AN ABSTRACT OF THE THESIS OF

Jeffrey Wade Dickinson for the degree of Master of Science in Nuclear Engineering presented on January 26, 1994.

Title: Computer Modeling and Analysis of Single and Multicell Thermionic Fuel Elements

Redacted for Privacy

Abstract Approved: _____

Andrew C. Klein

This paper presents the modeling efforts undertaken to develop a computer code that is able to perform coupled thermal-hydraulic and thermionic analysis for both single and multicell Thermionic Fuel Elements (TFE). The code developed, given the name MCTFE (for multicell thermionic fuel element), is a steady-state finite element code specifically designed to analyze cylindrical TFEs. It employs an iterative successive-over-relaxation solution technique to solve for the temperatures throughout the TFE and a coupled thermionic routine to determine the total TFE performance. The calculated results include: the temperature distributions in all regions of the TFE, the axial interelectrode voltages and current densities, and total TFE electrical output parameters including power, current and voltage.

To accurately model and determine a given TFE's performance, a detailed description of the TFE's configuration must be given. This information is supplied via an input deck that the prospective user creates for each TFE to be analyzed. The MCTFE code was designed for flexibility. Some of the "flexibility minded" options of the code include: the number of cells, the number of axial nodes for each cell, the number of fuel interior radial nodes, the power density distribution within the fuel, and the electrical boundary condition description needed for the thermionic analyses. Physical characteristics such as lengths, thicknesses, and material types of the various regions of the TFE are also designated by the user.

The thermionic modeling is performed by a coupling of the VOLTCALC routine, that determines interelectrode voltages based on current densities, and the CDEN function, that determines the current and electron cooling densities from the interelectrode voltage and electrode temperatures. The CDEN function comes from the CYLCON-6 routine developed by John McVey of Razor Associates Inc. Once a converged set of current densities and voltages are achieved, a redetermination of the temperatures (now necessary due to the new set of electron cooling values calculated) can be performed. This coupling process is continued until the temperatures, voltages and current densities are converged according to the prescribed criteria.

In order to benchmark the accuracy of the code methods, MCTFE results are compared to experimental data. For the single cell aspects of MCTFE, data from the Topaz-II type TFE are used to benchmark the code. The multicell dependent routines are verified by comparison to the 3H5 multicell TFE of General Atomics. In both cases, the results achieved have shown a good agreement between the code and experimental data.

Computer Modeling and Analysis of Single and Multicell Thermionic Fuel Elements

by

Jeffrey Wade Dickinson

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Completed January 26, 1994

Commencement June 1994

APPROVED:

Redacted for Privacy

Andrew C. Klein, Associate Professor, Nuclear Engineering

Redacted for Privacy

Alan H. Robinson, Head, Department of Nuclear Engineering

Redacted for Privacy

Dean of Graduate School

Date thesis is presented January 26, 1994

Typed by researcher for Jeffrey W. Dickinson

ACKNOWLEDGEMENT

The work presented here is by no means without the support of other groups and individuals. Many others provided assistance in various ways, and I offer my thanks and appreciation to all who have given of their time and/or means in support of this effort.

Specifically, I would like to acknowledge some of the more significant contributors to whom I am in debt: to Dr. Andrew C. Klein, my major advisor, for his assistance, direction and guidance, that have made this thesis possible; to Hsing H. Lee for his assistance with MCNP runs and his patience and willingness to discuss many questions and problems; to Ron Pawlowski for the TFEHX computer code that he developed and that provided a starting point and parallel comparison basis for the MCTFE code developed by this study; to John McVey who developed the CDEN function and associated subfunctions and subroutines that is used in and is an essential part of the MCTFE code; and also to Lester Begg, Liz Drees, and Joe Smith of General Atomics who provided experimental test data and information useful in comparison studies with the MCTFE code.

Acknowledgement is also given for the financial assistance provided to help make this work possible: the Oregon Space Grant Fellowship Program; and Universal Energy Systems Inc., Dayton, Ohio, M.L. Ramalingam, Program Director.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. INTRODUCTION

In 1992 A Thermionic Fuel Element (TFE) modeling code was developed by Ron A. Pawlowski of Oregon State University [1,2]. This code, which was called TFEHX (Thermionic Fuel Element Heat Transfer), was able to perform coupled thermal-hydraulic and thermionic analyses for single cell TFEs and, as such, was one of the first codes developed with this capability. For a general discussion and the historical development of the thermionic emission process and its use with nuclear fuel elements, the reader is referred to Chapter 2, "Literature Review", where these topics are addressed.

The initial goal of the work presented here was to improve and modify TFEHX such that it would be able to perform analysis of multicell TFEs. Early on in the effort, it became evident that a simple modification of TFEHX would not be feasible, but instead a new code (although making use of many of the TFEHX developments) would be necessary. One of the difficulties of using TFEHX for multicell TFEs was its use of Gaussian Elimination to solve for the converged temperature fields. With the hard wired 10 by 10 node system used by TFEHX the memory requirements were still tractable; however, when allowing for multiple cells and user specified node requirements (including a variable number of fuel interior nodes), the memory requirements can very quickly exceed the typical computers capability (i.e., the number of characters that are required to be stored goes as the square of the total number of nodes). Therefore, a standard successive-over-relaxation (SOR) iteration technique was chosen and implemented with the new code to solve for the converged temperature field [3]. Another major modification was necessary with the thermionic analysis of the system because TFEs of more than one cell utilize a different circuit arrangement. This required the development of a thermionic modeling subroutine equivalent to the CYLCON-6 subroutine used by TFEHX for single cell TFEs [4].

The code developed, which is called MCTFE (multicell thermionic fuel element), is able to perform coupled thermal-hydraulic and thermionic analysis of TFEs with a variable

number of cells (1-10). The MCTFE code was also written to allow flexibility. Some of the additional modifications made to improve flexibility include: a variety of choices to describe the power density distribution in the fuel, optional electrical boundary condition choices, and user designation of system resolution (i.e., number of nodes). The complete discussions of these modifications, including other minor changes that were made, are discussed in the detailed descriptions that follow in the remaining chapters of this report.

The specific modeling methodologies utilized to develop the MCTFE code, including the differences between single cell and multicell TFEs, are also discussed in detail in this report. As appropriate, each area of modeling is first discussed as applied to single cell TFEs. This is then followed by the additional considerations that a multicell system introduces to the modeling task. Benchmarking of the code was accomplished by comparison to the single cell Topaz-II TFE [5] and the multicell 3H5 TFE developed by General Atomics in their Thermionic Fuel Element Verification Program [6].

CHAPTER 2. LITERATURE REVIEW

2.1 INTRODUCTION

This chapter discusses the basic principles involved with the thermionic emission process and its use with nuclear fuel as a thermionic converter. In addition, a brief outline of the historical development of thermionic emission and converters is given. Explanations and comparisons of single and multicell Thermionic Fuel Elements (TFE) are provided to clarify the differences between these two types of TFEs and establish a base level of understanding for further discussions.

The focus of this chapter is in how thermionic technology can be applied to space nuclear reactor systems where its use appears to have the most promise. The unique requirements of few moving parts (which can transfer into high system reliability) and low system weight are necessary prerequisites for any potential space system design. The thermionic reactor systems with their high operating temperatures and direct conversion of heat to electricity are particularly advantageous to space system needs.

For further details on thermionic emission and conversion technology the interested reader is directed to the excellent books by Hatsopoulos and Gyftopoulos [7,8]. More information about the subject of space nuclear power, including the application of thermionic converters, can be found in the text by Angelo and Buden [9], while an overview of the historical development of thermionic reactors has been discussed by Ranken [10].

2.2 THERMIONIC EMISSION

Thermionic emission is the process where a metal at a very high temperature spontaneously emits electrons from its surface. The process has often been compared to the evaporation of a liquid [7]. In a liquid the molecules move at various energies and some of the near surface molecules have sufficient energy to escape the molecular forces that hold them to the bulk liquid and thus escape. Similarly, the electrons of a metal are held by the coulomb forces of the positively charged nuclei. If a given electron's velocity

is of sufficient magnitude and in the proper direction it is able to overcome the attractive forces holding it and it will then leave the surface of the metal. The velocity distribution of the metal's electrons is prescribed by the rules of quantum thermodynamics [8]. For the velocity components (v_x) in directions normal to the surface, the distribution is given by the half-Maxwellian function at the temperature of the emitting surface shown below [7]:

$$f(v_x) = (2m_e/\pi kT)^{1/2} \exp(m_e v_x^2/2kT), \quad \text{for } v_x \geq 0.$$

As the temperature of the metal is increased, a greater number of electrons have sufficient velocity (kinetic energy) to overcome the potential barrier holding them in the metal, and they will leave the metal surface.

The potential barrier holding the electrons within the metal is called the work function (ϕ). The emission rate or current of electrons leaving the surface (J) is described in terms of the work function, as well as the temperature of the surface, by the Richardson-Dushmann equation given below [11]:

$$J = A\,T^2 \exp(-\phi/kT), \quad \text{where} \quad A = 4m_e\pi k^2 e/h = 120 \text{ amp/cm}^2 - \text{K}^2.$$

Therefore, by heating a metal to a very high temperature, the emission of electrons can be significant and if a means of collecting the emitted electrons is provided the potential of an electrical power source is achieved.

The historical development of thermionic emission can be traced as far back as Edison's discovery in 1885 that electric current could be made to flow between two electrodes in a vacuum if one of the electrodes were heated [12]. The analysis and experimental investigation of the emission process was first performed by Richardson in 1912 [11]. And the thermionic emission process was first recognized as a potential source of electricity by Schlichter in 1915 [13].

2.3 THERMIONIC CONVERSION

The development of the potential use of the thermionic emission process as a source of electricity by thermionic conversion was slow until about the late 1950's when more

practical conversion efficiencies were obtained [10]. In 1957 several groups were able to achieve conversion efficiencies of 5 to 10% and electrical power densities of 3 to 10 watts/cm^2 [14].

To understand the thermionic conversion process it is useful to consider a simplified schematic. Figure 2.1 below illustrates the basic components of a vapor diode converter system [7]. As can be seen from the figure the basic components of the system include: the electrodes, an enclosure, a heat source, a heat sink, and a load that completes the electrical circuit.

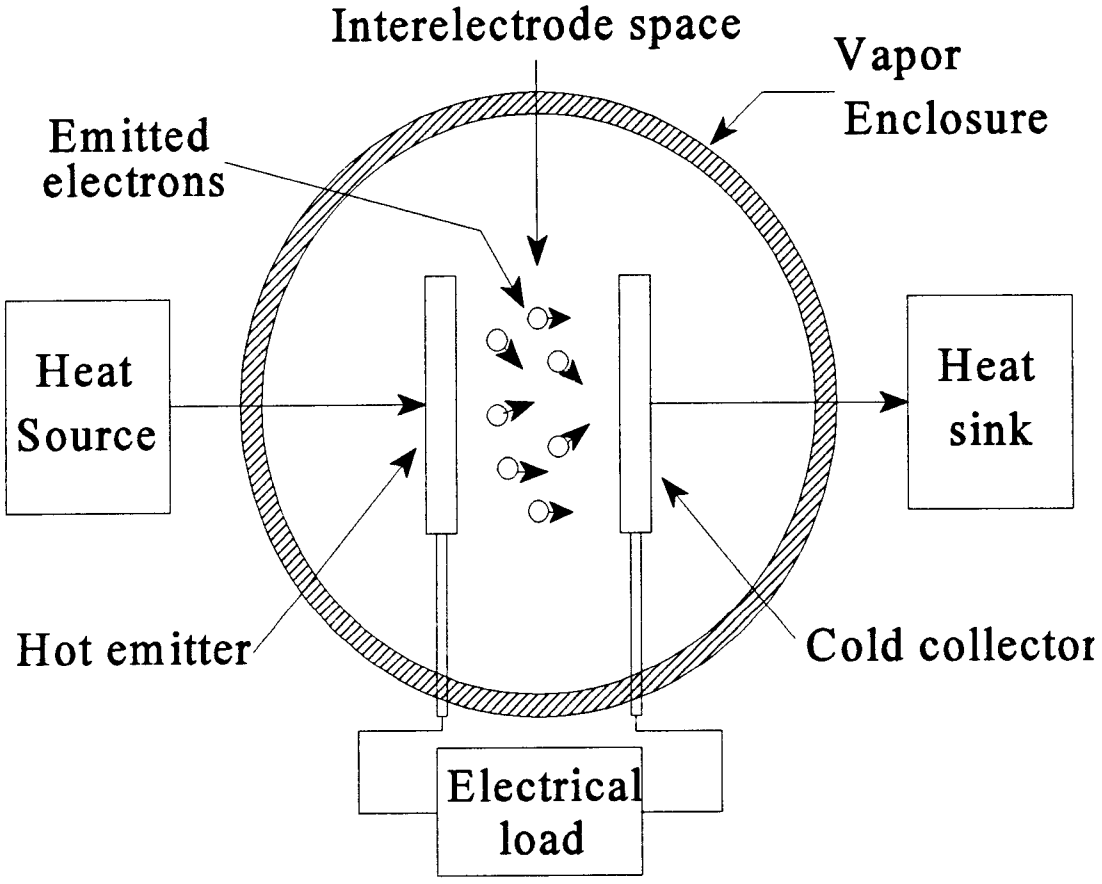


Figure 2.1 Schematic drawing of a basic vapor diode converter system [7].

The electrodes are made of refractory metals that provide an electrical conducting path and the emitting and collecting surfaces for the electrons. Some examples of commonly used materials for the electrodes are tungsten, molybdenum, and niobium. As one might guess by their function, the higher temperature electrode that the electrons are emitted or "evaporated" from is called the "emitter", and the cooler electrode that collects or "condenses" the electrons is given the name "collector".

The heat source supplies the energy necessary for the electrons to overcome the potential barrier and leave the metal surface, while the heat sink cools the collector thus providing a place for the electrons to "condense". It is thus seen that the thermionic converter is but a basic thermodynamic cycle whose efficiency will be limited by the magnitudes of the electrode temperatures.

Although the electrodes can operate in a vacuum, a rarefied vapor is often used in the interelectrode space. This vapor offers the benefit of enhancing the transport processes that occur in the interelectrode gap. While the emission process is ongoing, a negative space charge is present in the gap due to the concentration of the electrons traveling through the gap. This space charge inhibits the transport process by providing a resistive force to electron mobility (like charges repel). Therefore, a low vapor pressure of an easily ionizable gas is maintained in the gap to neutralized the ion field by the formation of positive ions. Because it is the most stable of all the easily ionizable gases, cesium is the gas usually chosen for this purpose [7].

The heat generator of the thermionic converter system can be any one of many possible technologies. Among the list of possible energy sources are: chemical, solar, radioisotope, and nuclear fission [9]. While chemical generators (e.g., fossil fuels) can supply the necessary power for high electrical demands, their limited lifetime prevents their use with space systems. Both the solar and radioisotope type converters can provide for long life but are limited in the power levels they can achieve. The nuclear reactor thermionic generator, however, can meet both the long life and high power requirements and has thus received the greatest attention as a future electrical supply source for high power spaced based systems.

2.4 THERMIONIC FUEL ELEMENTS

The Thermionic Fuel Element (TFE) is a diode converter using nuclear fuel as the heat source and is the basic building block of a thermionic reactor system. A typical reactor system will consist of an array of long cylindrical fuel elements where the converter electrodes are an integral part of the TFE (the emitter material is also used as the containment for the fuel). Figure 2.2 shows a cross-sectional view of a typical TFE indicating the various regions that the TFE is composed of, while Figure 2.3 illustrates a typical thermionic reactor [9].

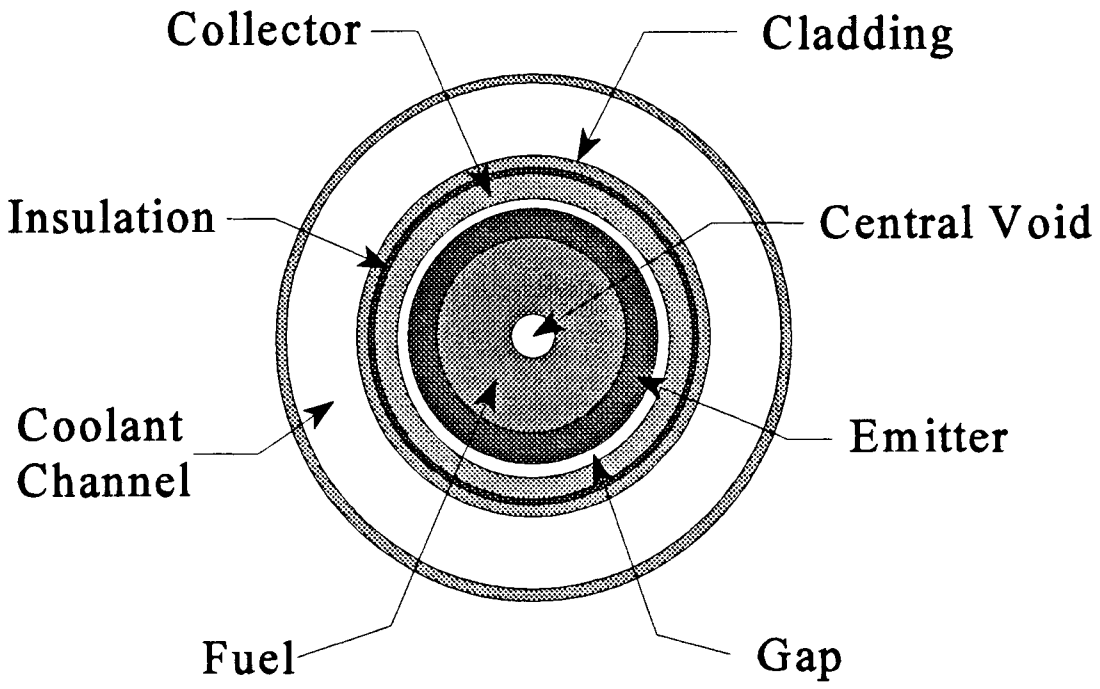


Figure 2.2 Cross-sectional view of cylindrical thermionic fuel element

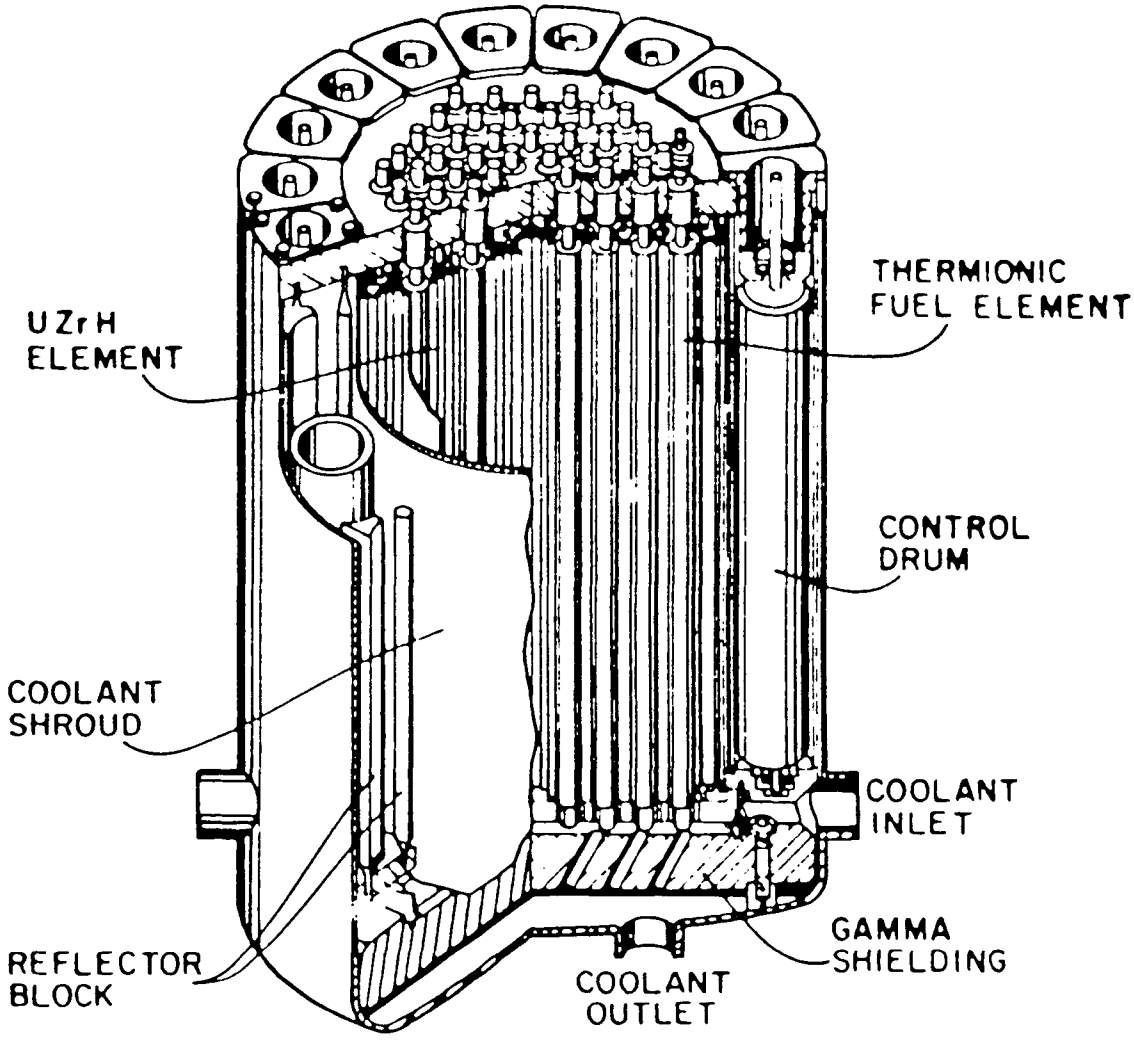


Figure 2.3 Thermionic reactor schematic [9]

Thermionic reactor development started in the late-1950's, but most of the early work on in-core thermionic conversion concepts was performed from 1963 to 1973 [9]. The main application, as previously mentioned, is for space environments with systems for unmanned satellites and manned space laboratories being projected from a few to hundreds of kilowatts of electric power. Out-of-core thermionic designs, where the reactor heat is transported to an external converter, were also examined from 1973 until 1982 [9], but, most of the current research is in the area of in-core systems.

The operating TFE accomplishes the direct conversion of heat into electricity without the need of an intermediate conversion system (e.g., a steam turbine). The obvious savings in system weight as well as minimal moving part requirements are a distinct advantage for the thermionic reactor concept's use in space. In addition, the high operating temperatures of the electrodes, necessary for good conversion efficiency, are an advantage to the heat rejection system. The heat removal systems typically dissipate heat radiatively to space, which is a process dependent on the fourth power of temperature. Higher system temperatures, therefore, require less radiator area for the same heat rejection rate and thus an additional weight savings.

As previously mentioned, Figure 2.2 illustrates the basic regions that are included in a typical TFE. These regions include: 1) the fuel, 2) the emitter, 3) the interelectrode gap, 4) the collector, 5) the insulator, 6) the outer cladding, and 7) the coolant channel. Also shown by the figure is a central void that is usually present in the fuel to facilitate the removal of the fission product gases that are formed by the fissioning process.

In order to maintain a net current flow from emitter to collector, the collector must be kept cool relative to the emitter (i.e., the collector also thermionically emits electrons). Typical electrode temperatures will be about 800 to 1000 K for the collector and 1600 to 2000 K for the emitter. Unfortunately, maintaining a large temperature difference between the two electrodes also promotes radiative and conductive heat transfer from the emitter to the collector. Therefore, unless this excess heat is removed, the collector temperature will continue to rise to that of the emitter. To remove the excess heat from the system, a coolant flows through the coolant channel and transfers this heat to the heat

rejection system. Due to the high system operating temperatures a NaK liquid metal is commonly used as the coolant.

2.5 COMPARISON OF SINGLE AND MULTICELL THERMIONIC FUEL ELEMENTS

The design of a cylindrical TFE can involve a long single cell element or a collection of shorter cells connected in series much like that of flashlight batteries. Figure 2.4 illustrates the single cell concept by showing a Topaz-II type TFE [15], and Figure 2.5 illustrates a multicell design being developed by the Thermionic Fuel Element Verification Program [16]. For the multicell configuration the emitter of one cell is electrically connected to the collector of an adjacent cell to complete the electrical circuit.

The output power of a TFE is given by the product of the voltage and current. For single cell TFEs, where a long element is used, a larger total current will be generated. With the multicell TFE, however, the result is a smaller current and a higher voltage due to the series connection of the shorter pieces. For example, a six-cell TFE could generate about six times the voltage and one sixth the current as a single cell TFE of equal length.

To compare the single cell and multicell TFEs it is useful to consider the advantages and disadvantages of each concept. The basic tradeoff between the two configurations is power and efficiency vs costs and reliability. The single cell TFE is a simple design that would have lower costs and higher reliability. However, due to the higher relative current, ohmic losses dictate operation at a less than optimum point on the volt-ampere curve of the converter that results in a lower power density and efficiency. The multicell TFE, by breaking up the converter into shorter pieces, limits the current and ohmic losses and operates at a more optimum voltage-current. But, this is at a cost of a less reliable and a more complicated design. Table 2.1 below lists the advantages and disadvantages of each as compared to the other. Although research is continuing with both TFE types [16,17], the current data would indicate that the single cell TFE is the better choice for lower power TFEs (e.g., about 5 - 50 kWe), while further research effort with the multicell converter should be pursued for higher level applications (e.g., 100 kW and above).

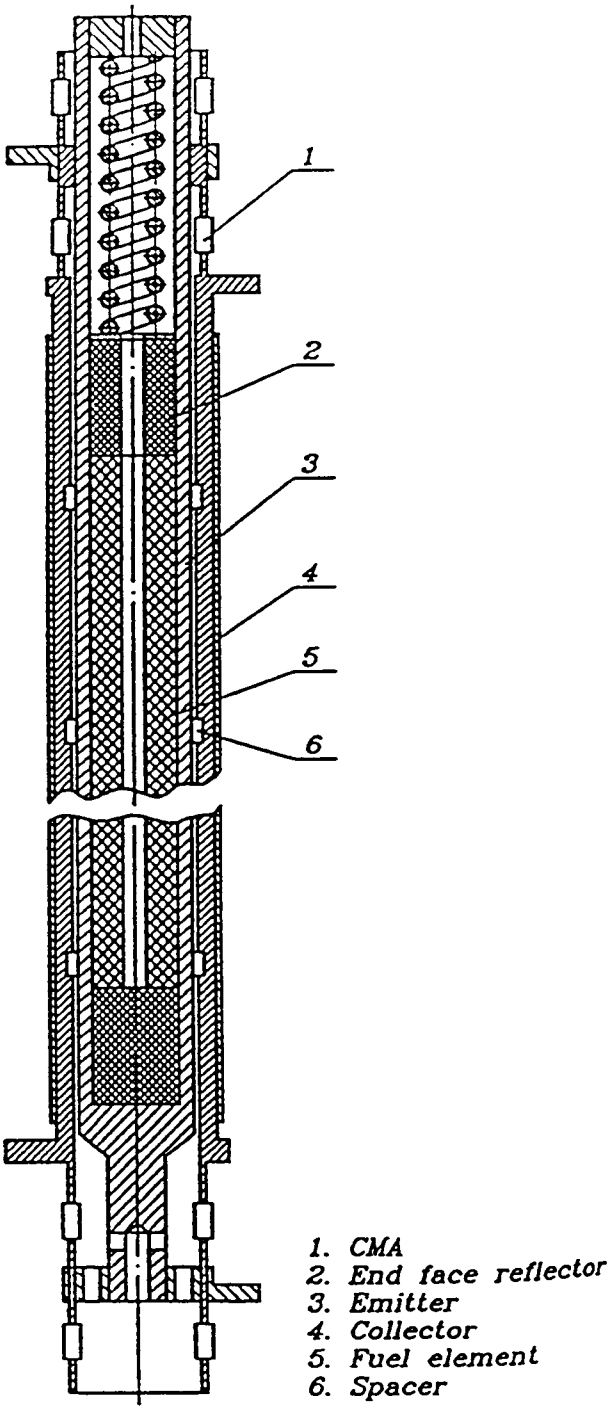


Figure 2.4 Topaz-II type single cell thermionic fuel element [15]

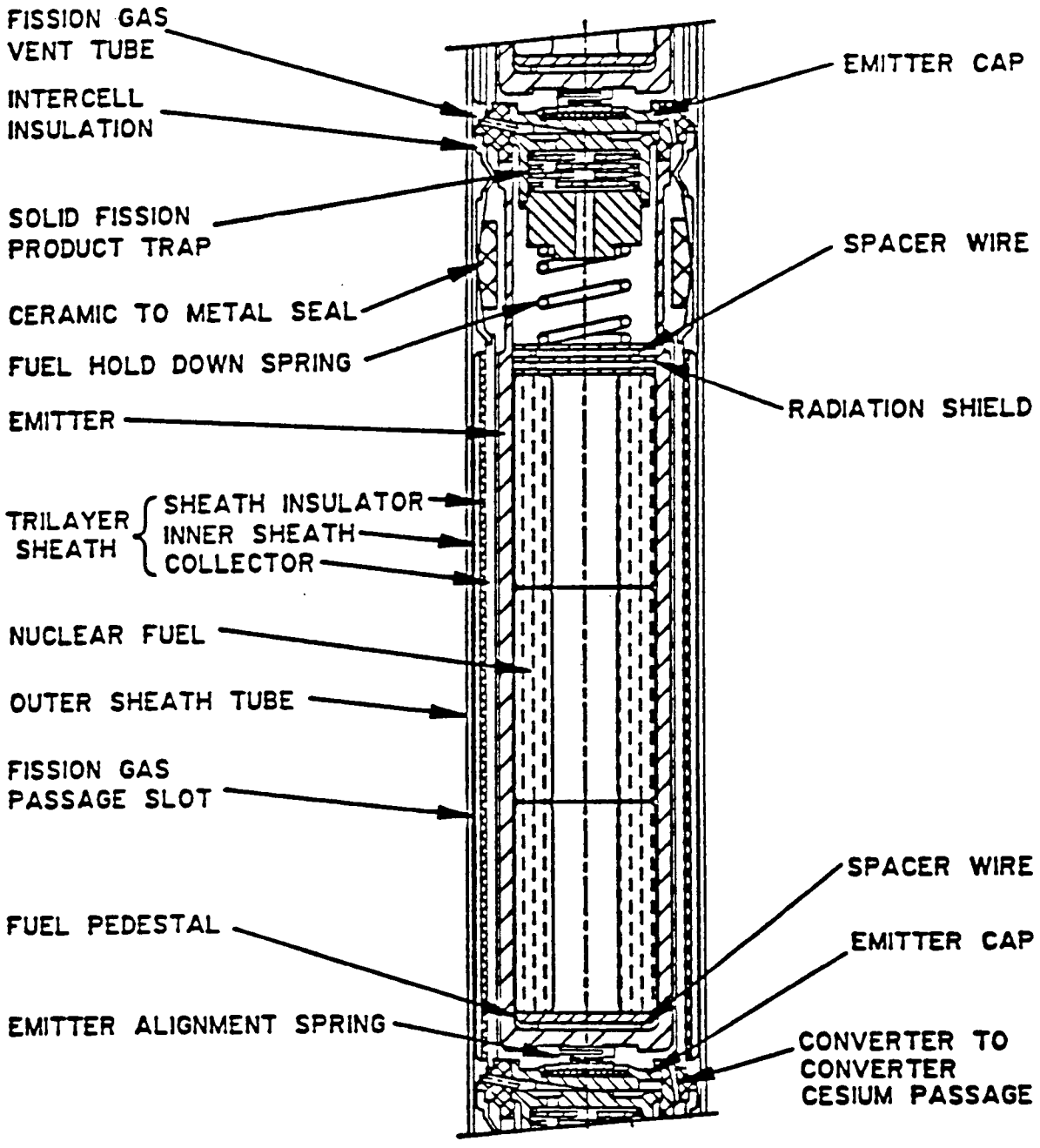
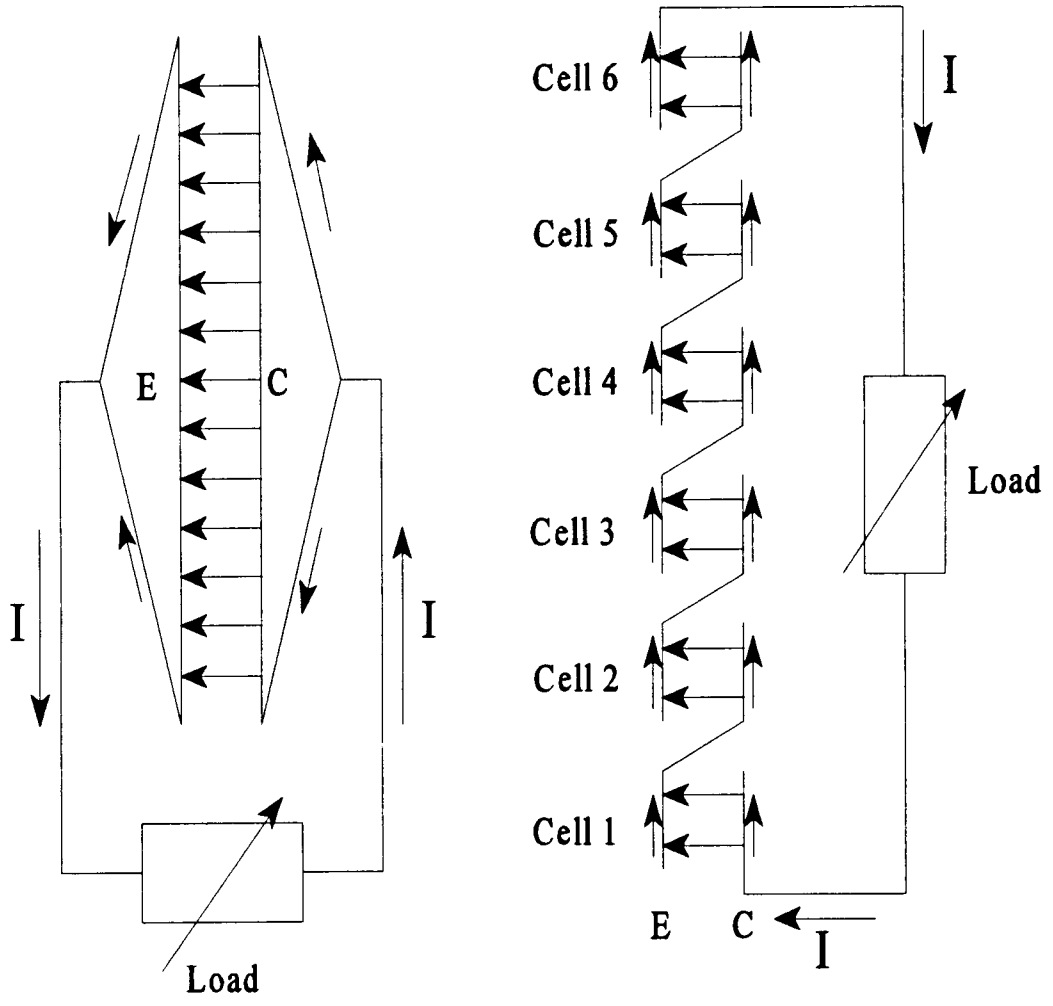


Figure 2.5 Schematic of a typical multicell thermionic fuel element [16]

As a final consideration between the two types of TFEs, the different electrical circuit arrangements between the two should be noted. For the single cell TFE, it is possible to make current connections at each end of each electrode of the TFE to reduce the average current in the electrodes (see Figure 2.6a). Thus, the system is, in effect, two cells operating in parallel. Therefore, although still higher than in the multicell TFE, the ohmic losses are reduced and the performance of the single cell TFE is enhanced. As previously mentioned, the multicell TFEs are connected in series. Figure 2.6b below illustrates the circuit arrangement that would be necessary for this TFE type. For further information on the comparison of single and multicell TFEs consult the paper by Ponomarev-Stepnoi et. al [15].

TABLE 2.1 Advantages and Disadvantages of Single and Multicell TFEs

	Advantages	Disadvantages
Single Cell TFE	- Easier removal of fission product gases - Allows testing by use of electric heater - Simpler to manufacture and assemble	- Low efficiency due to ohmic losses from high current - Low specific power production
Multicell TFE	- Higher efficiency (10 -12%) - Higher specific power, better for high power applications - Can easily accommodate axial enrichment zoning	- Complicated fission product removal system - Difficult construction and operation - Higher costs



a) Single cell TFE arrangement

b) A six cell TFE arrangement

Figure 2.6 Circuit diagrams of thermionic fuel elements

CHAPTER 3. SYSTEM MODELING

3.1 GENERAL CONSIDERATIONS

In order to facilitate the process of analyzing various thermionic fuel elements, without having to construct and test each possible configuration, it is useful to develop a system modeling code. This section details the approaches and methods used to create the MCTFE computer code that is written to perform coupled thermal-hydraulic and thermionic analyses of cylindrical thermionic fuel elements. MCTFE is a second generation code developed from the initial achievements made with TFEHX [1,2]. The MCTFE code is written in, as much as is practical, a general format allowing flexibility for the user to describe the particular TFE to be analyzed. This broad capability of the code includes the ability to analyze either single cell or multi-cell TFEs.

To accurately model and determine a given TFE's performance, a detailed description of the TFE's configuration must be given. This information is supplied via an input deck that the prospective user creates for each system to be analyzed. As mentioned before, the MCTFE code was designed for flexibility. Some of the "flexibility minded" options of the code include: the number of cells, the number of axial nodes for each cell, the number of fuel interior radial nodes, the power density distribution within the fuel, and the electrical boundary condition description needed for the thermionic analyses. Physical characteristics such as lengths, thickness, and material types of the various regions of the TFE are also designated. The beginning lines of the source code specifies the complete input deck content including a description of all the required input data.

The discussion of the modeling concepts utilized by MCTFE is broken into four areas: neutronic considerations (section 3.2), thermal-hydraulic considerations (section 3.3), thermionic considerations (section 3.4), and thermal-hydraulic and thermionic coupling (section 3.5). Each section gives an outline of the methodologies utilized by its topic and includes figures, equations, and discussion as necessary to explain the basis from which the applicable portions of the computer code are derived. The MCTFE source code is provided as Appendix A to answer any specific questions involving the MCTFE code.

Before proceeding to the following sections, it is useful at this point to describe the basic elements of the modeling process that are common to all areas. Any specific deviations from the following general descriptions will be addressed as necessary in the appropriate sections.

As previously stated, MCTFE is written to analyze cylindrical TFEs. To form a processing basis for cylindrical geometry, it is first necessary to describe a mesh or grid system from which finite elements can be formed. Figure 3.1 illustrates the basic mesh system employed by MCTFE. By assuming azimuthal symmetry, the description of the three-dimensional TFE is reduced to the two parameters: r and z. The index "i" is used to indicate the radial position (r) and varies from 1 at the inner most surfaces to N at the outer locations where N, the total number of radial nodes, is determined by user input. Similarly, the index "k" is used for the axial positions (z) and varies from 1 at the bottom of the first cell, to kmax at the top of the last cell where kmax, the total number of axial nodes, is also determined from the TFE input description given by the user.

With the skeletal framework defined, finite elements are then formed and described such that the various distributed parameters of the TFE can be calculated or specified. Some of the parameters so determined include: power densities (sect. 3.2), temperatures (sect. 3.3), and voltages, currents, and current densities (sect. 3.4).

The following sections now explain the calculational basis that MCTFE uses to generate the subroutines and their methods. By appropriately coupling the separate subroutines, which is discussed in section 3.5, the total performance of a TFE and thus a prediction of its thermal-hydraulic and thermionic behavior is obtained.

3.2 NEUTRONIC CONSIDERATIONS

3.2.1 General Description

The neutronic considerations of the MCTFE code are limited to the determination of the power densities that exist in the fueled region(s) of the TFE. Because MCTFE is a code that is designed to analyze a single TFE (which may contain one or more cells)

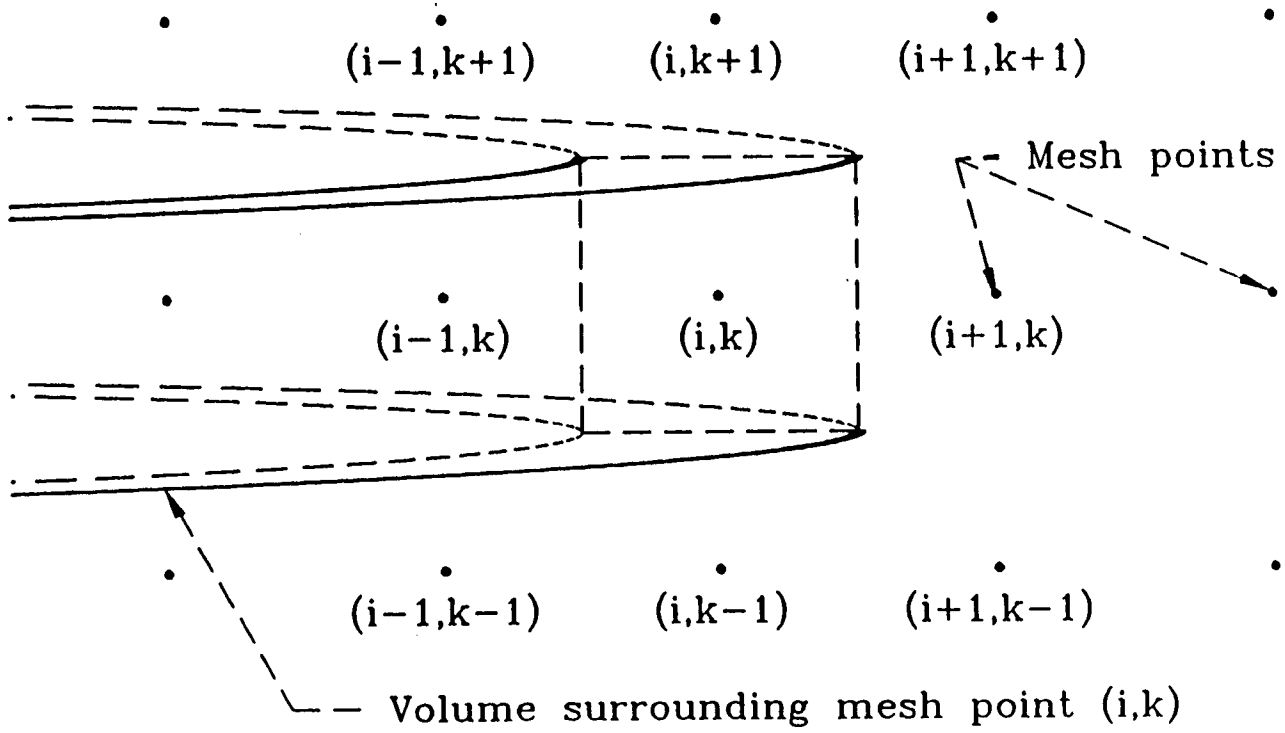


Figure 3.1 Cylindrical ring volume about the mesh point (i,k)

information as to how the TFE is to be used in a reactor system is not known. The full reactor system description, including materials present (e.g., moderator and reflectors) as well as various dimensions and the system geometry, would be necessary to independently determine the fuel power density. Therefore, there is a necessary input dependance to, in effect, describe how the particular TFE is used in a reactor concept. Necessary input parameters would include: the total power of the TFE; the peak-to-average linear power generation ratio; and, if the PDCALC solution method is selected, parameters designating the effective neutron energy and degree of axial reflection present. Further information on the PDCALC subroutine is given in section 3.2.6.

To give flexibility, a variety of options are available for the user to select to describe the power generation distribution. These options include:

1) a constant power density both radially and axially;
2) a constant power density radially with a chopped cosine axial distribution;
3) a constant power density radially with the use of tabular data input (e.g., from MCNP) to describe the axial variation;
4) a 2-D array of tabular data input describing both axial and radial power generation densities; or
5) an estimation of the true power distributions, both radially and axially, by the PDCALC routine.

The following sections now describe how MCTFE determines the power density as a function of position in the fuel for each of the above options. For each option, the method is described and possible applications of the option are given.

3.2.2 Constant Radial and Axial Power

For the case of a constant power density throughout the fuel, both radially and axially, the determination of the power density is very simple. The value of the volummetric power generation rate is given by the total TFE power (supplied by the user via the input

deck) divided by the TFE fuel volume. The volummetric power generation rate at a given axial and radial position, q(i,k), is therefore determined by the following equation:

$$q(i,k) = Pwth / [TFL * \pi * (OR(f)^2 - IR(f)^2)], \qquad \text{for all i and k in fuel,}$$

where, Pwth is the total thermal power of the TFE, TFL is the total fuel length (fuel length of a single cell TFE or the sum of cell fuel lengths for a multicell TFE), and IR(f) and OR(f) are the inner and outer fuel radii respectively.

Although this particular power distribution description is not achieved in real nuclear fueled TFEs, it is nonetheless a useful option to retain in the code. Many TFEs are tested with the use of electrical heating elements that have a constant heat generation rate within the heater element. Code prediction of experimental test results, as well as benchmarking studies, can therefore be performed on such electrically heated TFEs using this option.

3.2.3 Constant Radial Power with a Chopped Cosine Axial Distribution

For the chopped cosine axial distribution (with a constant radial density) the input specified TFE power and peak-to-average linear power generation ratio is used to determine the volummetric power generation rates. Figure 3.2 below illustrates the assumed axial linear power generation rate shape for this option.

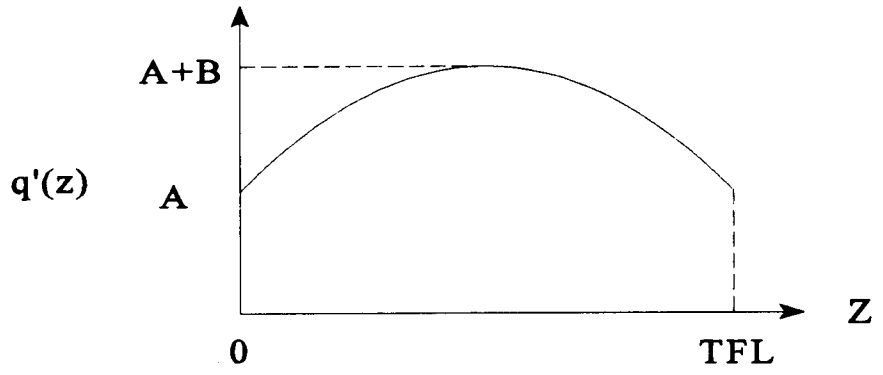
As seen by the figure, the linear power generation rate is dependent on the axial position (z) and is given by the following equation:

$$q'(z) = A+B \text{ SIN } (\pi z/TFL). \qquad (3.1)$$

To solve this equation, the constants A and B must be determined. These two unknowns are determined from the following two equations:

$$PktoAv = (A + B) / (Pwth/TFL), \text{ and} \qquad (3.2)$$

$$_0\int^{TFL} q'(z)\, dz = Pwth. \qquad (3.3)$$



$$q'(z) = A + B \, SIN \, (\pi Z / TFL)$$

Figure 3.2 Chopped cosine curve estimation of axial linear power generation rate

---

Solving equation 3.2 for A then yields the following equation:

$$A = (Pwth/TFL)*PktoAv - B. \qquad (3.4)$$

Substitution of equation 3.1 into 3.3 and integrating yields the following result:

$$Pwth = A*TFL + B*[TFL/\pi]*[-COS(\pi z/TFL|_0^{TFL}],$$

or

$$Pwth = A*TFL + 2*B*TFL/\pi. \qquad (3.5)$$

By next solving equation 3.5 for A, the following result is obtained:

$$A = Pwth/TFL - 2B/\pi. \qquad (3.6)$$

By setting equations 3.4 and 3.6 equal to each other, B can be shown to be:

$$B = (Pwth/TFL)*(PktoAv-1)/(1-2/\pi). \qquad (3.7)$$

The heat generation density at each axial position can now be determined by using equations 3.1, 3.6 and 3.7. The total power at a particular axial position is determined from the following equation:

$$TP(k) = \int_{k-1/2}^{k+1/2} q'(z)\, dz. \qquad (3.8)$$

Upon substituting equation 3.1 into 3.8 and performing the integration, the following result is obtained:

$$TP(k) = A*[z(k+1/2) - z(k-1/2)] + B*(TFL/\pi)*[-COS\{\pi*z(k+1/2)/TFL\} + COS\{\pi*z(k-1/2)/TFL\}]. \qquad (3.9)$$

The volummetric power generation rate density is then determined as follows:

q(i,k) = TP(k)/(fuel volume of node), for all i in fuel (since radial distribution is constant),

where,

$$\text{fuel volume of node} = \pi*\{OR(f)^2 - IR(f)^2\}*\{z(k+1/2) - z(k-1/2)\}.$$

This particular heat generation density description can be used when an estimate of the true axial distribution is desired without concern for the true radial distribution. In addition, it describes the distribution that is present when electrical heaters that simulate nuclear heating are used. This situation has been used with the testing of the Topaz-II type TFE [5], where a collection of heating elements are configured in such a way to produce the chopped cosine axial shape, while maintaining constant radial heat generation.

3.2.4 Tabular Data Input for Axial Density with Constant Radial Power Density

The option to input tabular data to define the fuel power density is provided in order to use MCNP data, or equivalent, as the power distribution description. In the input deck a column of entries is made that correspond to the axial values of the volummetric heat generation rates. The value at each axial position is assigned to each radial node at that

position (i.e., q(i,k) = q(k) for all i). The units of the input values can be any density type unit (e.g., MeV/g or Watts/cc). The MCTFE code will then normalize the values, weighted according to the volume of each respective node, such that the total power is as given by the input.

This particular option allows the user to be as accurate with the power density distribution as they are with the MCNP code that generates the values. Specific non-uniform effects, such as power peaking at the cell end faces, can then be taken into account, thus yielding more accurate fuel temperatures.

