postgraduate School Monterey, CA 93943-5004	3
5.	Professor A. D. Kraus, Code 62Ks Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5004	2
6.	Professor M. D. Kelleher, Code 69Kk Department of Mechanical Engineering Naval Postgraduate School Monterey, CA 93943-5004	1
7.	LCDR Karl R. Heinz SMC #1851 Naval Postgraduate School Monterey, CA 93943-5000	1
8.	LCDR John A. Watson 331 East T Street Benecia, CA 94510	2