3.2.5 Tabular Data Input for Axial and Radial Power Densities

This option is the same as described in section 3.2.4 with the exception that two-dimensional input may now be used. In this way accurate values of both the radial and axial power densities can be utilized. The data are entered in a two-dimensional array in the input deck, using any density type unit, with the MCTFE code again normalizing the values such that the total TFE power given in the input is achieved. This choice can potentially yield the most accurate fuel temperatures provided precise data is used because the true distribution (i.e., both the axial and radial) is accounted for.

The true axial distribution, as has already been mentioned, is a chopped cosine. This shape results from the axial leakage that occurs at the top and bottom surfaces of the reactor. The true radial distribution is also not constant (for thermal reactors) but is higher at the outer regions of the fuel. Since the neutrons born in the fuel are moderated outside of the fuel and then re-enter to cause fission, the outer regions of the fuel can effectively shield the inner regions. Therefore, this option allows the user to account for the non-constant heat generation densities that result in both the radial and axial dimensions.

3.2.6 PDCALC Subroutine

While the use of MCNP data to describe the power density distribution can produce acceptable results, the long run times associated with the MCNP code can limit its use.

Consequently, the motivation for PDCALC was to develop an approximate method of determining the true power density distribution in TFEs that can produce reasonable results with a simple routine. This section describes the method utilized by PDCALC to achieve this goal.

Because MCTFE is designed to analyze a single TFE, input parameters that give information as to how the TFE is used in a reactor system are necessary. These parameters are: the effective energy of the moderated neutrons, and the degree of axial reflection present at the top and bottom surfaces of the reactor. MCNP is utilized to "tune" the PDCALC routine to a given reactor configuration. Subsequent analysis can then be performed using the tuned parameters for different TFE configurations provided the overall reactor system is not altered significantly.

The fueled regions of TFEs can consist of a long single piece element or as multicell elements where many shorter pieces are used. Figure 3.3 illustrates the basic difference between the two concepts by showing the fuel regions of each. The PDCALC subroutine developed for MCTFE is able to determine the power density distribution of each of the two types of configurations. The code uses a finite difference numerical method with a successive-over-relaxation iterative solution technique to solve for the power densities [3].

To simplify the calculations and methods used in the PDCALC subroutine, assumptions and defining physical descriptions are made. Listed below are the basic assumptions or definitions used by the code:

1) The fuel region is non-multiplying. This is expected to be reasonable due to the fact that the neutrons are born in the fuel at high energies (about 1 MeV) and are moderated outside the fuel and then re-enter to cause fission;

2) The boundary condition for each of the cells is taken as a constant current on the external surfaces of the fuel. This current is assumed to be from mono-energetic (one-group approximation) neutrons whose energy is determined from the MCNP results;

3) Diffusion theory used to develop the equations to be used by the numerical method;

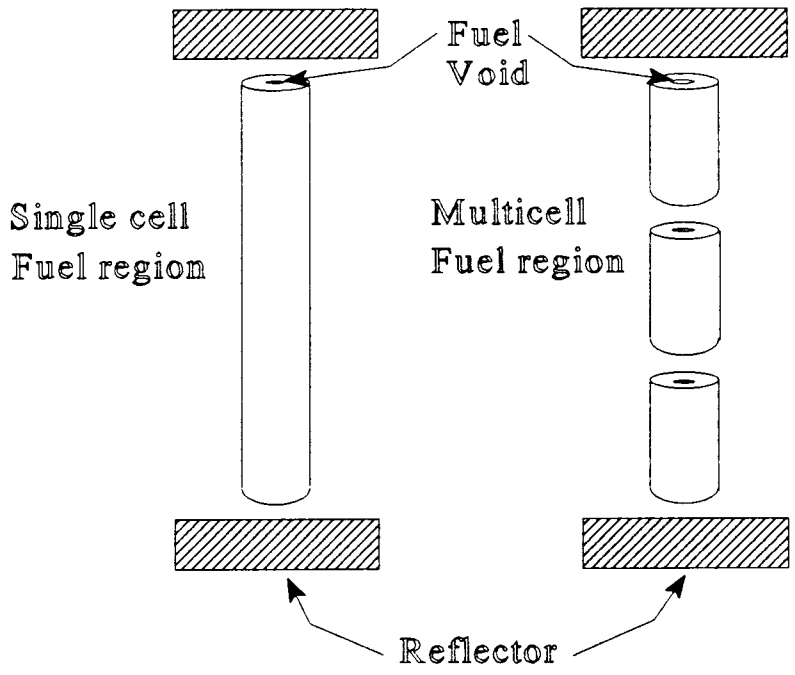


Figure 3.3 - Schematic illustration of single and multicell fuel regions

---

4) A constant and isotropic flux is assumed to exist outside of the fuel; and

5) Azimuthal symmetry.

As already stated, diffusion theory, which can be shown to be equivalent to a P_1 approximation to the neutron transport equation [18], is used to develop the numerical equations. The steady-state diffusion equation for non-multiplying media is therefore written as follows [19]:

$$\nabla^2 \phi - (1/L^2)\phi = 0, \qquad (3.10)$$

where $L^2 = D/\Sigma_a$ and $D = 1/(3\Sigma_{tr})$.

Now, in cylindrical coordinates ∇^2 is $\partial^2/\partial r^2 + 1/r\ \partial/\partial r + 1/r^2\ \partial^2/\partial\theta^2 + \partial^2/\partial z^2$, but, by the azimuthal symmetry assumption the third term involving angle is zero. And so equation

3.10 becomes:

$$\partial^2 \phi/\partial r^2 + 1/r\, \partial \phi/\partial r + \partial^2\phi/\partial z^2 - (1/L^2)\,\phi = 0.$$

The above equation is then used to determine the relative flux distribution, and thus the relative power distribution, within the fuel. Finite differences are used to approximate the derivatives in the equation and the MCNP determined neutron energy is used to determine the appropriate cross-sections which are corrected from 2200 m/s values [4] by assuming a $1/E^{1/2}$ dependance. It can be shown that the resulting difference equation for the interior nodes is as given by the following equation:

$$\phi(i,k) = \{1/[2/\Delta r^2 + 2/\Delta z^2 + 1/L^2]\} * \{[1/\Delta r^2 + 1/(2r\Delta r)]\, \phi(i+1,k) + [1/\Delta r^2 - 1/(2r\Delta r)]\, \phi(i-1,k) + 1/\Delta z^2\, \phi(i,k+1) + 1/\Delta z^2\, \phi(i,k-1)\}.$$

Which in iterative form becomes:

$$\phi(i,k) = \phi(i,k) + \text{Alpha} * \phi R(i,k),$$

where Alpha is the over relaxation parameter, and $\phi R(i,k)$ is the residual flux (i.e., incremental change in flux from one iteration to the next) given by,

$$\phi R(i,k) = \{1/[2/\Delta r^2 + 2/\Delta z^2 + 1/L^2]\} * \{[1/\Delta r^2 + 1/(2r\Delta r)]\, \phi(i+1,k) + [1/\Delta r^2 - 1/(2r\Delta r)]\, \phi(i-1,k) + 1/\Delta z^2\, \phi(i,k+1) + 1/\Delta z^2\, \phi(i,k-1)\} - \phi(i,k).$$

To determine the power density values at the boundaries the defined boundary conditions are used. At a given axial position a net current density exists and is assumed to be constant around the cell (i.e., azimuthal symmetry). A chopped cosine shape, determined from the user supplied peak-to-average linear power generation ratio, is used to calculate the relative boundary current axially. For the cell end face currents, the axial value of the chopped cosine curve at the end face axial position is used. These surfaces are also corrected for a radial dependance by a solid angle argument. This argument is based on the assumption of a constant and isotropic flux outside of the fuel such that the current density is proportional to the "non-fuel-shielded" solid angle subtended at a given

point. Therefore, the end face current will decrease over the face surface as the radial distance decreases.

The top and bottom end faces of the TFE at the reflector will experience a higher current density that is dependent on the reflector dimensions. Thus, an added parameter in the code that is "tuned" to the MCNP data was added to account for this higher current. The net current density of the fuel/void interface surface is taken as zero due to symmetry.

The resulting boundary condition equations are derived from the diffusion approximation [19], $J = -D \nabla \phi$, and are listed below:

1) For the fuel/void interface, with $J = 0$,

$$\phi(1,k) = \phi(2,k);$$

2) For the fuel/emitter interface (i.e., outer radial surface of the fuel),

$$\phi(i,k) = \phi(i-1,k) + J(k)\ \Delta r / D;$$

3) For the cell interior end faces (for multicell TFEs),

$$\phi(i,k) = \phi(i,k\pm1) + (SAC)\ J(k\pm1)\ \Delta r / D,$$

where SAC is the solid angle correction given as follows, with CSD = cell separation distance,

$$SAC = SQRT(CSD^2 + r^2)/SQRT(CSD^2 + r^2 + OR(f)^2);\ \text{and,}$$

4) For the top and bottom end faces,

$$\phi(i,k) = \phi(i,k\pm1) + (ARP)\ J(k\pm1)\ \Delta r / D,$$

where ARP is the axial reflection parameter.

With the relative boundary current densities defined, the power density distribution can be determined. This is accomplished by first initializing the boundary current values (at arbitrary scale but self-consistent with the defined relationships), and then adjusting

after each iteration until the total calculated power is in agreement with the TFE power given in the input by the user. The final result is a two-dimensional array of power density values that correspond to a given axial and radial position.

The PDCALC routine has shown the ability to give a good approximation of the axial and radial power density in the fuel. These results and a comparison to the MCNP data are provided in the results section of this report which is to follow.

3.3 THERMAL-HYDRAULIC CONSIDERATIONS

3.3.1 Introduction

To fully analyze the thermal-hydraulic behavior of a TFE, a detailed knowledge of the temperature distribution throughout the TFE is necessary. The first step in this process is to describe the various regions of the TFE that are to be considered. For the MCTFE code developed by this study, the regions that are defined and included in the modeling process are as follows: 1) void, 2) fuel, 3) emitter, 4) gap, 5) collector, 6) insulator, 7) cladding, and 8) coolant channel. These eight regions were illustrated in their relative positions by the cross-sectional view of Figure 2.2.

To determine the temperatures within all the defined regions of a TFE, each region is broken into a collection of finite elements. Each finite element is then examined to determine and define the heat transfer mechanisms associated with it. As appropriate, each of the primary modes of heat transfer (i.e., conduction, radiation, and convection) is used. In addition, a term accounting for the energy carried off by the electrons emitted by the thermionic emission process is included for the emitter and collector elements. The first law of thermodynamics for steady state systems (i.e., heat in = heat out), and the substitution of finite differences for the derivatives are then used to develop difference equations relating the temperature of the finite element to the temperatures of the neighboring elements. This process is repeated for all possible finite element types of each region and for all regions of the TFE. The result is a system of N equations and N unknowns where N is the total number of mesh points (one mesh point for each element).

In theory, the solution of the N equations can be done simultaneously, (e.g., Gaussian Elimination), but; in practical situations involving a large number of nodes, iterative methods are often implemented. Therefore, the MCTFE code utilizes a standard "successive over relaxation" (SOR) iterative solution technique to solve for the converged temperatures throughout the system by writing the difference equations generated for each node in iterative form [3]. After initializing each temperature, successive iterations are performed by sequentially sweeping throughout the field thus redetermining each node temperature based on its neighbors. This process is continued until convergence is obtained according to the prescribed convergence criteria.

To achieve reasonably accurate results, detailed considerations are made for various parameters. Included in these considerations are the thermal conductivities and electrical resistivities of the various materials as functions of temperature. The function KCOND, developed for TFEHX [1,2] and modified and expanded with more materials for use in MCTFE, is used to determine, node by node for each iteration, the thermal conductivity of the various materials. For element types on material interfaces, which include different materials within the element, an appropriately weighted average thermal conductivity is used. The function RHOW, which is a consolidation of the individual resistivity functions of TFEHX, is used similarly for the electrical resistivities.

The following sections are now provided to discuss how the finite elements are formed for each of the two types of TFEs modeled. These are: section 3.3.2, which describes the finite elements that exist in the analyses of single cell TFEs; and section 3.3.3, which discusses multi-cell TFEs. Much of the treatment of single cell TFEs also applies to the multi-cell cases. Therefore, the multi-cell section will discuss only the additional considerations that multi-cell systems incur while the basic elements that apply to both type systems will be examined in the first section.

3.3.2 Single cell TFEs

This section contains a description of all the possible finite elements of single cell TFEs. Included for each of the finite element types are: illustrations of the element

detailing the energy balance, the governing equations that characterize the element, and any specific assumptions made for the particular element.

3.3.2.1 Fuel Pellet/Central Void Interface - Top of the Fuel

Figure 3.4 illustrates the energy balance utilized for elements located at the top fuel surface and fuel/void interface. The q(i,k) term represents the power density (Watts/cc) generated within the fuel. These elements include the assumption that the central void and top fuel surfaces are adiabatic. Balancing the "heat in" and "heat out" terms yields the following equation:

$$q(i,k)\, A_2\, (z_k - z_{k-1/2}) = K_{i+1/2,k}\, A_1\, (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i,k-1/2}\, A_2\, (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1})$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1/[K_{i+1/2,k}\, A_1\, /(r_{i+1} - r_i) + K_{i,k-1/2}\, A_2\, /(z_k - z_{k-1})]\} * \{q(i,k)\, A_2\, (z_k - z_{k-1/2}) + [K_{i+1/2,k}\, A_1\, /(r_{i+1} - r_i)]\, T_{i+1,k} + [K_{i,k-1/2}\, A_2\, /(z_k - z_{k-1})]\, T_{i,k-1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

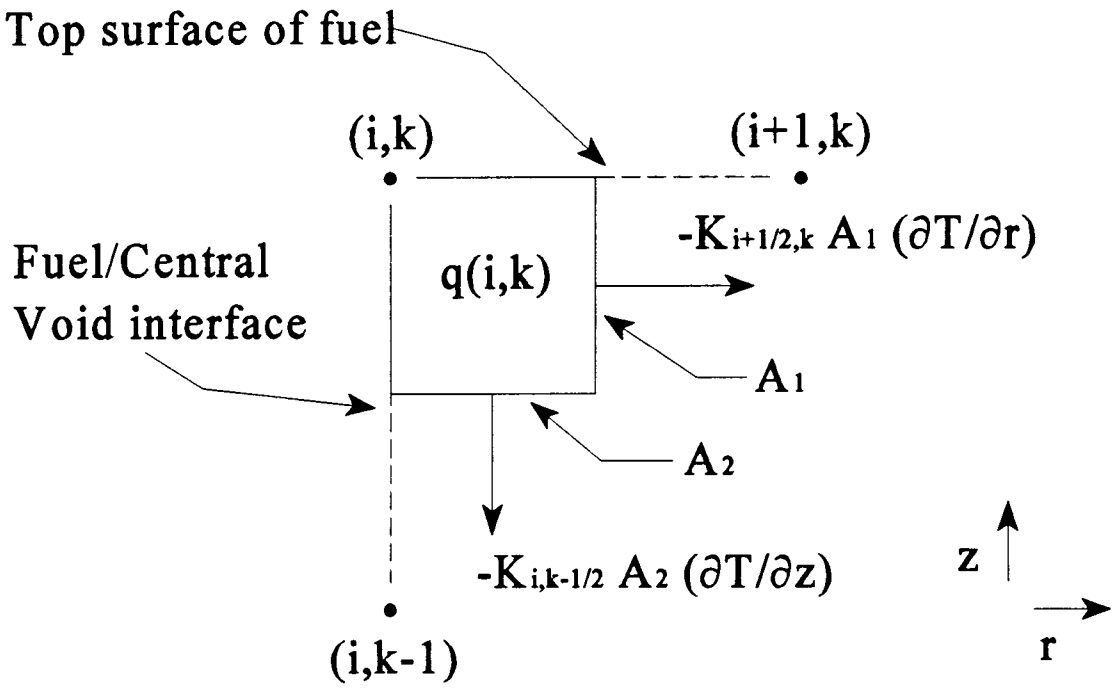
where Alpha is the over relaxation parameter and $R_{i,k}$ is the residual (i.e., incremental change in temperature from one iteration to the next) defined by

$$R_{i,k} = \{1/[K_{i+1/2,k}\, A_1\, /(r_{i+1} - r_i) + K_{i,k-1/2}\, A_2\, /(z_k - z_{k-1})]\} * \{q(i,k)\, A_2\, (z_k - z_{k-1/2}) + [K_{i+1/2,k}\, A_1\, /(r_{i+1} - r_i)]\, T_{i+1,k} + [K_{i,k-1/2}\, A_2\, /(z_k - z_{k-1})]\, T_{i,k-1}\} - T_{i,k}\,.$$

3.3.2.2 Fuel Pellet/Central Void Interface - Bottom of the Fuel

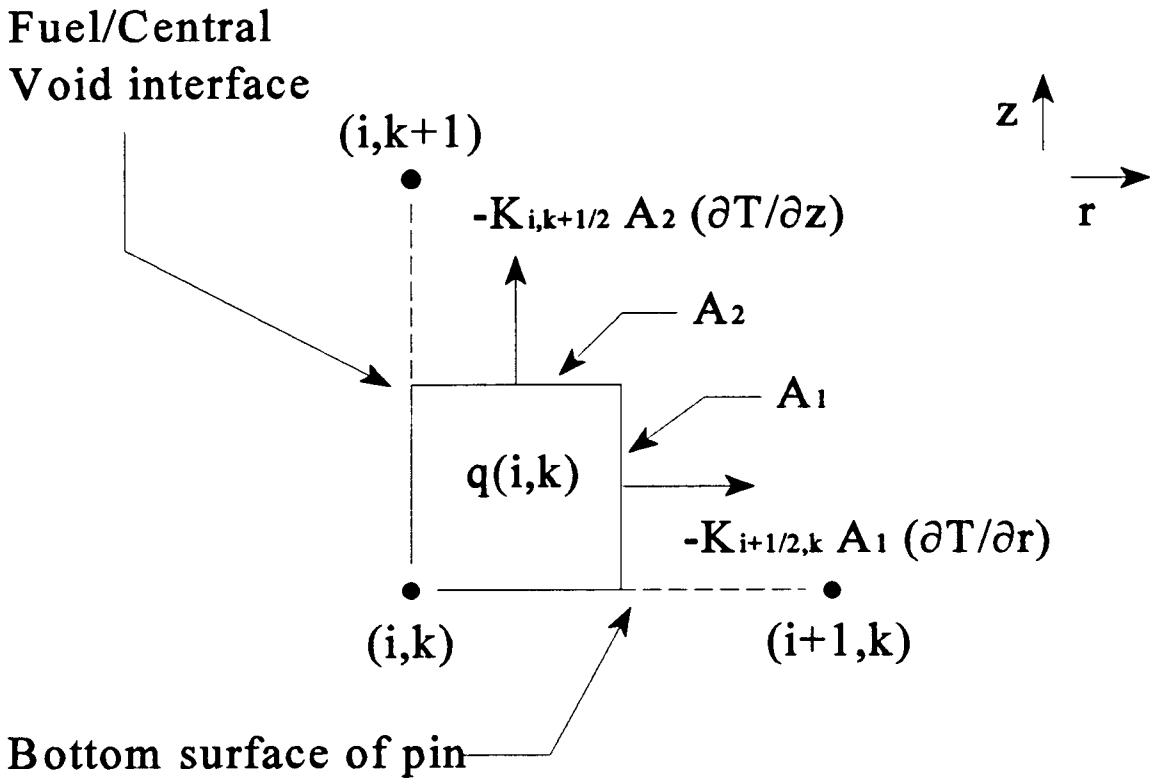
Figure 3.5 illustrates the energy balance utilized for elements located at the bottom fuel surface and fuel/void interface. These elements include the assumption that the central void and bottom fuel surfaces are adiabatic. Balancing the "heat in" and heat out" terms yields the following equation:

$$q(i,k)\, A_2\, (z_{k+1/2} - z_k) = K_{i+1/2,k}\, A_1\, (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i,k+1/2}\, A_2\, (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k)$$



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_k - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_i)$$

Figure 3.4 Energy balance for mesh points located at the top of the fuel pin and at the surface of the central void



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_{k+1/2} - z_k)$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_i)$$

Figure 3.5 Energy balance for mesh points located at the bottom of fuel pin and at the surface of the central void

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1/[K_{i+1/2,k} A_1 /(r_{i+1} - r_i) + K_{i,k+1/2} A_2 /(z_{k+1} - z_k)]\} * \{q(i,k) A_2 (z_{k+1/2} - z_k) + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1/[K_{i+1/2,k} A_1 /(r_{i+1} - r_i) + K_{i,k+1/2} A_2 /(z_{k+1} - z_k)]\} * \{q(i,k) A_2 (z_{k+1/2} - z_k) + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1}\} - T_{i,k}.$$

3.3.2.3 Fuel Pellet/Central Void Interface - Interior of the Fuel

Figure 3.6 illustrates the energy balance utilized for elements located within the fuel interior and at the fuel/void interface. These elements include the assumption that the central void surface is adiabatic. Balancing the "heat in" and heat out" terms yields the following equation:

$$q(i,k) A_2 (z_{k+1/2} - z_{k-1/2}) = K_{i+1/2,k} A_1 (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i,k+1/2} A_2 (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k) + K_{i,k-1/2} A_2 (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1})$$

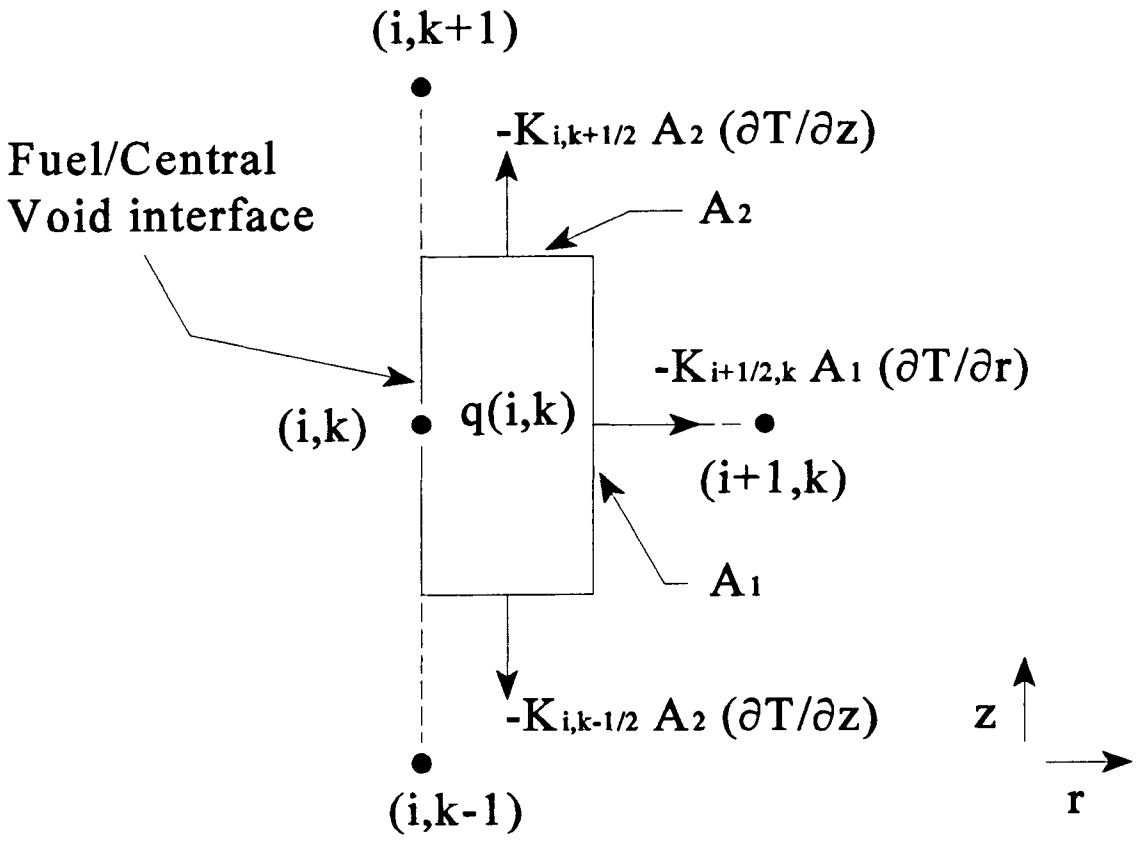
that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1/[K_{i+1/2,k} A_1 /(r_{i+1} - r_i) + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i,k-1/2} A_2 /(z_k - z_{k-1})]\} * \{q(i,k) A_2 (z_{k+1/2} - z_{k-1/2}) + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is



$$A_1 = 2\ \pi\ r_{i+1/2}\ (z_{k+1/2} - z_k)$$
$$A_2 = \pi\ (r^2_{i+1/2} - r^2_i)$$

Figure 3.6 Energy balance for mesh points located in the fuel interior and at the surface of the central void

$$R_{i,k} = \{1/[K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i) + K_{i,k+1/2}\, A_2 /(z_{k+1} - z_k) + K_{i,k-1/2}\, A_2 /(z_k - z_{k-1})]\} *$$
$$\{q(i,k)\, A_2\, (z_{k+1/2} - z_{k-1/2}) + [K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i)]\, T_{i+1,k} + [K_{i,k+1/2}\, A_2 /(z_{k+1} - z_k)]\, T_{i,k+1} +$$
$$[K_{i,k-1/2}\, A_2 /(z_k - z_{k-1})]\, T_{i,k-1}\} - T_{i,k}.$$

3.3.2.4 Fuel Pellet Interior - Top of the Fuel

Figure 3.7 illustrates the energy balance utilized for elements located within the fuel interior and at the top fuel surface. These elements include the assumption that the top fuel surface is adiabatic. Balancing the "heat in" and heat out" terms yields the following equation:

$$q(i,k)\, A_2\, (z_k - z_{k-1/2}) = K_{i+1/2,k}\, A_1\, (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i,k-1/2}\, A_2\, (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1}) +$$
$$K_{i-1/2,k}\, A_3\, (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1})$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1/[K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i) + K_{i,k-1/2}\, A_2 /(z_k - z_{k-1}) + K_{i-1/2,k}\, A_3 /(r_i - r_{i-1})]\} *$$
$$\{q(i,k)\, A_2\, (z_k - z_{k-1/2}) + [K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i)]\, T_{i+1,k} + [K_{i,k-1/2}\, A_2 /(z_k - z_{k-1})]\, T_{i,k-1} +$$
$$[K_{i-1/2,k}\, A_3 /(r_i - r_{i-1})]\, T_{i-1,k}\}$$

which in iterative form becomes

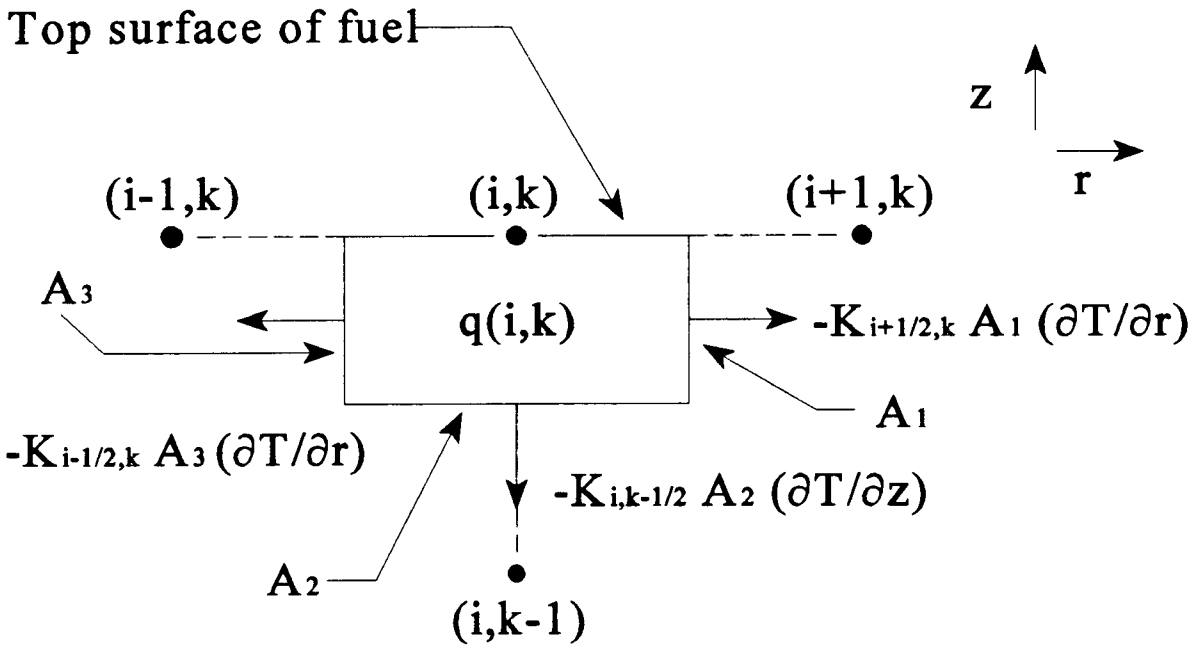
$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1/[K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i) + K_{i,k-1/2}\, A_2 /(z_k - z_{k-1}) + K_{i-1/2,k}\, A_3 /(r_i - r_{i-1})]\} *$$
$$\{q(i,k)\, A_2\, (z_k - z_{k-1/2}) + [K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i)]\, T_{i+1,k} + [K_{i,k-1/2}\, A_2 /(z_k - z_{k-1})]\, T_{i,k-1} +$$
$$[K_{i-1/2,k}\, A_3 /(r_i - r_{i-1})]\, T_{i-1,k}\} - T_{i,k}.$$

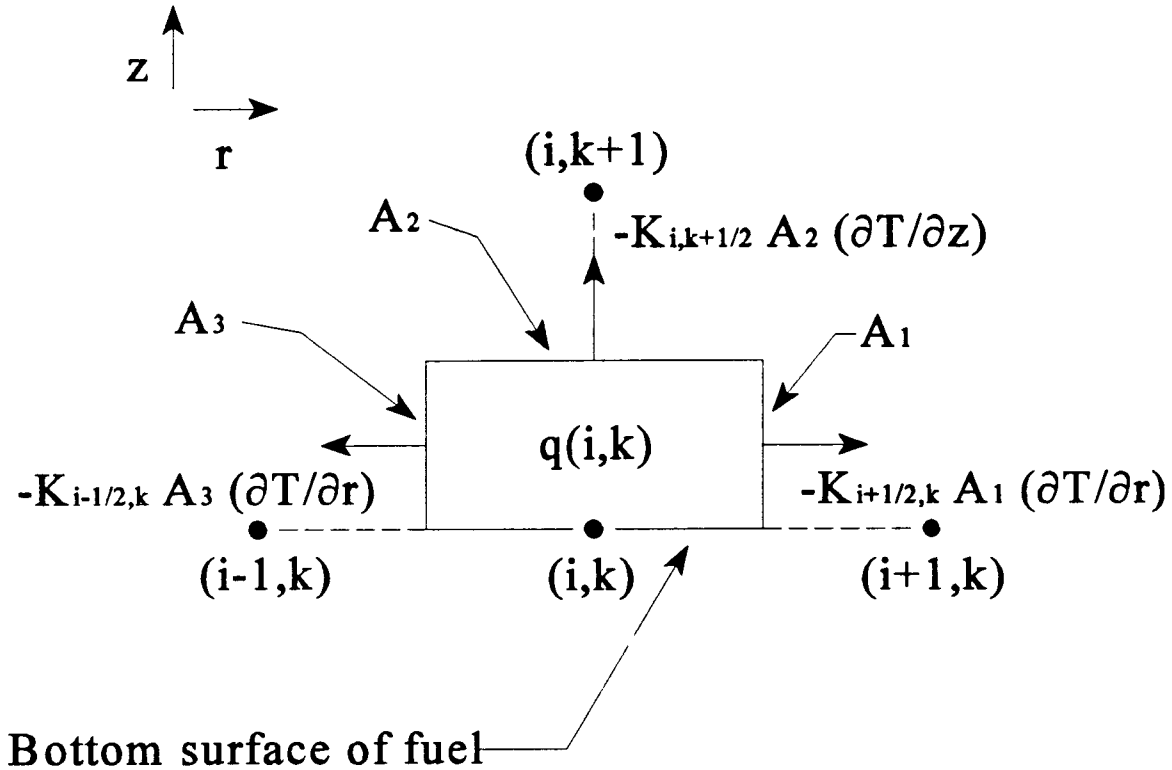
3.3.2.5 Fuel Pellet Interior - Bottom of the Fuel

Figure 3.8 illustrates the energy balance utilized for elements located within the fuel interior and at the bottom fuel surface. These elements include the assumption that the



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_k - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_k - z_{k-1/2})$$

Figure 3.7 Energy balance for mesh points located in the fuel interior and at the top surface of the fuel



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_{k+1/2} - z_k)$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_{k+1/2} - z_k)$$

Figure 3.8 Energy balance for mesh points located in the fuel interior and at the bottom surface of the fuel

bottom fuel surface is adiabatic. Balancing the "heat in" and heat out" terms yields the following equation:

$$q(i,k)\, A_2\, (z_{k+1/2} - z_k) = K_{i+1/2,k}\, A_1\, (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i,k+1/2}\, A_2\, (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k) + K_{i-1/2,k}\, A_3\, (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1})$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i) + K_{i,k+1/2}\, A_2 /(z_{k+1} - z_k) + K_{i-1/2,k}\, A_3 /(r_i - r_{i-1})]\} * \{q(i,k)\, A_2\, (z_{k+1/2} - z_k) + [K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i)]\, T_{i+1,k} + [K_{i,k+1/2}\, A_2 /(z_{k+1} - z_k)]\, T_{i,k+1} + [K_{i-1/2,k}\, A_3 /(r_i - r_{i-1})]\, T_{i-1,k}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i) + K_{i,k+1/2}\, A_2 /(z_{k+1} - z_k) + K_{i-1/2,k}\, A_3 /(r_i - r_{i-1})]\} * \{q(i,k)\, A_2\, (z_{k+1/2} - z_k) + [K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i)]\, T_{i+1,k} + [K_{i,k+1/2}\, A_2 /(z_{k+1} - z_k)]\, T_{i,k+1} + [K_{i-1/2,k}\, A_3 /(r_i - r_{i-1})]\, T_{i-1,k}\} - T_{i,k}\,.$$

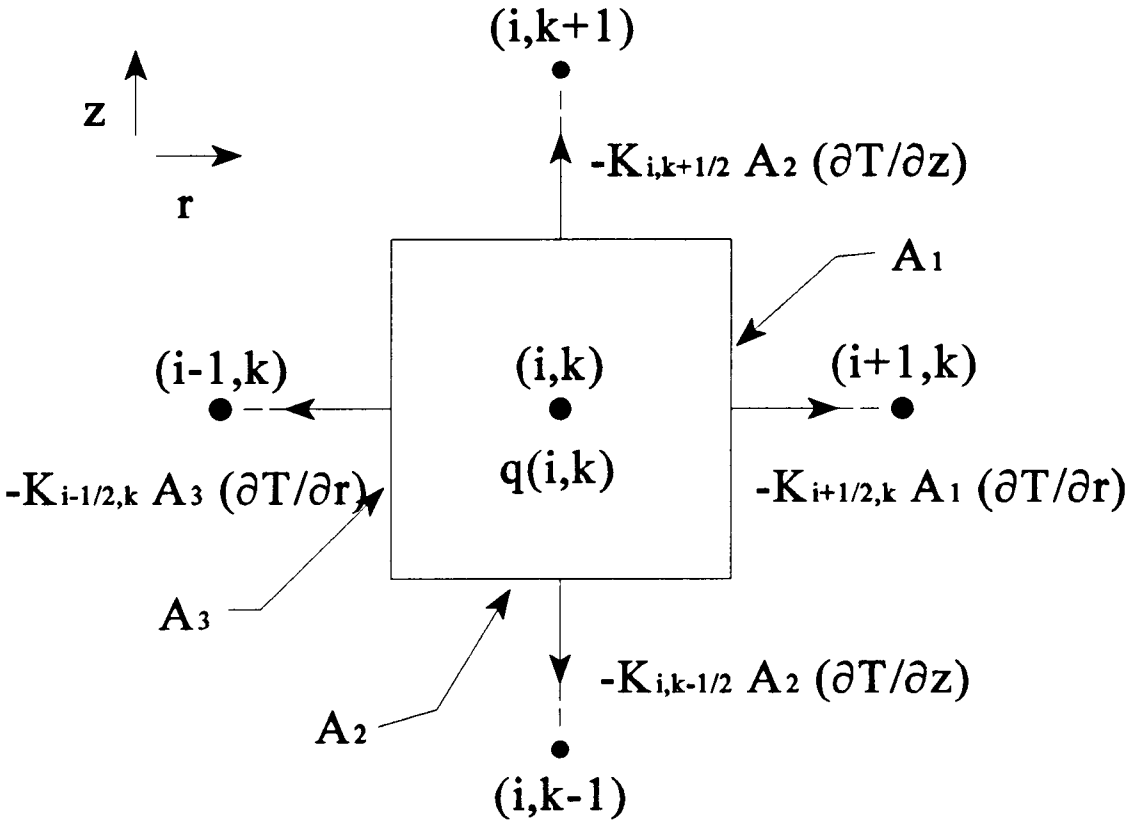
3.3.2.6 Fuel Pellet Interior

Figure 3.9 illustrates the energy balance utilized for elements located within the fuel interior. Balancing the "heat in" and "heat out" terms yields the following equation:

$$q(i,k)\, A_2\, (z_{k+1/2} - z_{k-1/2}) = K_{i+1/2,k}\, A_1\, (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i,k+1/2}\, A_2\, (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k) + K_{i-1/2,k}\, A_3\, (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k-1/2}\, A_2\, (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1})$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i) + K_{i,k+1/2}\, A_2 /(z_{k+1} - z_k) + K_{i-1/2,k}\, A_3 /(r_i - r_{i-1}) + K_{i,k-1/2}\, A_2 /(z_k - z_{k-1})]\} * \{q(i,k)\, A_2\, (z_{k+1/2} - z_{k-1/2}) + [K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i)]\, T_{i+1,k} + [K_{i,k+1/2}\, A_2 /(z_{k+1} - z_k)]\, T_{i,k+1} + [K_{i-1/2,k}\, A_3 /(r_i - r_{i-1})]\, T_{i-1,k} + [K_{i,k-1/2}\, A_2 /(z_k - z_{k-1})]\, T_{i,k-1}\}$$



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_{k+1/2} - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_{k+1/2} - z_{k-1/2})$$

Figure 3.9 Energy balance for mesh points located in the fuel interior regions

which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i+1/2,k} A_1 /(r_{i+1} - r_i) + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k-1/2} A_2 / (z_k - z_{k-1})]\}^* \{q(i,k) A_2 (z_{k+1/2} - z_{k-1/2}) + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k} + [K_{i,k+1/2} A_2 / (z_{k+1} - z_k)] T_{i,k+1} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1}\} - T_{i,k} .$$

3.3.2.7 Emitter/Fuel Interface - Top of the Cell

Figure 3.10 illustrates the energy balance utilized elements located at the top of the cell and at the emitter/fuel interface. The symbol w(i,k) represents the amount of work done within the emitter part of the element by the electrical current (i.e., I^2R losses). These elements include the assumption that the top cell surface is adiabatic. Balancing the "heat in" and "heat out" terms yields the following equation:

$$I^2_E (k) \rho_E(i,k) (z_k - z_{k-1/2}) \pi(r^2_{i+1/2} - r^2_i) + q(i,k) (z_k - z_{k-1/2}) \pi(r^2_i - r^2_{i-1/2}) = K_{i+1/2,k} A_1 (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i-1/2,k} A_3 (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k-1/2} A_2 (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1})$$

that can be solved for $T_{i,k}$ to yield

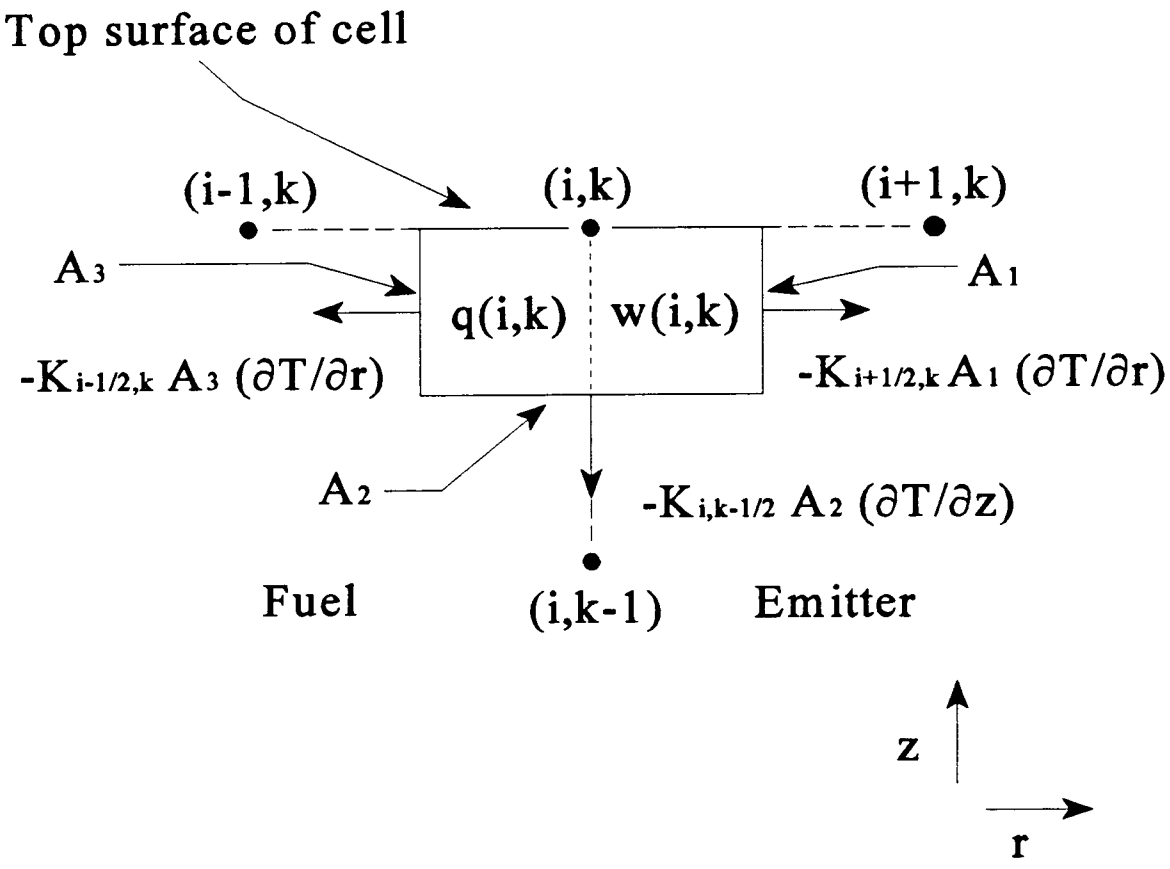
$$T_{i,k} = \{1 / [K_{i+1/2,k} A_1 /(r_{i+1} - r_i) + K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k-1/2} A_2 /(z_k - z_{k-1})]\} * \{I^2_E(k) \rho_E(i,k) (z_k - z_{k-1/2}) \pi(r^2_{i+1/2} - r^2_i) + q(i,k) (z_k - z_{k-1/2}) \pi(r^2_i - r^2_{i-1/2}) + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i+1/2,k} A_1 /(r_{i+1} - r_i) + K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k-1/2} A_2 /(z_k - z_{k-1})]\}^* \{I^2_E(k) \rho_E(i,k) (z_k - z_{k-1/2}) \pi(r^2_{i+1/2} - r^2_i) + q(i,k) (z_k - z_{k-1/2}) \pi(r^2_i - r^2_{i-1/2}) + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1}\} - T_{i,k} .$$



$$A_1 = 2\ \pi\ r_{i+1/2}\ (z_k - z_{k-1/2})$$
$$A_2 = \pi\ (r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\ \pi\ r_{i-1/2}\ (z_k - z_{k-1/2})$$

Figure 3.10 Energy balance for mesh points located at the top of the cell and at the emitter/fuel interface

3.3.2.8 Emitter/Fuel Interface - Bottom of the Cell

Figure 3.11 illustrates the energy balance utilized for elements located at the bottom of the cell and emitter/fuel interface. These elements include the assumption that the bottom cell surface is adiabatic. Balancing the "heat in" and "heat out" terms yields the following equation:

$$I^2_E(k)\,\rho_E(i,k)\,(z_{k+1/2} - z_k)\,\pi(r^2_{i+1/2} - r^2_i) + q(i,k)\,(z_{k+1/2} - z_k)\,\pi(r^2_i - r^2_{i-1/2}) = K_{i+1/2,k}\,A_1\,(T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i-1/2,k}\,A_3\,(T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k+1/2}\,A_2\,(T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k)$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i+1/2,k}\,A_1/(r_{i+1} - r_i) + K_{i-1/2,k}\,A_3/(r_i - r_{i-1}) + K_{i,k+1/2}\,A_2/(z_{k+1} - z_k)]\} * \{I^2_E(k)\,\rho_E(i,k)\,(z_{k+1/2} - z_k)\,\pi(r^2_{i+1/2} - r^2_i) + q(i,k)\,(z_{k+1/2} - z_k)\,\pi(r^2_i - r^2_{i-1/2}) + [K_{i+1/2,k}\,A_1/(r_{i+1} - r_i)]\,T_{i+1,k} + [K_{i-1/2,k}\,A_3/(r_i - r_{i-1})]\,T_{i-1,k} + [K_{i,k+1/2}\,A_2/(z_{k+1} - z_k)]\,T_{i,k+1}\}$$

which in iterative form becomes

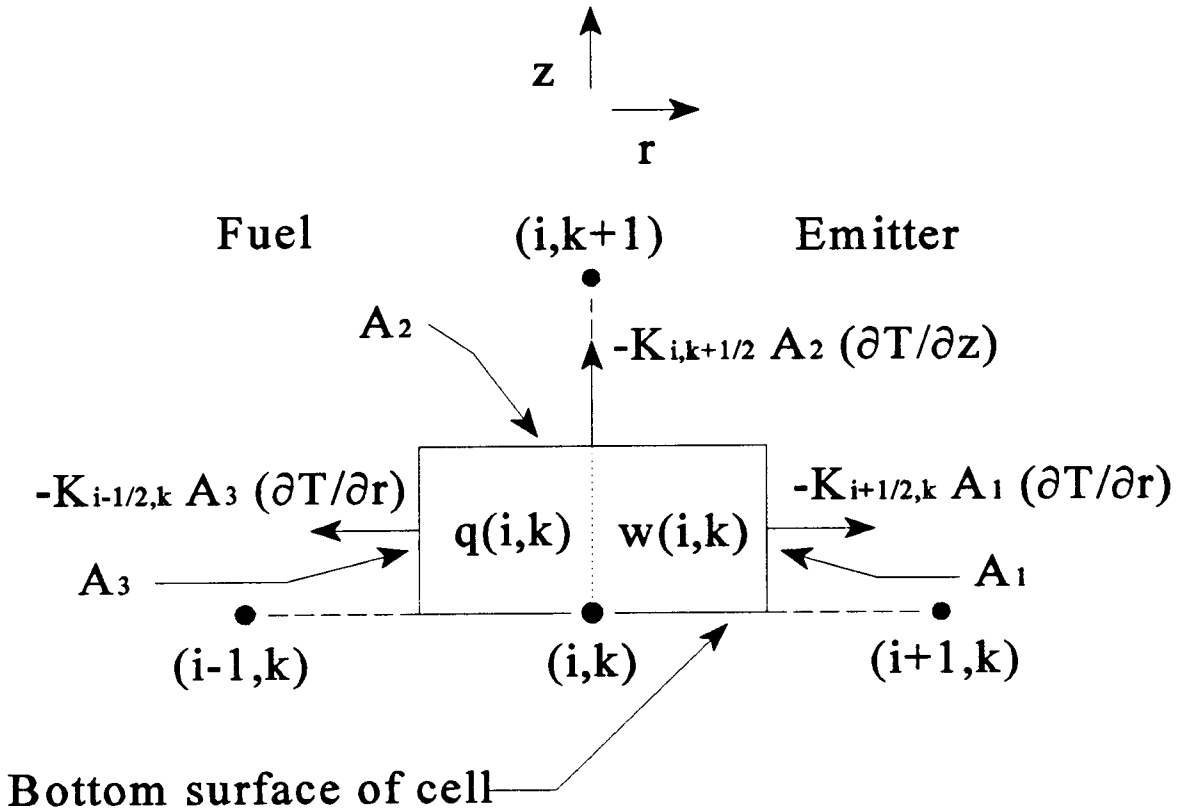
$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i+1/2,k}\,A_1/(r_{i+1} - r_i) + K_{i-1/2,k}\,A_3/(r_i - r_{i-1}) + K_{i,k+1/2}\,A_2/(z_{k+1} - z_k)]\} * \{I^2_E(k)\,\rho_E(i,k)\,(z_{k+1/2} - z_k)\,\pi(r^2_{i+1/2} - r^2_i) + q(i,k)\,(z_{k+1/2} - z_k)\,\pi(r^2_i - r^2_{i-1/2}) + [K_{i+1/2,k}\,A_1/(r_{i+1} - r_i)]\,T_{i+1,k} + [K_{i-1/2,k}\,A_3/(r_i - r_{i-1})]\,T_{i-1,k} + [K_{i,k+1/2}\,A_2/(z_{k+1} - z_k)]\,T_{i,k+1}\} - T_{i,k}\,.$$

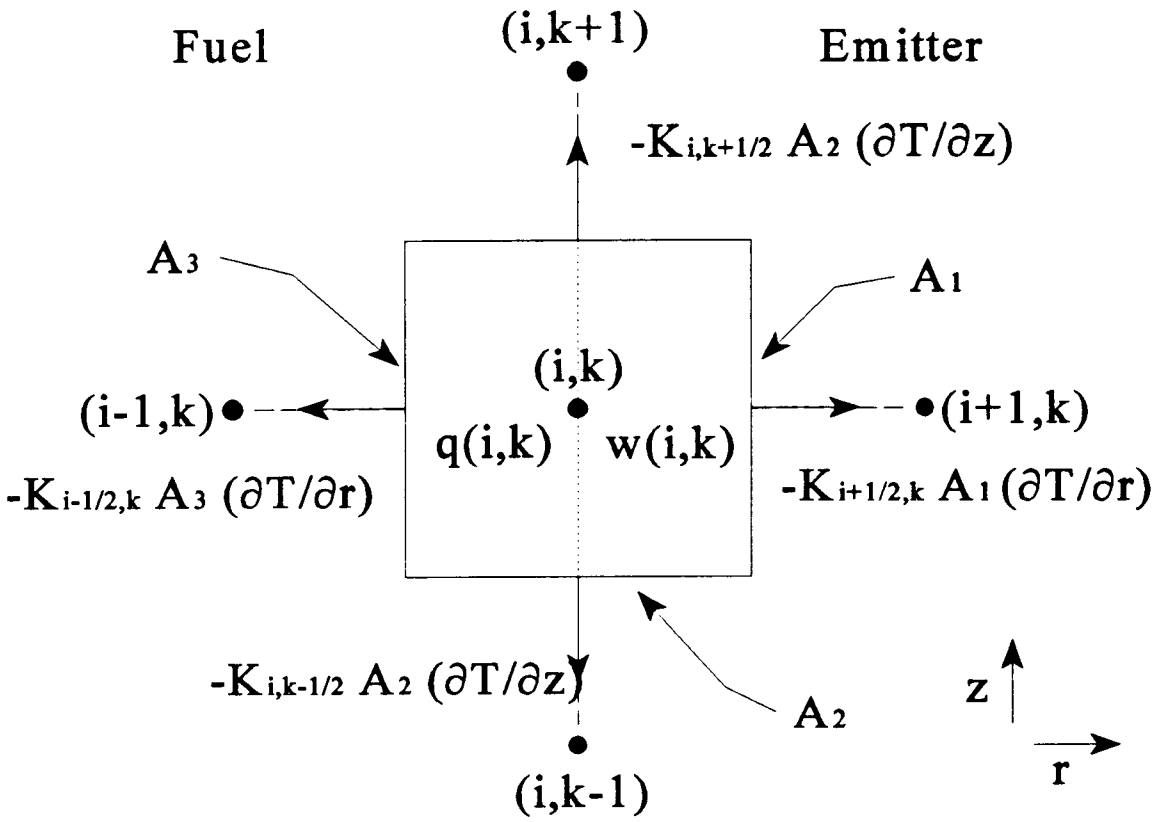
3.3.2.9 Emitter/Fuel Interface - Interior of the Cell

Figure 3.12 illustrates the energy balance utilized for elements located within the cell and at the emitter/fuel interface. Balancing the "heat in" and "heat out" terms yields the following equation:



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_{k+1/2} - z_k)$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_{k+1/2} - z_k)$$

Figure 3.11 Energy balance for mesh points located at the bottom of the cell and at the emitter/fuel interface



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_{k+1/2} - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_{k+1/2} - z_{k-1/2})$$

Figure 3.12 Energy balance for mesh points located within the cell and at the emitter/fuel interface

$$I^2_E(k)\,\rho_E(i,k)\,(z_{k+1/2} - z_{k-1/2})\,\pi(r^2_{i+1/2} - r^2_i) + q(i,k)\,(z_{k+1/2} - z_{k-1/2})\,\pi(r^2_i - r^2_{i-1/2}) = K_{i+1/2,k}\,A_1 \cdot (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i-1/2,k}\,A_3\,(T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k+1/2}\,A_2\,(T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k) + K_{i,k-1/2}\,A_2\,(T_{i,k} - T_{i,k-1})/(z_k - z_{k-1})$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i+1/2,k}\,A_1/(r_{i+1} - r_i) + K_{i-1/2,k}\,A_3/(r_i - r_{i-1}) + K_{i,k+1/2}\,A_2/(z_{k+1} - z_k) + K_{i,k-1/2}\,A_2/(z_k - z_{k-1})]\} * \{I^2_E(k)\,\rho_E(i,k)\,(z_{k+1/2} - z_{k-1/2})\,\pi(r^2_{i+1/2} - r^2_i) + q(i,k)\,(z_{k+1/2} - z_{k-1/2})\,\pi(r^2_i - r^2_{i-1/2}) + [K_{i+1/2,k}\,A_1/(r_{i+1} - r_i)]\,T_{i+1,k} + [K_{i-1/2,k}\,A_3/(r_i - r_{i-1})]\,T_{i-1,k} + [K_{i,k+1/2}\,A_2/(z_{k+1} - z_k)]\,T_{i,k+1} + [K_{i,k-1/2}\,A_2/(z_{k+1} - z_k)]\,T_{i,k-1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i+1/2,k}\,A_1/(r_{i+1} - r_i) + K_{i-1/2,k}\,A_3/(r_i - r_{i-1}) + K_{i,k+1/2}\,A_2/(z_{k+1} - z_k) + K_{i,k-1/2}\,A_2/(z_k - z_{k-1})]\} * \{I^2_E(k)\,\rho_E(i,k)\,(z_{k+1/2} - z_{k-1/2})\,\pi(r^2_{i+1/2} - r^2_i) + q(i,k)\,(z_{k+1/2} - z_{k-1/2})\,\pi(r^2_i - r^2_{i-1/2}) + [K_{i+1/2,k}\,A_1/(r_{i+1} - r_i)]\,T_{i+1,k} + [K_{i-1/2,k}\,A_3/(r_i - r_{i-1})]\,T_{i-1,k} + [K_{i,k+1/2}\,A_2/(z_{k+1} - z_k)]\,T_{i,k+1} + [K_{i,k-1/2}\,A_2/(z_{k+1} - z_k)]\,T_{i,k-1}\} - T_{i,k}\,.$$

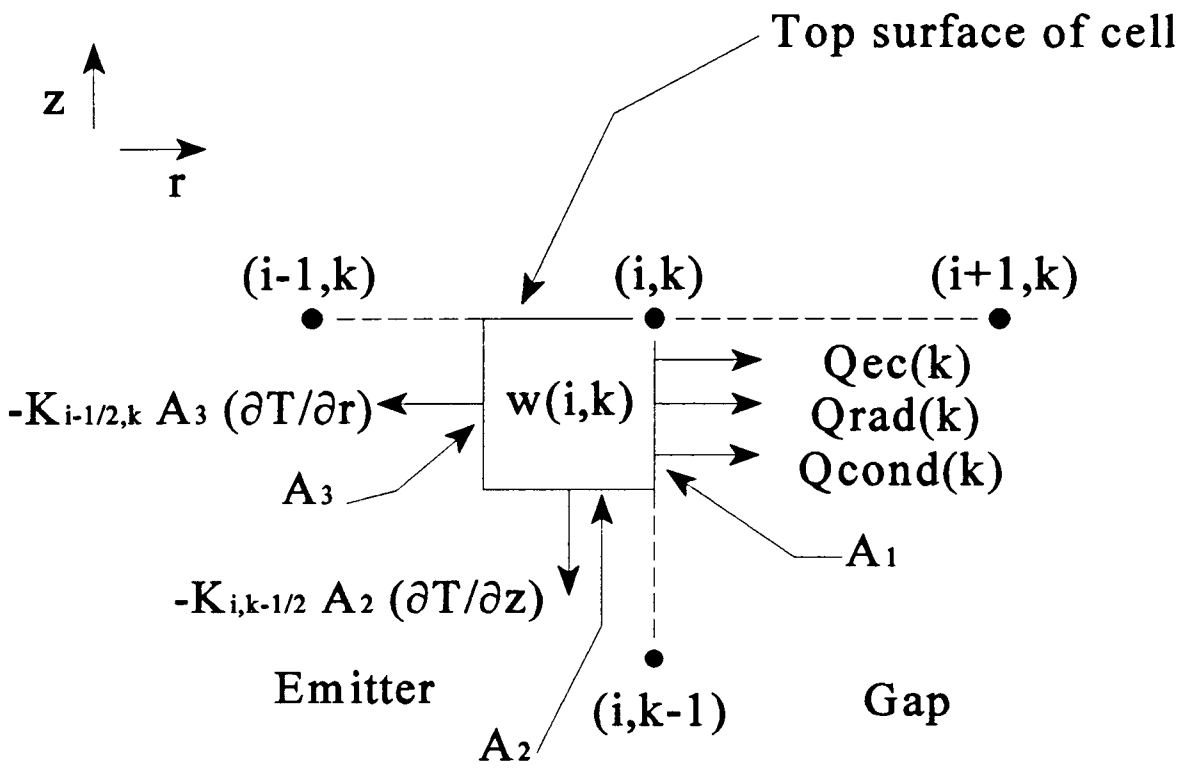
3.3.2.10 Emitter/Gap Interface - General Considerations

Figures 3.13 - 3.15 illustrates the energy balance utilized for the elements located at the emitter/gap interface. As shown by these figures, energy is transferred away from the surface of the emitter by the following modes:

1) Thermal conduction through the cesium vapor within the emitter/collector gap, Qcond (Watts/cm^2).

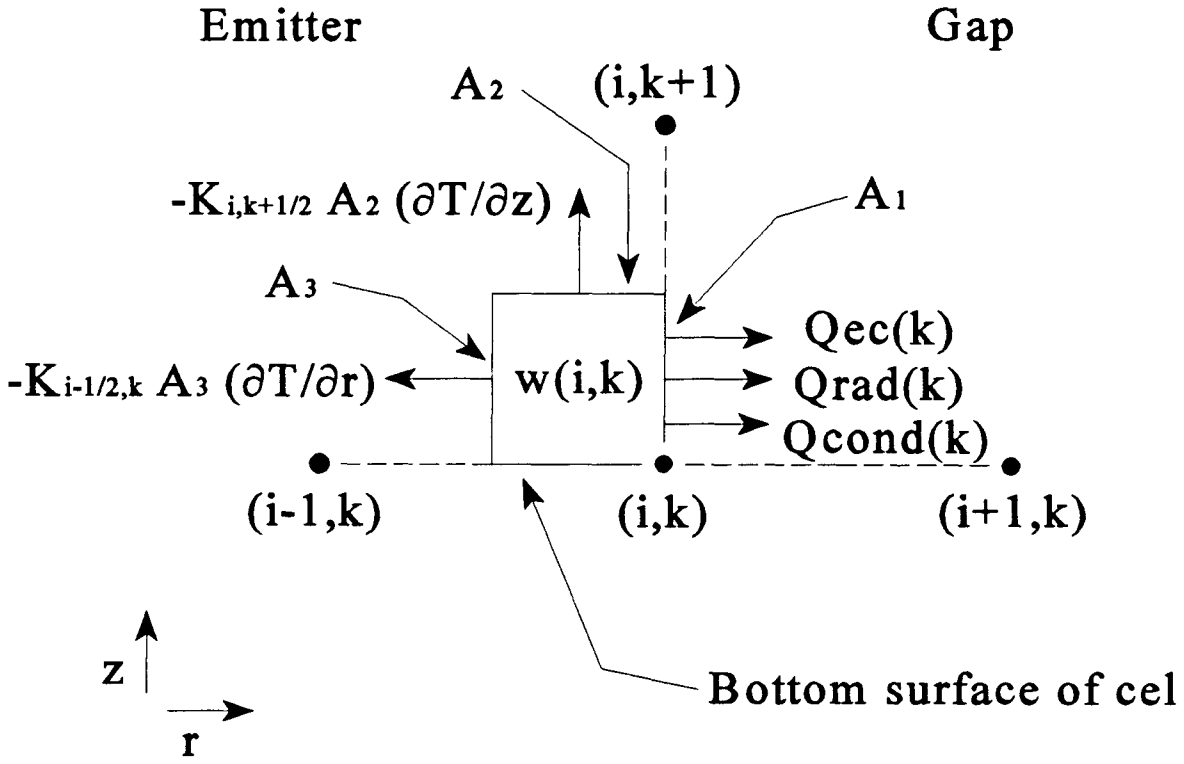
2) Thermal radiation between the emitter and the collector, Qrad (Watts/cm^2).

3) Net energy carried away by the electrons leaving the surface of the emitter, commonly called the electron cooling, Qec (Watts/cm^2).



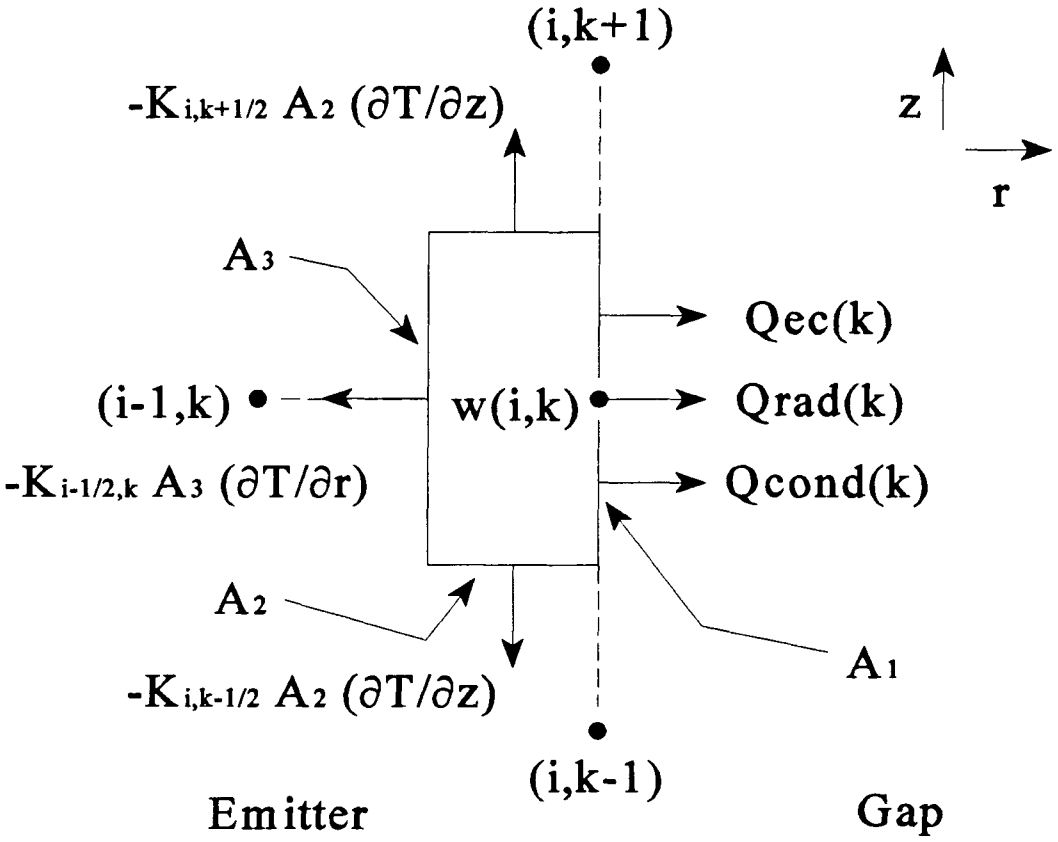
$$A_1 = 2\,\pi\, r_i\,(z_k - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_i - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_k - z_{k-1/2})$$

Figure 3.13 Energy balance for mesh points located at the top of the cell and at the emitter/gap interface



$$A_1 = 2\,\pi\,r_i\,(z_{k+1/2} - z_k)$$
$$A_2 = \pi\,(r^2_i - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\,r_{i-1/2}\,(z_{k+1/2} - z_k)$$

Figure 3.14 Energy balance for mesh points located at the bottom of the cell and at the emitter/gap interface



$$A_1 = 2\,\pi\, r_i\,(z_{k+1/2} - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_i - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_{k+1/2} - z_{k-1/2})$$

Figure 3.15 Energy Balance for mesh points located in the interrior of the cell and at the emitter/gap interface

The value of Qcond is computed using the correlation of Kitrilakis and Meeker [20]:

$$Qcond(k) = [\ K_{CS}\ (T_{E,k} - T_{C,k})\ ]\ /\ [\ d + 1.15E\text{-}5\ (T_{E,k} - T_{C,k})\ /\ Pcs\ ]$$

where $T_{E,k}$ is the temperature of the emitter, $T_{C,k}$ is the temperature of the collector, K_{CS} is the thermal conductivity of the cesium vapor (computed by the KCOND function), and Pcs is the pressure of the cesium vapor. The inclusion of cesium as a material type in the KCOND function is a modification made for MCTFE. A correlation for the cesium pressure (Pcs) is also used and is given as follows [7]:

$$Pcs = 2.45\ E\ 8\ \exp(8910\ /\ T_r)\ /\ SQRT\ (T_r)$$

where T_r is the cesium reservoir temperature.

The value of Qrad is determined by the standard thermal radiation equation:

$$Qrad(k) = \sigma\ \epsilon\ \ (T^4_{E,k} - T^4_{C,k})$$

where σ is the Stefan-Boltzmann constant (5.67E-12 Watts/cm^{2}K^4), and ϵ is the thermal emissivity of the emitter surface.

The electron cooling term, Qec, is computed using the TECMDL computer code [21,22], which is a subroutine of the function CDEN. This routine was developed for planar converters (parallel plates) and is assumed to be valid for cylindrical TFE's that typically involve a small inner-electrode gap to TFE radius ratio. The electron cooling term, as previously mentioned, is the net energy leaving the emitter surface due to the thermionic emission process. This term involves the energy carried away from the emitter surface by the "evaporating" electrons minus the portion of the cesium vapor de-excitation energy which returns to the emitter.

It is assumed that there is insignificant heat transfer occurring between emitter elements at one axial position and collector elements at a different axial position and vice versa. Therefore, each of the inner-electrode or gap heat transfer terms described above are calculated only for elements at the same axial position.

3.3.2.11 Emitter/Gap interface - Top of the Cell

Figure 3.13 illustrates the energy balance utilized for elements located at the top of the cell and at the emitter/gap interface. These elements include the assumption that the top surface of the cell is adiabatic. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$I^2_E(k)\ \rho_E(i,k)\ (z_k - z_{k-1/2}) / A_2 = K_{i-1/2,k}\ A_3\ (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k-1/2}\ A_2\ (T_{i,k} - T_{i,k-1}) /(z_k - z_{k-1}) + [Qcond(k) + Qrad(k) + Qec(k)]\ A_1$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1}) + K_{i,k-1/2}\ A_2 /(z_k - z_{k-1})]\} * \{I^2_E(k)\ \rho_E(i,k)\ (z_k - z_{k-1/2}) / A_2 - [Qcond(k) + Qrad(k) + Qec(k)]\ A_1 + [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1})]\ T_{i-1,k} + [K_{i,k-1/2}\ A_2 /(z_k - z_{k-1})]\ T_{i,k-1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1}) + K_{i,k-1/2}\ A_2 /(z_k - z_{k-1})]\} * \{I^2_E(k)\ \rho_E(i,k)\ (z_k - z_{k-1/2}) / A_2 - [Qcond(k) + Qrad(k) + Qec(k)]\ A_1 + [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1})]\ T_{i-1,k} + [K_{i,k-1/2}\ A_2 /(z_k - z_{k-1})]\ T_{i,k-1}\} - T_{i,k}\ .$$

3.3.2.12 Emitter/Gap Interface - Bottom of the Cell

Figure 3.14 illustrates the energy balance utilized for elements located at the bottom of the cell and at the emitter/gap interface. These elements include the assumption that the bottom surface of the cell is adiabatic. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$I^2_E(k)\ \rho_E(i,k)\ (z_{k+1/2} - z_k) / A_2 = K_{i-1/2,k}\ A_3\ (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k+1/2}\ A_2\ (T_{i,k} - T_{i,k+1})$$
$$/(z_{k+1} - z_k) + [Qcond(k) + Qrad(k) + Qec(k)]\ A_1$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1}) + K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k)]\} * \{I^2_E(k)\ \rho_E(i,k)\ (z_{k+1/2} - z_k) / A_2 -$$
$$[Qcond(k) + Qrad(k) + Qec(k)]\ A_1 + [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1})]\ T_{i-1,k} +$$
$$[K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k)]\ T_{i,k+1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1}) + K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k)]\} * \{I^2_E(k)\ \rho_E(i,k)\ (z_{k+1/2} - z_k) / A_2 -$$
$$[Qcond(k) + Qrad(k) + Qec(k)]\ A_1 + [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1})]\ T_{i-1,k} +$$
$$[K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k)]\ T_{i,k+1}\} - T_{i,k}\ .$$

3.3.2.13 Emitter/Gap Interface - Interior of the Cell

Figure 3.15 illustrates the energy balance utilized for elements located within the cell and at the emitter/gap interface. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$I^2_E(k)\ \rho_E(i,k)\ (z_{k+1/2} - z_{k-1/2}) / A_2 = K_{i-1/2,k}\ A_3\ (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k+1/2}\ A_2\ (T_{i,k} - T_{i,k+1})$$
$$/(z_{k+1} - z_k) + K_{i,k-1/2}\ A_2\ (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1}) + [Qcond(k) + Qrad(k) + Qec(k)]\ A_1$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1}) + K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k) + K_{i,k-1/2}\ A_2 /(z_k - z_{k-1})]\} * \{I^2_E(k)$$
$$\rho_E(i,k)\ (z_{k+1/2} - z_{k-1/2}) / A_2 - [Qcond(k) + Qrad(k) + Qec(k)]\ A_1 + [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1})]$$
$$T_{i-1,k} + [K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k)]\ T_{i,k+1} + [K_{i,k-1/2}\ A_2 /(z_k - z_{k-1})]\ T_{i,k-1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is

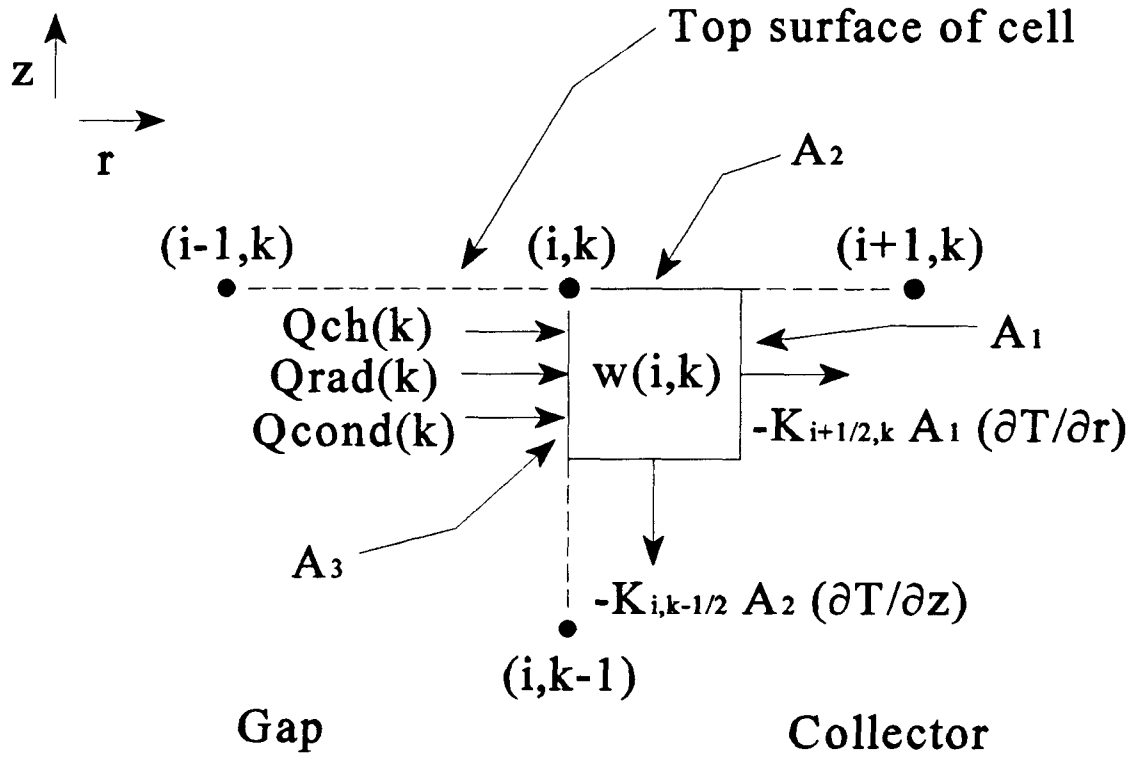
$$R_{i,k} = \{1 / [K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i,k-1/2} A_2 /(z_k - z_{k-1})]\} * \{I^2_E(k) \rho_E(i,k) (z_{k+1/2} - z_{k-1/2}) / A_2 - [Qcond(k) + Qrad(k) + Qec(k)] A_1 + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1}\} - T_{i,k}.$$

3.3.2.14 Collector/Gap Interface - General Considerations

Figures 3.16 - 3.18 illustrate the energy balance utilized for elements located at the collector/gap interface. As can be seen from these figures, there are three possible means that energy is transferred into the collector surface from the gap region. The radiation term, Qrad, and the conduction term, Qcond, are as described previously for emitter/gap surface elements. However, the electron cooling term, Qec, that leaves the emitter surface is replaced with Qch (representing collector heating) for the term entering the collector. In reality, all energy that leaves the emitter must enter the collector. However, because some of the electron cooling energy is converted to electrical potential energy within the gap (as it travels against an electrical potential) that is ultimately absorbed by the electrical load, it is considered to never have entered the collector. Of course, some of this potential energy will remain in the collector or be redistributed back to the emitter as electrical joule heating, but these "heat in" terms are already accounted for by the w(i,k) terms within each affected element. Accordingly, the Qch term is calculated by the following equation:

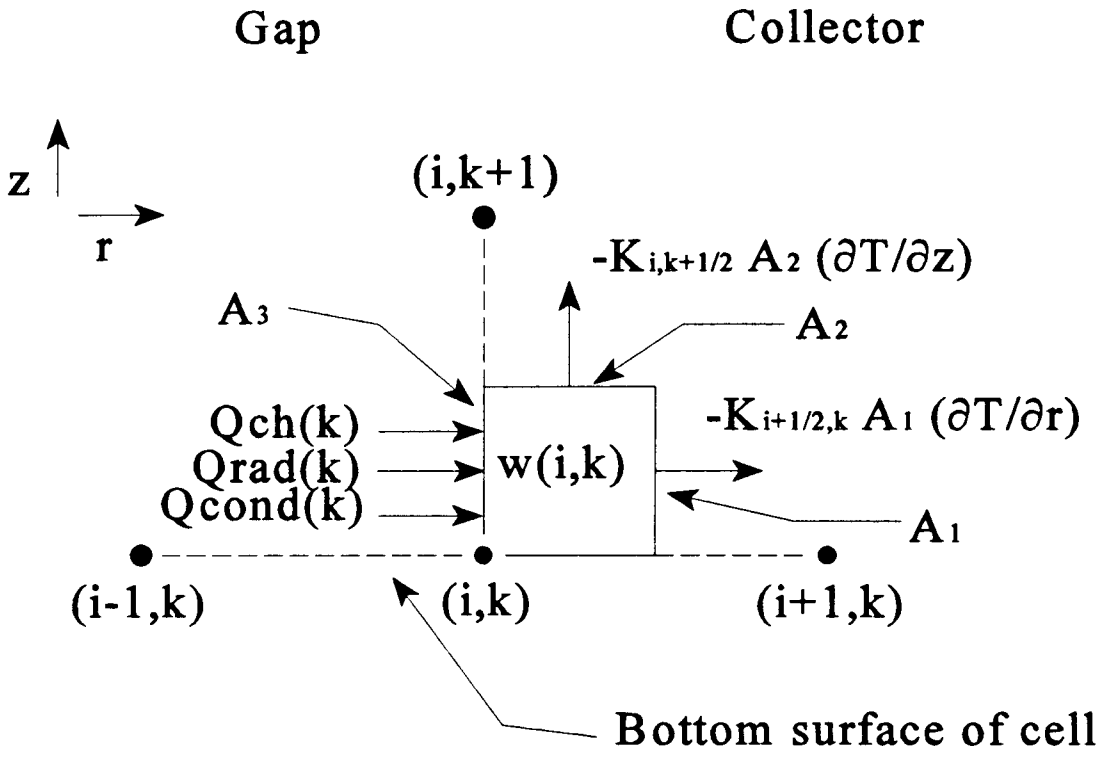
$$Qch(k) = Qec(k) - Jden(k) * V(k)$$

where Qec(k) is as given before, Jden(k) is the net current density from the emitter to the collector at a given axial position, and V(k) is the inner-electrode voltage that exists at that position.



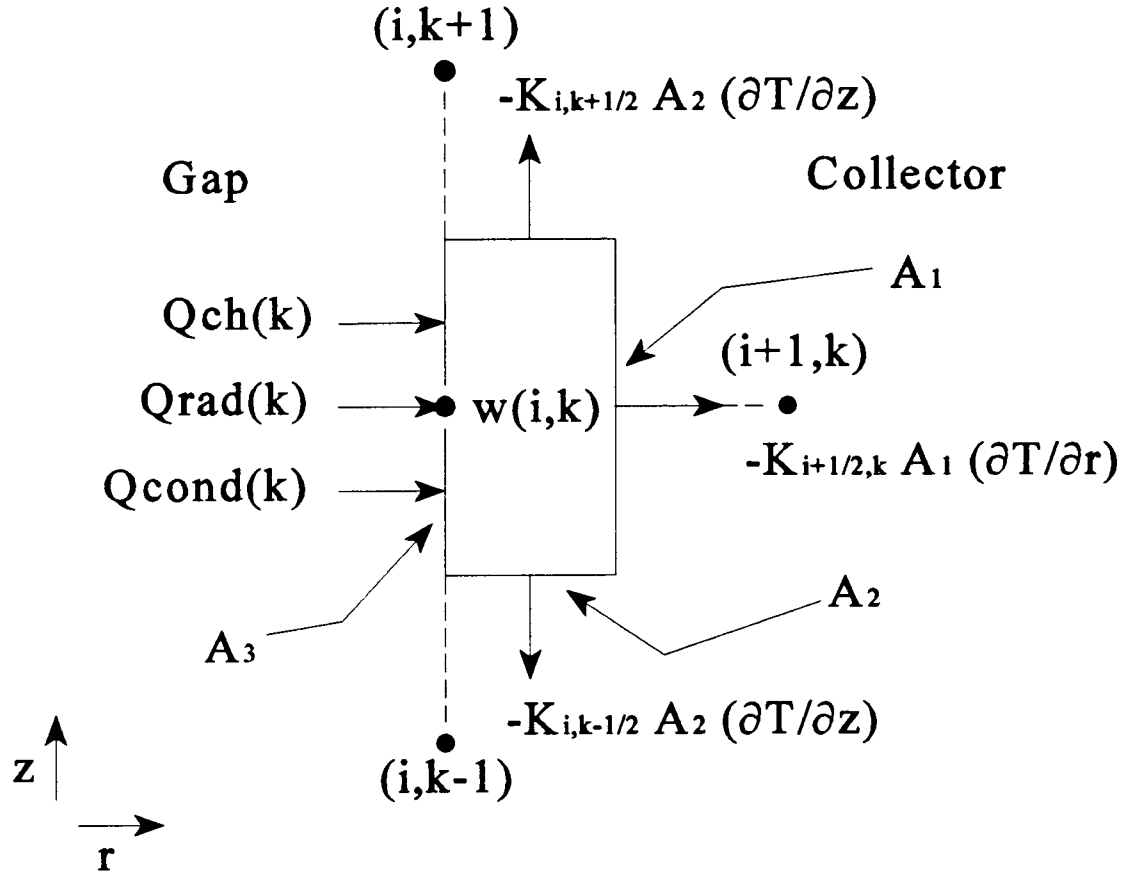
$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_k - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_i)$$
$$A_3 = 2\,\pi\, r_i\,(z_k - z_{k-1/2})$$

Figure 3.16 Energy balance for mesh points located at the top of the cell and at the collector/gap interface



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_{k+1/2} - z_k)$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_i)$$
$$A_3 = 2\,\pi\, r_i\,(z_{k+1/2} - z_k)$$

Figure 3.17 Energy balance for mesh points located at the bottom of the cell and at the collector/gap interface



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_{k+1/2} - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_i)$$
$$A_3 = 2\,\pi\, r_i\,(z_{k+1/2} - z_{k-1/2})$$

Figure 3.18 Energy balance for mesh points located within the cell interior and at the collector/gap interface

3.3.2.15 Collector/Gap Interface - Top of the Cell

Figure 3.16 illustrates the energy balance utilized for elements located at the top of the cell and at the collector/gap interface. These elements include the assumption that the top surface of the cell is adiabatic. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$I^2_C(k)\ \rho_C(i,k)\ (z_k - z_{k-1/2}) / A_2 + [Qcond(k) + Qrad(k) + Qch(k)]\ A_3 =$$
$$K_{i+1/2,k}\ A_1\ (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i,k-1/2}\ A_2\ (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1})$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i) + K_{i,k-1/2}\ A_2 /(z_k - z_{k-1})]\} * \{I^2_C(k)\ \rho_C(i,k)\ (z_k - z_{k-1/2}) / A_2 +$$
$$[Qcond(k) + Qrad(k) + Qec(k)]\ A_3 + [K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i)]\ T_{i+1,k} +$$
$$[K_{i,k-1/2}\ A_2 /(z_k - z_{k-1})]\ T_{i,k-1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i) + K_{i,k-1/2}\ A_2 /(z_k - z_{k-1})]\} * \{I^2_C(k)\ \rho_C(i,k)\ (z_k - z_{k-1/2}) / A_2 +$$
$$[Qcond(k) + Qrad(k) + Qec(k)]\ A_3 + [K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i)]\ T_{i+1,k} +$$
$$[K_{i,k-1/2}\ A_2 /(z_k - z_{k-1})]\ T_{i,k-1}\} - T_{i,k}\ .$$

3.3.2.16 Collector/Gap Interface - Bottom of the Cell

Figure 3.17 illustrates the energy balance utilized for elements located at the bottom of the cell and at the collector/gap interface. These elements include the assumption that the bottom surface of the cell is adiabatic. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$I^2_C(k)\ \rho_C(i,k)\ (z_{k+1/2} - z_k) / A_2 + [Qcond(k) + Qrad(k) + Qch(k)]\ A_3 =$$
$$K_{i+1/2,k}\ A_1\ (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i,k+1/2}\ A_2\ (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k)$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i) + K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k)]\} * \{I^2_C(k)\ \rho_C(i,k)\ (z_{k+1/2} - z_k) / A_2 +$$
$$[Qcond(k) + Qrad(k) + Qec(k)]\ A_3 + [K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i)]\ T_{i+1,k} +$$
$$[K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k)]\ T_{i,k+1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i) + K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k)]\} * \{I^2_C(k)\ \rho_C(i,k)\ (z_{k+1/2} - z_k) / A_2 +$$
$$[Qcond(k) + Qrad(k) + Qec(k)]\ A_3 + [K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i)]\ T_{i+1,k} +$$
$$[K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k)]\ T_{i,k+1}\} - T_{i,k}\ .$$

3.3.2.17 Collector/Gap Interface - Interior of the Cell

Figure 3.18 illustrates the energy balance utilized for elements located within the cell and at the collector/gap interface. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$I^2_C(k)\ \rho_C(i,k)\ (z_{k+1/2} - z_{k-1/2}) / A_2 + [Qcond(k) + Qrad(k) + Qch(k)]\ A_3 =$$
$$K_{i+1/2,k}\ A_1\ (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i,k+1/2}\ A_2\ (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k) +$$
$$K_{i,k-1/2}\ A_2\ (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1})$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i) + K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k) + K_{i,k-1/2}\ A_2 /(z_k - z_{k-1})]\} * \{I^2_C(k)$$
$$\rho_C(i,k)\ (z_{k+1/2} - z_{k-1/2}) / A_2 + [Qcond(k) + Qrad(k) + Qec(k)]\ A_3 + [K_{i+1/2,k}\ A_1 /$$
$$(r_{i+1} - r_i)]\ T_{i+1,k} + [K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k)]\ T_{i,k+1} + [K_{i,k-1/2}\ A_2 /(z_k - z_{k-1})]\ T_{i,k-1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i+1/2,k} A_1 /(r_{i+1} - r_i) + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i,k-1/2} A_2 /(z_k - z_{k-1})]\} * \{I^2_C(k) \rho_C(i,k) (z_{k+1/2} - z_{k-1/2}) / A_2 + [Qcond(k) + Qrad(k) + Qec(k)] A_3 + [K_{i+1/2,k} A_1 / (r_{i+1} - r_i)] T_{i+1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1}\} - T_{i,k}.$$

3.3.2.18 Collector/Insulation Interface - Top of the Cell

Figure 3.19 illustrates the energy balance utilized for elements located at the top of the cell and at the collector/insulation interface. These elements include the assumption that the top surface of the cell is adiabatic. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$I^2_C(k) \rho_C(i,k) (z_k - z_{k-1/2}) / \pi (r^2_i - r^2_{i-1/2}) = K_{i-1/2,k} A_3 (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k-1/2} A_2 (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1}) + K_{i+1/2,k} A_1 (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i)$$

that can be solved for $T_{i,k}$ to yield

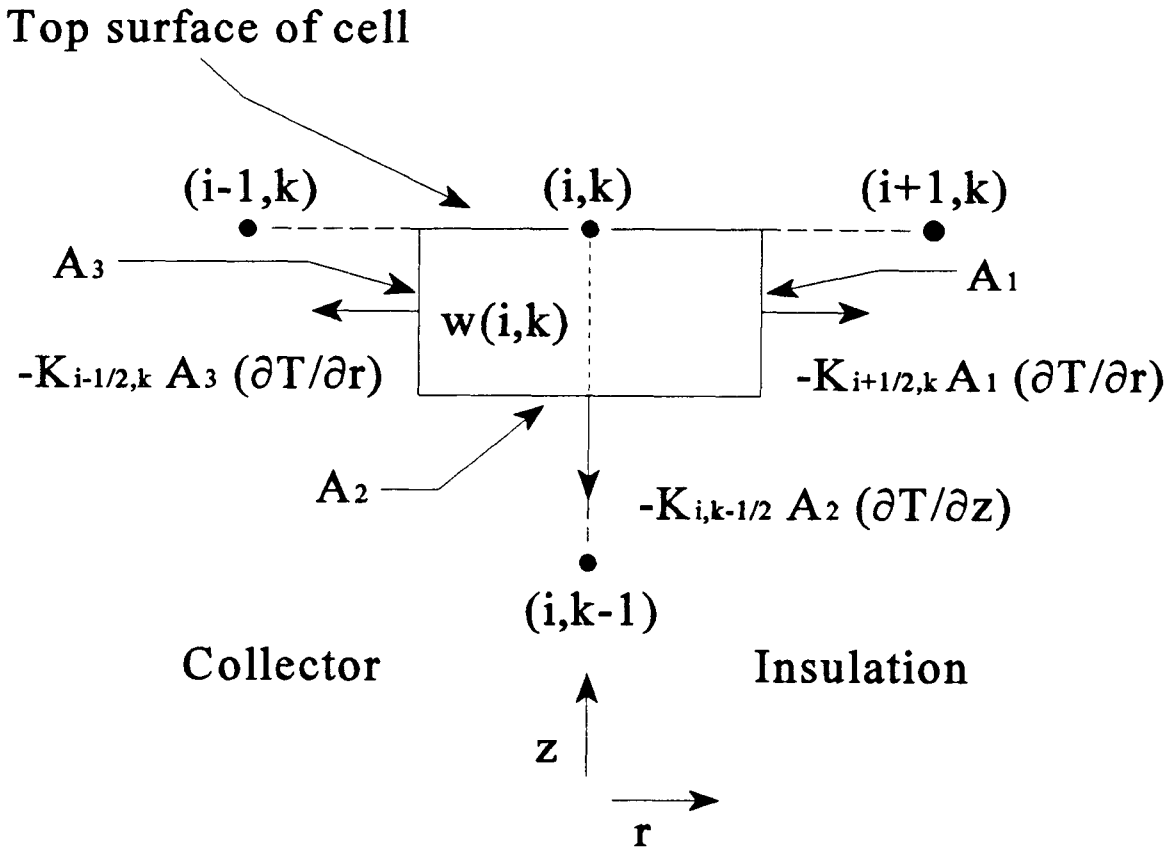
$$T_{i,k} = \{1 / [K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k-1/2} A_2 /(z_k - z_{k-1}) + K_{i+1/2,k} A_1 /(r_{i+1} - r_i)]\} * \{I^2_C(k) \rho_C(i,k) (z_k - z_{k-1/2}) / \pi (r^2_i - r^2_{i-1/2}) + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1} + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k-1/2} A_2 /(z_k - z_{k-1}) + K_{i+1/2,k} A_1 /(r_{i+1} - r_i)]\} * \{I^2_C(k) \rho_C(i,k) (z_k - z_{k-1/2}) / \pi (r^2_i - r^2_{i-1/2}) + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1} + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k}\} - T_{i,k}.$$



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_k - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_k - z_{k-1/2})$$

Figure 3.19 Energy balance for mesh points located at the top of the cell and at the collector/insulation interface

3.3.2.19 Collector/Insulation Interface - Bottom of the Cell

Figure 3.20 illustrates the energy balance utilized for elements located at the bottom of the cell and at the collector/insulation interface. These elements include the assumption that the bottom surface of the cell is adiabatic. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$I^2_C(k)\ \rho_C(i,k)\ (z_{k+1/2} - z_k) / \pi\ (r^2_i - r^2_{i-1/2}) = K_{i-1/2,k}\ A_3\ (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) +$$
$$K_{i,k+1/2}\ A_2\ (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k) + K_{i+1/2,k}\ A_1\ (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i)$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1}) + K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k) + K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i)]\} * \{I^2_C(k)$$
$$\rho_C(i,k)\ (z_{k+1/2} - z_k) / \pi\ (r^2_i - r^2_{i-1/2}) + [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1})]\ T_{i-1,k} +$$
$$[K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k)]\ T_{i,k+1} + [K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i)]\ T_{i+1,k}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

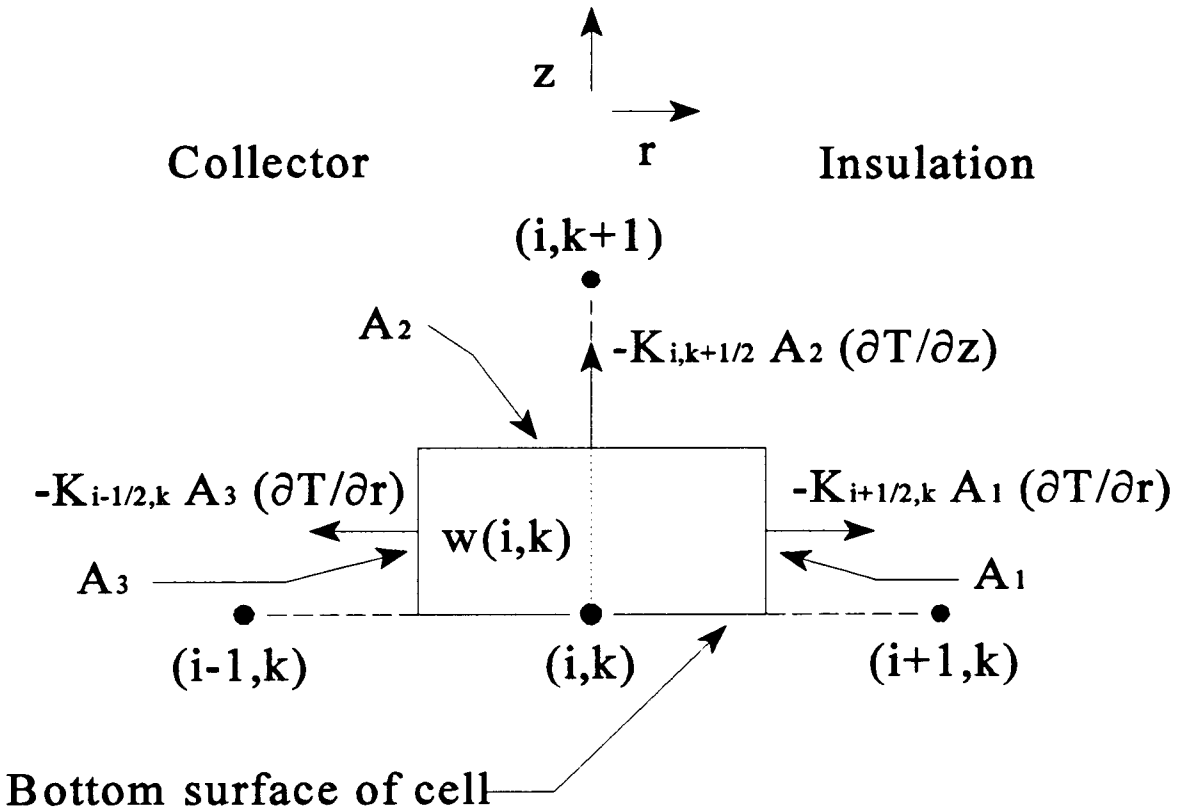
where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1}) + K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k) + K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i)]\} * \{I^2_C(k)$$
$$\rho_C(i,k)\ (z_{k+1/2} - z_k) / \pi\ (r^2_i - r^2_{i-1/2}) + [K_{i-1/2,k}\ A_3 /(r_i - r_{i-1})]\ T_{i-1,k} +$$
$$[K_{i,k+1/2}\ A_2 /(z_{k+1} - z_k)]\ T_{i,k+1} + [K_{i+1/2,k}\ A_1 /(r_{i+1} - r_i)]\ T_{i+1,k}\} - T_{i,k}\ .$$

3.3.2.20 Collector/Insulation Interface - Interior of the Cell

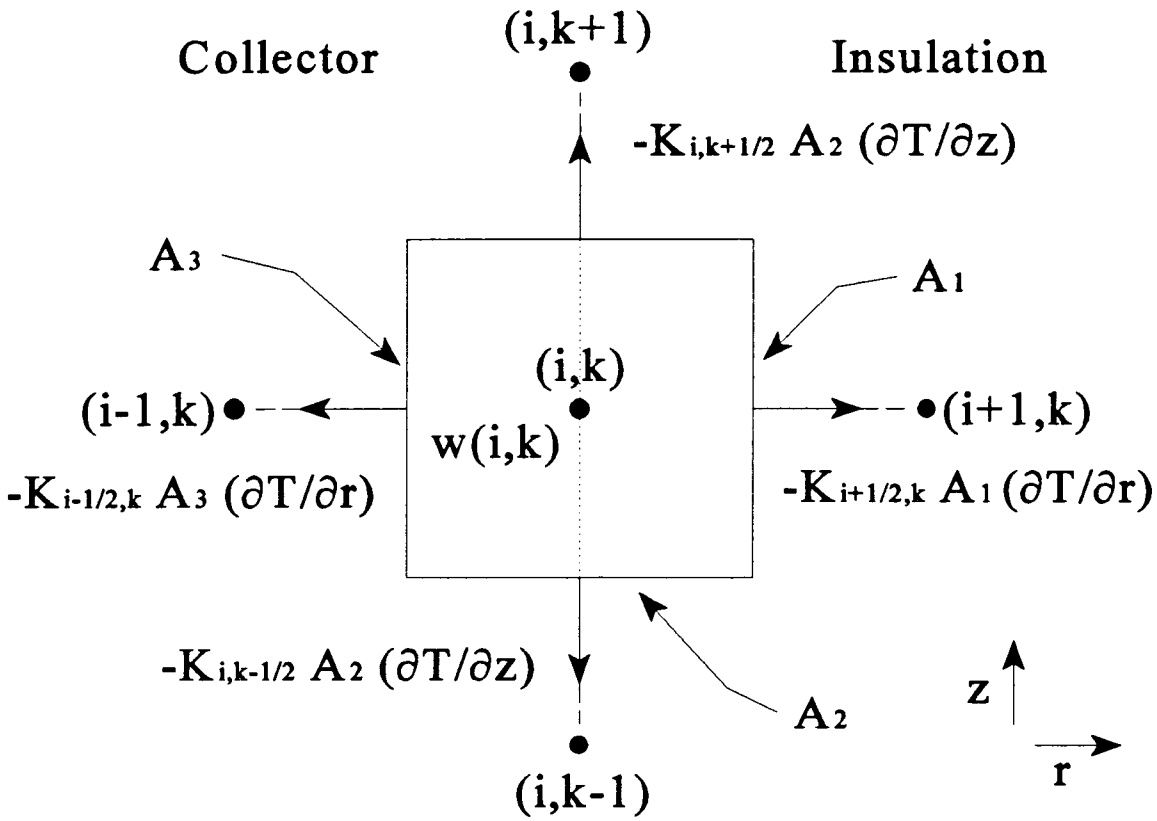
Figure 3.21 illustrates the energy balance utilized for elements located within the cell and at the collector/insulation interface. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$I^2_C(k)\ \rho_C(i,k)\ (z_{k+1/2} - z_{k-1/2}) / \pi\ (r^2_i - r^2_{i-1/2}) = K_{i-1/2,k}\ A_3\ (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) +$$
$$K_{i,k+1/2}\ A_2\ (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k) + K_{i+1/2,k}\ A_1\ (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) +$$
$$K_{i,k-1/2}\ A_2\ (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1})$$



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_{k+1/2} - z_k)$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_{k+1/2} - z_k)$$

Figure 3.20 Energy balance for mesh points located at the bottom of the cell and at the collector/insulation interface



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_{k+1/2} - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_{k+1/2} - z_{k-1/2})$$

Figure 3.21 Energy balance for mesh points located within the cell and at the collector/insulation interface

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i+1/2,k} A_1 /(r_{i+1} - r_i) + K_{i,k-1/2} A_2 /(z_k - z_{k-1})]\} * \{I^2_C(k) \rho_C(i,k) (z_{k+1/2} - z_{k-1/2}) / \pi (r^2_i - r^2_{i-1/2}) + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1}) T_{i,k-1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i+1/2,k} A_1 /(r_{i+1} - r_i) + K_{i,k-1/2} A_2 /(z_k - z_{k-1})]\} * \{I^2_C(k) \rho_C(i,k) (z_{k+1/2} - z_{k-1/2}) / \pi (r^2_i - r^2_{i-1/2}) + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1}) T_{i,k-1}\} - T_{i,k}.$$

3.3.2.21 Insulation/Cladding Interface - Top of the Cell

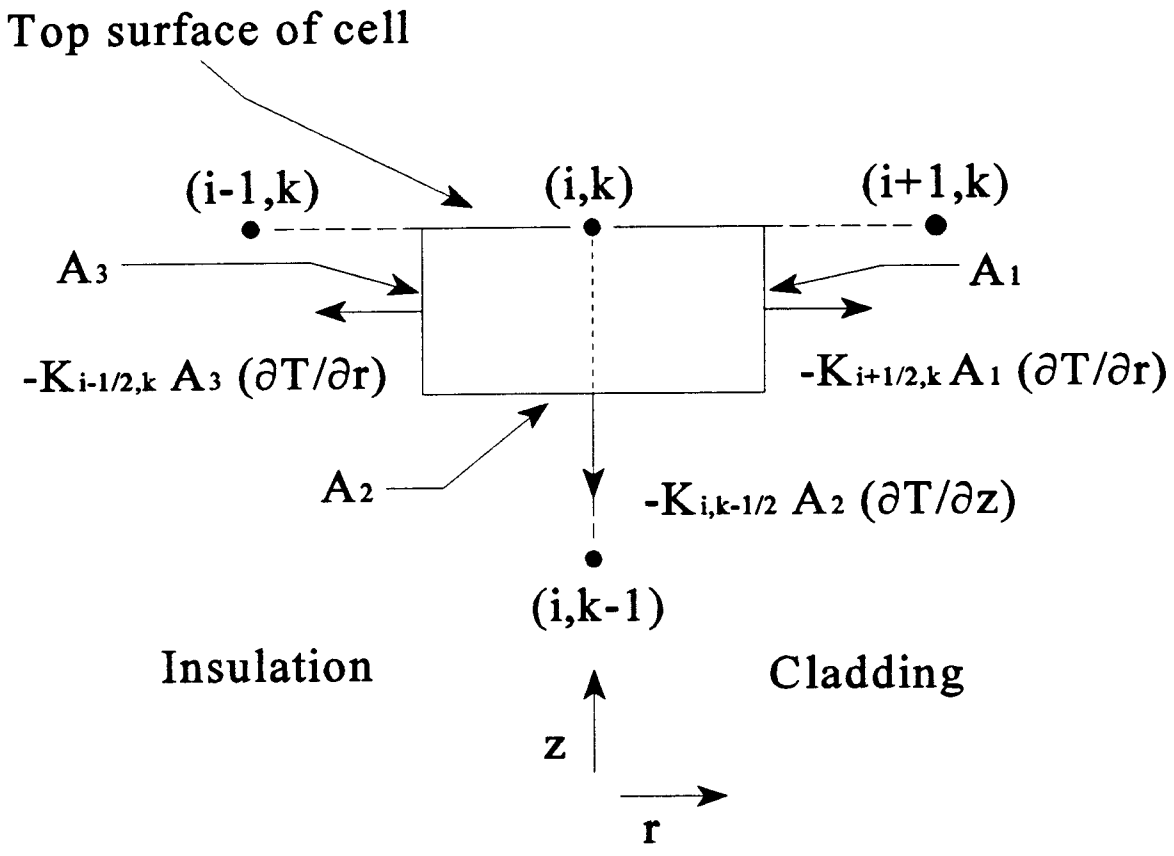
Figure 3.22 illustrates the energy balance utilized for elements located at the top of the cell and at the insulation/cladding interface. These elements include the assumption that the top surface of the cell is adiabatic. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$0 = K_{i-1/2,k} A_3 (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k-1/2} A_2 (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1}) + K_{i+1/2,k} A_1 (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i)$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k-1/2} A_2 /(z_k - z_{k-1}) + K_{i+1/2,k} A_1 /(r_{i+1} - r_i)]\} * \{[K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1} + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k}\}$$

which in iterative form becomes



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_k - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_k - z_{k-1/2})$$

Figure 3.22 Energy balance for mesh points located at the top of the cell and at the insulation/cladding interface

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k-1/2} A_2 /(z_k - z_{k-1}) + K_{i+1/2,k} A_1 /(r_{i+1} - r_i)]\} *$$
$$\{[K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1} + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k}\}$$
$$- T_{i,k} .$$

3.3.2.22 Insulation/Cladding Interface - Bottom of the Cell

Figure 3.23 illustrates the energy balance utilized for elements located at the bottom of the cell and at the insulation/cladding interface. These elements include the assumption that the bottom surface of the cell is adiabatic. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$0 = K_{i-1/2,k} A_3 (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k+1/2} A_2 (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k) +$$
$$K_{i+1/2,k} A_1 (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i)$$

that can be solved for $T_{i,k}$ to yield

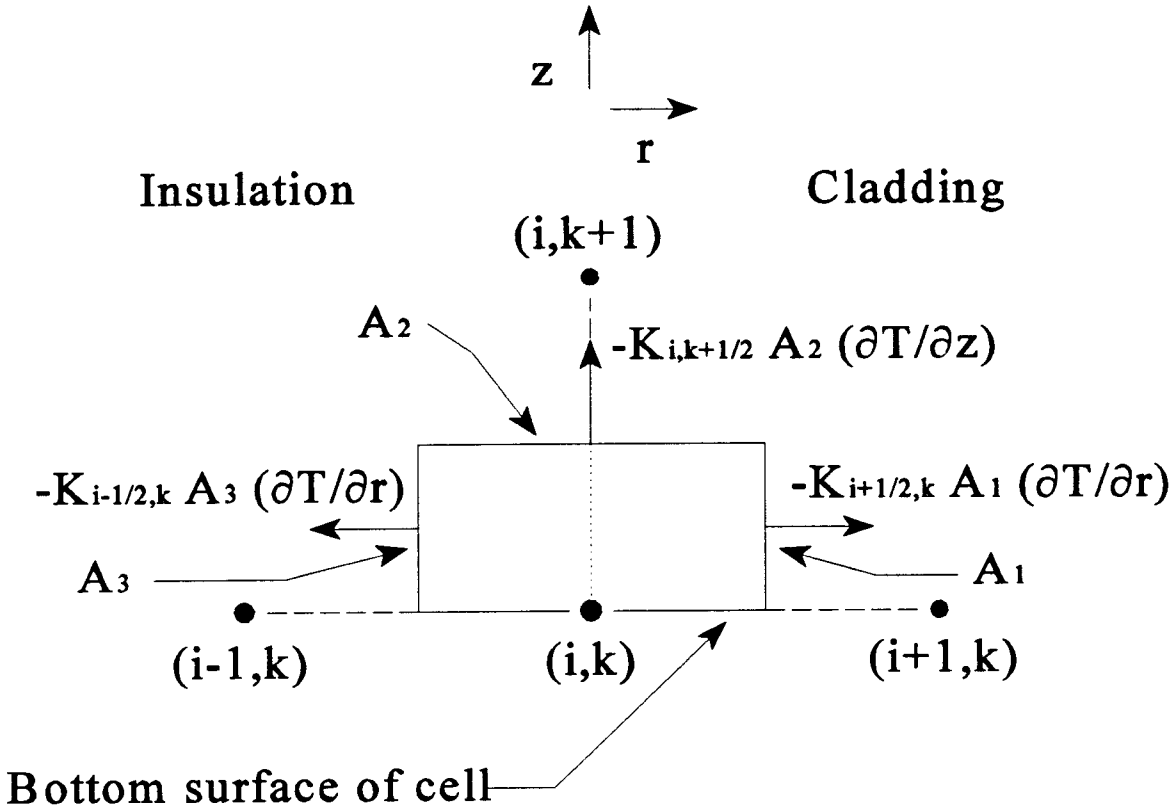
$$T_{i,k} = \{1 / [K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i+1/2,k} A_1 /(r_{i+1} - r_i)]\} *$$
$$\{[K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i+1/2,k} A_1 /(r_{i+1} - r_i)]\} *$$
$$\{[K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k-1} + [K_{i+1/2,k} A_1 /(r_{i+1} - r_i)] T_{i+1,k}\}$$
$$- T_{i,k} .$$



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_{k+1/2} - z_k)$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_{k+1/2} - z_k)$$

Figure 3.23 Energy balance for mesh points located at the bottom of the cell and at the insulation/cladding interface

3.3.2.23 Insulation/Cladding Interface - Interior of the Cell

Figure 3.24 illustrates the energy balance utilized for elements located within the cell and at the insulation/cladding interface. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$0 = K_{i-1/2,k}\, A_3\, (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k-1/2}\, A_2\, (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1}) + K_{i+1/2,k}\, A_1\, (T_{i,k} - T_{i+1,k})/(r_{i+1} - r_i) + K_{i,k+1/2}\, A_2\, (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k)$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [K_{i-1/2,k}\, A_3 /(r_i - r_{i-1}) + K_{i,k-1/2}\, A_2 /(z_k - z_{k-1}) + K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i) + K_{i,k+1/2}\, A_2 /(z_{k+1} - z_k)]\} * \{[K_{i-1/2,k}\, A_3 /(r_i - r_{i-1})]\, T_{i-1,k} + [K_{i,k-1/2}\, A_2 /(z_k - z_{k-1})]\, T_{i,k-1} + [K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i)]\, T_{i+1,k} + [K_{i,k+1/2}\, A_2 /(z_{k+1} - z_k)]\, T_{i,k+1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

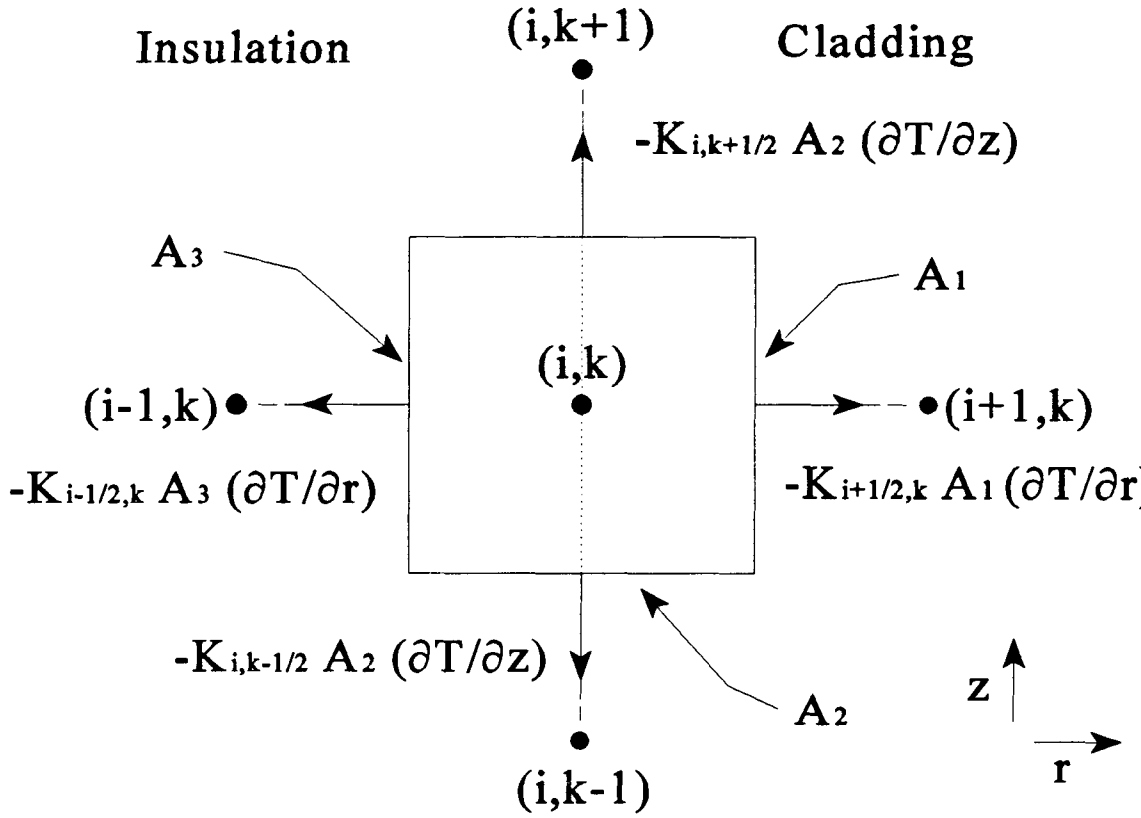
where $R_{i,k}$ is

$$R_{i,k} = \{1 / [K_{i-1/2,k}\, A_3 /(r_i - r_{i-1}) + K_{i,k-1/2}\, A_2 /(z_k - z_{k-1}) + K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i) + K_{i,k+1/2}\, A_2 /(z_{k+1} - z_k)]\} * \{[K_{i-1/2,k}\, A_3 /(r_i - r_{i-1})]\, T_{i-1,k} + [K_{i,k-1/2}\, A_2 /(z_k - z_{k-1})]\, T_{i,k-1} + [K_{i+1/2,k}\, A_1 /(r_{i+1} - r_i)]\, T_{i+1,k} + [K_{i,k+1/2}\, A_2 /(z_{k+1} - z_k)]\, T_{i,k+1}\} - T_{i,k}\,.$$

3.3.2.24 Cladding/Coolant Channel Interface - Top of the Cell

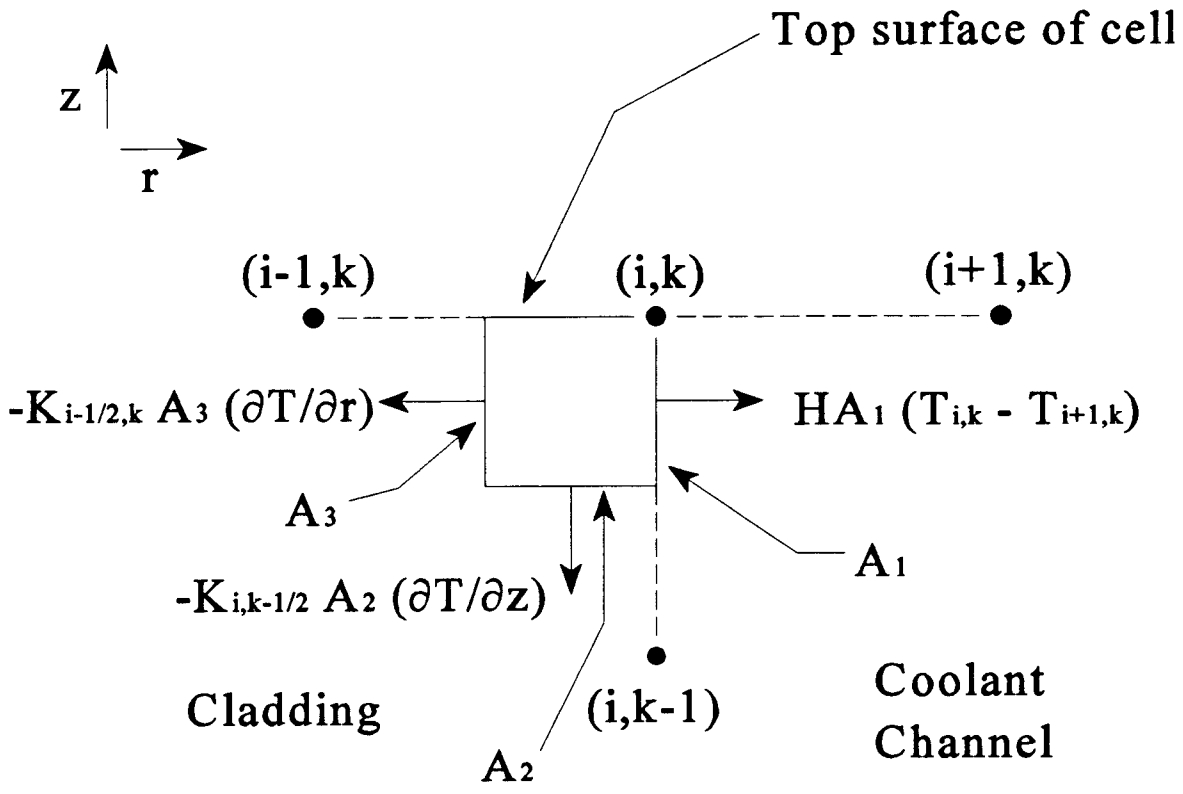
Figure 3.25 illustrates the energy balance utilized for elements located at the top of the cell and at the cladding/coolant interface. These elements include the assumption that the top surface of the cell is adiabatic. The symbol, "H", represents the convective heat transfer co-efficient determined by the relationship developed in TFEHX [23]. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$0 = H\, A_1\, (T_{i,k} - T_{i+1,k}) + K_{i,k-1/2}\, A_2\, (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1}) + K_{i-1/2,k}\, A_3\, (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1})$$



$$A_1 = 2\,\pi\, r_{i+1/2}\,(z_{k+1/2} - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_{k+1/2} - z_{k-1/2})$$

Figure 3.24 Energy balance for mesh points located within the cell and at the insulation/cladding interface



$$A_1 = 2\,\pi\, r_i\,(z_k - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_i - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_k - z_{k-1/2})$$

Figure 3.25 Energy balance for mesh points located at the top of the cell and at the cladding/coolant channel interface

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [H A_1 + K_{i,k-1/2} A_2 /(z_k - z_{k-1}) + K_{i-1/2,k} A_3 /(r_{i+1} - r_i)]\} * \{H A_1 T_{i+1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [H A_1 + K_{i,k-1/2} A_2 /(z_k - z_{k-1}) + K_{i-1/2,k} A_3 /(r_{i+1} - r_i)]\} * \{H A_1 T_{i+1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k}\} - T_{i,k} .$$

3.3.2.25 Cladding/Coolant Channel Interface - Bottom of the Cell

Figure 3.26 illustrates the energy balance utilized for elements located at the bottom of the cell and at the cladding/coolant interface. These elements include the assumption that the bottom surface of the cell is adiabatic. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$0 = H A_1 (T_{i,k} - T_{i+1,k}) + K_{i,k+1/2} A_2 (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k) + K_{i-1/2,k} A_3 (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1})$$

that can be solved for $T_{i,k}$ to yield

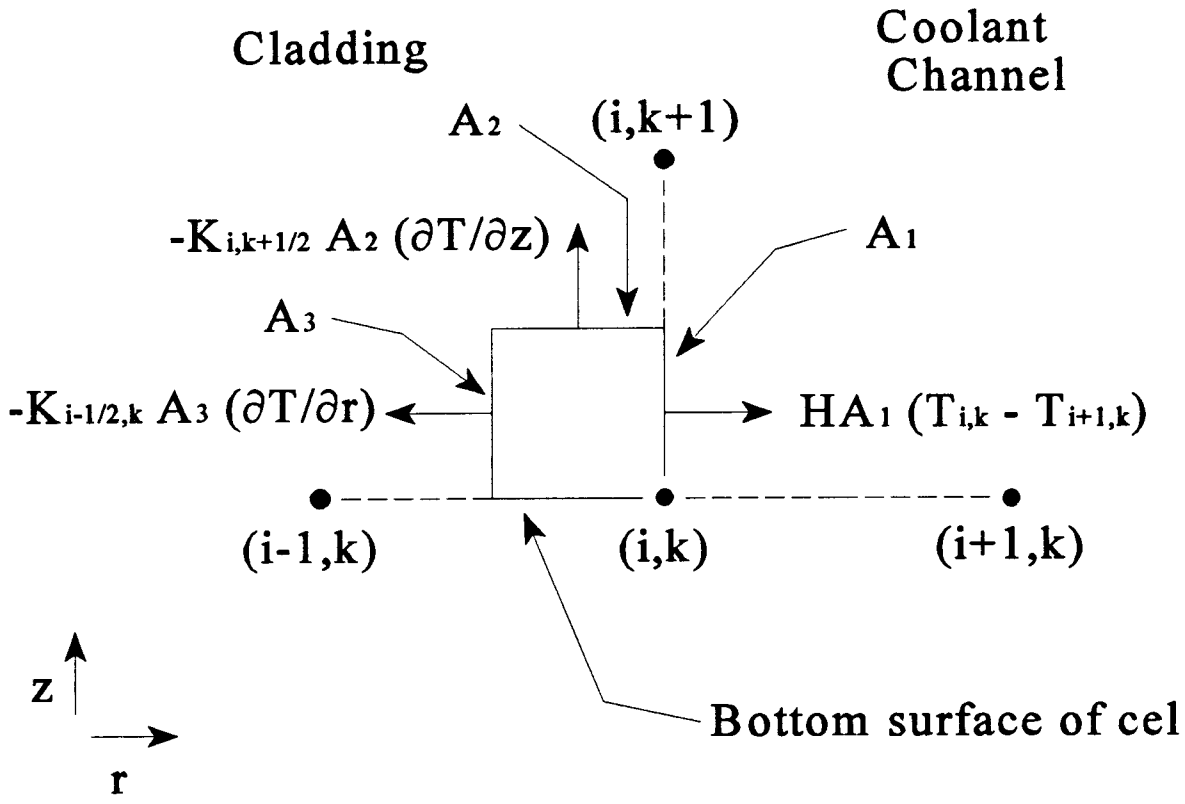
$$T_{i,k} = \{1 / [H A_1 + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i-1/2,k} A_3 /(r_i - r_{i-1})]\} * \{H A_1 T_{i+1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k}\} .$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [H A_1 + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i-1/2,k} A_3 /(r_i - r_{i-1})]\} * \{H A_1 T_{i+1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k}\} - T_{i,k} .$$



$$A_1 = 2\,\pi\, r_i\,(z_{k+1/2} - z_k)$$
$$A_2 = \pi\,(r^2_i - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_{k+1/2} - z_k)$$

Figure 3.26 Energy balance for mesh points located at the bottom of the cell and at the cladding/coolant channel interface

3.3.2.26 Cladding/Coolant Channel Interface - Interior of the Cell

Figure 3.27 illustrates the energy balance utilized for elements located within the cell and at the cladding/coolant interface. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$0 = H A_1 (T_{i,k} - T_{i+1,k}) + K_{i,k+1/2} A_2 (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k) + K_{i,k-1/2} A_2 (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1}) + K_{i-1/2,k} A_3 (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1})$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [H A_1 + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i,k-1/2} A_2 /(z_k - z_{k-1}) + K_{i-1/2,k} A_3 /(r_i - r_{i-1})]\} * \{H A_1 T_{i+1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k}\}$$

which in iterative form becomes

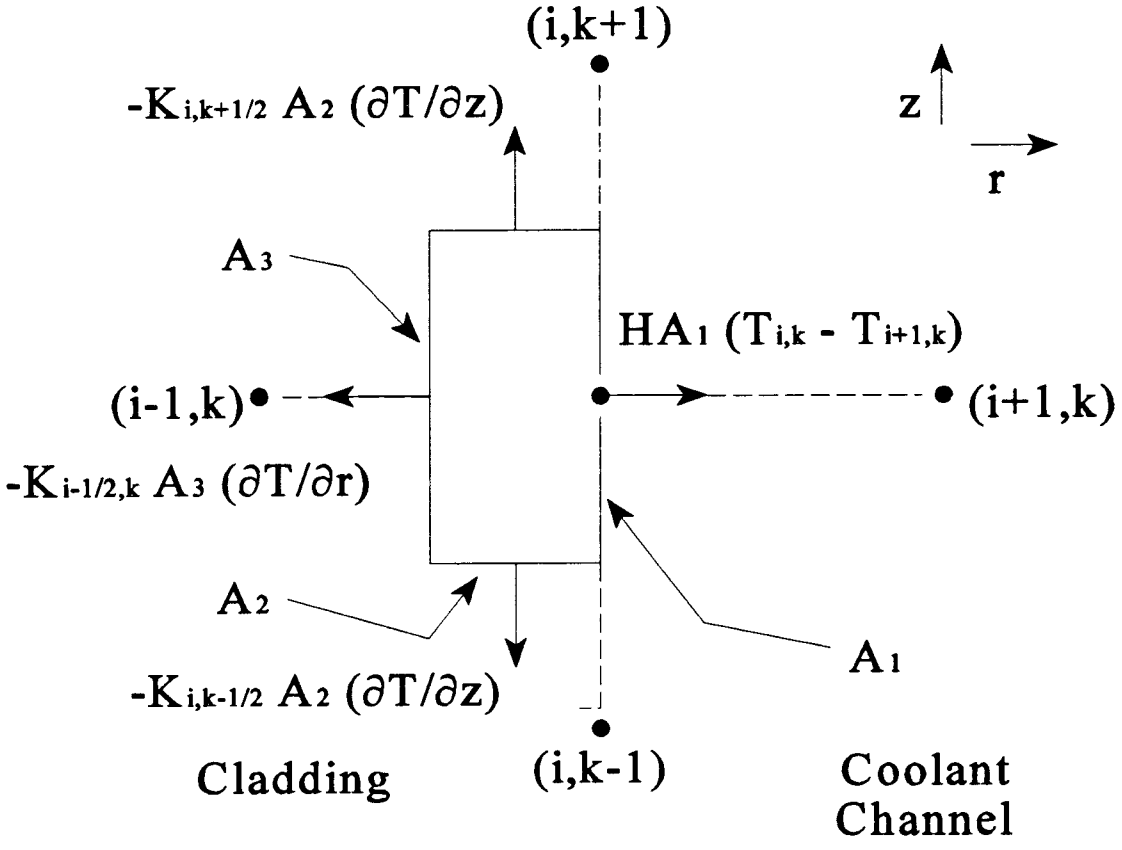
$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [H A_1 + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i,k-1/2} A_2 /(z_k - z_{k-1}) + K_{i-1/2,k} A_3 /(r_i - r_{i-1})]\} * \{H A_1 T_{i+1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k}\} - T_{i,k} .$$

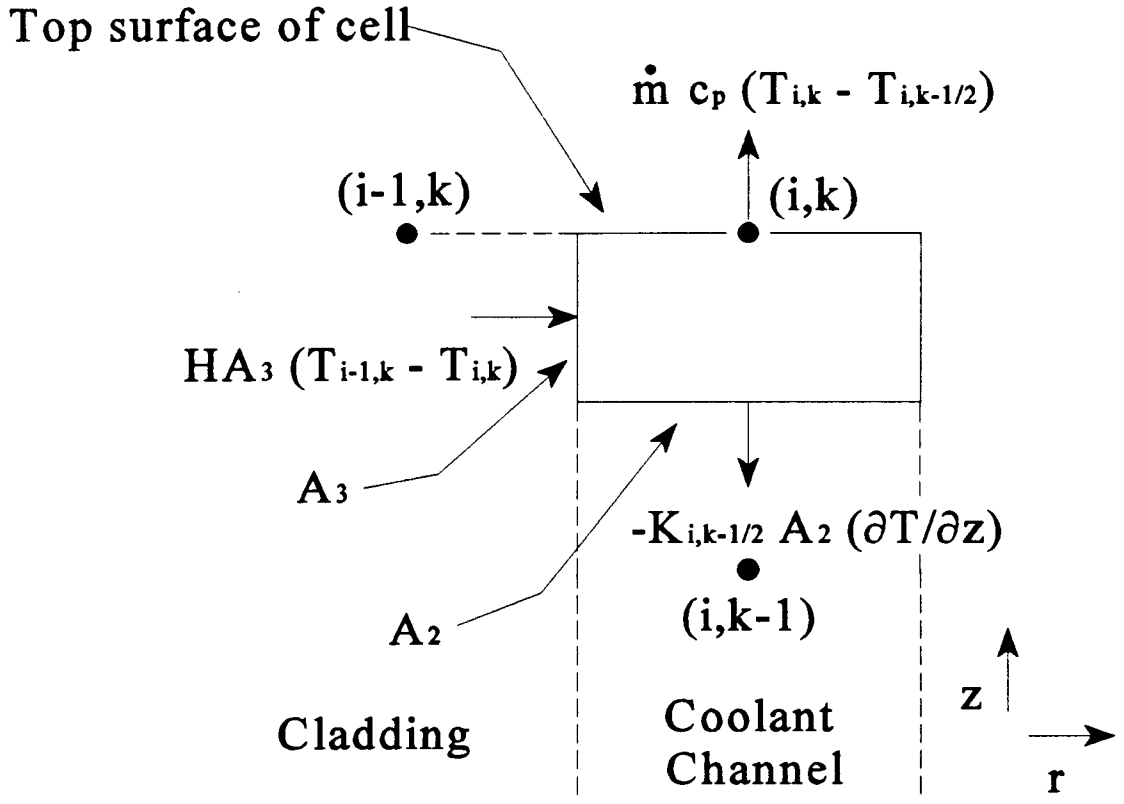
3.3.2.27 Coolant Channel - Top of the Cell

Figure 3.28 illustrates the energy balance utilized for coolant channel elements located at the top of the cell. These elements include the assumption that the top surface of the cell is adiabatic as far as heat conduction is concerned. It is also assumed that there is no net heat transfer occurring through the outer coolant channel wall. The net flow of energy out of the element due to the flowing coolant is accounted for by the $mc\Delta t$ term. Balancing the "heat in" and "heat out" terms then yields the following equation:



$$A_1 = 2\,\pi\,r_i\,(z_{k+1/2} - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_i - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\,r_{i-1/2}\,(z_{k+1/2} - z_{k-1/2})$$

Figure 3.27 Energy Balance for mesh points located in the interior of the cell and at the cladding/coolant channel interface



$$A_2 = \pi\ (r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\ \pi\ r_{i-1/2}\ (z_k - z_{k-1/2})$$

Figure 3.28 Energy balance for mesh points located at the top of the cell and in the coolant channel

$$H A_3 (T_{i-1,k} - T_{i,k}) = Mdot C_p (T_{i,k} - T_{i,k-1/2}) + K_{i,k-1/2} A_2 (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1})$$

that can be solved for $T_{i,k}$ using $(T_{i,k} - T_{i,k-1/2}) = (T_{i,k} - T_{i,k-1}) / 2$, to yield

$$T_{i,k} = \{1 / [H A_3 + Mdot C_p /2 + K_{i,k-1/2} A_2 /(z_k - z_{k-1})]\} * \{H A_3 T_{i-1,k} + [Mdot C_p T_{i,k-1}]/2 + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [H A_3 + Mdot C_p /2 + K_{i,k-1/2} A_2 /(z_k - z_{k-1})]\} * \{H A_3 T_{i-1,k} + [Mdot C_p T_{i,k-1}]/2 + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1}\} - T_{i,k} .$$

3.3.2.28 Coolant Channel - Bottom of the Cell

The mesh point of this element is a boundary condition whose value is the coolant inlet temperature specified by the user ($T_{i,k}$ = Tinlet).

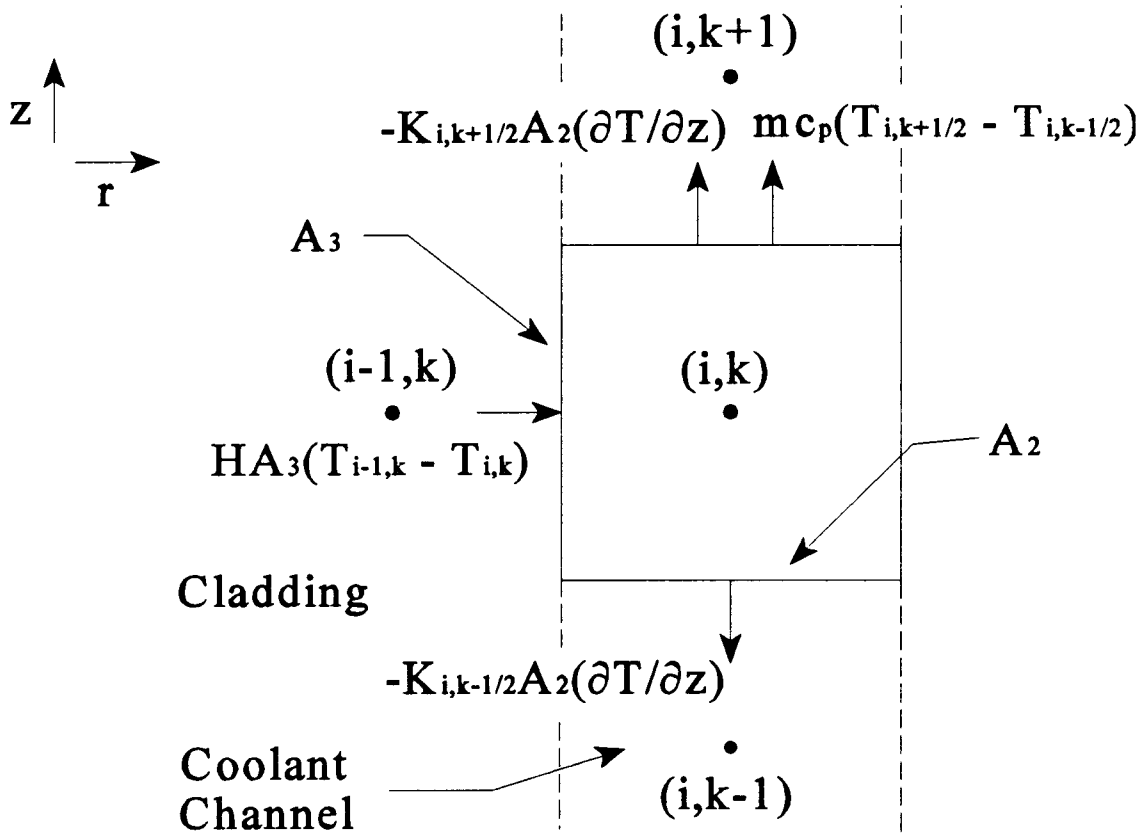
3.3.2.29 Coolant Channel - Interior of the Cell

Figure 3.29 illustrates the energy balance utilized for elements located within the coolant channel. These elements also include the assumption of no net heat transfer through the outer coolant channel wall. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$H A_3 (T_{i-1,k} - T_{i,k}) = Mdot C_p (T_{i,k+1/2} - T_{i,k-1/2}) + K_{i,k+1/2} A_2 (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k) + K_{i,k-1/2} A_2 (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1})$$

that can be solved for $T_{i,k}$ using $(T_{i,k+1/2} - T_{i,k-1/2}) = (T_{i,k+1} - T_{i,k-1})/2$, to yield

$$T_{i,k} = \{1 / [H A_3 + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i,k-1/2} A_2 /(z_k - z_{k-1})]\} * \{H A_3 T_{i-1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k) T_{i,k+1} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1} + Mdot C_p (T_{i,k-1} - T_{i,k+1})/2\}$$



$$A_2 = \pi\,(r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\,r_{i-1/2}\,(z_{k+1/2} - z_{k-1/2})$$

Figure 3.29 Energy balance for mesh points located within the interior of the cell and in the coolant channel

which in iterative form becomes

$$T_{i,k} = T_{i,k} + \text{Alpha} * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [H A_3 + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i,k-1/2} A_2 /(z_k - z_{k-1})]\} * \{H A_3 T_{i-1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k) T_{i,k+1} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1} + \text{Mdot } C_p (T_{i,k-1} - T_{i,k+1})/2\} - T_{i,k} .$$

3.3.3 Multicell TFEs

3.3.3.1 Introduction

To analyze the thermal-hydraulic behavior of multicell TFEs, the heat transfer occurring between cells must be addressed. To accommodate the cell to cell coupling, MCTFE includes the following intercell heat transfer mechanisms:

1) convection and conduction through the coolant channel,

2) conduction between the cladding of adjacent cells, and

3) conduction between the emitter of one cell and the collector of the adjacent cell, that occurs due to the electrical connection between the electrodes.

For the last mechanism listed, the heat transfer only occurs from the emitter of one cell to the "electrically downstream" collector of the next cell (i.e., not from the collector of one cell to the downstream emitter of the next cell). There is not considered to be any heat transfer between adjacent fuel, emitter, collector, or insulator regions which are effectively isolated from each other.

The following sections will now discuss how MCTFE includes the above three considerations by coupling the affected nodes together in the thermal-hydraulic routine. For each consideration the additional term(s) that a multicell TFE would include are examined and discussed.

3.3.3.2 Coolant channel coupling

The only possible finite elements affected by the coupling of the coolant channel from one cell to the next is of course the top and bottom elements of each cell. For the bottom element of the first cell and the top element of the last cell, the descriptions of sections 3.3.2 27 and 3.3.2.28 apply. However, for the top and bottom elements located at the cell interfaces additional terms are required.

Figure 3.30 illustrates the energy balance utilized for coolant channel elements located at the top of the cell and at the intercell region. It is assumed that there is no net heat transfer occurring through the outer coolant channel wall. The net flow of energy out of the element due to the flowing coolant is accounted for by the mcΔt term. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$H\,A_3\,(T_{i-1,k} - T_{i,k}) = Mdot\,C_p\,(T_{i,k+1/2} - T_{i,k-1/2}) + K_{i,k-1/2}\,A_2\,(T_{i,k} - T_{i,k-1})/(z_k - z_{k-1}) + K_{i,k+1/2}\,A_2\,(T_{i,k+1} - T_{i,k}) / CSD$$

where CSD = cell separation distance.

This equation can be solved for $T_{i,k}$ using $(T_{i,k+1/2} - T_{i,k-1/2}) = (T_{i,k+1} - T_{i,k-1}) / 2$, to yield:

$$T_{i,k} = \{1 / [H\,A_3 + K_{i,k-1/2}\,A_2 /(z_k - z_{k-1}) + K_{i,k+1/2}\,A_2 / CSD]\} * \{H\,A_3\,T_{i-1,k} + dot\,C_p\,(T_{i,k-1} - T_{i,k+1})/2 + [K_{i,k-1/2}\,A_2 /(z_k - z_{k-1})]\,T_{i,k-1} + [K_{i,k+1/2}\,A_2 / CSD]\,T_{i,k+1}\}$$

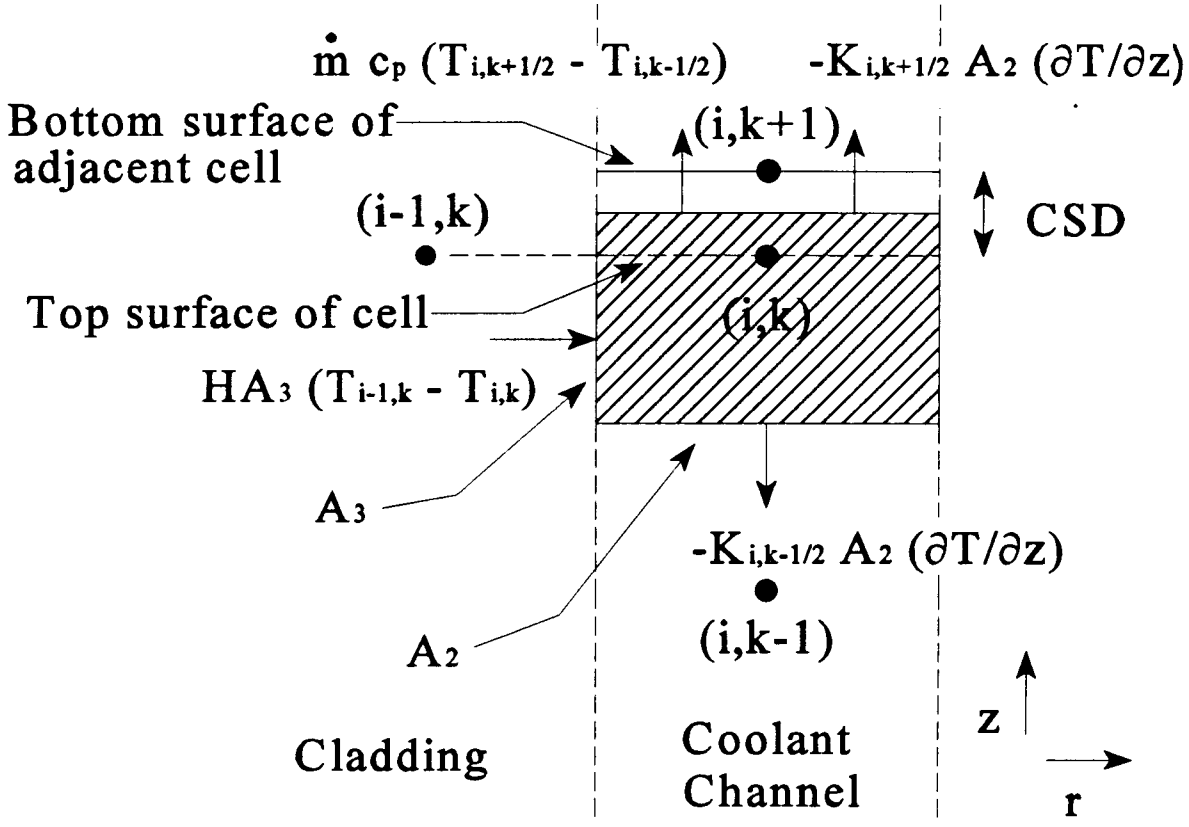
which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

where $R_{i,k}$ is

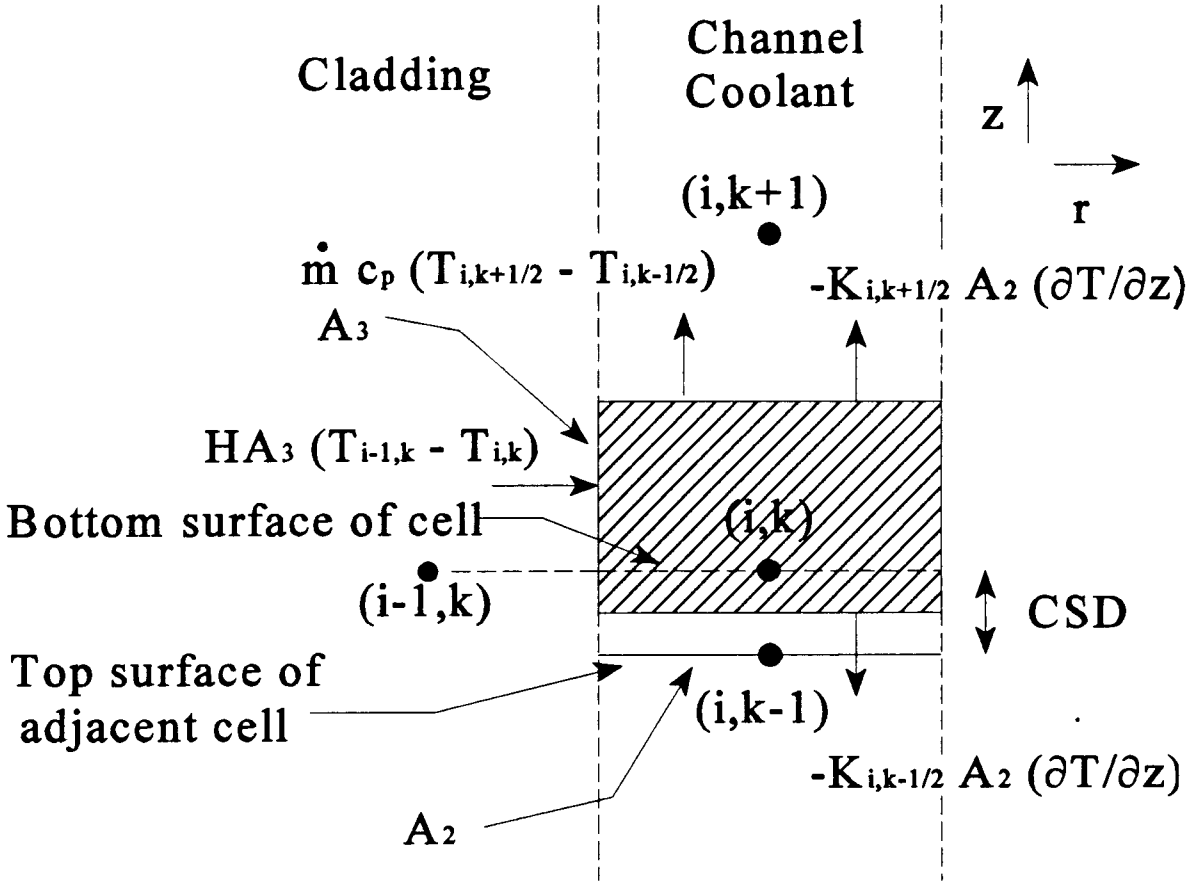
$$R_{i,k} = \{1 / [H\,A_3 + K_{i,k-1/2}\,A_2 /(z_k - z_{k-1}) + K_{i,k+1/2}\,A_2 / CSD]\} * \{H\,A_3\,T_{i-1,k} + Mdot\,C_p\,(T_{i,k-1} - T_{i,k+1})/2 + [K_{i,k-1/2}\,A_2 /(z_k - z_{k-1})]\,T_{i,k-1} + [K_{i,k+1/2}\,A_2 / CSD]\,T_{i,k+1}\} - T_{i,k}\,.$$

Figure 3.31 illustrates the energy balance utilized for coolant channel elements located at the bottom of the cell and at the intercell region. As before, it is assumed that there is



$$A_2 = \pi\ (r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\ \pi\ r_{i-1/2}\ (z_{k+1/2} - z_{k-1/2})$$

Figure 3.30 Energy balance for mesh points located at the top of the coolant channel and at a cell interface region



$$A_2 = \pi\ (r^2_{i+1/2} - r^2_{i-1/2})$$
$$A_3 = 2\ \pi\ r_{i-1/2}\ (z_{k+1/2} - z_{k-1/2})$$

Figure 3.31 Energy balance for mesh points located at the bottom of the coolant channel and at a cell interface region

no net heat transfer occurring through the outer coolant channel wall and the net flow of energy out of the element due to the flowing coolant is accounted for by the mcΔt term. Balancing the "heat in" and "heat out" terms then yields the following equation:

$$H A_3 (T_{i-1,k} - T_{i,k}) = Mdot C_p (T_{i,k+1/2} - T_{i,k-1/2}) + K_{i,k+1/2} A_2 (T_{i,k+1} - T_{i,k})/(z_{k+1} - z_k) + K_{i,k-1/2} A_2 (T_{i,k} - T_{i,k-1}) / CSD$$

where CSD = cell separation distance.

This equation can be solved for $T_{i,k}$ using $(T_{i,k+1/2} - T_{i,k-1/2}) = (T_{i,k+1} - T_{i,k-1}) / 2$, to yield:

$$T_{i,k} = \{1 / [H A_3 + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i,k-1/2} A_2 / CSD]\} * \{H A_3 T_{i-1,k} + Mdot C_p (T_{i,k-1} - T_{i,k+1})/2 + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i,k-1/2} A_2 / CSD] T_{i,k-1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

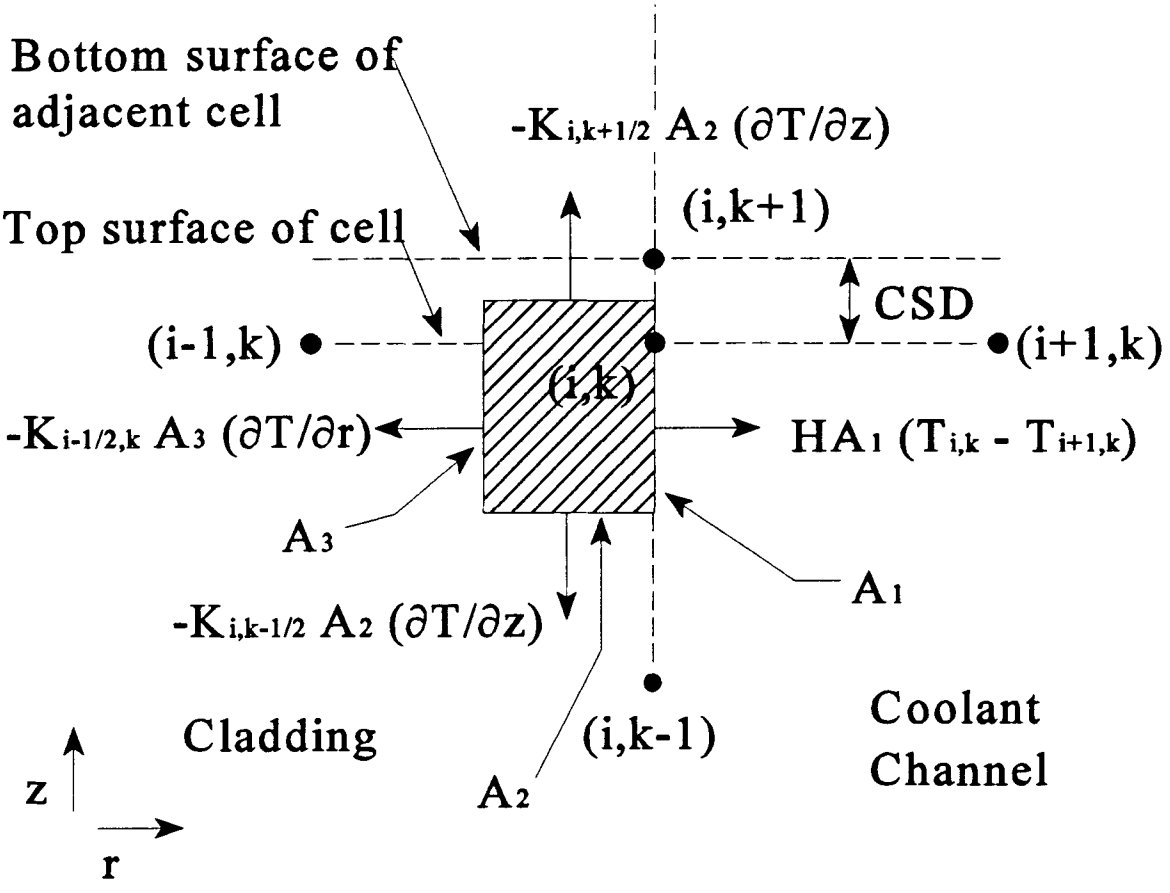
where $R_{i,k}$ is

$$R_{i,k} = \{1 / [H A_3 + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i,k-1/2} A_2 / CSD]\} * \{H A_3 T_{i-1,k} + Mdot C_p (T_{i,k-1} - T_{i,k+1})/2 + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i,k-1/2} A_2 / CSD] T_{i,k-1}\} - T_{i,k} .$$

3.3.3.3 Cladding coupling

As was the case for the coolant channel, the only finite elements effected by the coupling of the cladding of one cell to the next is the top and bottom elements of each cell. For the bottom element of the first cell and the top element of the last cell, the descriptions of sections 3.3.2 24 and 3.3.2.25 apply. However, for the top and bottom elements located at the cell interfaces additional terms are required.

Figure 3.32 illustrates the energy balance utilized for cladding elements located: at the top of the cell at an intercell region and at the coolant channel and cladding interface. Balancing the "heat in" and "heat out" terms then yields the following equation:



$$A_1 = 2\,\pi\,r_i\,(z_{k+1/2} - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_i - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\,r_{i-1/2}\,(z_{k+1/2} - z_{k-1/2})$$

Figure 3.32 Energy balance for mesh points located at the top of the cell at an intercell region and at the cladding/coolant channel interface.

$$0 = H A_1 (T_{i,k} - T_{i+1,k}) + K_{i,k-1/2} A_2 (T_{i,k} - T_{i,k-1})/(z_k - z_{k-1}) + K_{i-1/2,k} A_3 (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k+1/2} A_2 (T_{i,k+1} - T_{i,k})/CSD$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [H A_1 + K_{i,k-1/2} A_2 /(z_k - z_{k-1}) + K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k+1/2} A_2 /CSD]\} * \{H A_1 T_{i+1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k+1/2} A_2 /CSD] T_{i,k+1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [H A_1 + K_{i,k-1/2} A_2 /(z_k - z_{k-1}) + K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k+1/2} A_2 /CSD]\} * \{H A_1 T_{i+1,k} + [K_{i,k-1/2} A_2 /(z_k - z_{k-1})] T_{i,k-1} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k+1/2} A_2 /CSD] T_{i,k+1}\} - T_{i,k} .$$

Figure 3.33 illustrates the energy balance utilized for cladding elements located: at the bottom of the cell at an intercell region and at the coolant channel and cladding interface. Balancing the "heat in" and "heat out" terms then yields the following equation:

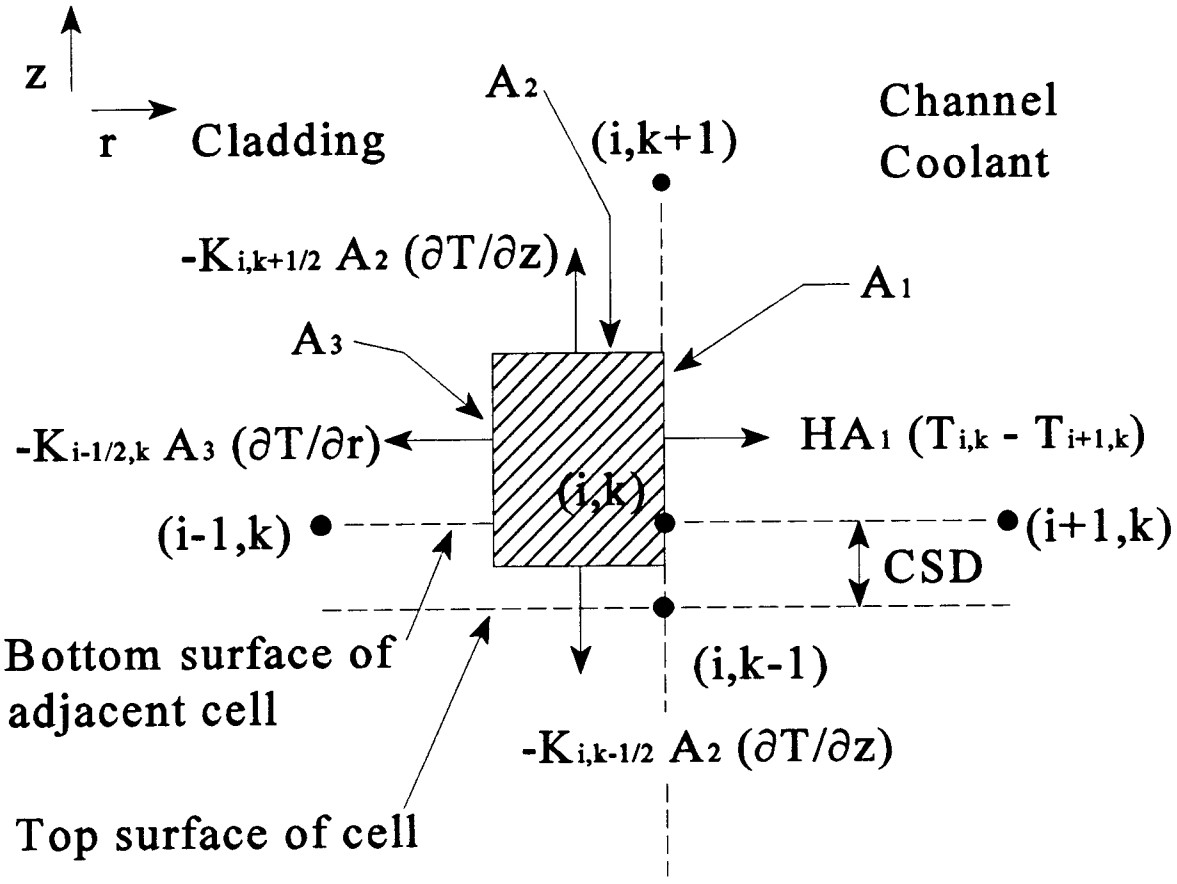
$$0 = H A_1 (T_{i,k} - T_{i+1,k}) + K_{i,k+1/2} A_2 (T_{i,k} - T_{i,k+1})/(z_{k+1} - z_k) + K_{i-1/2,k} A_3 (T_{i,k} - T_{i-1,k})/(r_i - r_{i-1}) + K_{i,k-1/2} A_2 (T_{i,k} - T_{i,k-1})/CSD$$

that can be solved for $T_{i,k}$ to yield

$$T_{i,k} = \{1 / [H A_1 + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k-1/2} A_2 /CSD]\} * \{H A_1 T_{i+1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} + [K_{i,k-1/2} A_2 /CSD] T_{i,k-1}\}$$

which in iterative form becomes

$$T_{i,k} = T_{i,k} + Alpha * R_{i,k}$$



$$A_1 = 2\,\pi\, r_i\,(z_{k+1/2} - z_{k-1/2})$$
$$A_2 = \pi\,(r^2_i - r^2_{i-1/2})$$
$$A_3 = 2\,\pi\, r_{i-1/2}\,(z_{k+1/2} - z_{k-1/2})$$

Figure 3.33 Energy balance for mesh points located at the bottom of the cell at an intercell region and at the cladding/coolant channel interface.

where $R_{i,k}$ is

$$R_{i,k} = \{1 / [H A_1 + K_{i,k+1/2} A_2 /(z_{k+1} - z_k) + K_{i-1/2,k} A_3 /(r_i - r_{i-1}) + K_{i,k-1/2} A_2 /CSD]\} *$$
$$\{H A_1 T_{i+1,k} + [K_{i,k+1/2} A_2 /(z_{k+1} - z_k)] T_{i,k+1} + [K_{i-1/2,k} A_3 /(r_i - r_{i-1})] T_{i-1,k} +$$
$$[K_{i,k-1/2} A_2 /CSD] T_{i,k-1}\} - T_{i,k} .$$

3.3.3.4 Emitter-collector coupling

As previously shown by Figure 2.6(b), successive cells of a multicell TFE are connected in series. By electrically connecting the emitter of one cell to the collector of it's adjacent (electrically downstream) cell, a heat conducting path is also established. To determine the rate of heat transfer that occurs between these electrodes the effective conductance (K) is determined for the intercell connection. From the heat conduction equation, $Q = -kA \Delta T/\Delta X$, where $\Delta T/\Delta X$ is substituted for dT/dX, the conductance is defined as: $K = kA/\Delta X$. Therefore, the heat transfer rate can be given in terms of the conductance as: $Q = -K \Delta T$.

For series connected conductors the overall conductance is determined by summing the reciprocals of the individual conductors (i.e., $1/K_{eff} = 1/K_1 + 1/K_2 + 1/K_3 +$ etc.). With the lengths and areas for each segment of the intercell connection designated by the user in the input, the effective conductance is then calculated for the connection and multiplied by the electrode temperature difference to determine the heat transfer rate for the affected nodes. To calculate the intercell effective conductance MCTFE includes the function CONINCEL which is utilized at each iteration because the material conductivities are a function of temperature. For the emitter of the top cell, which is connected to the collector of the bottom cell through the electrical load, no heat transfer is assumed to occur due to expected small values of lead conductance.

The finite elements affected by the electrical connections of each cell of a multicell TFE include the top cell surfaces of the emitter and the bottom cell surfaces of the collector. For each of these elements an extra term (i.e., "heat in" term for collector, and "heat out" term for emitter) is included in the elements temperature equation. This term is

the simple product of the intercell effective conductance and the temperature difference between the element and the opposite electrode.

3.4 THERMIONIC CONSIDERATIONS

3.4.1 Introduction

MCTFE is a coupled thermal-hydraulic and thermionic modeling code. The thermionic analysis that is performed by MCTFE consists of determining the voltages, current densities, and electron cooling densities that exist at each axial position of the TFE. The thermionic analysis is accomplished by the VOLTCALC subroutine, which is also a coupled routine. VOLTCALC consists of a routine that determines the voltages based on a given set of current densities and the CDEN function [4,21,22] that determines the current densities based on voltages and electrode temperatures. These two components of VOLTCALC then iterate back and forth until a converged set of voltages and current densities are obtained.

CDEN is a function that is part of the CYLCON-6 subroutine used in TFEHX and was originally developed by John McVey of Rasor Associates Inc. [4]. Although the voltage calculation method has been modified in MCTFE the CDEN function continues to be used as originally written.

The section that now follows will describe the methodologies utilized by the VOLTCALC subroutine. This analysis will include the differences that are encountered between single cell and multicell TFEs that utilize different circuit arrangements.

3.4.2 VOLTCALC Subroutine

The VOLTCALC subroutine calculates the voltages (V), current densities (A/cm2) and electron cooling densities (Watts/cm2) that exist at each axial position of the TFE. The current and electron cooling densities, as already mentioned, are provided by the CDEN function. Therefore the remaining task is to compute the interelectrode voltage from the given current densities, which can be performed by the use of Ohms Law (i.e., voltage = current * resistance) [24]. Because single cell and multicell TFEs utilize a

different circuit arrangement, each type TFE will be examined and discussed separately in the following sections.

3.4.2.1 Single Cell TFEs

As previously illustrated with Figure 2.6(a), single cell TFEs typically use a double ended connection to each electrode. Therefore, to determine the interelectrode voltage at each axial position the following analysis is considered. To begin, the voltage drop across a small slice of an electrode of thickness dz is given as follows:

$$dV_E(z) = I_E(z)\,\rho_E(z)\,dz / \pi(R^2_{Eo} - R^2_{Ei}) \quad \text{for the emitter, and}$$

$$dV_C(z) = I_C(z)\,\rho_C(z)\,dz / \pi(R^2_{Co} - R^2_{Ci}) \quad \text{for the collector}$$

where ρ = resistivity, and R_{Eo}, R_{Ei}, R_{Co}, R_{Ci}, are the outer and inner electrode radii for the emitter and collector respectively. The preceding equations are simple expressions of Ohm's Law with the resistance given as the product of the resistivity and length divided by the area of the slice.

The interelectrode voltage, V(z), is given by the difference between the emitter and collector voltage as follows:

$$V(z) = V_E(z) - V_C(z).$$

Now, the emitter and collector voltage can be given as:

$$V_E(z) = V_E(0) + {}_0\int^z dV_E(z) \quad \text{and} \quad V_C(z) = V_C(0) - {}_0\int^z dV_C(z), \text{ or}$$

$$V(z) = [V_E(0) - V_C(0)] + {}_0\int^z dV_E(z) + {}_0\int^z dV_C(z).$$

But, $V_E(0) - V_C(0)$ = Itotal * R_L (load resistance), therefore, substituting for $dV_E(z)$ and $dV_C(z)$ then yields the following:

$$V(z) = \text{Itotal} * R_L + {}_0\int^z I_E(z)\,\rho_E(z)\,dz / \pi(R^2_{Eo} - R^2_{Ei}) + {}_0\int^z dV\, I_C(z)\,\rho_C(z)\,dz / \pi(R^2_{Co} - R^2_{Ci}),$$

that for finite element analysis with k as the axial parameter is

$$V(k) = Itotal * R_L + \sum_{(1,k)} [I_E(k)\, \rho_E(k) / \pi(R^2_{Eo} - R^2_{Ei}) + I_C(k)\, \rho_C(k) / \pi(R^2_{Co} - R^2_{Ci})]\, dk. \quad (3.11)$$

To determine the voltages from equation 3.11, the currents and resistivities for both the emitter and collector at each axial position must be known. The resistivities are determined in the RHOW function, while the currents are determined by a summing of the current densities. For the emitter and collector, $I_E(k)$ and $I_C(k)$ are given by the following equations (note: each equation involves the emitter radius because that is where the current densities are referenced):

$$I_E(k) = I_E\,bottom - [\sum_{(1,k)} Jden(k)*dk]*2*\pi*R_{Eo}, \text{ and}$$

$$I_C(k) = I_C\,bottom - [\sum_{(1,k)} Jden(k)*dk]*2*\pi*R_{Eo},$$

where, I_E bottom and I_C bottom are the currents leaving and entering the emitter and collector electrodes respectively at the bottom of the cell. The total current (Itotal) generated by the cell is also needed in equation 3.11 and is calculated by the following:

$$Itotal = [\sum_{(1,kmax)} Jden(k)*dk]*2*\pi*R_{Eo}.$$

In TFEHX, it was assumed that the bottom currents for both the emitter and collector would be one half the total current. However, this would not be the case unless exact symmetry about the axial midpoint is present. The higher temperatures at the upper ends of the collector (which follows the coolant temperature profile), and to a lesser extent with the emitter, result in higher resistivities of the electrodes at their upper ends. This situation should result in more current entering the electrodes from the bottom than the top due to the lower resistance. With MCTFE a simple solution method is used to determine the analytically correct bottom currents.

Because current enters (collector) or leaves (emitter) at both ends of the electrode, somewhere in the middle of the electrode the current will be equal to zero. At this point the emitter electrode voltage will at a maximum and the collector electrode voltage at a

minimum. Since the voltage at each end of the electrode is the same, the sum of voltages rises and drops through the entire electrode should sum to zero. Therefore, MCTFE guesses the bottom currents and then iteratively adjusts them until a zero voltage sum is obtained for both electrodes over their length.

3.4.2.2 Multicell TFEs

For multicell TFEs the cells of the TFE are connected in series as was previously shown by Figure 2.6(b). With the voltage drops across a thin slice (dV) as given in the last section, the interelectrode voltage can be determined for each cell of the multicell as follows.

$$V(z) = V_E(z) - V_C(z), \quad \text{with}$$

$$V_E(z) = V_E(L) + {}_z\int^L dV_E(z) \qquad \text{and} \qquad V_C(z) = V_C(0) - {}_0\int^z dV_C(z)$$

where L is the cell length, or

$$V(z) = [V_E(L) - V_C(0)] + {}_z\int^L dV_E(z) + {}_0\int^z dV_C(z)$$

but, $V_E(L) - V_C(0) = Itotal * R_{eff}$, where R_{eff} is the effective resistance seen by the cell defined as: $R_{eff} = Vcell / Itotal$. Upon substitution for $dV_E(z)$ and $dV_C(z)$ the following is obtained:

$$V(z) = Itotal * R_{eff} + {}_z\int^L I_E(z)\,\rho_E(z)\,dz / \pi(R^2_{Eo} - R^2_{Ei}) + {}_0\int^z dV\, I_C(z)\,\rho_C(z)\,dz / \pi(R^2_{Co} - R^2_{Ci}),$$

which for finite element analysis with k as the axial parameter is

$$V(k) = Itotal * R_{eff} + \sum_{(k,kmax)} [I_E(k)\,\rho_E(k)\,dk] / \pi(R^2_{Eo} - R^2_{Ei}) + \sum_{(kmin,k)} [I_C(k)\,\rho_C(k)\,dk] / \pi(R^2_{Co} - R^2_{Ci}) \qquad (3.12)$$

where kmin and kmax are the values of k at the bottom and top surfaces of the cell.

As was the case with single cell TFEs, the currents and resistivities for both the emitter and collector at each axial position must be known to determine the voltages from

equation 3.12. For the multicell case the collector and emitter currents, $I_E(k)$ and $I_C(k)$, would be given by the following equations:

$$I_E(k) = \sum_{(kmin,k)} [\, Jden(k)*dk]*2*\pi*R_{Eo}, \quad \text{and} \quad I_C(k) = Itotal - I_E(k).$$

The total current (Itotal) generated by the cell is calculated by summing the products of the current densities and areas over all axial nodes of the cell as before.

3.4.2.3 Boundary conditions

With each of the methods described in the previous two sections the electrical boundary condition must be known in order to determine the results. MCTFE allows the user to specify the load resistance, TFE total voltage, or TFE total current as this condition. In both equations 3.11 and 3.12 the bounding resistance (i.e., R_L for the single cell case and R_{eff} for each cell of multicell TFEs) is required in the calculation. The following paragraphs now describe how VOLTCALC infers this resistance from the input boundary condition (except for the load resistance option of single cell cases where R_L is directly given).

For single cell TFEs, where total voltage is given as the boundary condition, R_L is simply given by Vtotal / Itotal. In fact, it is the total voltage across the cell that equation 3.11 is calculating with the load resistance. When total current is specified by the user as the boundary condition an initial guess of the load resistance is made. This load resistance is subsequently modified with each iteration until the total current determined in VOLTCALC (i.e., summing the cell current densities) equals the load current specified by the user.

With multicell TFEs the situation is a little more complicated. The common thread for each cell of the multicell TFE is the total current. Conservation of current demands that the total current generated in each cell be equal. Therefore, VOLTCALC guesses the effective resistance (R_{eff}) for each cell and then modifies with each iteration until the current of each cell is equal to the others. When load current is specified, the modification of each cells effective resistance continues until cell current equals the load current.

The specification of load resistance or TFE voltage as the boundary condition for multicell TFEs constrains the total TFE (i.e., all the cells). For these cases, the effective resistance for each cell is modified subject to the constraint. Because the sum of the voltage rises in the circuit equals the sum of the voltage drops, the following equation can be used:

$$\text{Itotal} * [\ \textstyle\sum_{(1,I)} R_{eff}(i)\ ] = \text{Itotal} * [\ \text{RICTOT} + R_L\ ]$$

where I is the number of cells and RICTOT is the total resistance of all the intercell regions (i.e., the voltage drop between cells). Canceling the currents then yields:

$$\textstyle\sum_{(1,I)} R_{eff}(I) = \text{RICTOT} + R_L$$

If the load resistance is the specified boundary condition, then subsequent effective resistance modifications are made subject to the constraint of the preceding equation. When TFE voltage is used as the boundary condition, the above equation is used with the load resistance given as shown by the following equation:

$$R_L = V_L / \text{Itotal}.$$

The intercell resistance (RICTOT) used in the above equations is determined by summing the resistances of each intercell region. The resistance of each intercell region is determined as was previously described for the intercell conductance except with resistances the total resistance of a series connection is calculated by a simple sum of the resistances of each piece. The function RESINCEL in MCTFE redetermines this resistance with each iteration since the resistivities of the materials are functions of temperature.

3.5 THERMAL-HYDRAULIC TO THERMIONIC COUPLING AND CONVERGENCE

Sections 3.3 and 3.4 have described the modeling and analysis methods for the thermal-hydraulic and thermionic routines used by MCTFE. In order to determine the

total performance of the TFE, these routines must be coupled. The coupling is required because of the electron cooling (and collector heating) that occurs by the emitted electrons. The emitted electrons remove energy when they are emitted from the surface of the emitter and deposit this energy in the collector. Consequently the temperatures throughout the TFE are dependant upon the amount of electron cooling present, which is dependent on the thermionic processes. The thermionic processes, however, are dependent on the temperatures of the electrodes thus necessitating the coupling of the routines.

MCTFE couples it's SOR iterative thermal-hydraulic method with the thermionic analysis performed in VOLTCALC. These two routines cycle alternately until both the temperatures and the voltages (which are dependent on the current densities) are converged. To aid in the mutual convergence of the routines, the successive electron cooling values that are determined by each outer iteration with VOLTCALC are "under-relaxed" by changing the old values incrementally. The electron cooling values are modified with each outer iteration by adding a fraction (set by the under-relaxation parameter BETA, usually about 1/2) of the difference between the new and old values to the old value. This technique has demonstrated the ability in MCTFE to prevent oscillatory behavior between the two routines, that otherwise can occur.

The convergence criteria for both temperatures and voltages is hard wired in MCTFE, but can be changed by modifying the source code. Current requirements for convergence are set at 0.05 K for the temperature routine and 0.5 mV for the voltage routine. Convergence within each routine is achieved when the root-mean-squared (RMS) error for both temperatures and voltages are less than the designated criteria. Convergence between the coupled routines is considered to have occurred when the temperature RMS is below the temperature criteria after a single iteration (i.e., the last outer iteration with VOLTCALC has changed the electron cooling values so little than the temperatures haven't changed significantly).

CHAPTER 4. RESULTS AND ANALYSIS

4.1 INTRODUCTION

This chapter will discuss the results and analysis completed with the study presented by this report. The main goal of the study was to improve upon and modify TFEHX [1,2] (a coupled thermal-hydraulic and thermionic modeling code for single cell TFEs) such that analysis of TFEs with a multiple number of cells could be performed.

The predominant result of the study has been the development of the MCTFE computer code that can analyze both single and multicell TFEs. Besides expanding the cell number capability of TFEHX, additional modifications have been made with MCTFE. These modifications include adding more flexibility for the user to describe and analyze a particular TFE. Some of the major modifications include: optional boundary condition descriptions, a user designated number of axial and fuel radial nodes, and added power density description choices. Table 4.1 summarizes the major modifications made from TFEHX with MCTFE by comparing some major features of each code.

The additional sections of this chapter will now present and discuss the particular findings and analysis made with MCTFE. Included in these discussions will be the comparisons made with experimental data of physical systems to provide benchmarking of MCTFE. Results achieved with the PDCALC subroutine, developed to approximate the power generation densities in the fuel, will also be given. The chapter will then conclude with some general comments about TFE performance as predicted by MCTFE.

4.2 BENCHMARKING AND COMPARISON STUDIES OF MCTFE

This section will present the comparison studies performed with MCTFE to benchmark and verify the accuracy of the code methods. The comparison studies have been made with experimental results from the Russian built Topaz-II type single cell TFE [5], and data from General Atomics and the developments made in their Thermionic Fuel Element Verification Program [25,26,27]. The General Atomics data involves

Table 4.1 TFEHX and MCTFE Code Comparisons

	TFEHX Code	MCTFE Code
Cell type capability:	single	multiple (1-10)
Input power profile choices:	- chopped cosine (axial) constant (radial)	- constant (axial and radial) - chopped cosine (axial) constant (radial) - tabular data (axial) constant (radial) - tabular data (axial and radial) - PDCALC solution
Thermal-hydraulic modeling:		
- solution method	Gaussian Elimination	SOR iterative scheme
- axial cell nodes	10	user specified
- radial fuel interior nodes	3	user specified
Thermionic modeling:		
- subroutine	CYLCON-6 (utilizes single cell type circuit arrangement)	VOLTCALC (utilizes user specified circuit arrangement)
- boundary condition choice	Total current	Total current Load resistance Total voltage

experimental work with multicell TFEs that is used to compare with MCTFE results and verify the MCTFE multicell dependent routines.

The following two sections will present the results achieved from these benchmarking studies. The Topaz-II experimental data and MCTFE results are examined in section 4.2.1, and the comparisons made between MCTFE and the General Atomics TFEs are discussed in section 4.2.2.

4.2.1 MCTFE Comparison with Topaz-II Data

As was previously done with TFEHX, MCTFE is compared to the data generated by the Russian studies of the Topaz-II TFEs [5]. Figure 4.1 below illustrates the initial results that were achieved with TFEHX. These results were generated using a constant current (153 amps) and a set cesium reservoir temperature (575 K) throughout the range. At first, MCTFE was compared in like manner, with the exception that the cesium reservoir temperature was optimized at each input power level. The results obtained by MCTFE are illustrated in Figure 4.2. As can be seen from the figure, the MCTFE results presented the same basic shape as the TFEHX data, only slightly higher (especially at the lower input power levels).

The differences between TFEHX and MCTFE, however, can be explained. First of all, the 575 K cesium temperature used by TFEHX is the optimum temperature for an input power level of about 3100 Watts. If TFEHX had used the optimum cesium temperature at each input power, the smaller input power level results would have shown a larger electrical power output. Secondly, there is a difference in the value of the emissivity used by TFEHX and MCTFE. TFEHX used a constant value of 0.2 as the emissivity. MCTFE, however, uses the temperature correlation for emissivity developed by the Russians in their work [5]. This correlation results in emissivities of about 0.15 to 0.17 over the range of calculated emitter temperatures. Consequently, both of the above factors would yield higher, especially at lower input powers, electrical power outputs for MCTFE than TFEHX, which was observed.

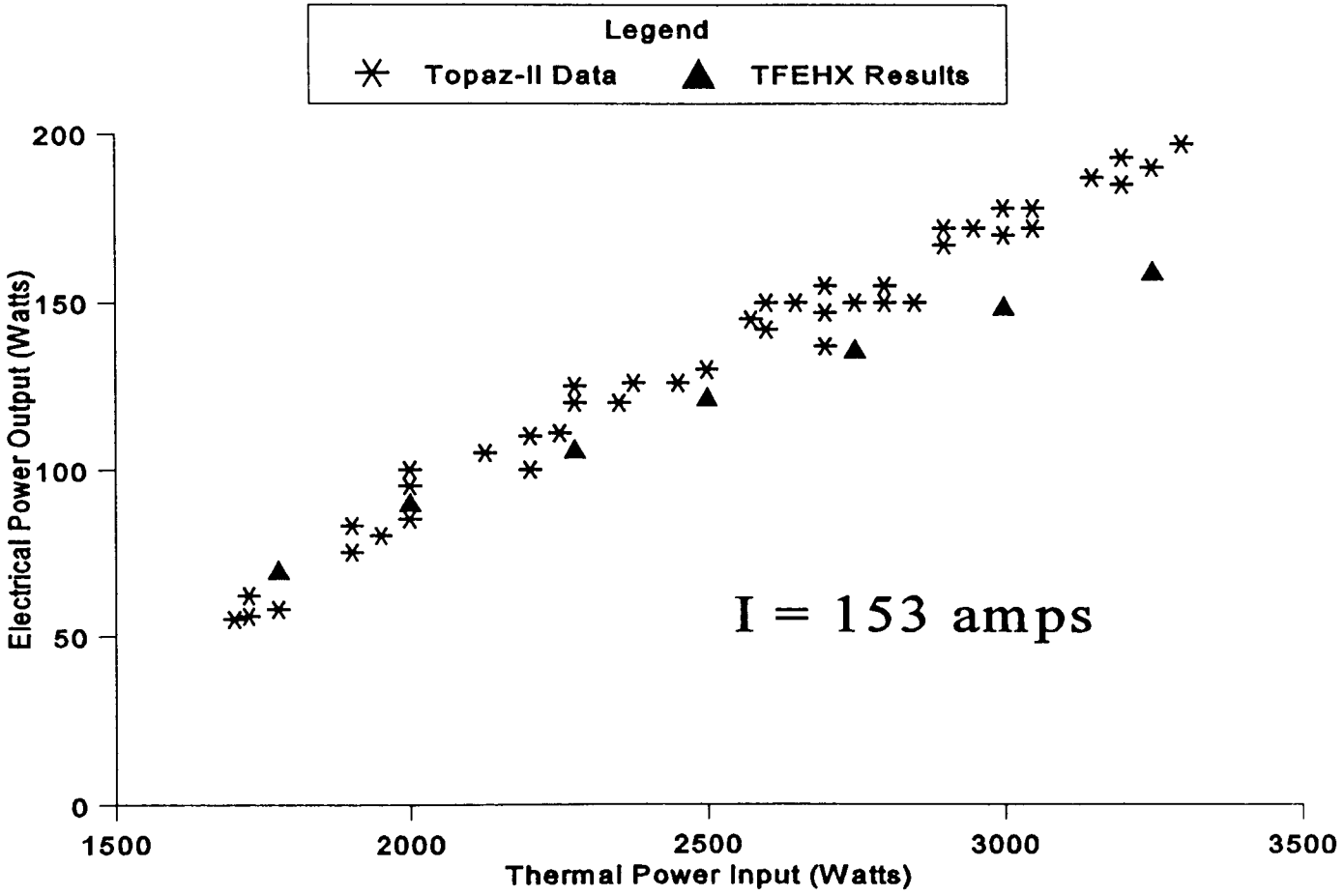


Figure 4.1 Comparison of Topaz-II data with TFEHX code results at constant current

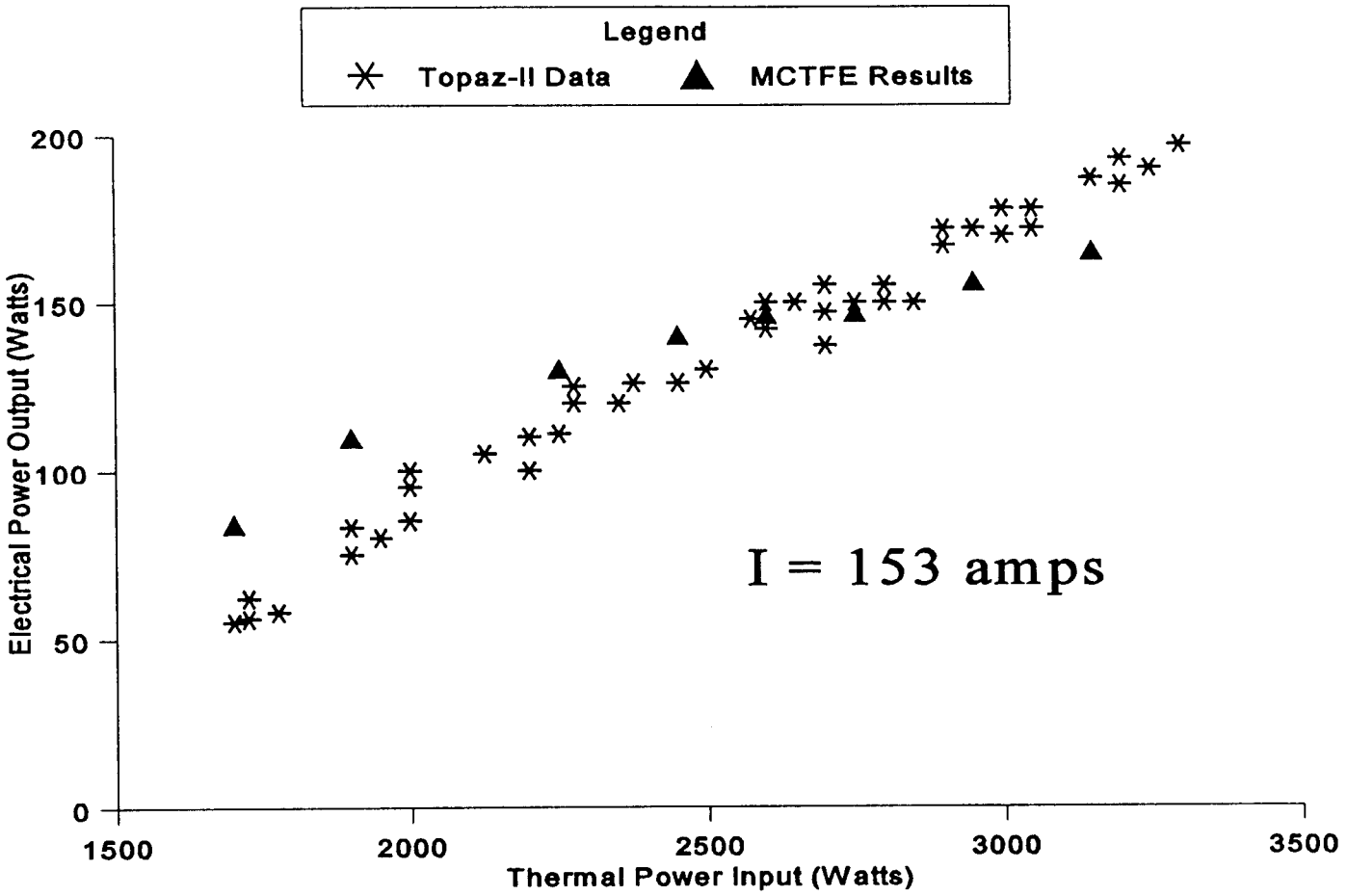


Figure 4.2 Comparison of Topaz-II data with MCTFE code results at constant current

Although the MCTFE to TFEHX discrepancies were explained, further investigation was performed to determine why both codes apparently mis-predicted the response curve shape. As already alluded to, the original Topaz-II data was thought to have been taken at constant current (i.e., 153 amps). Upon further review, however, it was noticed that some specific data were taken at constant voltage (i.e., 0.87 volts). For these experimental results, which are given in Tables 4.2 and 4.3, both the current and voltage were given such that the load resistance could be determined. Analysis of this data with MCTFE was then performed using the associated load resistance as the boundary condition. Plotting the MCTFE results for the these cases (including addition points taken at 0.87 volts) with the Topaz-II data as before then yielded the results of Figure 4.3 below. Interestingly enough there is now very good agreement. As can be determined by comparing Figures 4.2 and 4.3, MCTFE expects there to be a lower performance at a thermal input power 3150 Watts if a constant current of 153 A is assumed than would be the case for 204 A. However, MCTFE agrees with the data at 204 A but not at 153 A. At the low thermal power input levels the situation is reversed. Now, MCTFE overpredicts the electrical power if it is assumed that 153 A is the current, but agrees if 118 A (the current at 2050 W and 0.87 V) is the current. In other words, if the Topaz data of Figures 4.1, 4.2 and 4.3 were "truly" generated at 153 A, then the total Topaz data (i.e., the scatter data of Figures 4.1, 4.2 and 4.3 and the tabular data of Tables 4.2 and 4.3) are saying that the TFE will respond the same no matter what the current, or equivalently the load resistance, is (at least over the load resistance range corresponding to 118 A to 204 A). However, it is not believed that this is the case, but, that the Topaz-II data of Figures 4.1, 4.2 and 4.3 were not generated at a constant 153 A over the entire range of results as were seemingly reported [5]. It should also be noted that the data of Tables 4.2 and 4.3 did involve some cases at 153 A, and that the MCTFE results of Figure 4.2 "coincidentally" crossed the Topaz-II data at the input power levels corresponding to these cases of 153 A.

Figure 4.4 below is also provided to illustrate how the MCTFE results compared specifically to the data of Tables 4.2 and 4.3. The comparison provide by this figure doesn't appear to be as good as the Figure 4.3 results seemed to indicate. However,

Figure 4.4 does not indicate the degree of data scatter that was experimentally observed, which can be seen with Figure 4.3.

Table 4.2 Topaz-II Data With Electric Heating [5]

TFE Output Power (W)	TFE Output Voltage (V)	TFE Output Current (A)	TFE Input Thermal Power (W)
103	0.87	118	2050
118	0.87	136	2250
132	0.87	153	2450
147	0.87	170	2600

Table 4.3 Topaz-II Data With Nuclear Heating [5]

TFE Output Power (W)	TFE Output Voltage (V)	TFE Output Current (A)	TFE Input Thermal Power (W)
132	0.87	153	2600
147	0.87	170	2750
162	0.87	187	2950
177	0.87	204	3150

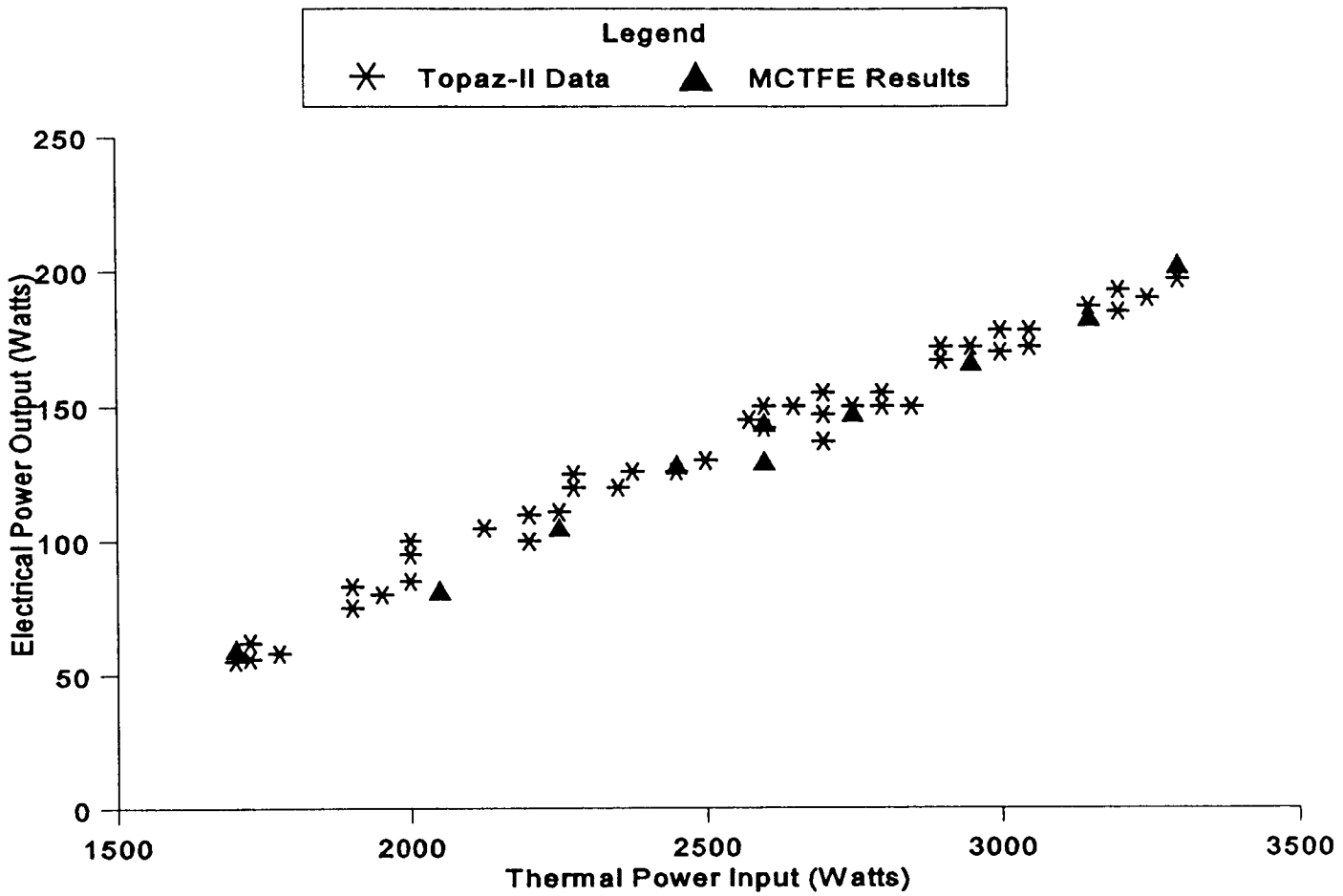


Figure 4.3 Comparison of Topaz-II data with MCTFE code results at 0.87 volts

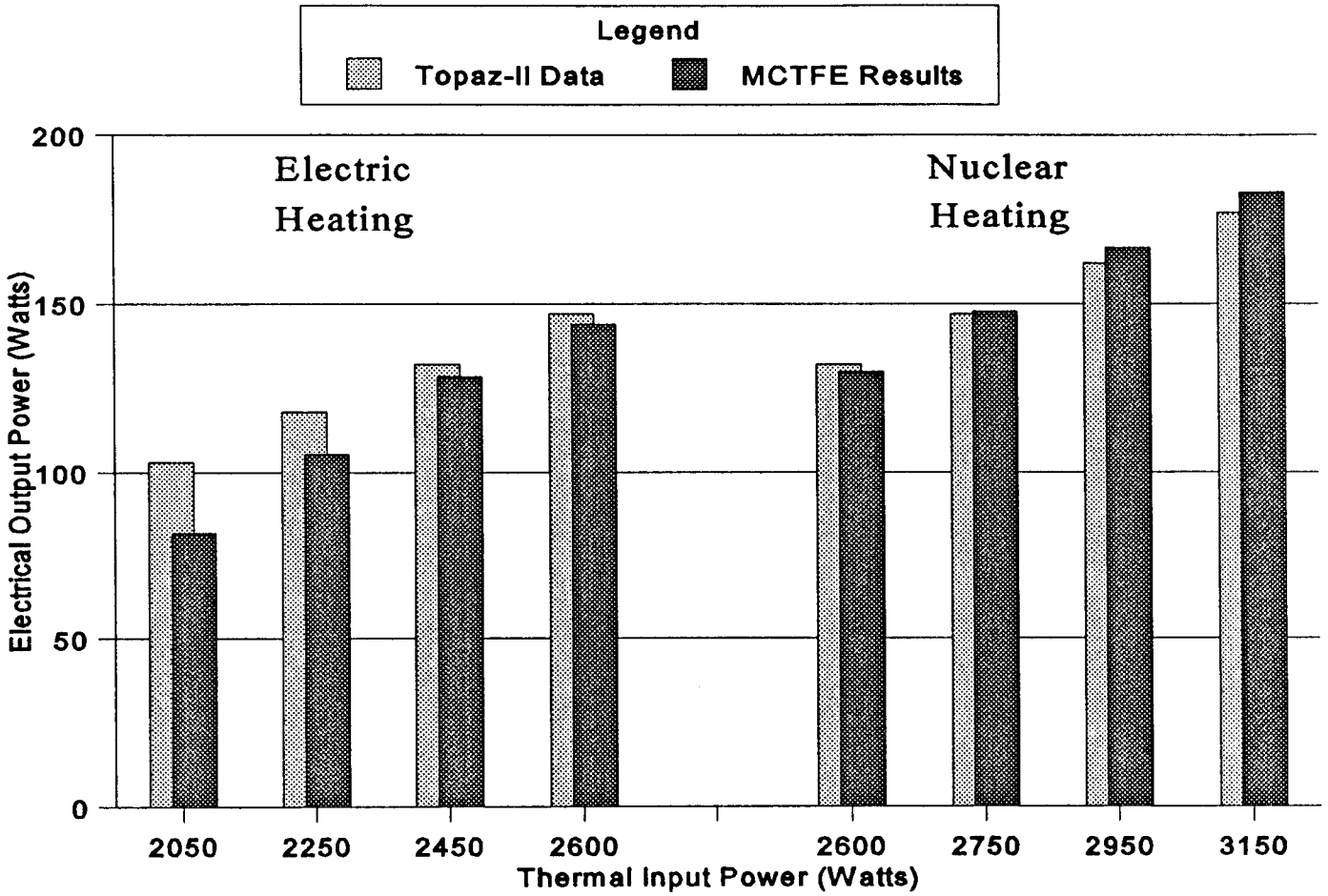


Figure 4.4 Comparison of specific Topaz-II data with MCTFE code results

The results between the Topaz-II experimental data and the MCTFE code, provided the correct boundary condition situation is observed, were in very good agreement. The assumptions and methods of the code are therefore considered to be reasonable as the predicted electrical power outputs by MCTFE were within the scatter of the measured data observed by the Topaz-II experiments (about 10-15%). Because the electrical performance of a TFE is closely related to the temperatures of the electrodes, thermionic agreement would also imply system temperature agreement.

4.2.2 MCTFE Comparison with General Atomics Data

The evaluation of MCTFE's multicell dependent routines is made by comparison to the experimental results obtained by General Atomics in their Thermionic Fuel Element Verification Program [16]. Because of the unique testing procedures used by General Atomics, a definitive benchmark comparison was not possible. Lack of precise information of the thermal input power to the TFE and the cesium pressure in the gap, as well as a different cooling configuration than what MCTFE was designed for, were a few of the difficulties. Nonetheless, a general comparison was made and is considered of value to confirm the proper functioning of MCTFE with multicell TFEs, and to provide some degree of confidence in the code results.

The particular configuration tested was the 3H5 TFE developed by General Atomics. This TFE is a three cell device that was tested in a Triga reactor. Instead of liquid metal (NaK) forced convection cooling of the TFE (as assumed with MCTFE), the 3H5 TFE was immersed in the reactor pool to be cooled by natural circulation. An additional modeling difficulty was the different cesium reservoir construction used by the 3H5 TFE. MCTFE assumes a liquid cesium reservoir in order to calculate the cesium pressure in the gap. The 3H5 TFE utilized a graphite reservoir to supply the cesium to the gap and it was not known how to correlate the given graphite reservoir temperature to an equivalent liquid reservoir temperature.

In order to make comparisons an inlet NaK coolant temperature and flowrate was assumed and adjusted until the collector temperatures were approximately as given in the

data [25]. In this way, a comparison could be made from the collector inward between the experimental results and MCTFE. The thermal input power levels were inferred by General Atomics based on the temperatures of the electrodes and assumed loss rates, therefore; these values were used as input to MCTFE. To determine the cesium pressure in the interelectrode gap the cesium reservoir temperature of 605 K as reported for the 1H3 TFE (a one cell TFE) was used. The 1H3 TFE, which was tested at too high a temperature to model with MCTFE, utilizes a liquid cesium reservoir as assumed in MCTFE.

Using the preceding assumptions, the 3H5 TFE was modeled with MCTFE. Table 4.4 compares the results achieved with MCTFE to the reported test data obtained by General Atomics. Although the exact thermal power input and cesium reservoir temperatures were not known, the approximate agreement displayed is considered to offer reasonable assurance of the MCTFE code and methods. To illustrate the critical effect of the cesium reservoir temperature, additional results are shown in the table with the reservoir temperature set at 600 K.

Table 4.4 Comparison of MCTFE results to General Atomics 3H5 experimental data (CT = Cesium Reservoir Temperature)

	MCTFE (CT = 600 K)	MCTFE (CT = 605 K)	General Atomics Data [24]
TFE Total Current (Amps)	100.4	117.2	107.6
TFE Total Voltage (Volts)	1.257	1.467	1.347
TFE Power (Watts)	126.3	171.9	144.9
TFE Efficiency (%)	6.99	9.52	8.03

The comparison study performed with MCTFE on the 3H5 TFE developed by General Atomics indicates proper operation of the code's multicell dependent routines. However, more precise data and further studies with systems tested in a configuration more compatible with MCTFE's design methodology should be conducted as such information becomes available. Despite the difficulties encountered it is believed that the results achieved have offered some support of MCTFE and the accuracy of its methods.

4.3 PDCALC SUBROUTINE RESULTS

The PDCALC subroutine was created to determine the power density distribution in single or multicell cylindrical TFEs. In developing PDCALC, the subroutine's results were compared to MCNP data for both radial and axial variation. This section presents these results that illustrate how a relatively simple routine can accomplish a good approximation to the true power density distribution.

To perform the comparison with MCNP, a six cell TFE was created. The geometry of this TFE was fully described to the MCNP code so the power generated within the TFE at all positions could be determined. The PDCALC subroutine was then used to determine the power generation densities and the results are compared to those achieved with MCNP. Figures 4.5 and 4.6 below illustrate the radial variation of the power density for an internal node and a surface node respectively. Figure 4.7 displays the axial variation of the power generated in the fuel. For this latter plot the radial node powers were summed for each axial position to illustrate the overall axial variation. In each of these figures higher power densities are seen in the fuel surface regions consistent with the self-shielding concept. This is seen to occur both at the outer radial regions and axially at the end face surfaces. In view of the rough approximations assumed the results are considered to be very good for a "first-order" type of determination.

Although two parameters were "tuned" by MCNP, namely the effective energy of the neutrons entering the fuel and a parameter accounting for the degree of TFE end reflection, it is still believed to be a useful comparison. No curve fitting using only two parameters could ever hope to predict the power density distribution within a collection of

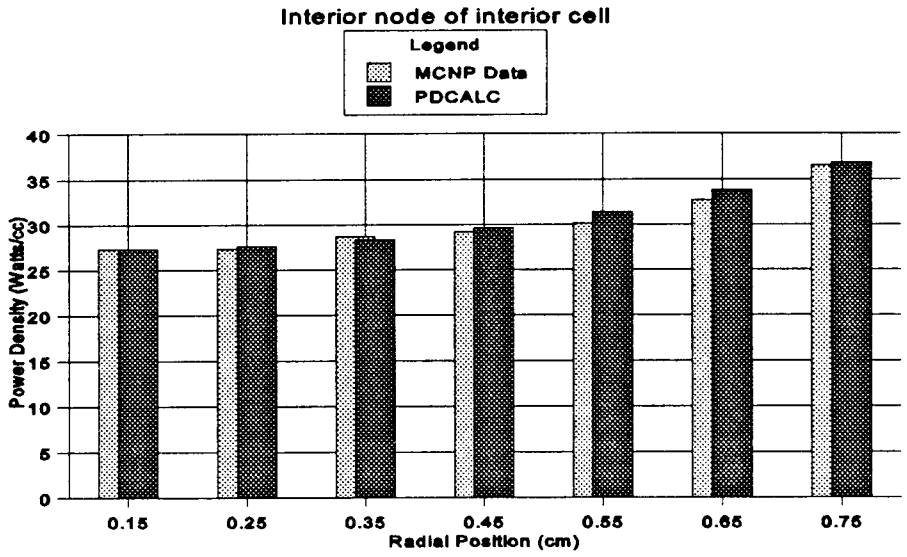


Figure 4.5 MCNP comparison to PDCALC results for interior node of interior cell

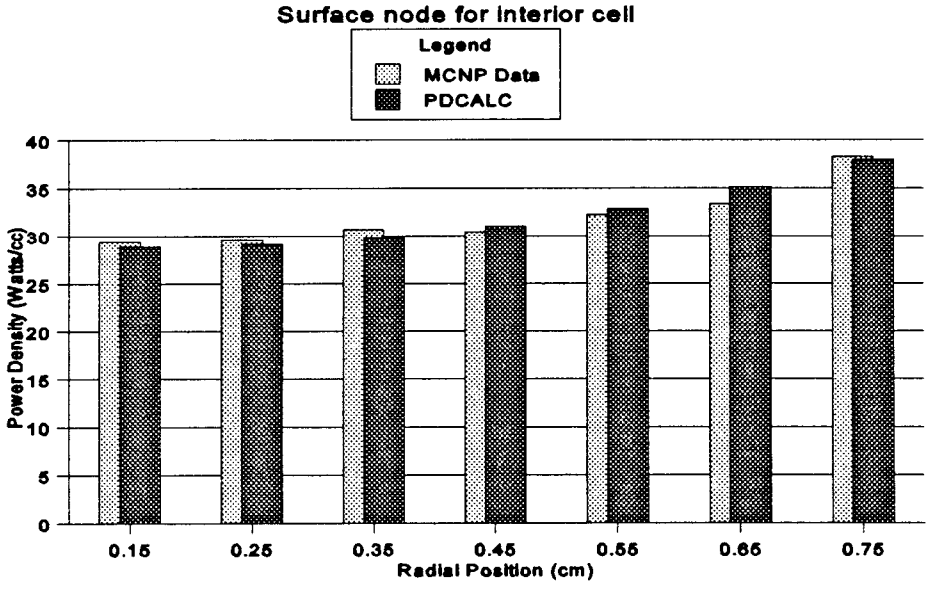


Figure 4.6 MCNP comparison to PDCALC results for surface node of interior cell

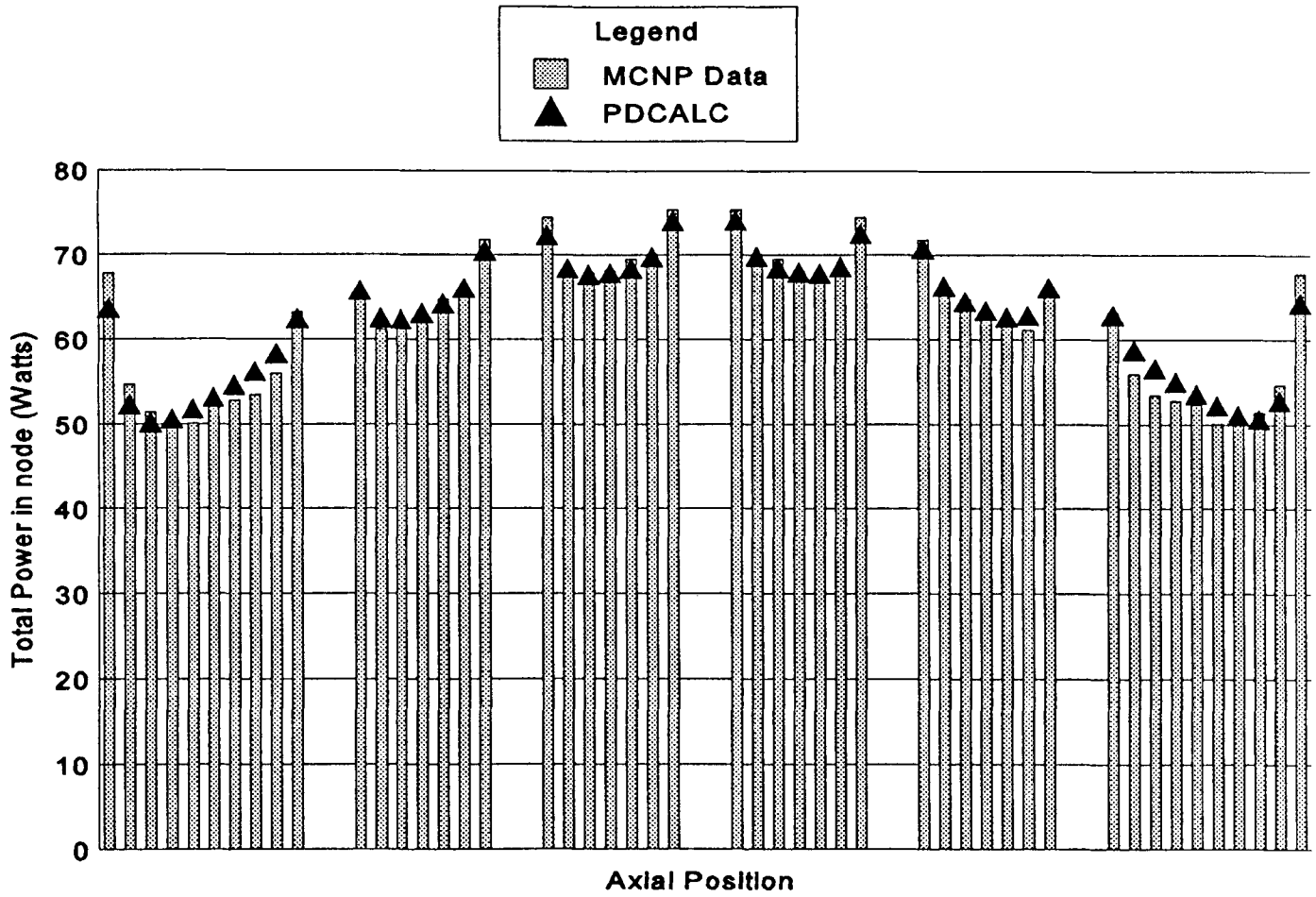


Figure 4.7 Axial comparison of MCNP data and PDCALC results

cylindrical fuel segments as complicated as the achieved results indicate. Therefore, the MCNP data does serve as an internal consistency check against the PDCALC subroutine even though some of the boundary value factors were adjusted to the MCNP results. It is also believed to be a useful subroutine to have in MCTFE because a reasonable approximation of the power densities within the fuel can be obtained without requiring a time consuming MCNP run. For a given general TFE reactor system, different TFE configurations can be analyzed based on a single MCNP run performed to adjust the parameters for that configuration.

Another comparison between MCNP and the PDCALC subroutine was made with the single cell Topaz-II type TFE. For this comparison both MCNP generated power densities and use of the PDCALC option were used in MCTFE to analyze the TFE. To illustrate the results Figure 4.8 is provided. In this figure the peak fuel temperatures (which occur at the fuel/void interface of each axial position) are plotted. Also plotted in the figure are results obtained using the constant power and chopped cosine distribution descriptions.

The results in the figure indicate that assuming a constant radial power profile with the chopped cosine axial variation will conservatively over-estimate the peak fuel temperature of the Topaz-II type TFE by about 90 degrees K. This effect is understood by the fact that the constant radial profile assumes a greater than actual amount of heat is generated closer to the fuel center thus necessitating higher fuel temperatures. Also observed is the fact that constant power, both radially and axially, actually underestimates the peak fuel temperature as would be expected. The results for the MCNP and PDCALC options have yielded similar results again indicating the usefulness of the PDCALC approximation.

4.4 MCTFE PREDICTION OF TFE PERFORMANCE

In the course of developing and using the MCTFE code to model and analyze the performance of TFEs a noteworthy effect has been observed. Specifically, this is the critical nature of the cesium pressure that is maintained in the interelectrode gap. The

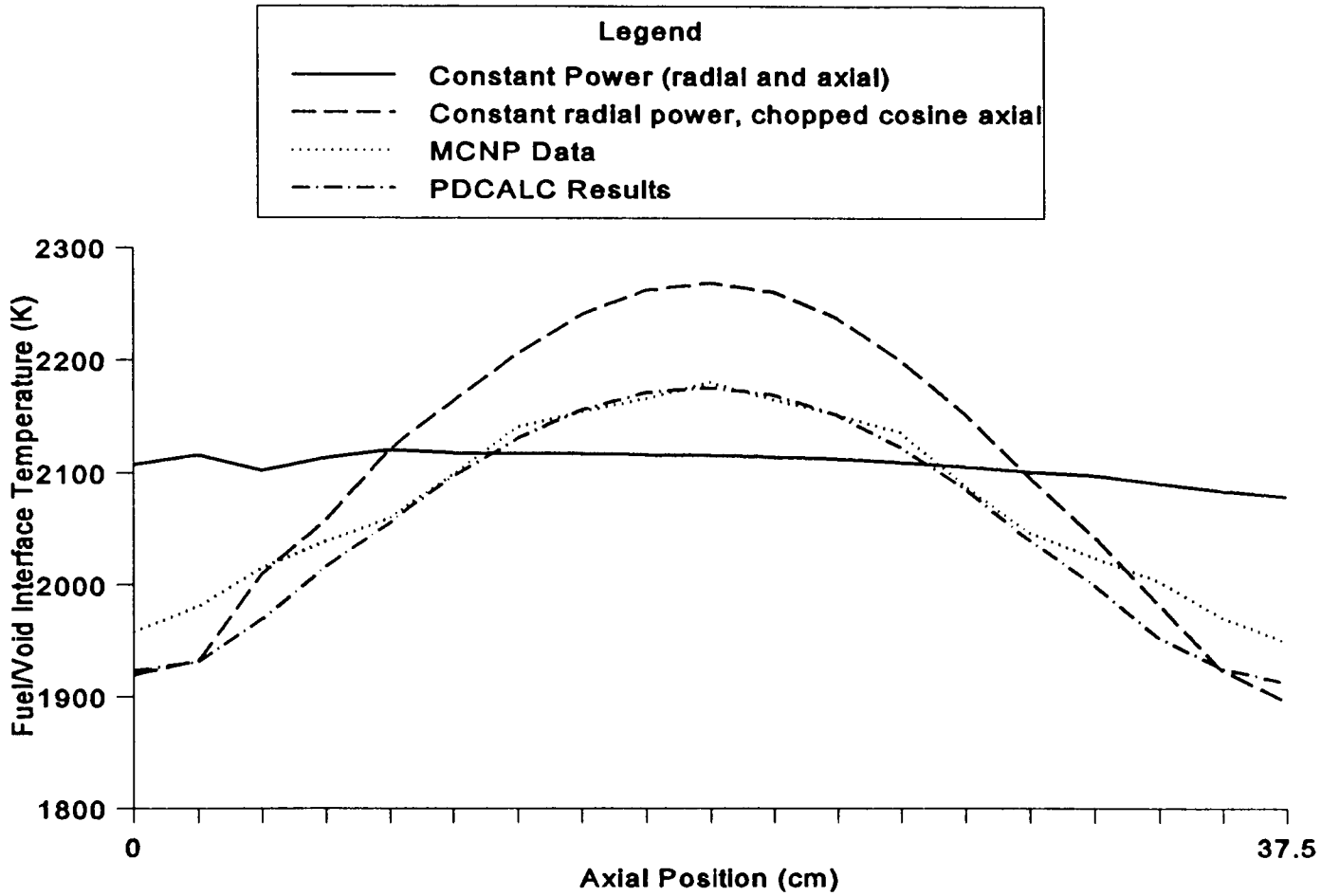


Figure 4.8 Comparison of Fuel/Void interface Temperatures with Power Density Description (Topaz-II type TFE at 3150 Watts)

cesium gas is required to form positive ions that neutralize the concentration of electrons that are traveling through the gap. If the cesium pressure is too low for a given current insufficient cesium is present to reduce the large counter electric fields that develop. To demonstrate this effect Figures 4.9 and 4.10 have been prepared to show the electrical power dependance upon the cesium pressure (or cesium reservoir temperature).

Figure 4.9 illustrates how the optimum cesium reservoir temperature varies with input power level as predicted by MCTFE. This figure was prepared using the Topaz-II type TFE. As seen in the figure the higher power levels, which will tend to generate more current, require a higher cesium reservoir temperature. This is the result expected and these optimum values of cesium reservoir temperature were used when the Topaz-II benchmarking studies were performed as previously discussed.

To illustrate TFE performance when non-optimum cesium pressures or reservoir temperatures are used, Figure 4.10 is given. The data of this figure was generated from the General Atomics 3H5 multicell TFE by comparing the results achieved with MCTFE using different cesium reservoir temperatures. These results indicate how the performance of the TFE will drop off when not at the optimum pressure, especially when the pressure is too low. It should also be noted how only small changes in the cesium reservoir temperature will produce significant changes in the TFEs response.

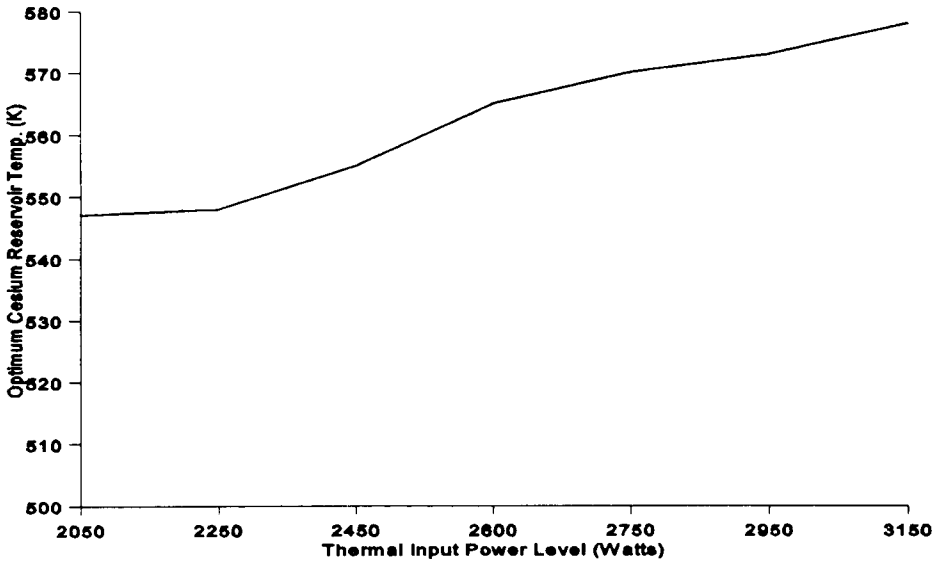


Figure 4.9 Optimum Cesium Pressure vs. Thermal Input Power for Topaz-II TFE

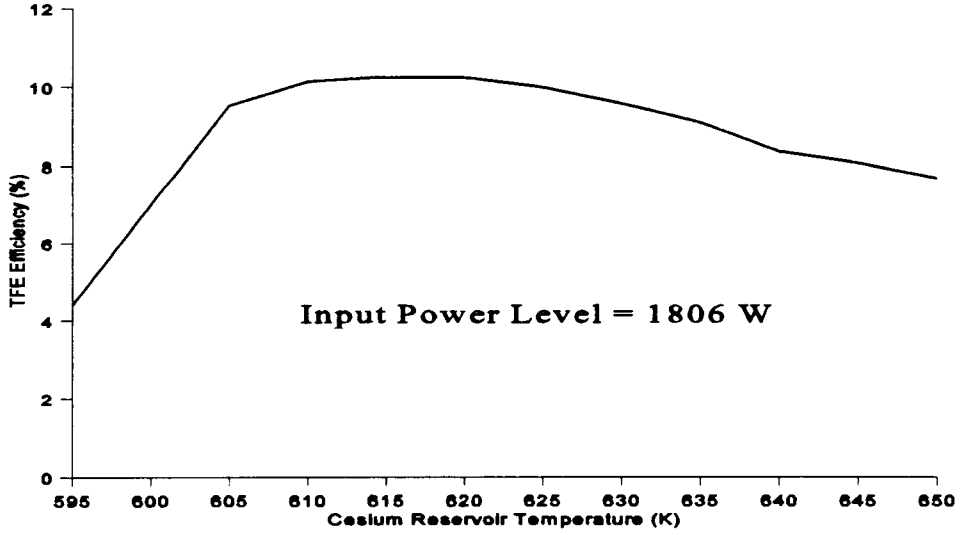


Figure 4.10 General Atomics 3H5 TFE response vs. Cesium Reservoir Temperature

CHAPTER 5. CONCLUSIONS AND RECOMMENDATIONS

This report has been written to describe the recent modifications of TFEHX that have been made to expand and improve its capabilities. This new code, which has been given the name MCTFE (for multicell thermionic fuel element), is able to perform coupled thermal-hydraulic and thermionic analysis of single or multicell TFEs. MCTFE has been specifically designed to analyzed cylindrical liquid metal (NaK) cooled cesium vapor TFEs. The MCTFE code has been developed with flexibility that includes: user defined system node sizes, fuel power density description options, and electrical boundary condition choices.

To benchmark MCTFE, Topaz-II and General Atomics experimental data have been compared with results obtained from MCTFE. This report has presented these results in the preceding chapter. When compared to Topaz-II experimental data, MCTFE has shown very good agreement provided the correct boundary condition situation is observed. For the General Atomics data precise modeling was not possible, but the approximations used have indicated that MCTFE can function with multicell TFEs and produce reasonable results.

Future developments that can still be made to the MCTFE code include: the addition of more material choices for the electrodes, insulation and cladding; allowance for a variable number of TFE regions; and modifications that account for the effects of single cell TFE spacers. In addition, incorporating MCTFE into a total reactor system code, that predicts the performance of an entire thermionic reactor, could be undertaken. This would enable an effective tri-coupling of the thermal-hydraulic, thermionic and neutronic considerations to take place, thus giving a total performance evaluation of a thermionic reactor system. The PDCALC subroutine developed by this study has in effect been a first step in this direction.

The critical nature of the cesium pressure in the interelectrode gap on a TFEs performance emphasizes the need to further investigate this important factor. Allowance for internal graphite type reservoirs (especially in the area of multicell TFE development)

would be a valuable addition to MCTFE. As currently written, MCTFE could be easily modified to allow for a separate cesium reservoir temperature to be specified for each cell.

Additional improvements that might also be added to MCTFE may include expanding the cooling mode options for the TFE. Currently, it is assumed that forced convection by the liquid metal NaK is the cooling method being used. Alternative mechanisms (e.g., natural circulation) could also be added to expand the range of application of MCTFE.

The major effort of this study was in the development of the MCTFE code. While some analysis has been performed with the code, further investigations into the design and operation of TFEs can now be performed. Comparison studies that examine the potential use of single or multicell TFEs with a given design criteria can now be undertaken with the use of the MCTFE code. Parametric examinations can also be addressed to further explore the performance of single or multicell TFEs for various configurations.

When more experimental data become available, MCTFE should continue to be tested and modified as necessary to improve its accuracy and document its performance. MCTFE has intended to take another step forward from the initial achievements of TFEHX. As one who has benefited greatly from the starting point provided by TFEHX, the author here would encourage any interested individuals to likewise improve, modify, or alter the methods used by MCTFE as future developments make possible or future needs dictate.

REFERENCES

1. Pawlowski, R.A., and Klein, A.C., "Coupled Thermionic and Thermal-hydraulic Analysis of Thermionic Fuel Elements", 26th Intersociety Energy Conversion Engineering Conference, Vol. 3, pp. 99, Boston, MA, 1991.

2. Klein, A.C., Lee H. H., Lewis, B. R., Pawlowski, R.A. and Abdul-Hamid, S. "Advanced Single Cell Thermionic Reactor System Design Studies", OSU-NE-9209, Oregon State University, Corvallis, OR, 1992.

3. Chapra, S.C., and Canale, R.P., "Numerical Methods for Engineers", McGraw-Hill, Inc., New York, 1988.

4. McVey, J.B., "Preliminary Technical Report - CYLCON Semi-2D Cylindrical Converter Model", Rasor Associates, Inc., Sunnyvale, CA, 1990.

5. Ponomarev-Stepnoi, N.N. et al., NPS TOPAZ-II Description, JV Inertek report, Moscow, Russia, 1991.

6. Samuelson, A.L., "Thermionic Fuel Element Verification Program - Summary of Results", Proceedings of 7th Symposium on Space Nuclear Power Systems, CONF-900109, Albuquerque, NM, 1990.

7. Hatsopoulos, G.N. and Gyftopoulos, E.P., "Thermionic Energy Conversion, Vol. I: Processes and Devices", MIT Press, Massachusetts, 1973.

8. Hatsopoulos, G.N. and Gyftopoulos, E.P., "Thermionic Energy Conversion, Vol. II: Theory, Technology, and Application", MIT Press, Massachusetts, 1973.

9. Angelo Jr., J.A. and Buden, D., "Space Nuclear Power", Orbit Book Company, Inc., Malabar, FL, 1985.

10. Ranken, W. A., "Historical View of Thermionic Reactor Development", Proceedings of 7th Symposium on Space Nuclear Power Systems, CONF-900109, Albuquerque, NM, 1990.

11. Richardson, O.W., "Some Applications of the Electron Theory of Matter", Phil. Mag., 23: 594-627, 1912

12. Preece, W.H., "On Peculiar Behavior of Glow-Lamps When Raised To High Incandescence", Proc. Roy. Soc. Ser. A, 38: 219, London, 1885.

13. Schlichter, W., "Die spontane Electronenemission gluhender Metalle und das gluhelek-trische Element, Ann. Physik, 47(4): 573-640, 1915

14. Hernqvist, K.G., Kanefsky, M., and Norman, E., Thermionic Energy Converter, RCA Rev., 19: 244, 1958.

15. Ponomarev-Stepnoi, N.N. et al., "Comparative Analysis of Concepts of Single-Cell and Multi-Cell TFE of Thermionic NPS", Proceedings of 10th Symposium on Space Nuclear Power and Propulsion, CONF-930103, Albuquerque, NM, 1993.

16. Houts, M.G., Wharton, W.R., Begg, L.L., and Lawrence, L.A., "The Thermionic Fuel Element Verification Program:Technical Progress and Future Plans", Proceedings of 10th Symposium on Space Nuclear Power and Propulsion, CONF-930103, Albuquerque, NM, 1993.

17. Morris, D.B., "The Thermionic System Evaluation Test (TSET): Descriptions, Limitations, and the Involvement of the Space Nuclear Power Community", Proceedings of 10th Symposium on Space Nuclear Power and Propulsion, CONF-930103, Albuquerque, NM, 1993.

18. Bell, G.I., and Glasstone, S., "Nuclear Reactor Theory", Krieger Publishing Company, Malabar, FL, 1985.

19. Duderstadt, J.J., and Hamilton, L.J., "Nuclear Reactor Analysis", John Wiley & Sons, Inc., New York, N.Y., 1976.

20. Kitrilakis, S., and Meeker, M., "Experimental Determination of the Heat Conduction of Cesium Gas", Advanced Energy Conversion, Vol. 3, pp. 59-68, Pergamon Press 1963.

21. McVey, J.B., "Planar Converter Standard Model Documentation - Supplementary Description of TECMDL Converter Physics", E-563-002-B-063087, Rasor Associates, Inc., Sunnyvale, CA, 1990.

22. McVey, J.B., "TECMDL - Ignited Mode Planar Converter Model", E-563-004-C-082988, Rasor Associates, Inc., Sunnyvale, CA, 1990.

23. Faust, O. ed., Sodium-NaK Engineering Handbook, Vol. 1, Gordon and Breach, New York, N.Y., 1972.

24. Brophy J.J., "Basic Electronics for Scientists" 4th Edition, McGraw-Hill Book Company, New York, N.Y., 1983.

25. Begg L., FAX Correspondence containing Incore TFE Data, General Atomics, San Diego, CA, October, 11, 1993.

26. Begg L., FAX Correspondence containing test information on TFE-1H2, 1H3, and 3H5, General Atomics, San Diego, CA, December, 18, 1993.

27. Drees L., Letter Correspondence containing General Atomics TFE blueprints, General Atomics, San Diego, CA, December, 21, 1993.

APPENDIX A: MCTFE SOURCE CODE

```
      Program MCTFE
C
******* The following program is written to determine the coupled ******
******* thermal-hydraulic and thermionic performance of cylindrical ****
******* NaK cooled, single or multi-celled thermionic fuel elements ****
************************************************************
*     INPUT FILE NAME : 'MCTFE.INP'

*     DESCRIPTION OF THE INPUT DECK:

*

*                    ID            DESCRIPTION

* IMAGE 1 (A80)
*              1.1   TITLE      Title for the run
*
* IMAGE 2
*              2.1   Tin        Inlet temperature of NaK coolant
*                                 (K)
*
*              2.2   Mdot       Mass flow rate of the NaK coolant
*                                 (Kg/S)
*
*              2.3   W          Weight fraction of K in the NaK
*                                 coolant
*
*              2.4   Tr         Temperature of the cesium
*                                 reservoir (K)
*
*              2.5   BCSet       Boundary condition  (1, 2, or 3)
*                              1 - Load resistance to be specified
*                              2 - Total current to be specified
*                              3 - TFE voltage to be specified
*
*              2.6   BP         Value of boundary parameter
*                                 (in milli-ohms if BCSet = 1)
*                                 (in Amps if BCSet = 2)
*                                 (in Volts if BCset = 3)
*
*              2.7   Pwth       Total fuel thermal power of the
*                                 TFE (Watts)
*
*              2.8   EFF        Expected electrical power efficiency
```

```
*                          (used only for temperature and
*                           voltage initialization routines)
*
*
* IMAGE 3
*          3.1  NmCell     Number of Cells in the TFE
*
*          3.2  IHKUP      1 or 2
*                             (1 if current connections are at
*                              the same end)
*                             (2 if current connections are at
*                              opposite ends)
*
*               (Note: IHKUP must be 2 if NmCell > 1)
*
*          3.3  FIN        Number of radial interior nodes
*                            for each axial position of fuel
*
*          3.4  PwrSet     1, 2, 3, 4, or 5
*
*               (1 for constant axial power)
*               (2 for a chopped sinusoidal profile)
*               (3 for a single axial column of tabular
*                 values such as mcnp output)
*               (4 for FIN+1 columns of tabular values, i.e.
*                 for each axial and radial region of fuel)
*               (5 for PDCALC solution)
*
*     ************************************************
*     * If NmCell = 1, enter zero  for the remaining                *
*     * entry in IMAGE 3 and all entries in 4 and 5                 *
*     ************************************************
*
*          3.5  TPM        Intercell transition piece
*                            material number
*
* IMAGE 4                  (Only required if NmCell > 1, if
*                            NmCell = 1  enter zeros)
*
*          4.1  CSD        Cell separation distance (cm)
*                            (distance from top of fueled
*                            region of one cell to the
*                            bottom of fueled region of the
*                            next cell)
```

```
*
*          4.2  NFEL     Non-fueled emitter length (cm)
*
*          4.3  NFCL     Non-fueled collector length (cm)
*
*          4.4  ESL      Length of emitter stem (cm)
*
*          4.5  TPL      Average length of transition piece
*
*                        (Note: NFEL+NFCL+ESL+TPL=CSD)
*
* IMAGE 5                (Only required if NmCell > 1, if
*                          NmCell = 1 enter zeros)
*
*          5.1  EST      Thickness of emitter stem (cm)
*
*          5.2  ESR      Average radius of emitter stem (cm)
*
*          5.3  TPT      Average thickness of transition
*                          piece (cm)
*          5.4  TPR      Average radius of transition piece
*                          (cm)
*
* IMAGE 6          Do for all cells I=1,NmCell
*
*          6.1  CL(I)    Cell length for the Ith cell (cm)
*                          (length of fueled region)
*
*          6.2  CAN(I)   Number of axial nodes for the
*                          Ith cell
*
* IMAGE 7          Do for all regions (I=1,7)
*
*                  Regions to be specified
*                     1) fuel
*                     2) emitter
*                     3) emitter-collector gap
*                     4) collector
*                     5) insulator
*                     6) cladding
*                     7) coolant channel
*
*          7.1  IR(I)    Inside radius of region (cm)
*
```

```
*
*          7.2  OR(I)     Outside radius of region (cm)
*
*
* IMAGE 8
*          8.1  M1        Material number for fuel
*
*          8.2  M2        Material number for emitter
*
*          8.3  M3        Material number for gap
*
*          8.4  M4        Material number for collector
*
*          8.5  M5        Material number for insulation
*
*          8.6  M6        Material number for cladding
*
*               Identification numbers are defined as
*               follows:
*
*                    #    Material
*                    1    UO2
*                    2    Tungsten
*                    3    Niobium
*                    4    Nb1Zr
*                    5    Molybdenum
*                    6    Stainless steel 316
*                    7    Cesium
*                    8    Al2O3
*                    9    Tantalum
*
*
* IMAGE 9 (1F10.4)    (Only required if PwrSet = 2, 3, 4, or 5)
*
*          9.1  PktoAv    (Enter If PwrSet = 2 or 5)
*                            Ration of Peak to Average linear
*                            power generation values
*           or
*
*         *9.1  Pwr(K)     (Enter If PwrSet = 3)
*                            Power density for each axial fuel
*                            region (Mev/g) enter values
*           or               in top to bottom order
*
```

```
*          *9.1  Pwr       (Enter If PwrSet = 4)
*             (FIN+1,K)    Power density for each axial and
*                             radial region of fuel (Mev/g) enter
*                             values in outer (emitter side) to
*                             inner, top to bottom order
*
*           9.2  E         (Enter If PwrSet = 5)
*                           Effective energy (in eV) of neutrons
*                           used in PDCALC subroutine
*
*           9.3  ARP        (Enter If PwrSet = 5)
*                             Axial reflection parameter used
*                             in PDCALC subroutine
*
*           9.4  FuelEnr    (Enter If PwrSet = 5)
*                             Fuel Enrichment (%)
*
*           9.5  Fuelden    (Enter If PwrSet = 5)
*                             Fuel Density
*                             (fraction of theoretical)
*
*             *(Do for all axial regions of all cells, the
*               number of regions for each cell will be one
*               less than the number of axial nodes for that
*               cell, enter zeros for intercell regions)
*
*
************************************************************************
      IMPLICIT DOUBLE PRECISION (A-H)
      IMPLICIT DOUBLE PRECISION (P-Z)
      CHARACTER*80 Title
      REAL*8 Kcond1,Kcond2,Kcond3,Kcond4,KC1(11),KC2(11),KC3(11)
      REAL*8 KC4(11),IR(10),OR(10),CL(10),Mdot,AREA(10),R(20,100)
      REAL*8 Jden(100),V(100),VR(100),Itot,T(20,100),q(11,100)
      REAL*8 CONIN(10),CONOUT(10),D(100),DZ(100),FZ(100),QEC(100)
      REAL*8 RPwr(11,100),VOL(11,100),Pwr(100),TP(100),IL,Reff(10)
      REAL*8 NFEC,NFEP,NFCP,NFEL,NFCL,CIC(10),RIC(10),Tm(100)
      REAL*8 QRAD(100),QCOND(100),IC(100),IE(100),RI(11),RO(11)
      REAL*8 AR1(11),AR2(11),AR3(11),Kcond,MAXERR,qel(100)
      REAL*8 qelold(100),Icell(10),QCH(100)
      REAL*8 F(10,100),FR(10,100),RF(10)
      REAL*8 NP(10,100),Jgs(100),TNP(100)
      INTEGER M1,M2,M3,M4,M5,M6,CAN(10),HIGH,PwrSet,TPM
      INTEGER FIN,TIT,TITmax,BCSet
```

```
      DATA SIG /5.67D-12/
      DATA Phi0, TITmax /4.9D0, 1500/
      DATA CONVERGT /0.05/
      COMMON /Cooldat/ OR,IR,Mdot,PI,W,AREA
      COMMON /Inceldat/ TPP,ESP,NFEP,NFCP,TPM,M2,M4
      PI=DACOS(-1.D0)
      ITA=0.05
      EA=0.05
      AFOLD=1.
      SUMOLD=1.
      BETA=0.5
C
C **** Reading the input deck and echoing to the output file ***********
C
      OPEN(1,FILE='MCTFE.INP')
      OPEN(2,FILE='MCTFE.OUT')
      WRITE(2,9)
      READ(1,11)Title
      WRITE(2,11)Title
      READ(1,*)Tin, Mdot, W, Tr, BCSet, BP, Pwth, EFF
      WRITE(2,12)Tin, Mdot, W, Tr, BCSet, BP, Pwth, EFF
      IF(BCSet.eq.1)THEN
        RL=BP*1.0D-3
      ENDIF
      IF(BCSet.eq.2)THEN
        IL=BP
      ENDIF
      IF(BCSet.eq.3)THEN
        VL=BP
      ENDIF
      READ(1,*)NmCell,IHKUP,FIN,PwrSet,TPM
      WRITE(2,13)NmCell,IHKUP,FIN,PwrSet,TPM
      READ(1,*)CSD,NFEL,NFCL,ESL,TPL
      WRITE(2,14)CSD,NFEL,NFCL,ESL,TPL
      READ(1,*)EST,ESR,TPT,TPR
      WRITE(2,15)EST,ESR,TPT,TPR
      DO 10 I=1,NmCell
        READ(1,*)CL(I), CAN(I)
        WRITE(2,16)CL(I), CAN(I)
 10   CONTINUE
      DO 25 I=1,7
        READ(1,*)IR(I),OR(I)
        WRITE(2,17)IR(I),OR(I)
 25   CONTINUE
```

```
      READ(1,*)M1,M2,M3,M4,M5,M6
      WRITE(2,18)M1,M2,M3,M4,M5,M6
      IF(PwrSet.eq.2) THEN
        READ(1,*)PktoAv
        WRITE(2,19)PktoAv
      ENDIF
      IF(PwrSet.eq.5) THEN
        READ(1,*)PktoAv,E,ARP,FuelEnr,Fuelden
        WRITE(2,20)PktoAv,E,ARP,FuelEnr,Fuelden
      ENDIF
      Kmax=0
      TFL=0.D0
      DO 30 I=1,NmCell
      Kmax=Kmax+CAN(I)
      TFL=TFL+CL(I)
   30 CONTINUE
      IF(PwrSet.eq.3) THEN
        DO 40 K=Kmax-1,1,-1
          READ(1,*)Pwr(K)
          WRITE(2,19)Pwr(K)
   40   CONTINUE
      ENDIF
      IF(PwrSet.eq.4) THEN
        DO 45 K=Kmax-1,1,-1
          READ(1,*)(RPwr(I,K),I=FIN+1,1,-1)
          WRITE(2,21)(RPwr(I,K),I=FIN+1,1,-1)
   45   CONTINUE
      ENDIF
    9 FORMAT(80('*'),'ECHO OF INPUT DECK'/80('*'))
   11 FORMAT(A80)
   12 FORMAT(4F10.4,1I10,3F10.4)
   13 FORMAT(5I10)
   14 FORMAT(5F10.4)
   15 FORMAT(4F10.4)
   16 FORMAT(1F10.4,1I10)
   17 FORMAT(2F10.4)
   18 FORMAT(6I10)
   19 FORMAT(1F10.4)
   20 FORMAT(5F10.4)
   21 FORMAT(11F10.4)
      CONIN(1)=0.
      CONOUT(NmCell)=0.
C
      IF(NmCell.eq.1)THEN
```

```
        ALPHA = 1.75
      ELSE
        ALPHA = 0.5
      ENDIF
C **** Determination of input dependent parameters **********************
C
C **** Determining the thickness of each region ****
C
      thf=OR(1)-IR(1)
      the=OR(2)-IR(2)
      thg=OR(3)-IR(3)
      thc=OR(4)-IR(4)
      thi=OR(5)-IR(5)
      thcl=OR(6)-IR(6)
C
C **** Determination of cross-sectional areas for each region ****
C
      DO 50 I=1,7
      AREA(I)=PI*(OR(I)**2-IR(I)**2)
 50   CONTINUE
C
C **** Determining intercell constants ****
C
      IF(NmCell.ne.1) THEN
        TPP=2*PI*TPR*TPT/TPL
        ESP=2*PI*ESR*EST/ESL
        NFEP=AREA(2)/NFEL
        NFCP=AREA(4)/NFCL
      ENDIF
C
C **** Determining initial conditions for voltage subroutine ****
C
      IF(BCSet.eq.2)THEN
        RL=EFF*Pwth/IL**2
      ENDIF
      IF(BCSet.eq.3)THEN
        RL=VL**2/EFF/Pwth
      ENDIF
C
C **** Determining the cesium pressure ****
C
      PCS=2.45D8*DEXP(8910.D0/Tr)/(Tr)**0.5D0
C
C **** Determining the axial distance for each node ****
```

```
C
      FZ(1)=0.D0
      LOW=1
      HIGH=CAN(1)-1
      DO 55 I=1,NmCell
C
C **** Determination of DZ(K) as will be used by the temperature ****
C **** initialization equations and PDCALC and D(K) as will be used ****
C **** by remainder of the code for the axial distance of each node ****
C
        DO 60 K=LOW,HIGH
          DZ(K)=CL(I)/(CAN(I)-1)
          D(K)=DZ(K)
          FZ(K+1)=FZ(K)+DZ(K)
   60   CONTINUE
        DZ(HIGH+1)=CSD
        FZ(HIGH+2)=FZ(HIGH+1)
        D(HIGH+1)=D(HIGH)/2
        D(LOW)=D(HIGH+1)
        LOW=LOW+CAN(I)
        HIGH=HIGH+CAN(I+1)
   55 CONTINUE
C
C **** Determination of fuel power (watts) vs axial position ****
C
C **** If PwrSet = 1 ****
C
      IF(PwrSet.eq.1) THEN
        DO 61 K=1,Kmax-1
           TP(K)=Pwth/TFL*DZ(K)
          IF(DZ(K).ne.0.)THEN
            q(1,K)=TP(K)/AREA(1)/DZ(K)
          ELSE
            q(1,K)=0.
          ENDIF
   61   CONTINUE
      ENDIF
C
C **** If PwrSet = 2 or 5 ****
C
      IF(PwrSet.eq.2.or.PwrSet.eq.5) THEN
        B=(Pwth/TFL)*(PktoAv-1.D0)/(1.D0-2.D0/PI)
        A=Pwth/TFL*PktoAv-B
        DO 63 K=1,Kmax-1
```

```
          TP(K)=A*DZ(K)+B*TFL/PI*(-DCOS(PI*FZ(K+1)/TFL)
     a           +DCOS(PI*FZ(K)/TFL))
          IF(DZ(K).ne.0.)THEN
            q(1,K)=TP(K)/AREA(1)/DZ(K)
          ELSE
            q(1,K)=0.
          ENDIF
   63   CONTINUE
      ENDIF
C
C **** If PwrSet = 3 ****
C
      IF(PwrSet.eq.3) THEN
        PwrSum=0.D0
        DO 65 K=1,Kmax-1
          PwrSum=PwrSum+Pwr(K)*DZ(K)
   65   CONTINUE
        DO 67 K=1,Kmax-1
          TP(K)=Pwr(K)*DZ(K)/PwrSum*Pwth
          IF(DZ(K).ne.0.)THEN
            q(1,K)=TP(K)/AREA(1)/DZ(K)
          ELSE
            q(1,K)=0.
          ENDIF
   67   CONTINUE
      ENDIF
C
C **** PDCALC Solution Method
*************************************************
C
      IF(PwrSet.eq.5)THEN
        Print*,'Entering PDCALC .....'
        ACC=1.75
        sigaU25=678*sqrt(0.0253/E)
        sigaU28=2.73*sqrt(0.0253/E)
        sigaO=0.0002*sqrt(0.0253/E)
        sigsU25=15*sqrt(0.0253/E)
        sigsU28=13.8*sqrt(0.0253/E)
        sigsO=4.2*sqrt(0.0253/E)
        Fuelden=0.95
        FuelEnr=0.95
        SIGS=(FuelEnr*sigsU25+(1-FuelEnr)*sigsU28+2*sigsO)*0.0223*
     a       Fuelden
        SIGA=(FuelEnr*sigaU25+(1-FuelEnr)*sigaU28+2*sigaO)*0.0223*
```

```
     a       Fuelden
          DIFCOFF=1./3./(SIGA+0.9887*SIGS)
          DR=thf/(FIN+1)
          DL=SQRT(DIFCOFF/SIGA)
          RF(1)=IR(1)
          DO 70 I=2,FIN+2
            RF(I)=RF(I-1)+DR
   70     CONTINUE
          DO 71 I=1,FIN+1
          DO 71 K=1,Kmax-1
            VOL(I,K)=PI*(RF(I+1)**2-RF(I)**2)*DZ(K)
   71     CONTINUE
          Do 72 K=1,Kmax
           Jgs(K)=A+B*SIN(PI*FZ(K)/TFL)
   72     CONTINUE
C
C **** Initialization of the field ****
C
          DO 73 I=1,FIN+2
          DO 73 J=1,Kmax
           F(I,J)=Pwth/TFL/AREA(1)
   73     CONTINUE
   75     CONTINUE
C
C **** Iterative solution of the field ****
C
          LOW=2
          HIGH=CAN(1)-1
          DO 74 K=1,NmCell
            DO 76 J=LOW,HIGH
              DO 77 I=2,FIN+1
              FR(I,J)=1/(2/DR**2+2/DZ(J)**2+1/DL**2)*((1/DR**2+1/2/RF(I)
     a              /DR)*F(I+1,J)+(1/DR**2-1/2/RF(I)/DR)*F(I-1,J)+1/
     b              DZ(J)**2*F(I,J+1)+1/DZ(J)**2*F(I,J-1))-F(I,J)
              F(I,J)=F(I,J)+ACC*FR(I,J)
   77         CONTINUE
              F(1,J)=F(2,J)
              F(FIN+2,J)=F(FIN+1,J)+Jgs(J)*DR/DL
   76       CONTINUE
            DO 78 I=1,FIN+2
              IF(K.eq.1) THEN
                F(I,LOW-1)=F(I,LOW)+0.2*Jgs(LOW-1)*DR/DL
              ELSE
                dd=SQRT(CSD**2+RF(I)**2)
```

```
            F(I,LOW-1)=F(I,LOW)+(dd/SQRT(dd**2+OR(1)**2))
     a                  *Jgs(LOW-1)*DR/DL
           ENDIF
           IF(K.eq.NmCell) THEN
            F(I,HIGH+1)=F(I,HIGH)+0.2*Jgs(HIGH+1)*DR/DL
           ELSE
            dd=SQRT(CSD**2+RF(I)**2)
            F(I,HIGH+1)=F(I,HIGH)+(dd/SQRT(dd**2+OR(1)**2))
     a                  *Jgs(HIGH+1)*DR/DL
           ENDIF
  78      CONTINUE
          LOW=LOW+CAN(K)
          HIGH=HIGH+CAN(K+1)
  74    CONTINUE
C
C **** Check for convergence ****
C
        SUM=0.
        DO 79 I=1,FIN+2
        DO 79 K=1,Kmax
          SUM=SUM+FR(I,K)**2
  79    CONTINUE
        FRMS=SQRT(SUM/Kmax/(FIN+2))
        IF(FRMS.gt.0.0001)THEN
          GO TO 75
        ENDIF
C
        LOW=1
        HIGH=CAN(1)-1
        DO 80 K=1,NmCell
        DO 81 J=LOW,HIGH
        DO 82 I=1,FIN+1
          NP(I,J)=(F(I,J)+F(I+1,J)+F(I,J+1)+F(I+1,J+1))/4*VOL(I,J)
          q(I,J)=NP(I,J)/VOL(I,J)
  82    CONTINUE
  81    CONTINUE
        LOW=LOW+CAN(K)
        HIGH=HIGH+CAN(K+1)
  80    CONTINUE
        sum=0.
        DO 83 I=1,FIN+1
        DO 83 J=1,Kmax-1
        sum=sum+NP(I,J)
  83    continue
```

```
      IF(ABS(SUM-Pwth).LT.0.1) THEN
        GO TO 100
      ELSE
        DO 84 K=1,Kmax
          Jgs(K)=Jgs(K)+0.5*(Jgs(K)*Pwth/SUM-Jgs(K))
 84     CONTINUE
      ENDIF
      GO TO 75
100   CONTINUE
      Print*,'.....Exiting PDCALC'
    ENDIF
C
C **** End of PDCALC method
*************************************************
C
C
C **** If PwrSet = 4 ****
C
    IF(PwrSet.eq.4) THEN
      DO 95 I=1,FIN+1
      DO 95 K=1,Kmax-1
        FROR=IR(1)+I*thf/(FIN+1)
        FRIR=IR(1)+(I-1)*thf/(FIN+1)
        VOL(I,K)=DZ(K)*PI*(FROR**2-FRIR**2)
        RPwrSum=RPwrSum+RPwr(I,K)*VOL(I,K)
 95   CONTINUE
      DO 105  I=1,FIN+1
      DO 105  K=1,Kmax-1
        q(I,K)=RPwr(I,K)/RPwrSum*Pwth
105   CONTINUE
    ENDIF
C
C **** Volumetric power generation at each node if PwrSet = 1, 2, or 3 ****
C
C **** If PwrSet = 1, 2, or 3 ****
C
    IF(PwrSet.lt.4) THEN
      DO 110 K=1,Kmax-1
        DO 115 I=1,FIN+1
          q(I,K)=q(1,K)
115     CONTINUE
110   CONTINUE
    ENDIF
C
```

```
C **** Initializing the temperature fields ******************************
C
      T(FIN+8,1)=Tin
C
C **** Coolant channel temperatures ****
C
      DO 160 K=2,Kmax
        T(FIN+8,K)=TP(K-1)/Mdot/Cp(T(FIN+8,K-1),W)+T(FIN+8,K-1)
 160  CONTINUE
C
      LOW=2
      HIGH=CAN(1)
      DO 165 I=1,NmCell
        DO 170 K=LOW,HIGH
          Totpwr=(TP(K)+TP(K-1))/2
C
C **** Outer cladding wall temperatures ****
C
          T(FIN+7,K)=Totpwr/H(T(FIN+8,K-1))/2/PI/OR(6)/D(K)+T(FIN+8,K)
C
C **** Outer insulation wall temperatures ****
C
          T(FIN+6,K)=Totpwr*thcl/Kcond(M6,T(FIN+7,K))/PI/(OR(5)
     a          +OR(6))/D(K)+T(FIN+7,K)
C
C **** Outer collector wall temperatures ****
C
          T(FIN+5,K)=Totpwr*thi/Kcond(M5,T(FIN+6,K))/PI/(OR(5)
     a          +IR(5))/D(K)+T(FIN+6,K)
C
C **** Inner collector wall temperatures ****
C
          T(FIN+4,K)=Totpwr*thc/Kcond(M4,T(FIN+5,K))/PI/(OR(4)
     a          +IR(4))/D(K)+T(FIN+5,K)
C
C **** Outer emitter wall temperatures ****
C
          T(FIN+3,K)=(0.67D0*Totpwr/SIG/0.15/2/PI/OR(2)/D(K)
     a          +T(FIN+4,K)**4)**(0.25D0)
C
C **** Inner emitter wall temperature ****
C
          T(FIN+2,K)=Totpwr*the/Kcond(M2,T(FIN+3,K))/PI/(OR(2)
     a          +IR(2))/D(K)+T(FIN+3,K)
```

```
C
C **** Fuel temperatures ****
C
      CNP=Totpwr*(1-(OR(1)**2-(OR(1)-thf/2/(FIN+1))**2)*
     a    PI/AREA(1))
      DO 173 J=FIN+1,1,-1
      FROR=IR(1)+(J-1)*thf/(FIN+1)+thf/2/(FIN+1)
      IF(J.eq.1)THEN
        FRIR=IR(1)
      ELSE
        FRIR=FROR-thf/(FIN+1)
      ENDIF
      T(J,K)=CNP*thf/(FIN+1)/Kcond(M1,T(J+1,K))
     a       /2/PI/FROR/D(K)+T(J+1,K)
 173   CONTINUE
C
 170  CONTINUE
C
      DO 175 K=1,FIN+8
       T(K,HIGH+1)=T(K,HIGH)
 175  CONTINUE
C
      LOW=LOW+CAN(I)
      HIGH=HIGH+CAN(I+1)
C
 165  CONTINUE
C
C **** K=1 temperatures for all regions ****
C
     DO 180 I=1,10
       T(I,1)=2*T(I,2)-T(I,3)
  180  CONTINUE
     DO 185 K=2,5
      T(FIN+K,Kmax+1)=T(FIN+K,1)
      T(FIN+K,0)=T(FIN+K,Kmax)
  185 CONTINUE
C
C **** Initialization of TIC values ****
C
     RICTOT=0.D0
     IF(NmCell.ne.1) THEN
     K=0
     DO 195 I=1,NmCell-1
       K=K+CAN(I)
```

```
      TIC1=(T(FIN+3,K)+T(FIN+2,K))/2-1.D1
      TIC2=(T(FIN+5,K+1)+T(FIN+4,K+1))/2+2.D1
      TIC3=(T(FIN+5,K+1)+T(FIN+4,K+1))/2+1.D1
C
C **** Initialization of CIC(I) and RIC(I) values ****
C
      CIC(I)=CONINCEL((T(FIN+3,K)+T(FIN+2,K))/2,TIC1,TIC2,TIC3,
     a       (T(FIN+5,K+1)+T(FIN+4,K+1))/2)
      RIC(I)=RESINCEL((T(FIN+3,K)+T(FIN+2,K))/2,TIC1,TIC2,TIC3,
     a       (T(FIN+5,K+1)+T(FIN+4,K+1))/2)
      CONIN(I+1)=CIC(I)
      CONOUT(I)=CIC(I)
      RICTOT=RICTOT+RIC(I)
 195  CONTINUE
    ENDIF
C
C **** Multi-celled effective resistance initialization ****
C
    IF(NmCell.gt.1)THEN
      DO 198 I=1,NmCell
       Reff(I)=(RL+RICTOT)/NmCell
 198   CONTINUE
    ENDIF
C
C **** Current density field initialization ************************************
C
    LOW=1
    HIGH=CAN(1)
    DO 200 I=1,NmCell
      DO 205 K=LOW,HIGH
       Jden(K)=SQRT(EFF*Pwth/RL)/(CL(I)*2*PI*OR(2))
 205   CONTINUE
      LOW=LOW+CAN(I)
      HIGH=HIGH+CAN(I+1)
 200  CONTINUE
C
C ***************************** Main code loop
**********************************
C
 2000 CONTINUE
C
C **** Determine voltages and current densities ****
C
    CALL VOLTCALC(Kmax,Jden,D,OR,NmCell,CAN,T,RICTOT,VR,V,IC,
```

```
     a              IE,AREA,M2,M4,RL,Itot,IHKUP,FIN,Tr,Phi0,
     b              Icell,thg,qel,IL,VL,BCSet,Reff,Beta,CL)
C
      DO 300 K=1,Kmax
        qel(K)=0.5*(qel(K)-qelold(K))+qelold(K)
         qelold(K)=qel(K)
 300    CONTINUE
      TIT=1
C
C *************** SOR determination of system temperatures
*********************
C
      Print*,'Entering SOR LOOP.....'
 1000 CONTINUE
C
C ** Determine gap heat x-fer values **
C
      DO 350 K=1,Kmax
      Tm(K)=4.D0/9.D0*((T(FIN+3,K)**(1.5D0)-T(FIN+4,K)**(1.5D0))/
     a      (T(FIN+3,K)-T(FIN+4,K)))**2
      EMIS=0.0325+6.4D-5*T(FIN+3,K)
      QRAD(K)=SIG*EMIS*(T(FIN+3,K)**4-T(FIN+4,K)**4)*2*PI*OR(2)*D(K)
      QCOND(K)=Kcond(M3,Tm(K))*(T(FIN+3,K)-T(FIN+4,K))*2*PI*OR(2)
     a         *D(K)/(thg+1.15D-5*(T(FIN+3,K)+T(FIN+4,K))/PCS)
      QEC(K)=qel(K)*2*PI*OR(2)*D(K)
      QCH(K)=QEC(K)-Jden(K)*V(K)*2*PI*OR(2)*D(K)
 350  CONTINUE
C
C **** Coolant channel temperatures ************************************
C
      K1=1
      K2=0
      CAN(0)=0
      LOW=2
      HIGH=CAN(1)-1
      DO 400 I=1,NmCell
C
C ** Coolant channel cell bottom **
C
      K1=K1+CAN(I-1)
      K=K1
      A2=AREA(7)
      HS=H(T(FIN+8,K))
      IF(K.ne.1) THEN
```

```
      A1=2*PI*OR(6)*(D(K)+CSD/2)
      Kcond1=Kcond(10,(T(FIN+8,K+1)+T(FIN+8,K))/2)
      Kcond2=Kcond(10,(T(FIN+8,K-1)+T(FIN+8,K))/2)
      R(FIN+8,K)=1/(HS*A1+Kcond1*A2/D(K+1)+Kcond2*A2/CSD)*
     a     (HS*A1*T(FIN+7,K)+Kcond1*A2/D(K+1)*T(FIN+8,K+1)+
     b     Kcond2*A2/CSD*T(FIN+8,K-1)+0.5D0*Mdot*Cp(T(FIN+8,K),W)*
     c     (T(FIN+8,K-1)-T(FIN+8,K+1)))-T(FIN+8,K)
      T(FIN+8,K)=T(FIN+8,K)+Alpha*R(FIN+8,K)
      ENDIF
C
C ** Coolant channel cell interior **
C
      A1=4*PI*OR(6)*D(K)
      DO 410 K=LOW,HIGH
      HS=H(T(FIN+8,K))
      Kcond1=Kcond(10,(T(FIN+8,K+1)+T(FIN+8,K))/2)
      Kcond2=Kcond(10,(T(FIN+8,K)+T(FIN+8,K-1))/2)
      R(FIN+8,K)=1/(HS*A1+Kcond1*A2/D(K)+Kcond2*A2/D(K))*
     a        (HS*A1*T(FIN+7,K)+Kcond1*A2/D(K)*T(FIN+8,K+1)
     b        +Kcond2*A2/D(K)*T(FIN+8,K-1)+0.5D0*Mdot*Cp(T(FIN+8,K),W)
     c        *(T(FIN+8,K-1)-T(FIN+8,K+1)))-T(FIN+8,K)
      T(FIN+8,K)=T(FIN+8,K)+Alpha*R(FIN+8,K)
 410  CONTINUE
      LOW=LOW+CAN(I)
      HIGH=HIGH+CAN(I+1)
C
C ** Coolant channel cell top **
C
      K2=K2+CAN(I)
      K=K2
      A1=A1/2
      Kcond2=Kcond(10,(T(FIN+8,K-1)+T(FIN+8,K))/2)
      HS=H(T(FIN+8,K))
      CpS=Cp(T(FIN+8,K),W)
      IF(K.ne.Kmax) THEN
      A1=A1*(D(K)+CSD/2)/D(K)
      Kcond1=Kcond(10,(T(FIN+8,K+1)+T(FIN+8,K))/2)
      R(FIN+8,K)=1/(HS*A1+Kcond1*A2/CSD+Kcond2*A2/D(K-1))*
     a     (HS*A1*T(FIN+7,K)+Kcond1*A2/CSD*T(FIN+8,K+1)+
     b     Kcond2*A2/D(K-1)*T(FIN+8,K-1)+0.5D0*Mdot*CpS*
     c     (T(FIN+8,K-1)-T(FIN+8,K+1)))-T(FIN+8,K)
      T(FIN+8,K)=T(FIN+8,K)+Alpha*R(FIN+8,K)
      ELSE
      R(FIN+8,K)=1/(HS*A1+Kcond2*A2/D(K-1)+0.5D0*Mdot*CpS)*(HS*A1*
```

```
     a     T(FIN+7,K)+Kcond2*A2/D(K-1)*T(FIN+8,K-1)+0.5D0*Mdot*CpS*
     c     T(FIN+8,K-1))-T(FIN+8,K)
      T(FIN+8,K)=T(FIN+8,K)+Alpha*R(FIN+8,K)
      ENDIF
 400  CONTINUE
C
C **** Cladding outer wall temperatures ********************************
C
      K1=1
      K2=0
      CAN(0)=0
      LOW=2
      HIGH=CAN(1)-1
      DO 420 I=1,NmCell
C
C ** Cladding outer wall cell bottom **
C
      K1=K1+CAN(I-1)
      K=K1
      HS=H(T(FIN+8,K))
      A2=PI*(OR(6)**2-(IR(6)+thcl/2)**2)
      Kcond1=Kcond(M6,(T(FIN+7,K)+T(FIN+6,K))/2)
      Kcond2=Kcond(M6,(T(FIN+7,K)+T(FIN+7,K+1))/2)
      IF(K.ne.1)THEN
      A1=2*PI*(IR(6)+thcl/2)*(D(K)+CSD/2)
      A3=2*PI*OR(6)*(D(K)+CSD/2)
      Kcond3=Kcond(M6,(T(FIN+7,K)+T(FIN+7,K-1))/2)
      R(FIN+7,K)=1/(HS*A3+Kcond1*A1/thcl+Kcond2*A2/D(K+1)+
     a      Kcond3*A2/CSD)*(Kcond1*A1/thcl*T(FIN+6,K)+Kcond2*A2/D(K+1)*
     b      T(FIN+7,K+1)+Kcond3*A2/CSD*T(FIN+7,K-1)+HS*A3*
     c      T(FIN+8,K))-T(FIN+7,K)
      T(FIN+7,K)=T(FIN+7,K)+Alpha*R(FIN+7,K)
      ELSE
      A1=2*PI*(IR(6)+thcl/2)*D(K)
      A3=2*PI*OR(6)*D(K)
      R(FIN+7,K)=1/(HS*A3+Kcond1*A1/thcl+Kcond2*A2/D(K+1))*
     a      (Kcond1*A1/thcl*T(FIN+6,K)+Kcond2*A2/D(K+1)*T(FIN+7,K+1)+
     b      HS*A3*T(FIN+8,K))-T(FIN+7,K)
      T(FIN+7,K)=T(FIN+7,K)+Alpha*R(FIN+7,K)
      ENDIF
C
C ** Cladding outer wall cell interior **
C
      A1=4*PI*(IR(6)+thcl/2)*D(K)
```

```
      A3=4*PI*OR(6)*D(K)
      DO 430 K=LOW,HIGH
      HS=H(T(FIN+8,K))
      Kcond1=Kcond(M6,(T(FIN+7,K)+T(FIN+6,K))/2)
      Kcond2=Kcond(M6,(T(FIN+7,K+1)+T(FIN+7,K))/2)
      Kcond3=Kcond(M6,(T(FIN+7,K-1)+T(FIN+7,K))/2)
      R(FIN+7,K)=1/(HS*A3+Kcond1*A1/thcl+Kcond2*A2/D(K)+
     a        Kcond3*A2/D(K))*(Kcond1*A1/thcl*T(FIN+6,K)+Kcond2*A2/
     b        D(K)*T(FIN+7,K+1)+Kcond3*A2/D(K)*T(FIN+7,K-1)+
     c        HS*A3*T(FIN+8,K))-T(FIN+7,K)
      T(FIN+7,K)=T(FIN+7,K)+Alpha*R(FIN+7,K)
  430 CONTINUE
      LOW=LOW+CAN(I)
      HIGH=HIGH+CAN(I+1)
C
C ** Cladding outer wall cell top **
C
      K2=K2+CAN(I)
      K=K2
      HS=H(T(FIN+8,K))
      A1=A1/2
      A3=A3/2
      Kcond1=Kcond(M6,(T(FIN+7,K)+T(FIN+6,K))/2)
      Kcond2=Kcond(M6,(T(FIN+7,K-1)+T(FIN+7,K))/2)
      IF(K.ne.Kmax)THEN
      A1=A1*(D(K)+CSD/2)/D(K)
      A3=2*PI*OR(6)*D(K)
      Kcond3=Kcond(M6,(T(FIN+7,K)+T(FIN+7,K+1))/2)
      R(FIN+7,K)=1/(HS*A3+Kcond1*A1/thcl+Kcond2*A2/D(K-1)+
     a        Kcond3*A2/CSD)*(Kcond1*A1/thcl*T(FIN+6,K)+Kcond2*A2/D(K-1)
     b        *T(FIN+7,K-1)+Kcond3*A2/CSD*T(FIN+7,K+1)+HS*A3
     c        *T(FIN+8,K))-T(FIN+7,K)
      T(FIN+7,K)=T(FIN+7,K)+Alpha*R(FIN+7,K)
      ELSE
      R(FIN+7,K)=1/(H(T(FIN+8,K))*A3+Kcond1*A1/thcl+Kcond2*A2/D(K-1))*
     a        (Kcond1*A1/thcl*T(FIN+6,K)+Kcond2*A2/D(K-1)*T(FIN+7,K-1)+
     b        H(T(FIN+8,K))*A3*T(FIN+8,K))-T(FIN+7,K)
      T(FIN+7,K)=T(FIN+7,K)+Alpha*R(FIN+7,K)
      ENDIF
C
  420 CONTINUE
C
C **** Insulation outer wall cell temperatures ************************
C
```

```
      K1=1
      K2=0
      CAN(0)=0
      LOW=2
      HIGH=CAN(1)-1
      DO 440 I=1,NmCell
C
C ** Insulation outer wall cell bottom **
C
      K1=K1+CAN(I-1)
      K=K1
      A1=2*PI*(IR(5)+thi/2)*D(K)
      A2=PI*((IR(6)+thcl/2)**2-(IR(5)+thi/2)**2)
      A3=2*PI*(IR(6)+thcl/2)*D(K)
      Kcond1=Kcond(M5,(T(FIN+6,K)+T(FIN+5,K))/2)
      Kcond2=0.5D0*Kcond(M5,(T(FIN+6,K)+T(FIN+6,K+1))/2)+
     a      0.5D0*Kcond(M6,(T(FIN+6,K)+T(FIN+6,K+1))/2)
      Kcond3=Kcond(M6,(T(FIN+6,K)+T(FIN+7,K))/2)
      R(FIN+6,K)=1/(Kcond1*A1/thi+Kcond2*A2/D(K+1)+Kcond3*A3/thcl)*
     a      (Kcond1*A1/thi*T(FIN+5,K)+Kcond2*A2/D(K+1)*T(FIN+6,K+1)+
     b      Kcond3*A3/thcl*T(FIN+7,K))-T(FIN+6,K)
      T(FIN+6,K)=T(FIN+6,K)+Alpha*R(FIN+6,K)
C
C ** Insulation outer wall cell interior **
C
      A1=2*A1
      A3=2*A3
      DO 450 K=LOW,HIGH
      Kcond1=Kcond(M5,(T(FIN+6,K)+T(FIN+5,K))/2)
      Kcond2=0.5D0*Kcond(M5,(T(FIN+6,K)+T(FIN+6,K+1))/2)+
     a      0.5D0*Kcond(M6,(T(FIN+6,K)+T(FIN+6,K+1))/2)
      Kcond3=Kcond(M6,(T(FIN+6,K)+T(FIN+7,K))/2)
      Kcond4=0.5D0*Kcond(M5,(T(FIN+6,K)+T(FIN+6,K-1))/2)+
     a      0.5D0*Kcond(M6,(T(FIN+6,K)+T(FIN+6,K-1))/2)
      R(FIN+6,K)=1/(Kcond1*A1/thi+Kcond2*A2/D(K)+Kcond3*A3/thcl+
     a      Kcond4*A2/D(K))*(Kcond1*A1/thi*T(FIN+5,K)+Kcond2*A2
     b      /D(K)*T(FIN+6,K+1)+Kcond3*A3/thcl*T(FIN+7,K)+Kcond4*A2/
     c      D(K)*T(FIN+6,K-1))-T(FIN+6,K)
      T(FIN+6,K)=T(FIN+6,K)+Alpha*R(FIN+6,K)
 450  CONTINUE
      LOW=LOW+CAN(I)
      HIGH=HIGH+CAN(I+1)
C
C ** Insulation outer wall cell top **
```

```
C
      K2=K2+CAN(I)
      K=K2
      A1=A1/2
      A3=A3/2
      Kcond1=Kcond(M5,(T(FIN+6,K)+T(FIN+5,K))/2)
      Kcond2=0.5D0*Kcond(M5,(T(FIN+6,K)+T(FIN+6,K-1))/2)+
     a      0.5D0*Kcond(M6,(T(FIN+6,K)+T(FIN+6,K-1))/2)
      Kcond3=Kcond(M6,(T(FIN+6,K)+T(FIN+7,K))/2)
      R(FIN+6,K)=1/(Kcond1*A1/thi+Kcond2*A2/D(K-1)+Kcond3*A3/thcl)*
     a      (Kcond1*A1/thi*T(FIN+5,K)+Kcond2*A2/D(K-1)*T(FIN+6,K-1)+
     b      Kcond3*A3/thcl*T(FIN+7,K))-T(FIN+6,K)
      T(FIN+6,K)=T(FIN+6,K)+Alpha*R(FIN+6,K)
C
 440  CONTINUE
C
C **** Collector outer wall cell temperatures **************************
C
      K1=1
      K2=0
      CAN(0)=0
      LOW=2
      HIGH=CAN(1)-1
      DO 460 I=1,NmCell
C
C ** Collector outer wall cell bottom **
C
      K1=K1+CAN(I-1)
      K=K1
      A1=2*PI*(IR(4)+thc/2)*D(K)
      A2=PI*((IR(5)+thi/2)**2-(IR(4)+thc/2)**2)
      A3=2*PI*(IR(5)+thi/2)*D(K)
      A4=PI*(OR(4)**2-(OR(4)-thc/2)**2)
      Kcond1=Kcond(M4,(T(FIN+5,K)+T(FIN+4,K))/2)
      Kcond2=0.5D0*Kcond(M4,(T(FIN+5,K)+T(FIN+5,K+1))/2)+
     a      0.5D0*Kcond(M5,(T(FIN+5,K)+T(FIN+5,K+1))/2)
      Kcond3=Kcond(M5,(T(FIN+5,K)+T(FIN+6,K))/2)
      R(FIN+5,K)=1/(Kcond1*A1/thc+Kcond2*A2/D(K+1)+Kcond3*A3/thi+
     a      0.5D0*CONIN(I))*(Kcond1*A1/thc*T(FIN+4,K)+Kcond2*A2/
     b      D(K+1)*T(FIN+5,K+1)+Kcond3*A3/thi*T(FIN+6,K)+(IC(K)/
     c      AREA(4))**2*rhow(M4,T(FIN+5,K))*D(K)*A4+0.25D0*CONIN(I)
     d      *(T(FIN+3,K-1)+T(FIN+2,K-1)))-T(FIN+5,K)
      T(FIN+5,K)=T(FIN+5,K)+Alpha*R(FIN+5,K)
C
```

```
C ** Collector outer wall cell interior **
C
      A1=2*A1
      A3=2*A3
      DO 470 K=LOW,HIGH
      Kcond1=Kcond(M4,(T(FIN+5,K)+T(FIN+4,K))/2)
      Kcond2=0.5D0*Kcond(M4,(T(FIN+5,K)+T(FIN+5,K+1))/2)+
     a      0.5D0*Kcond(M5,(T(FIN+5,K)+T(FIN+5,K+1))/2)
      Kcond3=Kcond(M5,(T(FIN+5,K)+T(FIN+6,K))/2)
      Kcond4=0.5D0*Kcond(M4,(T(FIN+5,K)+T(FIN+5,K-1))/2)+
     a      0.5D0*Kcond(M5,(T(FIN+5,K)+T(FIN+5,K-1))/2)
      R(FIN+5,K)=1/(Kcond1*A1/thc+Kcond2*A2/D(K)+Kcond3*A3/thi+
     a      Kcond4*A2/D(K))*(Kcond1*A1/thc*T(FIN+4,K)+Kcond2*A2
     b      /D(K)*T(FIN+5,K+1)+Kcond3*A3/thi*T(FIN+6,K)+Kcond4*A2/
     c      D(K)*T(FIN+5,K-1)+(IC(K)/AREA(4))**2*rhow(M4,T(FIN+5,K))*
     d      D(K)*A4)-T(FIN+5,K)
      T(FIN+5,K)=T(FIN+5,K)+Alpha*R(FIN+5,K)
 470  CONTINUE
      LOW=LOW+CAN(I)
      HIGH=HIGH+CAN(I+1)
C
C ** Collector outer wall cell top **
C
      K2=K2+CAN(I)
      K=K2
      A1=A1/2
      A3=A3/2
      Kcond1=Kcond(M4,(T(FIN+5,K)+T(FIN+4,K))/2)
      Kcond2=0.5D0*Kcond(M4,(T(FIN+5,K)+T(FIN+5,K-1))/2)+
     a      0.5D0*Kcond(M5,(T(FIN+5,K)+T(FIN+5,K-1))/2)
      Kcond3=Kcond(M5,(T(FIN+5,K)+T(FIN+6,K))/2)
      R(FIN+5,K)=1/(Kcond1*A1/thc+Kcond2*A2/D(K-1)+Kcond3*A3/thi)*
     a      (Kcond1*A1/thc*T(FIN+4,K)+Kcond2*A2/D(K-1)*T(FIN+5,K-1)+
     b      Kcond3*A3/thi*T(FIN+6,K)+(IC(K)/AREA(4))**2*rhow(M4,
     c      T(FIN+5,K))*D(K)*A4)-T(FIN+5,K)
      T(FIN+5,K)=T(FIN+5,K)+Alpha*R(FIN+5,K)
 460  CONTINUE
C
C ** Collector inner wall cell temperatures ****************************
C
      K1=1
      K2=0
      CAN(0)=0
      LOW=2
```

```
      HIGH=CAN(1)-1
      DO 480 I=1,NmCell
C
C ** Collector inner wall cell bottom **
C
      K1=K1+CAN(I-1)
      K=K1
      A1=2*PI*(IR(4)+thc/2)*D(K)
      A2=PI*((IR(4)+thc/2)**2-IR(4)**2)
      Kcond1=Kcond(M4,(T(FIN+5,K)+T(FIN+4,K))/2)
      Kcond2=Kcond(M4,(T(FIN+4,K)+T(FIN+4,K+1))/2)
      R(FIN+4,K)=1/(A1*Kcond1/thc+A2*Kcond2/D(K+1)+0.5D0*CONIN(I))*
     a      (A1*Kcond1/thc*T(FIN+5,K)+A2*Kcond2/D(K+1)*T(FIN+4,K+1)+
     b      QRAD(K)+QCOND(K)+QCH(K)+(IC(K)/AREA(4))**2*A2*D(K)*
     c      rhow(M4,T(FIN+4,K))+0.25D0*CONIN(I)*(T(FIN+3,K-1)+
     d      T(FIN+2,K-1)))-T(FIN+4,K)
      T(FIN+4,K)=T(FIN+4,K)+Alpha*R(FIN+4,K)
C
C ** Collector inner wall cell interior **
C
      A1=2*A1
      DO 490 K=LOW,HIGH
      Kcond1=Kcond(M4,(T(FIN+5,K)+T(FIN+4,K))/2)
      Kcond2=Kcond(M4,(T(FIN+4,K)+T(FIN+4,K+1))/2)
      Kcond3=Kcond(M4,(T(FIN+4,K)+T(FIN+4,K-1))/2)
      R(FIN+4,K)=1/(A1*Kcond1/thc+A2*Kcond2/D(K)+A2*Kcond3/D(K))*
     a      (A1*Kcond1/thc*T(FIN+5,K)+A2*Kcond2/D(K)*T(FIN+4,K+1)+A2*
     b      Kcond3/D(K)*T(FIN+4,K-1)+QRAD(K)+QCOND(K)+QCH(K)+
     c      (IC(K)/AREA(4))**2*A2*D(K)*rhow(M4,T(FIN+4,K)))
     d      -T(FIN+4,K)
      T(FIN+4,K)=T(FIN+4,K)+Alpha*R(FIN+4,K)
 490  CONTINUE
      LOW=LOW+CAN(I)
      HIGH=HIGH+CAN(I+1)
C
C ** Collector inner wall cell top **
C
      K2=K2+CAN(I)
      K=K2
      A1=A1/2
      Kcond1=Kcond(M4,(T(FIN+5,K)+T(FIN+4,K))/2)
      Kcond2=Kcond(M4,(T(FIN+4,K)+T(FIN+4,K-1))/2)
      R(FIN+4,K)=1/(A1*Kcond1/thc+A2*Kcond2/D(K-1))*(A1*Kcond1/thc*
     a      T(FIN+5,K)+A2*Kcond2/D(K-1)*T(FIN+4,K-1)+QRAD(K)+QCOND(K)
```

```
     b      +QCH(K)+(IC(K)/AREA(4))**2*A2*D(K)*rhow(M4,T(FIN+4,K)))
     c      -T(FIN+4,K)
      T(FIN+4,K)=T(FIN+4,K)+Alpha*R(FIN+4,K)
 480  CONTINUE
C
C **** Emitter outer wall cell temperatures ****************************
C
      K1=1
      K2=0
      CAN(0)=0
      LOW=2
      HIGH=CAN(1)-1
      DO 500 I=1,NmCell
C
C ** Emitter outer wall cell bottom **
C
      K1=K1+CAN(I-1)
      K=K1
      A1=2*PI*(IR(2)+the/2)*D(K)
      A2=PI*(OR(2)**2-(IR(2)+the/2)**2)
      Kcond1=Kcond(M2,(T(FIN+2,K)+T(FIN+3,K))/2)
      Kcond2=Kcond(M2,(T(FIN+3,K)+T(FIN+3,K+1))/2)
      R(FIN+3,K)=1/(A1*Kcond1/the+A2*Kcond2/D(K+1))*(A1*Kcond1/the*
     a      T(FIN+2,K)+A2*Kcond2/D(K+1)*T(FIN+3,K+1)-QEC(K)
     b      -QRAD(K)-QCOND(K)+(IE(K)/AREA(2))**2*A2*D(K)*
     c      rhow(M2,T(FIN+3,K)))-T(FIN+3,K)
      T(FIN+3,K)=T(FIN+3,K)+Alpha*R(FIN+3,K)
C
C ** Emitter outer wall cell interior **
C
      A1=2*A1
      DO 510 K=LOW,HIGH
      Kcond1=Kcond(M2,(T(FIN+2,K)+T(FIN+3,K))/2)
      Kcond2=Kcond(M2,(T(FIN+3,K)+T(FIN+3,K+1))/2)
      Kcond3=Kcond(M2,(T(FIN+3,K)+T(FIN+3,K-1))/2)
      R(FIN+3,K)=1/(A1*Kcond1/the+A2*Kcond2/D(K)+A2*Kcond3/D(K))*
     a      (A1*Kcond1/the*T(FIN+2,K)+A2*Kcond2/D(K)*T(FIN+3,K+1)+A2*
     b      Kcond3/D(K)*T(FIN+3,K-1)-QEC(K)-QRAD(K)-QCOND(K)+
     c      (IE(K)/AREA(2))**2*A2*D(K)*rhow(M2,T(FIN+3,K)))
     d      -T(FIN+3,K)
      T(FIN+3,K)=T(FIN+3,K)+Alpha*R(FIN+3,K)
 510  CONTINUE
      LOW=LOW+CAN(I)
      HIGH=HIGH+CAN(I+1)
```

```
C
C ** Emitter outer wall cell top **
C
      K2=K2+CAN(I)
      K=K2
      A1=A1/2
      Kcond1=Kcond(M2,(T(FIN+2,K)+T(FIN+3,K))/2)
      Kcond2=Kcond(M2,(T(FIN+3,K)+T(FIN+3,K-1))/2)
      R(FIN+3,K)=1/(A1*Kcond1/the+A2*Kcond2/D(K-1)+0.5D0*CONOUT(I))*
     a          (A1*Kcond1/the*T(FIN+2,K)+A2*Kcond2/D(K-1)*T(FIN+3,
     b          K-1)-QEC(K)-QRAD(K)-QCOND(K)+(IE(K)/AREA(2))**2
     c          *A2*D(K)*rhow(M2,T(FIN+3,K))+0.25D0*CONOUT(I)*
     d          (T(FIN+4,K+1)+T(FIN+5,K+1)))-T(FIN+3,K)
      T(FIN+3,K)=T(FIN+3,K)+Alpha*R(FIN+3,K)
  500 CONTINUE
C
C **** Fuel outer wall cell temperatures ******************************
C
      K1=1
      K2=0
      CAN(0)=0
      LOW=2
      HIGH=CAN(1)-1
      DO 520 I=1,NmCell
C
C ** Fuel outer wall cell bottom **
C
      K1=K1+CAN(I-1)
      K=K1
      A1=2*PI*(OR(1)-thf/2/(FIN+1))*D(K)
      A2=PI*((OR(1)+the/2)**2-(OR(1)-thf/2/(FIN+1))**2)
      A3=2*PI*(OR(1)+the/2)*D(K)
      A4=PI*((OR(1)+the/2)**2-OR(1)**2)
      V1=PI*(OR(1)**2-(OR(1)-thf/2/(FIN+1))**2)*D(K)
      Kcond1=Kcond(M1,(T(FIN+2,K)+T(FIN+1,K))/2)
      Kcond2=0.5D0*Kcond(M1,(T(FIN+2,K)+T(FIN+2,K+1))/2)+
     a       0.5D0*Kcond(M2,(T(FIN+2,K)+T(FIN+2,K+1))/2)
      Kcond3=Kcond(M2,(T(FIN+2,K)+T(FIN+3,K))/2)
      R(FIN+2,K)=1/(A1*Kcond1/thf*(FIN+1)+A2*Kcond2/D(K+1)+A3*Kcond3/
     a          the)*(A1*Kcond1/thf*(FIN+1)*T(FIN+1,K)+A2*Kcond2/D(K+1)
     b          *T(FIN+2,K+1)+A3*Kcond3/the*T(FIN+3,K)+(IE(K)/AREA(2))
     c          **2*A4*D(K)*rhow(M2,T(FIN+2,K))+q(FIN+1,K)*V1)
     d          -T(FIN+2,K)
      T(FIN+2,K)=T(FIN+2,K)+Alpha*R(FIN+2,K)
```

```
C
C ** Fuel outer wall cell interior **
C
      A1=2*A1
      A3=2*A3
      DO 530 K=LOW,HIGH
      Kcond1=Kcond(M1,(T(FIN+2,K)+T(FIN+1,K))/2)
      Kcond2=0.5D0*Kcond(M1,(T(FIN+2,K)+T(FIN+2,K+1))/2)+
     a      0.5D0*Kcond(M2,(T(FIN+2,K)+T(FIN+2,K+1))/2)
      Kcond3=Kcond(M2,(T(FIN+2,K)+T(FIN+3,K))/2)
      Kcond4=0.5D0*Kcond(M1,(T(FIN+2,K)+T(FIN+2,K-1))/2)+
     a      0.5D0*Kcond(M2,(T(FIN+2,K)+T(FIN+2,K-1))/2)
      R(FIN+2,K)=1/(A1*Kcond1/thf*(FIN+1)+A2*Kcond2/D(K)+A3*Kcond3/the+
     a      A2*Kcond4/D(K))*(A1*Kcond1/thf*(FIN+1)*T(FIN+1,K)
     b      +A2*Kcond2/D(K)*T(FIN+2,K+1)+A3*Kcond3/the*T(FIN+3,K)+A2
     c      *Kcond4/D(K)*T(FIN+2,K-1)+(IE(K)/AREA(2))**2*A4*D(K)
     d      *rhow(M2,T(FIN+2,K))+q(FIN+1,K)*V1+q(FIN+1,K-1)*V1)-
     e      T(FIN+2,K)
      T(FIN+2,K)=T(FIN+2,K)+Alpha*R(FIN+2,K)
  530 CONTINUE
      LOW=LOW+CAN(I)
      HIGH=HIGH+CAN(I+1)
C
C ** Fuel outer wall cell top **
C
      K2=K2+CAN(I)
      K=K2
      A1=A1/2
      A3=A3/2
      Kcond1=Kcond(M1,(T(FIN+2,K)+T(FIN+1,K))/2)
      Kcond2=0.5D0*Kcond(M1,(T(FIN+2,K)+T(FIN+2,K-1))/2)+
     a      0.5D0*Kcond(M2,(T(FIN+2,K)+T(FIN+2,K-1))/2)
      Kcond3=Kcond(M2,(T(FIN+2,K)+T(FIN+3,K))/2)
      R(FIN+2,K)=1/(A1*Kcond1/thf*(FIN+1)+A2*Kcond2/D(K-1)+A3*Kcond3/
     a      the+0.5D0*CONOUT(I))*(A1*Kcond1/thf*(FIN+1)*T(FIN+1,K)+A2*
     b      Kcond2/D(K-1)*T(FIN+2,K-1)+A3*Kcond3/the*T(FIN+3,K)+(IE(K)
     c      /AREA(2))**2*A4*D(K)*rhow(M2,T(FIN+2,K))+0.5D0*CONOUT(I)*
     d      (T(FIN+4,K+1)+T(FIN+5,K+1))/2+q(FIN+1,K-1)*V1)-T(FIN+2,K)
      T(FIN+2,K)=T(FIN+2,K)+Alpha*R(FIN+2,K)
  520 CONTINUE
C
C **** Fuel interior cell temperatures ********************************
C
      DO 540 J=FIN+1,2,-1
```

```
      K1=1
      K2=0
      CAN(0)=0
      LOW=2
      HIGH=CAN(1)-1
      DO 550 I=1,NmCell
C
C ** Fuel interior cell bottom **
C
      K1=K1+CAN(I-1)
      K=K1
      RI(J)=IR(1)+(J-1)*thf/(FIN+1)-thf/2/(FIN+1)
      RO(J)=RI(J)+thf/(FIN+1)
      AR1(J)=2*PI*RI(J)*D(K)
      AR2(J)=PI*(RO(J)**2-RI(J)**2)
      AR3(J)=2*PI*RO(J)*D(K)
      V1=PI*(RO(J)**2-((RO(J)+RI(J))/2)**2)*D(K)
      V2=PI*(((RO(J)+RI(J))/2)**2-RI(J)**2)*D(K)
      KC1(J)=Kcond(M1,(T(J,K)+T(J-1,K))/2)
      KC2(J)=Kcond(M1,(T(J,K)+T(J,K+1))/2)
      KC3(J)=Kcond(M1,(T(J,K)+T(J+1,K))/2)
      R(J,K)=1/(AR1(J)*KC1(J)*(FIN+1)/thf+AR2(J)*KC2(J)/D(K+1)+
     b       AR3(J)*KC3(J)*(FIN+1)/thf)*(AR1(J)*KC1(J)*(FIN+1)/
     c       thf*T(J-1,K)+AR2(J)*KC2(J)/D(K+1)*T(J,K+1)+AR3(J)*
     d       KC3(J)*(FIN+1)/thf*T(J+1,K)+q(J,K)*V1+q(J-1,K)*V2)
     e       -T(J,K)
      T(J,K)=T(J,K)+Alpha*R(J,K)
C
C ** Fuel interior cell interior **
C
      AR1(J)=2*AR1(J)
      AR3(J)=2*AR3(J)
      DO 560 K=LOW,HIGH
      KC1(J)=Kcond(M1,(T(J,K)+T(J-1,K))/2)
      KC2(J)=Kcond(M1,(T(J,K)+T(J,K+1))/2)
      KC3(J)=Kcond(M1,(T(J,K)+T(J+1,K))/2)
      KC4(J)=Kcond(M1,(T(J,K)+T(J,K-1))/2)
      R(J,K)=1/(AR1(J)*KC1(J)*(FIN+1)/thf+AR2(J)*KC2(J)/D(K)+AR3(J)*
     a       KC3(J)*(FIN+1)/thf+AR2(J)*KC4(J)/D(K))*(AR1(J)*KC1(J)*
     b       (FIN+1)/thf*T(J-1,K)+AR2(J)*KC2(J)/D(K)*T(J,K+1)+AR3(J)*
     c       KC3(J)*(FIN+1)/thf*T(J+1,K)+AR2(J)*KC4(J)/D(K)*T(J,K-1)
     d       +q(J,K)*V1+q(J-1,K)*V2+q(J,K-1)*V1+q(J-1,K-1)*V2)-T(J,K)
      T(J,K)=T(J,K)+Alpha*R(J,K)
  560 CONTINUE
```

```
      LOW=LOW+CAN(I)
      HIGH=HIGH+CAN(I+1)
C
C ** Fuel interior cell top **
C
      K2=K2+CAN(I)
      K=K2
      AR1(J)=AR1(J)/2
      AR3(J)=AR3(J)/2
      KC1(J)=Kcond(M1,(T(J,K)+T(J-1,K))/2)
      KC2(J)=Kcond(M1,(T(J,K)+T(J,K-1))/2)
      KC3(J)=Kcond(M1,(T(J,K)+T(J+1,K))/2)
      R(J,K)=1/(AR1(J)*KC1(J)*(FIN+1)/thf+AR2(J)*KC2(J)/D(K-1)+
     b       AR3(J)*KC3(J)*(FIN+1)/thf)*(AR1(J)*KC1(J)*(FIN+1)/thf
     c       *T(J-1,K)+AR2(J)*KC2(J)/D(K-1)*T(J,K-1)+AR3(J)*KC3(J)
     d       *(FIN+1)/thf*T(J+1,K)+q(J,K-1)*V1+q(J-1,K-1)*V2)-T(J,K)
      T(J,K)=T(J,K)+Alpha*R(J,K)
  550 CONTINUE
  540 CONTINUE
C
C **** Fuel inner wall temperatures ************************************
C
      K1=1
      K2=0
      CAN(0)=0
      LOW=2
      HIGH=CAN(1)-1
      DO 570 I=1,NmCell
C
C ** Fuel inner wall cell bottom **
C
      K1=K1+CAN(I-1)
      K=K1
      A1=2*PI*(IR(1)+thf/2/(FIN+1))*D(K)
      A2=PI*((IR(1)+thf/2/(FIN+1))**2-IR(1)**2)
      V1=A2*D(K)
      Kcond1=Kcond(M1,(T(1,K)+T(2,K))/2)
      Kcond2=Kcond(M1,(T(1,K)+T(1,K+1))/2)
      R(1,K)=1/(Kcond1*A1/thf*(FIN+1)+Kcond2*A2/D(K+1))*(Kcond1*A1/
     a       thf*(FIN+1)*T(2,K)+Kcond2*A2/D(K+1)*T(1,K+1)+q(1,K)*V1)
     b       -T(1,K)
      T(1,K)=T(1,K)+Alpha*R(1,K)
C
C ** Fuel inner wall cell interior **
```

```
C
      A1=2*A1
      DO 580 K=LOW,HIGH
      Kcond1=Kcond(M1,(T(1,K)+T(2,K))/2)
      Kcond2=Kcond(M1,(T(1,K)+T(1,K+1))/2)
      Kcond3=Kcond(M1,(T(1,K)+T(1,K-1))/2)
      R(1,K)=1/(Kcond1*A1/thf*(FIN+1)+Kcond2*A2/D(K)+Kcond3*A2/D(K))
     a       *(Kcond1*A1/thf*(FIN+1)*T(2,K)+Kcond2*A2/D(K)*T(1,K+1)
     b       +Kcond3*A2/D(K)*T(1,K-1)+q(1,K)*V1+q(1,K-1)*V1)-T(1,K)
      T(1,K)=T(1,K)+Alpha*R(1,K)
 580  CONTINUE
      LOW=LOW+CAN(I)
      HIGH=HIGH+CAN(I+1)
C
C ** Fuel inner wall cell top **
C
      K2=K2+CAN(I)
      K=K2
      A1=A1/2
      Kcond1=Kcond(M1,(T(1,K)+T(2,K))/2)
      Kcond2=Kcond(M1,(T(1,K)+T(1,K-1))/2)
      R(1,K)=1/(Kcond1*A1/thf*(FIN+1)+Kcond2*A2/D(K-1))*(Kcond1*A1/
     a       thf*(FIN+1)*T(2,K)+Kcond2*A2/D(K-1)*T(1,K-1)+q(1,K-1)*V1)
     b       -T(1,K)
      T(1,K)=T(1,K)+Alpha*R(1,K)
C
 570  CONTINUE
C
C **** Update TIC values ****
C
      IF(NmCell.ne.1) THEN
      K=0
      RICTOT=0.D0
      DO 590 I=1,NmCell-1
        K=K+CAN(I)
        TPC=Kcond(TPM,(TIC3+TIC2)/2)*TPP
        ESC=Kcond(M2,(TIC2+TIC1)/2)*ESP
        NFEC=Kcond(M2,((T(FIN+2,K)+T(FIN+3,K))/2+TIC1)/2)*NFEP
        TIC1=(T(FIN+3,K)+T(FIN+2,K))/2-(T(FIN+3,K)+T(FIN+2,K)-
     a       T(FIN+4,K+1)-T(FIN+5,K-1))*CIC(I)/NFEC/2
        TIC2=TIC1-(T(FIN+3,K)+T(FIN+2,K)-T(FIN+4,K+1)-T(FIN+5,K+1))
     a          *CIC(I)/ESC/2
        TIC3=TIC2-(T(FIN+3,K)+T(FIN+2,K)-T(FIN+4,K+1)-T(FIN+5,K+1))
     a          *CIC(I)/TPC/2
```

```
C
C **** Update of CIC(I) and RIC(I) values ****
C
      CIC(I)=CONINCEL((T(FIN+3,K)+T(FIN+2,K))/2,TIC1,TIC2,TIC3,
     a          (T(FIN+5,K+1)+T(FIN+4,K+1))/2)
      RIC(I)=RESINCEL((T(FIN+3,K)+T(FIN+2,K))/2,TIC1,TIC2,TIC3,
     a          (T(FIN+5,K+1)+T(FIN+4,K+1))/2)
      CONIN(I+1)=CIC(I)
      CONOUT(I)=CIC(I)
      RICTOT=RICTOT+RIC(I)
 590  CONTINUE
     ENDIF
C
C ******************* End of SOR loop
*****************************************
C
C
C **** Check for temperature convergence ****
C
     SUMR=0.D0
     MAXERR=0.D0
     DO 600 I=1,FIN+8
     DO 600 K=1,Kmax
       SUMR=SUMR+R(I,K)**2
       MAXERR=MAX(MAXERR,abs(R(I,K)))
 600  CONTINUE
     TRMS=(SUMR/Kmax/(FIN+8))**(0.5D0)
     EIG=SQRT(SUMR/SUMOLD)
     SUMOLD=SUMR
     AF=1/(1-EIG)
     ITA=ITA+1
     TestAF=abs((AF-AFOLD)/AFOLD)
     IF(TRMS.lt.CONVERGT.and.TIT.eq.1)THEN
       Print*,'.....Exiting SOR LOOP'
       Print*,'Temperatures converged after ',TIT,' iteration'
       Print*,'TRMS = ',TRMS,' Max Error = ',MAXERR
       GO TO 3000
     ENDIF
     IF(TRMS.lt.CONVERGT.and.ITA.gt.1)THEN
       Print*,'.....Exiting SOR LOOP'
       Print*,'Temperatures converged after ',TIT,' iterations'
       Print*,'TRMS = ',TRMS,' Max Error = ',MAXERR
       GO TO 2000
     ENDIF
```

```
      IF(TIT.gt.TITmax) THEN
        Print *,'Unable to converge temperatures'
        Print *,'after ',TITmax,' iterations, program terminated'
        GO TO 3000
      ENDIF
      IF(TestAf.lt.EA.and.ITA.gt.9)THEN
        DO 810 I=1,FIN+8
        DO 810 K=1,Kmax
          T(I,K)=T(I,K)+AF*R(I,K)
 810    CONTINUE
        ITA=0
      ENDIF
      AFOLD=AF
      PRINT*,'TRMS =',TRMS,' After iteration ',TIT
      TIT=TIT+1
      GO TO 1000
C
 3000 CONTINUE
C
C **** Print and Write final results ****
C
      Print*,' '
      IF(NmCell.ne.1.and.BCSet.eq.2)THEN
        RL=0.
        DO 1600 I=1,NmCell
          RL=RL+Reff(I)
 1600   CONTINUE
        RL=RL-RICTOT
      ENDIF
      Print 1702,RL*1000
      Print 1703,Itot
      Print 1704,Itot*RL
      Print 1705,Itot**2*RL
      Print 1706,Itot**2*RL/Pwth*100
      Print*,' '
      Print*,'See output file (MCTFE.OUT) for complete run results'
      Write(2,1707)
      Write(2,1708)
      DO 1700 K=Kmax,1,-1
        Write(2,1701)(T(I,K),I=FIN+8,1,-1)
 1700 CONTINUE
      Write(2,1709)
      Write(2,1710)
      LOW=Kmax-CAN(NmCell)+1
```

```
      HIGH=Kmax
      DO 1750 I=NmCell,1,-1
        DO 1760 K=HIGH,LOW,-1
          AREAK=2*PI*OR(2)*D(K)
          Write(2,1714)FZ(K),QRAD(K)/AREAK,QCOND(K)/AREAK,QEC(K)
     a  /AREAK,QCH(K)/AREAK
1760  CONTINUE
      Write(2,*)' '
      LOW=LOW-CAN(I-1)
      HIGH=HIGH-CAN(I)
1750  CONTINUE
      Write(2,1711)
      Write(2,1712)
      LOW=Kmax-CAN(NmCell)+1
      HIGH=Kmax
      DO 1800 I=NmCell,1,-1
        DO 1810 K=HIGH,LOW,-1
          Write(2,1713)FZ(K),Jden(K),V(K),Jden(K)*V(K)
1810    CONTINUE
        Write(2,*)' '
        LOW=LOW-CAN(I-1)
        HIGH=HIGH-CAN(I)
1800  CONTINUE
      Write(2,*)' '
      Write(2,1702)RL*100
      Write(2,1703)Itot
      Write(2,1704)Itot*RL
      Write(2,1705)Itot**2*RL
      Write(2,1706)Itot**2*RL/Pwth*100
      IF(NmCell.gt.1)THEN
        Write(2,1715)
        DO 1820 I=NmCell,1,-1
          Write(2,1716)I,Icell(I)*Reff(I)
1820    CONTINUE
      ENDIF
1701  FORMAT(11F7.1)
1702  FORMAT(' TFE Load Resistance  (mOhms) =',1F10.3)
1703  FORMAT(' TFE Current          (Amps)  =',1F10.1)
1704  FORMAT(' TFE Voltage          (Volts) =',1F10.3)
1705  FORMAT(' TFE Electrical Power (Watts) =',1F10.1)
1706  FORMAT(' TFE Efficiency       (%)     =',1F10.2)
1707  FORMAT(/,80('*'),'OUTPUT RESULTS'/,80('*')/)
1708  FORMAT('Final Temperatures (K), (Coolant to Fuel - Left',
     a' to Right)'/,80('-'))
```

```
 1709 FORMAT(/'Interelectrode Gap Heat Transfer Densities'/,80('-'))
 1710 FORMAT(3X,'Axial Position (cm)',2X,'QRAD (W/cm2)',2X,
    a  'QCOND (W/cm2)',2X,'QEC (W/cm2)',2X,'QCH (W/cm2)',3X)
 1711 FORMAT(//,'Thermionic Results'/,80('-'))
 1712 FORMAT('Axial Position (cm)',2X,'Current Density (A/cm2)',2X
    a ,'Voltage (V)',2X,'Power Density (W/cm2)')
 1713 FORMAT(3X,1F10.3,12X,1F10.3,9X,1F10.3,8X,1F10.3)
 1714 FORMAT(4X,1F10.3,9X,1F10.3,4X,1F10.3,5X,1F10.3,3X,1F10.3)
 1715 FORMAT(//,'Individual Cell Voltages (top to bottom)',
    a  /,80('-'))
 1716 FORMAT('Voltage for cell ',I1,' =',1F10.3)
      CLOSE(1)
      CLOSE(2)
      END
C
**********************************************************************
********
      Real*8 Function Cp(T,W)
C       (Units Joules/Kg/K)
**********************************************************************
********
*       Uses correlations from the Sodium-NaK Engineering      *
*       Handbook (O. Foust, ed.; vol. 1 pp. 52-53) to return   *
*       the value of the heat capacity of the NaK coolant for  *
*       a given temperature T and potassium weight fraction W  *
*       for the coolant (e.g. eutectic NaK-78 has W=78.)        *
*       Only single phase coolants are modeled.  If the        *
*       temperature of the coolant is higher than the boiling  *
*       point of NaK at the given sodium-potassium composition,*
*       this routine reports the error and halts the program.  *
*                                                              *
****************************************************************
C
      Real*8 T, W, BoilingPt, CpNa, CpK
C
      BoilingPt = ((756.5-881.4)*W + 881.4) + 273.1
      If (T.GT.BoilingPt) then
          Write(*,100) T,BoilingPt
          Write(2,100) T,BoilingPt
          Stop
      EndIf
C
      CpNa=(1.43612D0-5.80237D-4*T+4.62081D-7*T**2)*1000.D0
      CpK=(0.83850D0-3.67230D-4*T+4.58980D-7*T**2)*1000.D0
```

```
C
      Cp=CpNa*(1.0D0-W)+CpK*W
C
  100 Format(' The temperature T=',F7.1,' degrees K is higher than '/
     a       ' the boiling point for NaK-',F7.1,//
     b       ' Execution terminated.'//)
      Return
      End
C
**********************************************************************
********
      Real*8 Function H(T)
C       (Units = W/cm**2/K)
**********************************************************************
********
C
C*************************************************************
C                                                  *
C   Uses the convective heat transfer correlations given     *
C   for liquid metal flows through annular channels, as      *
C   presented in M.M. El-Wakil, Nuclear Heat Transport,      *
C   p. 269, equations 10-6 and 10-7.                         *
C                                                  *
C*************************************************************
      Real*8 De,OR(10),IR(10),Pe,Mdot,T,W,PI,Nu,Cp,Kcond,AREA(10)
      COMMON /Cooldat/ OR,IR,Mdot,PI,W,AREA
C
      De=2*(OR(7)-IR(7))
      Pe = De*Mdot*Cp(T,W)/Kcond(10,T)/AREA(7)
C
      If (OR(7)/IR(7).LT.1.4) then
         Nu = 5.8D0 + 0.02D0*(Pe**0.8D0)
        Else
         Nu = 5.25D0 + 0.0188D0*(Pe**0.8D0)*(OR(7)/IR(7))**0.3D0
        EndIf
C
      H = Nu*Kcond(10,T)/De
C
      Return
      End
C
**********************************************************************
********
      Real*8 Function Kcond(I,T)
```

```
C       (Units = W/cm/K)
*************************************************************************
********
      Real*8 T, Kdata(5,11)
C
C     Material  #1 = UO2
C     Material  #2 = W
C     Material  #3 = Nb
C     Material  #4 = Nb1Zr
C     Material  #5 = Mo
C     Material  #6 = S.S.316
C     Material  #7 = Cs (Tm < or = 1280 K)
C     Material  #8 = Al2O3
C     Material  #9 = Tanalum
C     Material #10 = NaK
C     Material #11 = Cs (Tm > 1280 K)
C
      Data Kdata/2.414D-2,
     a          -2.478D-5, 1.094D-8,  -1.67D-12,     0.0D0,
     b  0.4886D0, -3.057D-4, 1.237D-7,  -1.72D-11,     0.0D0,
     c 0.11104D0,  4.870D-6, 1.281D-8,      0.0D0,     0.0D0,
     d 0.11104D0,  4.870D-6, 1.281D-8,      0.0D0,     0.0D0,
     e  0.3602D0, -1.141D-4, 2.050D-8,      0.0D0,     0.0D0,
     f  0.0758D0,  1.890D-4,    0.0D0,      0.0D0,     0.0D0,
     g 4.9214D-3,-1.7726D-5,2.3987D-8,-1.4427D-11,3.2552D-15,
     h  0.1858D0, -4.169D-4, 3.469D-7,  -9.74D-11,      0.D0,
     i 0.12443D0,2.67567D-5,-3.583D-8, 3.4419D-11,-7.381D-15,
     j   6.20D-2,  7.204D-4,-8.343D-7,  3.060E-10,     0.0D0,
     k -7.908D-4, 1.7170D-6,-1.221D-9, 2.8969D-13,     0.0D0/
C
      Kcond=(Kdata(1,I)+Kdata(2,I)*T+Kdata(3,I)*T**2+Kdata(4,I)*
     a          T**3+Kdata(5,I)*T**4)*4.184D0
      IF(I.eq.7.and.T.gt.1280) THEN
      Kcond=Kcond(11,T)
      ENDIF
      Return
      End
C
*************************************************************************
********
      REAL*8 FUNCTION CONINCEL(T1,T2,T3,T4,T5)
C       (Units = Watts/K)
*************************************************************************
********
```

```
C
      REAL*8 T1,T2,T3,T4,T5,TPP,ESP,NFEP,NFCP,TPC,ESC,NFEC,NFCC
      REAL*8 Kcond
      INTEGER TPM,M2,M4
      COMMON /Inceldat/ TPP,ESP,NFEP,NFCP,TPM,M2,M4
      TPC=Kcond(TPM,(T4+T3)/2)*TPP
      ESC=Kcond(M2,(T3+T2)/2)*ESP
      NFEC=Kcond(M2,(T1+T2)/2)*NFEP
      NFCC=Kcond(M4,(T5+T4)/2)*NFCP
      CONINCEL=1/(1/TPC+1/ESC+1/NFCC+1/NFEC)
      RETURN
      END
C
*************************************************************************
********
      REAL*8 FUNCTION RESINCEL(T1,T2,T3,T4,T5)
C           (Units = Ohms)
*************************************************************************
********
C
      REAL*8 T1,T2,T3,T4,T5,TPP,ESP,NFEP,NFCP,TPR,ESRe,NFER,NFCR
      REAL*8 rhow
      INTEGER TPM,M2,M4
      COMMON /Inceldat/ TPP,ESP,NFEP,NFCP,TPM,M2,M4
      TPR=rhow(TPM,(T4+T3)/2)/TPP
      ESRe=rhow(M2,(T3+T2)/2)/ESP
      NFER=rhow(M4,(T1+T2)/2)/NFEP
      NFCR=rhow(M2,(T5+T4)/2)/NFCP
      RESINCEL=TPR+ESRe+NFER+NFCR
      RETURN
      END
C
*************************************************************************
********
      SUBROUTINE
VOLTCALC(Kmax,Jden,D,OR,NmCell,CAN,T,RICTOT,VR,V,IC,
     a                 IE,AREA,M2,M4,RL,Itot,IHKUP,FIN,Tr,Phi0,
     b                 Icell,thg,qel,IL,VL,BCSet,Reff,Beta,CL)
*************************************************************************
********
C
      REAL*8 Jden(100),D(100),Itot,OR(10),IC(100),IE(100),CL(10)
      REAL*8 SUM,T(20,100),rhow,Beta,ICbot,Jdenold(100),Reff(10)
      REAL*8 ReffT,RICTOT,SUM1,SUM2,VR(100),V(100),PI,IL,VL,IEbot
```

```
      REAL*8 AREA(10),RL,Tr,Phi0,thg,qel(100),cden,Icell(10)
      REAL*8 GAMMA(10),Icellold(10)
      INTEGER CAN(10),HIGH,FIN,VIT,BVIT,BCSet
      DATA CONVERGV /0.0005/
      PI=DACOS(-1.D0)
      BVIT=1
      VIT=1
      MaxErr=0.
      PRINT*,'Entering VOLTCALC.....'
C
C ** Determination of total current **
C
 100  CONTINUE
      sum=0.D0
      sumI=0.D0
      checkI=0.
      LOW=1
      HIGH=CAN(1)
      DO 150 I=1,NmCell
        Icell(I)=0.
        DO 120 K=LOW,HIGH
          Icell(I)=Icell(I)+Jden(K)*D(K)*2*PI*OR(2)
 120    CONTINUE
        checkI=checkI+abs(Icell(I)-Icellold(I))
        Icellold(I)=Icell(I)
        sum=sum+Icell(I)
        sumI=sumI+(Icell(I)-IL)**2
        LOW=LOW+CAN(I)
        HIGH=HIGH+CAN(I+1)
 150  CONTINUE
      IRMS=sumI**0.5
      IF(checkI.lt.0.01)THEN
        IRMS=0.
      ENDIF
      Itot=sum/NmCell
C
C **** Set up boundary conditions ****
C
      IF(BCSet.eq.1)THEN
        IL=Itot
      ENDIF
      IF(BCSet.eq.2)THEN
        RL=RL+0.25*(RL*Itot/IL-RL)
      ENDIF
```

```
      IF(BCSet.eq.3)THEN
        RL=VL/ITOT
        IL=Itot
      ENDIF
      IF(NmCell.gt.1)THEN
        ReffT=0.
        DO 170 I=1,NmCell
          GAMMA(I)=MIN(1,100*abs(Icell(I)-IL)/Icell(I))
          Reff(I)=Reff(I)+GAMMA(I)*(Reff(I)*Icell(I)/IL-Reff(I))
          ReffT=ReffT+Reff(I)
  170   CONTINUE
        IF(BCSet.ne.2)THEN
          DO 180 I=1,NmCell
            Reff(I)=Reff(I)*(RICTOT+RL)/ReffT
  180     CONTINUE
        ENDIF
      ENDIF
C
      IF(IHKUP.eq.1)THEN
        ICbot=Itot/2
        IEbot=-ICbot
  200   CONTINUE
        SUM=Jden(1)*D(1)
        IC(1)=ICbot-SUM*PI*OR(2)
        IE(1)=IEbot+SUM*PI*OR(2)
        DO 220 K=2,Kmax-1
          SUM=SUM+Jden(K)*D(K)
          IC(K)=ICbot-(SUM-Jden(K)*D(K)/2)*2*PI*OR(2)
          IE(K)=IEbot+(SUM-Jden(K)*D(K)/2)*2*PI*OR(2)
  220   CONTINUE
        IC(Kmax)=ICbot-Itot+Jden(Kmax)*D(Kmax)*PI*OR(2)
        IE(Kmax)=IEbot+Itot-Jden(Kmax)*D(Kmax)*PI*OR(2)
        SUM1=0.
        SUM2=0.
        DO 230 K=1,Kmax
          SUM1=SUM1+IC(K)*rhow(M4,(T(FIN+4,K)+T(FIN+5,K))/2)*D(K)
     a         /AREA(4)
          SUM2=SUM2+IE(K)*rhow(M2,(T(FIN+2,K)+T(FIN+3,K))/2)*D(K)
     a         /AREA(2)
  230   CONTINUE
        test1=SUM1*AREA(4)/CL(1)/rhow(M4,(T(FIN+5,1)+
     a       T(FIN+4,1+Kmax/2))/2)
        test2=SUM2*AREA(2)/CL(1)/rhow(M2,(T(FIN+3,1)+
     a       T(FIN+2,1+Kmax/2))/2)
```

```
        ICbot=ICbot-test1
        IEbot=IEbot-test2
        IF(abs(test1).gt.0.1.or.abs(test2).gt.0.1)THEN
          GO TO 200
        ENDIF
      ENDIF
      IF(IHKUP.eq.2)THEN
C
C ** Cell top node currents **
C
        K=0
        DO 240 I=1,NmCell
        K=K+CAN(I)
        IC(K)=Jden(K)*D(K)*PI*OR(2)
        IE(K)=Itot-IC(K)
 240    CONTINUE
C
C ** Cell bottom node currents **
C
        K=1
        DO 250 I=1,NmCell
        IE(K)=Jden(K)*D(K)*PI*OR(2)
        IC(K)=Itot-IE(K)
        K=K+CAN(I)
 250    CONTINUE
C
C ** Cell interrior node currents **
C
        K=1
        LOW=2
        HIGH=CAN(1)-1
        DO 260 I=1,NmCell
        SUM=Jden(K)*D(K)
          DO 270 K=LOW,HIGH
          SUM=SUM+Jden(K)*D(K)
          IE(K)=(SUM-Jden(K)*D(K)/2)*2*PI*OR(2)
          IC(K)=Itot-IE(K)
 270      CONTINUE
        K=HIGH+2
        LOW=LOW+CAN(I)
        HIGH=HIGH+CAN(I+1)
 260    CONTINUE
      ENDIF
C
```

```
C **** Determination of voltages (each point of each cell) ****
C
      IF(IHKUP.eq.1)THEN
        DO 300 K=1,CAN(1)
          SUM1=0.D0
          SUM2=0.D0
          IF(K.NE.1) THEN
            DO 310 J=1,K
              SUM1=SUM1+IC(J)*rhow(M4,(T(FIN+4,J)+T(FIN+5,J))/2)*D(J)
              SUM2=SUM2+IE(J)*rhow(M2,(T(FIN+2,J)+T(FIN+3,J))/2)*D(J)
 310        CONTINUE
            IF(K.lt.CAN(1)) THEN
              SUM1=SUM1-IC(K)*rhow(M4,(T(FIN+4,K)+T(FIN+5,K))/2)*D(K)/2
              SUM2=SUM2-IE(K)*rhow(M2,(T(FIN+2,K)+T(FIN+3,K))/2)*D(K)/2
            ENDIF
          ENDIF
          VR(K)=Itot*RL+SUM1/AREA(4)-SUM2/AREA(2)-V(K)
          V(K)=V(K)+Beta*VR(K)
 300    CONTINUE
      ENDIF
C
      IF(IHKUP.eq.2)THEN
        LOW=1
        HIGH=CAN(1)
        DO 320 I=1,NmCell
 325      CONTINUE
          DO 330 K=LOW,HIGH
            SUM1=0.D0
            SUM2=0.D0
            IF(K.NE.LOW) THEN
              DO 340 J=LOW,K
                SUM1=SUM1+IC(J)*rhow(M4,(T(FIN+4,J)+T(FIN+5,J))/2)*D(J)
 340          CONTINUE
              IF(K.lt.HIGH) THEN
                SUM1=SUM1-IC(K)*rhow(M4,(T(FIN+4,K)+T(FIN+5,K))/2)*D(K)
     a             /2
              ENDIF
            ENDIF
            IF(K.NE.HIGH) THEN
              DO 350 J=K,HIGH
                SUM2=SUM2+IE(J)*rhow(M2,(T(FIN+4,J)+T(FIN+5,J))/2)*D(J)
 350          CONTINUE
              IF(K.gt.LOW) THEN
                SUM2=SUM2-IE(K)*rhow(M2,(T(FIN+2,K)+T(FIN+3,K))/2)*D(K)
```

```
     a             /2
                 ENDIF
               ENDIF
               VR(K)=Reff(I)*Itot+SUM1/AREA(4)+SUM2/AREA(2)-V(K)
               V(K)=V(K)+VR(K)
 330         CONTINUE
             LOW=LOW+CAN(I)
             HIGH=HIGH+CAN(I+1)
 320       CONTINUE
       ENDIF
C
       SUM=0.
       DO 400 K=1,Kmax
         SUM=SUM+VR(K)**2
         MaxErr=MAX(MaxErr,abs(VR(K)))
 400   CONTINUE
       VRMS=DSQRT(SUM/Kmax)
C
C **** Check for voltage convergence ****
C
       PRINT*,' '
       PRINT*,'VRMS = ',VRMS,'    Iteration = ',VIT
       PRINT*,' '
       DO 410 K=Kmax,1,-1
         PRINT 411, K, V(K)
 410   CONTINUE
 411   FORMAT(' V(',I2,') =',F10.3)
       IF(NmCell.gt.1)THEN
         DO 420 I=NmCell,1,-1
           PRINT 421, I, Icell(I), IL
 420       CONTINUE
 421       FORMAT (' cell(',I1,') current =',F10.3,'  target current ='
     a     ,F10.3)
       ENDIF
       IF(VRMS.lt.CONVERGV.and.VIT.gt.1.and.IRMS.lt.0.1)THEN
         GO TO 500
       ENDIF
       VIT=VIT+1
       BVIT=BVIT+1
       IF(BVIT.gt.21)THEN
        Beta=MAX(0.25,Beta/2)
        BVIT=1
       ENDIF
C
```

```
C **** Update current densities ****
C
      DO 450 K=1,Kmax
      Jden(K)=cden(T(FIN+3,K),T(FIN+4,K),Tr,Phi0,thg,V(K),qel(K),
     a        Jdenold(K))
      Jden(K)=Jdenold(K)+Beta*(Jden(K)-Jdenold(K))
      Jdenold(K)=Jden(K)
 450  CONTINUE
      GO TO 100
 500  CONTINUE
      PRINT*,'Voltages are converged after ',VIT,' iterations'
      PRINT*,'VRMS = ',VRMS,' Max error =',MaxErr
      PRINT*,'.....Exiting VOLTCALC'
      RETURN
      END
C
**********************************************************************
********
      Real*8 Function rhow(I,T)
C       (Units = Ohms-cm)
**********************************************************************
********
      Real*8 T,Rdata(4,9),r
C
C     Material  #1 = UO2
C     Material  #2 = W
C     Material  #3 = Nb
C     Material  #4 = Nb1Zr
C     Material  #5 = Mo
C     Material  #6 = S.S.316
C     Material  #7 = Cs (Tm < or = 1280 K)
C     Material  #8 = Al2O3
C     Material  #9 = Tanalum
C
      Data Rdata/0,
     a                    0,          0,          0,
     b -.2285507D0,0.01808205D0,  6.64431D-6,-7.479135D-10,
     c -1.451331D0,0.04999382D0,-4.867492D-6,        0.0D0,
     d -1.451331D0,0.04999382D0,-4.867492D-6,        0.0D0,
     e     -.506D0,     0.022D0,       0.0D0,         0.0D0,
     f           0,           0,          0,          0,
     g           0,           0,          0,          0,
     h           0,           0,          0,          0,
     i -.9355648D0, 4.634023D-2,-1.402214D-6, -8.64408D-10/
```

```
C
      r=(Rdata(1,I)+Rdata(2,I)*T+Rdata(3,I)*T**2+Rdata(4,I)*T**3)
      rhow=r*1.D-6
      Return
      End
C
*****************************************************************************
********
      REAL*8 FUNCTION cden(te,tc,tr,phi0,d,v,qel,jguess)
C               (Units = Amps/CM**2)
*****************************************************************************
********
      REAL*8 te,tc,tr,phi0,d,v,qel,jguess
C
C  Uses jvbrac, jvfind, ndsphi
C
C*****************************************************************
C                                                               *
C  The function cden returns current density as a function of  *
C  voltage utilizing thermionic models which compute voltage   *
C  as a function of current density.  A combination of the     *
C  TECMDL and UNIG thermionic models are used, which are called*
C  by the routines jvbrac and jvfind.                          *
C                                                             *
C     Input values -                                           *
C     te        Emitter temperature (K)                        *
C     tc        Collector temperature (K)                      *
C     tr        Cesium reservoir temperature (K)               *
C     phi0      Emitter bare work function (eV)                *
C     d         Interelectrode spacing (cm)                    *
C     v         Output voltage                                 *
C     jguess    Guess for current density (amps/cm2)           *
C                                                              *
C     Output values -                                          *
C     cden      Current density (amps/cm2)                     *
C     qel       Emitter electron cooling (watts/cm2)           *
C                                                              *
C*****************************************************************
C
      REAL*8 jvfind,phie,phic,j1,j2,f1,f2,ndsphi
      LOGICAL success
C
C ....Get the cesiated work functions....
      phie=ndsphi(te,tr,phi0)
```

```
      phic=1.9104+(tc/tr)*(2.2963+(tc/tr)*(-3.1364+
     &  (tc/tr)*.98039))
      if ((te.le.1300.d0).and.(v.ge.0.5d0)) then
        cden=0.d0
        qel=0.d0
        return
      endif
C ....Try to bracket the desired current density
      j1=jguess
      call jvbrac(te,tc,tr,phie,phic,d,v,j1,j2,f1,f2,success)
C.....Zero in on the correct current density value.....
      if (success) then
        cden=jvfind(te,tc,tr,phie,phic,d,v,j1,j2,f1,f2,qel)
      else
        pause ' no match'
      end if
C
      return
      END
C
C
      SUBROUTINE jvbrac(te,tc,tr,phie,phic,d,v0,x1,x2,f1,f2,success)
C     IMPLICIT NONE
      REAL*8 te,tc,tr,phie,phic,d,v0,x1,x2,f1,f2,XKE,FACTOR
      LOGICAL success
      PARAMETER(XKE=8.6175d-5,FACTOR=1.6d0)
C
C Uses jvcurve
C
C*****************************************************************
C                                                      *
C   The subroutine jvbrac searches for two current density      *
C   values which will bracket the desired solution for output   *
C   voltage.                                             *
C                                                       *
C      Input values -                                     *
C      te        Emitter temperature (K)                    *
C      tc        Collector temperature (K)                  *
C      tr        Cesium reservoir temperature (K)             *
C      phie       Cesiated emitter work function (eV)         *
C      phic       Cesiated collector work function (eV)       *
C      d         Interelectrode spacing (cm)                *
C      v0         Desired value for output voltage           *
C      x1         Input as first guess for current density    *
```

```
C                                                              *
C     Output values -                                           *
C     x1        Output as one of the bracketing values of     *
C               current density                                 *
C     x2        The other bracketing value of current density*
C     f1        The value of v - v0 at x1                       *
C     f2        The value of v - v0 at x2                       *
C                                                              *
C***************************************************************
C
      REAL*8 dx,x3,f3,v,qel,xjc
      INTEGER bad
C
C.....First set the search step to 1 A/cm2.....
      dx=1.
C.....Call the combined thermionic model.....
      call jvcurve(te,tc,tr,phie,phic,d,x1,v,qel)
      f1=v-v0
C.....Increment current density in the correct direction...
      x2=x1+dsign(dx,f1)
      x2=dmax1(x2,0.d0)
C.....Compute voltage at new current density.....
      call jvcurve(te,tc,tr,phie,phic,d,x2,v,qel)
      f2=v-v0
      bad=0
C.....Find the back emission current density for lower search limit..
      xjc=120.d0*tc*tc*dexp(-11604.5d0*phic/tc)
C.....Continue searching until solution is bracketed.....
      do while (f1*f2.gt.0.d0)
C.....Increase size of search step.....
        dx=dx*FACTOR
        x3=x2+dsign(dx,f2)
C.....Count number of times that current density tries to go
C       below the back emission level.  If 2 or more, return
C       without a successful solution.....
        if (x3.lt.-xjc) bad=bad+1
        if (bad.ge.2) then
          success=.false.
          return
        end if
        x3=dmax1(x3,-xjc)
        call jvcurve(te,tc,tr,phie,phic,d,x3,v,qel)
        f3=v-v0
        x1=x2
```

```
        f1=f2
        x2=x3
        f2=f3
      end do
      success=.true.
      return
      END
C
C
      REAL*8 FUNCTION jvfind(te,tc,tr,phie,phic,d,v0,j1,j2,f1,f2,qel)
C     IMPLICIT NONE
      REAL*8 te,tc,tr,phie,phic,d,v0,j1,j2,f1,f2,qel,TOL2
      PARAMETER(TOL2=1.d-5)
C
C  Uses jvcurve
C
C*****************************************************************
C                                                               *
C   The function jvfind uses the modified regula falsi method   *
C   to find a value for current density which yields a desired  *
C   voltage.  The solution must already have been bracketed.    *
C                                                               *
C      Input values -                                             *
C      te        Emitter temperature (K)                        *
C      tc        Collector temperature (K)                      *
C      tr        Cesium reservoir temperature (K)               *
C      phie       Cesiated emitter work function (eV)           *
C      phic       Cesiated collector work function (eV)         *
C      d         Interelectrode spacing (cm)                    *
C      v0         Desired value for output voltage              *
C      j1        One of the bracketing values of current       *
C                density.                                       *
C      j2        The other bracketing value of current density*
C      f1        The value of v - v0 at j1                      *
C      f2        The value of v - v0 at j2                      *
C                                                               *
C      Output values -                                            *
C      jvfind     The solution for the current density          *
C      qel       The electron cooling at the solution point    *
C                                                               *
C*****************************************************************
C
      REAL*8 tol1,save_f,j3,f3,v
C
```

```
      tol1=1.d-5
      save_f=f1
   10 continue
        j3=j2-f2*(j2-j1)/(f2-f1)
        call jvcurve(te,tc,tr,phie,phic,d,j3,v,qel)
        f3=v-v0
C.....Re-assign whichever endpoint has the same sign of f as the most
C     recent point.  If an endpoint has been stagnant for 2 passes,
C     replace f with f/2 there.....
        if (f3*f1.lt.0.d0) then
          j2=j3
          f2=f3
          if (f3*save_f.gt.0.d0) f1=f1/2.d0
        else
          j1=j3
          f1=f3
          if (f3*save_f.gt.0.d0) f2=f2/2.d0
        end if
        save_f=f3
      if (.not.((dabs(j1-j2).le.tol1).or.(dabs(f3).le.TOL2))) goto 10
      jvfind=j3
      return
      END
C
C
      SUBROUTINE jvcurve(te,tc,tr,phie,phic,dcm,j,v,qel)
C     IMPLICIT NONE
      REAL*8 te,tc,tr,phie,phic,dcm,j,v,qel,JLOW,JHIGH
      PARAMETER(JLOW=0.1d0, JHIGH=3.d0)
C  Uses tecmdl, unig
C
C*********************************************************************
C                                                        *
C  The routine jvcurve combines the outputs of TECMDL and UNIG  *
C  to produce a single, physically reasonable, well-behaved      *
C  volt-ampere curve.  The limits JHIGH and JLOW are used to     *
C  save computational effort by only calling both models in the  *
C  interval bounded by these limits.  Above JHIGH only TECMDL    *
C  is called, below JLOW only UNIG is called.                    *
C                                                      *
C      Input values -                                    *
C      te          Emitter temperature (K)                 *
C      tc          Collector temperature (K)               *
C      tr          Cesium reservoir temperature (K)          *
```

```
C      phie        Cesiated emitter work function (eV)          *
C      phic        Cesiated collector work function (eV)        *
C      dcm          Interelectrode spacing (cm)                 *
C      j           Current density (amps/cm2)                   *
C                                                               *
C      Output values -                                          *
C      v           Output voltage                               *
C      qel         Emitter electron cooling (watts/cm2)         *
C                                                               *
C***************************************************************
      INTEGER sheaths
      REAL*8 dmm,vig,qelig,vun,qelun,ji,jel,old
C
      dmm=dcm*10.d0
      if (j.lt.JLOW) then
        jel=j
C.....A simple iteration is necessary when calling unig as it
C     accepts the electron current as an independent variable,
C     whereas the total current = electron current - ion current....
  10    continue
          old=jel
          call unig(te,tc,tr,dcm,phie,phic,jel,ji,v,qel,sheaths)
          jel=j+ji
        if (dabs((jel-old)/jel).gt.1.d-5) go to 10
C         print*,'v =',v,'unig j =',jel-ji
      else if (j.gt.JHIGH) then
        call tecmdl(te,tc,tr,phie,phic,dmm,j,v,qel)
C         print*,'v =',v,'tecmdl j =',j
      else
        jel=j
  20    continue
          old=jel
          call unig(te,tc,tr,dcm,phie,phic,jel,ji,vun,qelun,sheaths)
          jel=j+ji
        if (dabs((jel-old)/jel).gt.1.d-6) go to 20
C         print*,'v =',v,'unig j =',jel-ji
        call tecmdl(te,tc,tr,phie,phic,dmm,j,vig,qelig)
C         print*,'v =',v,'tecmdl j =',j
C.....Select whichever voltage (and corresponding electron cooling)
C     is higher.....
        if (vig.ge.vun) then
          v=vig
          qel=qelig
        else
```

```
        v=vun
        qel=qelun
       end if
C       print*,'v =',v,'chosen j =',j
     end if
     return
     END
     SUBROUTINE tecmdl(te,tc,tr,phie,phic,d,j,vout,qel)
C    IMPLICIT NONE
     REAL*8 te,tc,tr,phie,phic,d,j,vout,qel,VI,B,H,BP,XK,TWO,
    & AR,HALF
     PARAMETER (VI=3.2d0,B=30.d0,H=5.d0,BP=50.d0,XK=8.6175d-5,
    &          TWO=2.d0,AR=120.d0,HALF=0.5d0)
C
C    Uses ndsphi,obstr,obstr2,satur,satur2
C
C*************************************************************
C                                                  *
C     C-563-002-G-082988                             *
C     Written by                                   *
C     John B. McVey                                  *
C     Rasor Associates, Inc.                       *
C     (408) 734-1622 X-315                           *
C                                                  *
C     TECMDL is an implementation of the "phenomenological *
C     model" of the ignited mode converter described in    *
C     N.S. Rasor, "Thermionic Energy Conversion", Chapter  *
C     5 of Applied Atomic Collision Physics, vol. 5,       *
C     Massey, McDaniel, and Bederson eds., Academic Press, *
C     1982. The article is available on request from       *
C     Rasor Associates. The following physics has been     *
C     added to the model:                                  *
C           A. Plasma energy loss by radiation.            *
C           B. Improved description of saturation          *
C              region.                                   *
C           C. Provision for ion-retaining collector     *
C              sheath.                                   *
C     The equations are documented in the Rasor document   *
C     E-563-002-A-063087, which is available on request.   *
C                                                  *
C     This routine calls four subroutines, two for the     *
C     obstructed mode (positive and negative collector     *
C     sheath) and two for the saturation region (positive  *
C     and negative collector sheath) as required. It       *
```

```
C     calculates the output voltage and emitter electron   *
C     cooling.                                              *
C                                                           *
C     Input values -                                         *
C     TE          Emitter temperature (K)                    *
C     TC          Collector temperature (K)                  *
C     TR          Cesium reservoir temperature (K)           *
C     PHIE        Emitter work function (eV).                *
C     PHIC        Collector work function (eV).              *
C     D           Interelectrode spacing (mm)                *
C     J           Current density (amps/cm2)                 *
C                                                           *
C     Output values -                                        *
C     VOUT        Output voltage (volts)                     *
C     QEL         Net emitter electron cooling (watts/cm2)   *
C                                                           *
C   Version G is special for use in CYLCON6.                 *
C   - calculation of cesiated work functions removed         *
C   - changed to double precision                            *
C   - parameter jconfdnc removed                             *
C                                                           *
C***********************************************************
C
      REAL*8 jsp,jc,jcj,pcs,ta,
     &       pd,tee,tec,ve,vc,vd,vrad,jej,je,dv,
     &       jsj,jij,js,phis
C
C.....Calculate saturation current density.....
      jsp=AR*te*te*dexp(-phie/(XK*te))
C
C.....Calculate back emission current density and ratio.....
      jc=AR*tc*tc*dexp(-phic/(XK*tc))
      jcj=jc/j
C
C.....Calculate cesium pressure in torr.....
      pcs=2.45d+8*dexp(-8910.d0/tr)/dsqrt(tr)
C
C.....Average neutral and ion temperature.....
      ta=(te+tc)/TWO
C
C.....Calculate Pd.....
      pd=pcs*d
C
C.....Call the obstructed region calculation
```

```
C       (can't be obstructed if current density is
C       above saturation).....
      if (j.le.jsp) then
C
C.....Calculation for positive collector sheath.....
         call obstr(VI,B,H,j,jcj,te,tc,tr,pd,d,ta,tee,tec,ve,vc,
     &     vd,vrad,jej)
C
C.....If collector sheath was negative in previous calculation,
C       use appropriate calculation.....
         if (vc.lt.0.d0) call obstr2(VI,B,H,j,jcj,te,tc,tr,pd,d,ta,
     &     tee,tec,ve,vc,vd,vrad,jej)
C
C.....Calculate effective emitted current density.....
        je=jej*j
C
C.....Calculate sheath barrier height.....
         dv=XK*te*dlog(jsp/je)
C
C.....Calculate output voltage.....
        vout=phie-phic-vd+dv
C
C.....Calculate net electron cooling from emitter
C       (includes plasma radiation).....
         qel=j*(phie+dv+TWO*XK*tee)-je*TWO*XK*(tee-te)-
     &         HALF*j*vrad
C
      endif
C
C.....Call the saturation region calculation if above saturation
C       or if obstructed region calculation failed.....
      if ((j.gt.jsp).or.(dv.lt.0.d0)) then
C
C.....Calculation for positive collector sheath.....
         call satur(VI,B,BP,H,j,jcj,jsp,te,tc,tr,pd,d,ta,tee,tec,
     &      ve,vc,vd,vrad,jsj,jij)
C
C.....Calculation for negative collector sheath if previous
C       calculation gave negative value.....
         if (vc.lt.0.d0) call satur2(VI,B,BP,H,j,
     &     jcj,jsp,te,tc,tr,pd,d,ta,tee,tec,ve,vc,vd,vrad,jsj,
     &     jij)
C
C.....Calculate effective Schottky saturation current density.....
```

```
      js=jsj*j
C
C.....Calculate Schottky reduced emitter work function.....
       phis=XK*te*dlog(AR*te*te/js)
C
C.....Calculate output voltage.....
       vout=phis-phic-vd
C
C.....Calculate net emitter electron cooling (includes
C      ion heating and plasma radiation).....
       qel=j*(phis+TWO*XK*tee)-js*TWO*XK*(tee-te)+j*jij*(ve+3.89d0+
     &     TWO*XK*tee)-HALF*j*vrad
C
      endif
      return
      END
C
C
      SUBROUTINE obstr(vi,b,h,j,jcj,te,tc,tr,pd,d,ta,tee,tec,ve,vc,
     &    vd,vrad,jej)
C     IMPLICIT NONE
      INTEGER MAXITR
      REAL*8 vi,b,h,j,jcj,te,tc,tr,pd,d,ta,tee,tec,ve,vc,vd,vrad,
     &    jej,XK,HALF,TOL,AR,EMIS,ONE,TWO,THREE,TINY
      PARAMETER (XK=8.6175d-5,TWO=2.d0,HALF=.5d0,TOL=1.d-5,AR=120.d0,
     &           EMIS=.4d0,MAXITR=50,ONE=1.d0,THREE=3.d0,TINY=1.d-32)
C
C     Uses tsc, ltec
C
C*************************************************************
C                                                  *
C      OBSTR is called by TECMDL to implement the          *
C      phenomenological equations for                  *
C      the obstructed region of the ignited mode volt-      *
C      ampere curve with a positive (electron retaining)    *
C      sheath at the collector. This is the formulation     *
C      described in Massey, McDaniel, and Bederson.          *
C      Equations (24)-(29) and (31) are used. The emitter   *
C      side and collector side electron temperatures are    *
C      subject to LTE (Local Thermodynamic Equilibrium)      *
C      constraints which are implemented by the functon      *
C      TSC and the subroutine LTE. Equations (24) for        *
C      JE/J and (25) for Vd are coupled, and are solved      *
C      iteratively using a secant method search.            *
```

```
C                                                              *
C     Input values -                                            *
C     VI        Effective ionization energy (eV)                  *
C     B         Ionizability factor                               *
C     H         Collector current factor                          *
C     J         Current density (amps/cm2)                        *
C     JCJ       Ratio of back emission to current density         *
C     TE        Emitter temperature (K)                           *
C     TC        Collector temperature (K)                         *
C     TR        Cesium reservoir temperature (K)                  *
C     PD        Pressure-spacing product (torr-mm)                *
C     D         Interelectrode spacing (mm)                       *
C     TA        Average neutral and ion temperature (K)           *
C                                                              *
C     Output values -                                           *
C     TEE       Emitter side electron temperature (K)             *
C     TEC       Collector side electron temperature (K)           *
C     VE        Emitter sheath height (eV)                        *
C     VC        Collector sheath height (eV)                      *
C     VD        Arc drop (eV)                                     *
C     VRAD      Plasma radiation component of arc drop (eV)*
C     JEJ       Ratio JE/J of effective emitted current           *
C               density to working current density                *
C                                                              *
C*************************************************************
C
      INTEGER iter
      REAL*8 dlea,phinc,jnc,hs,tsc,telect,dl,r,ans,dif,
     &    oldj,olddif,newj,param1,param2,param3,param4
      LOGICAL first,ltec
C
C.....Calculate ratio of spacing to electron-neutral
C       mean free path.....
      dlea=35.d0/((te+tc)/2000.d0)*pd
C
C.....Calculate emitter side electron temperature.....
      tee=VI/(TWO*XK*dlog(B*dlea))
C
C.....Calculate collector side electron temperature.....
      tec=(THREE*tee+TWO*tc*jcj)/(dlog((H+HALF)/
     &    (ONE+jcj))+TWO*jcj+THREE)
C
C.....Calculate neutralization potential and current density.....
      phinc=1.7d0+.383d0*tec/tr
```

```
      jnc=AR*tec*tec*dexp(-phinc/(XK*tec))
C
C.....Assign value to LTE limit for H and check
C       to see if LTE limits the electron temperature
C       at the collector edge - if so, calculate a new
C       TEC value.....
      hs=jnc/j
      if (hs.lt.h) then
        tec=tsc(tee,tc,tr,hs,j,jcj)
        ltec=.true.
      else
        ltec=.false.
      endif
C
C.....Calculate average electron temperature.....
      telect=(tee+tec)/TWO
C
C.....Calculate ratio of spacing to total
C       electron mean free path, including
C       ion scattering.....
      dl=dlea+3.4d+7*j*d/(telect**2.5d0)
C
C.....If LTE has occured at the collector edge, the routine
C       LTE is called to check that the average electron
C       temperature is above the bulk LTE limit.  If not,
C       the LTE routine will calculate new emitter side,
C       collector side,  and average electron temperatures
C       The ratio of spacing to total mean free path is also
C       recalculated.....
      if (ltec) call lte(tee,tec,telect,tc,tr,hs,j,jcj,dl,dlea,d)
C
C.....Calculate collector sheath height.....
      vc=THREE*XK*(tee-tec)-TWO*xk*(tec-tc)*jcj
C
C.....Calculate collector reflection factor.....
      r=(ONE+jcj)*dexp(vc/(XK*tec))-ONE
C
C.....Calculate radiation component of arc drop.....
      vrad=9.65d+5*pd/(j*ta)*dexp(-2.d0/(XK*telect))*(ONE+.069d0
     &  *dexp(.58d0/(XK*telect))*(EMIS/dsqrt(d/10.d0)+HALF))
C
C.....Guess JEJ and enter secant method iteration.....
      jej=TWO
      first=.true.
```

```
C
C.....First compute some parameters in order to save time in the
C         iteration loop.....
      param1=TWO*XK*(tec-te+(tec-tc)*jcj)+vrad
      param2=TWO*XK*(tee-te)
      param3=.75d0*dl+r
      param4=-ONE/(XK*tee)
C.....Start iteration.....
      do iter=1,MAXITR
C.....Calculate arc drop.....
        vd=param2*(jej-ONE)+param1
C
C.....Calculate emitter sheath height.....
       ve=vd+vc
C
C.....Calculate answer for JEJ and compute difference from
C      guess for JEJ.....
        if (ve*param4.le.dlog(TINY)) then
C.....Case for ve so large that the exp function would underflow..
          ans=ONE
        else
C.....Normal case.....
          ans=ONE+param3*dexp(ve*param4)
        endif
       dif=jej-ans
     if (dabs(dif).lt.TOL) go to 10
C
C.....Update value of JEJ until convergence.....
       if (first) then
         oldj=jej
         olddif=dif
          jej=jej-dsign(.2d0,dif)
         first=.false.
       else
         newj=(oldj*dif-jej*olddif)/(dif-olddif)
         oldj=jej
         olddif=dif
         jej=newj
       endif
     if (dabs(jej-oldj).lt.1.d-5*jej) go to 10
     enddo
 10   if (iter.gt.MAXITR) pause 'Exceeded maximum iterations in
    &obstr'
     return
```

```
      END
C
C
      SUBROUTINE satur(vi,b,bp,h,j,jcj,jsp,te,tc,tr,pd,d,ta,tee,tec,
     &     ve,vc,vd,vrad,jsj,jij)
C     IMPLICIT NONE
      INTEGER MAXITR
      REAL*8 vi,b,bp,h,j,jcj,jsp,te,tc,tr,pd,d,ta,tee,tec,ve,vc,vd,
     &     vrad,jsj,jij,XK,TWO,HALF,TOL1,TOL2,AR,EMIS,ONE,THREE,TINY
      PARAMETER (XK=8.6175d-5,TWO=2.d0,HALF=.5d0,TOL1=1.d-6,TOL2=1.d-5,
     &          MAXITR=100,AR=120.d0,EMIS=.4d0,ONE=1.d0,THREE=3.d0,
     &          TINY=1.d-32)
C
C     Uses tsc, ltec
C
C*************************************************************
C                                                            *
C     SATUR is called by TECMDL to implement the phenomen- *
C     ological model equations in the saturation region,    *
C     with a positive collector sheath. The formulation     *
C     given by eqs. (33) to (35) in Massey, McDaniel, and   *
C     Bederson has been improved so that is consistent      *
C     with the level of complexity used in the obstructed   *
C     region calculation. The ion current into the          *
C     emitter is now included in all equations in the form  *
C     of the parameter JIJ (Ji/J). An additional            *
C     iteration over what was needed in the obstructed      *
C     mode calculation is required for finding the value    *
C     of JIJ. A modified linear interpolation method is     *
C     used. The iteration for finding VD and JEJ is         *
C     nested within this new iteration.                     *
C                                                           *
C     Input values -                                        *
C     VI      Effective ionization energy (eV)              *
C     B       Ionizability factor                           *
C     BP      Temperature increase parameter                *
C     H       Collector current factor                      *
C     J       Current density (amps/cm2)                    *
C     JCJ     Ratio of back emission to current density     *
C     TE      Emitter temperature (K)                       *
C     TC      Collector temperature (K)                     *
C     TR      Cesium reservoir temperature (K)              *
C     PD      Pressure-spacing product (torr-mm)            *
C     D       Interelectrode spacing (mm)                   *
```

```
C     TA        Average neutral and ion temperature (K)    *
C                                                          *
C     Output values -                                      *
C     TEE       Emitter side electron temperature (K)      *
C     TEC       Collector side electron temperature (K)    *
C     VE        Emitter sheath height (eV)                 *
C     VC        Collector sheath height (eV)               *
C     VD        Arc drop (eV)                              *
C     VRAD      Plasma radiation component of arc drop (eV)*
C     JSJ       Ratio JS/J of effective emitted current    *
C               density to working current density         *
C     JIJ       Ratio Ji/J of additional ion current to    *
C               the emitter to working current density     *
C                                                          *
C************************************************************
C
      INTEGER iwhch,iter1,iter2
      REAL*8 dlea,phinc,jnc,hs,tsc,telect,dl,r,ans,f,oldj,
     &       oldf,newj,param1,param2,param3,param4,js,g,
     &       x1,x2,x3,y1,y2,y3,ys
      LOGICAL first,ltec
C
C.....Guess ion current ratio.....
      jij=0.d0
C
C.....Set iteration counter.....
      iwhch=1
C
C.....Calculate ratio of gap to electron-neutral mean free path.....
      dlea=35.d0/((te+tc)/2000.d0)*pd
C
C.....Begin modified linear interpolation search for JIJ.....
C
      do iter1=1,MAXITR
C.....Calculate emitter side electron temperature.....
         tee=vi/(TWO*XK*dlog(b*dlea)-XK*dlog(ONE-bp*jij))
C
C.....Calculate collector side electron temperature.....
         tec=(THREE*tee+TWO*tc*jcj)/(dlog((h+HALF)/
     &    (ONE+jcj))+TWO*jcj+THREE)
C
C.....Calculate neutralization potential and current density.....
         phinc=1.7d0+.383d0*tec/tr
         jnc=AR*tec*tec*dexp(-phinc/(XK*tec))
```

```
C
C.....Assign value to LTE limit for H and check
C       to see if LTE limits the electron temperature
C       at the collector edge - if so, calculate a new
C       TEC value.....
        hs=jnc/j
        if (hs.lt.h) then
          tec=tsc(tee,tc,tr,hs,j,jcj)
          ltec=.true.
        else
          ltec=.false.
        endif
C
C.....Calculate average electron temperature.....
         telect=(tee+tec)/TWO
C
C.....Calculate ratio of spacing to total
C       electron mean free path, including
C       ion scattering.....
         dl=dlea+3.4d+7*j*d/(telect**2.5d0)
C
C.....If LTE has occured at the collector edge, the routine
C       LTE is called to check that the average electron
C       temperature is above the bulk LTE limit.  If not,
C       the LTE routine will calculate new emitter side,
C       collector side,  and average electron temperatures
C       The ratio of spacing to total mean free path is also
C       recalculated.....
         if (ltec) call lte(tee,tec,telect,tc,tr,hs,j,jcj,dl,dlea,d)
C
C.....Calculate collector sheath height.....
         vc=THREE*XK*(tee-tec)-TWO*XK*(tec-tc)*jcj
C
C.....Calculate collector reflection factor.....
         r=(ONE+jcj)*dexp(vc/(XK*tec))-ONE
C
C.....Calculate radiation component of arc drop.....
         vrad=9.65d+5*pd/(j*ta)*dexp(-2.d0/(XK*telect))*
     &    (ONE+.069d0*dexp(.58d0/(XK*telect))*
     &    (EMIS/dsqrt(d/10.d0)+HALF))
C
C.....Guess JSJ and enter secant method iteration.....
        jsj=2.d0
        first=.true.
```

```
C
C.....First compute some parameters in order to save time
C         in the iteration loop
          param1=TWO*XK*(tec-te+(tec-tc)*jcj)+vrad-jij*(vc+
     &          3.89d0+TWO*XK*tee)
          param2=TWO*XK*(tee-te)
          param3=.75d0*dl+r
          param4=-ONE/(XK*tee)
C.....Start iteration.....
          do iter2=1,MAXITR
C.....Calculate arc drop.....
            vd=(param2*(jsj-ONE)+param1)/(ONE+jij)
C
C.....Calculate emitter sheath height.....
           ve=vd+vc
C
C.....Calculate answer for JSJ and compute difference from
C       guess for JSJ.....
            if (ve*param4.le.dlog(TINY)) then
C.....Case for ve so large that exp function would underflow.....
              ans=ONE+jij
            else
C.....Normal case.....
              ans=ONE+(param3-HALF*jij)*dexp(ve*param4)+jij
            endif
           f=jsj-ans
          if (dabs(f).lt.TOL1) go to 20
C
C.....Update value of JSJ until convergence.....
           if (first) then
             oldj=jsj
             oldf=f
              jsj=jsj-dsign(.2d0,f)
             first=.false.
           else
             newj=(oldj*f-jsj*oldf)/(f-oldf)
             oldj=jsj
             oldf=f
             jsj=newj
            endif
          if (dabs(jsj-oldj).lt.1.d-5*jsj) go to 20
          enddo
 20       if (iter2.gt.MAXITR) pause 'Exceeded maximum iterations in
     &SATUR for finding current ratio'
```

```
C
C.....Calculate value of JS from eqn. (35) of Massey,
C       McDaniel, and Bederson.....
         js=jsp*dexp(612.d0*dsqrt(dsqrt(-j*jij*dsqrt(ve)))/te)
C
C.....Compute error term.....
        g=js/j-jsj
      if (dabs(g).lt.TOL2) go to 30
C
C.....Update JIJ to make error small.....
        if (iwhch.eq.1) then
          if ((g.gt.0.d0).and.(vc.le.0.d0)) return
         x1=jij
         y1=g
          jij=-.1d0
          x2=jij
         iwhch=2
       else if (iwhch.eq.2) then
         x2=jij
         y2=g
          if (y1*y2.gt.0.d0) then
           x1=x2
           y1=y2
            jij=jij+dsign(.1d0,g)
C.....Prevent jij from becoming equal to -1.  Make it the
C          nearest larger number.....
             if (jij.le.-ONE) jij=-.999999d0
          else
            iwhch=3
            ys=y2
            jij=(x1*y2-x2*y1)/(y2-y1)
           endif
       else if (iwhch.eq.3) then
         x3=jij
         y3=g
          if (y3*y1.lt.0.d0) then
           x2=x3
           y2=y3
            if (y3*ys.gt.0.d0) y1=y1/TWO
          else
            x1=x3
            y1=y3
             if (y3*ys.gt.0.d0) y2=y2/TWO
           endif
```

```
        ys=y3
        jij=(x1*y2-x2*y1)/(y2-y1)
       endif
     enddo
 30   if (iter1.gt.MAXITR) then
c      write(*,'(a)') ' Maximum iterations exceeded in
c    &SATUR2 for finding ion current'
       write(6,'(a)') ' Maximum iterations exceeded in
     &SATUR2 for finding ion current'
c      write(*,*) j,tr
       write(6,*) j,tr
       stop
     endif
     return
     END
C
C
     REAL*8 FUNCTION tsc(tee,tc,tr,hs,j,jcj)
C     IMPLICIT NONE
     INTEGER MAXITR
     REAL*8 tee,tc,tr,hs,j,jcj,XK,ONE,TWO,THREE,TOL,AR,HALF
     PARAMETER (XK=8.6175d-5,ONE=1.d0,TWO=2.d0,THREE=3.d0,TOL=
    &          1.d-5,AR=120.d0,MAXITR=50,HALF=.5d0)
C
C*************************************************************
C                                                  *
C     The function TSC is called by the subroutines OBSTR  *
C     and SATUR in order to compute the collector side     *
C     electron temperature when LTE conditions exist at the*
C     collector.  A secant method iteration is used.       *
C                                                  *
C     Input values -                                 *
C     TEE          Emitter side electron temperature (K)   *
C     TC           Collector temperature (K)               *
C     TR           Cesium reservoir temperature (K)        *
C     HS           Ratio of neutralization current Jn to   *
C                  Current density J                       *
C     J            Current density (amps/cm2)              *
C     JCJ          Ratio of back emission to current den-  *
C                  sity                                    *
C                                                  *
C     Output values -                                *
C     TSC          LTE value for electron temperature at   *
C                  collector edge of plasma                *
```

```
C                                                              *
C**************************************************************
C
      INTEGER iter
      REAL*8 r1,dh,phinc,jnc,param1,param2,hss,dif,oldh,
     &       olddif,newh
      LOGICAL first,goon
C
C.....Calculate numerator.....
      r1=THREE*tee+TWO*tc*jcj
C
C.....Enter iteration
C
      first=.true.
      goon=.false.
      dh=ONE
      param1=TWO*jcj+THREE
      param2=ONE+jcj
      do iter=1,MAXITR
C.....Calculate collector edge electron temperature.....
        tsc=r1/(dlog((hs+HALF)/param2)+param1)
C
C.....Calculate neutralization work function and current density.....
        phinc=1.7d0+.383d0*tsc/tr
        jnc=AR*tsc**2*dexp(-phinc/(XK*tsc))
C
C.....Find answer for HS, difference between guess and answer.....
        hss=jnc/j
        dif=hs-hss
      if (dabs(dif).lt.TOL) go to 40
C
C.....Update HS to make difference small.....
        if (first) then
          oldh=hs
          olddif=dif
           hs=hs-dsign(dh,dif)
           hs=dmax1(hs,1.0d-12)
          first=.false.
           dh=dh*1.6d0
        else
          if (.not.goon) then
            if (dif*olddif.gt.0.d0) then
              newh=hs-dsign(dh,dif)
              newh=dmax1(newh,1.d-12)
```

```
            dh=1.6d0*dh
          else
            goon=.true.
            newh=(oldh*dif-hs*olddif)/(dif-olddif)
          endif
        else
          newh=(oldh*dif-hs*olddif)/(dif-olddif)
        endif
        oldh=hs
        olddif=dif
        hs=newh
      endif
    if (dabs(hs-oldh).lt.1.d-5*hs) go to 40
    enddo
 40  if (iter.gt.MAXITR) pause 'Exceeded maximum iterations in TSC'
    return
    END
C
C
    SUBROUTINE lte(tee,tec,tav,tc,tr,hs,j,jcj,dl,dlea,d)
C    IMPLICIT NONE
    INTEGER MAXITR
    REAL*8 tee,tec,tav,tc,tr,hs,j,jcj,dl,dlea,d,XK,
   &       ONE,TWO,THREE,TOL,AR,HALF
    PARAMETER (XK=8.6175d-5,ONE=1.d0,TWO=2.d0,THREE=3.d0,TOL=
   &          1.d-5,AR=120.d0,MAXITR=50,HALF=.5d0)
C
C*************************************************************
C                                                  *
C    The routine LTE is called by OBSTR and SATUR in order *
C    to check, and possibly recalculate, the electron      *
C    temperatures in order to keep the average electron    *
C    temperature above the LTE limit for the bulk plasma.  *
C    This is briefly discussed in Appendix B of Chapter 5  *
C    in Massey, McDaniel, and Bederson.                    *
C                                                  *
C    Input values -                                  *
C    TEE       Electron temperature at emitter edge (K)   *
C    TEC       Electron temperature at collectore edge (K)*
C    TAV       Average electron temperature (K)           *
C    TC        Collector temperature (K)                  *
C    TR        Cesium reservoir temperature (K)           *
C    HS        Ratio of neutalization current to current  *
C              density                                    *
```

```
C     J          Current density (amps/cm2)                    *
C     JCJ         Ratio of back emission to current density    *
C     DL          Ratio of gap to total electron mean free     *
C                 path                                         *
C     DLEA          Ratio of gap to electron-neutral mean      *
C                 free path                                    *
C     D           Interelectrode gap (mm)                      *
C                                                              *
C     Output values (recalculated) -                           *
C     TEE                                                      *
C     TEC                                                      *
C     TAV                                                      *
C     DL                                                       *
C                                                              *
C**************************************************************
C
      INTEGER iter1,iter2
      REAL*8 ts,dl1,tss,dif,oldt,olddif,newt,r1,dh,
     &       phinc,jnc,hss,oldh,newh,tsc,param1,param2
      LOGICAL first,goon
C
C.....First guess for TS.....
      ts=tav
      first=.true.
      dh=TWO
C
C.....Enter secant method search for TS.....
C
C.....Calculate new value for ratio of gap to mean free path.....
      do iter1=1,MAXITR
        dl1=dlea+3.4d+7*j*d/ts**2.5d0
C
C.....Calculate an answer for TS.....
        tss=1.7d0/(XK*dlog((AR*ts*ts)/(j*dl1))-.383d0/tr)
C
C.....Find difference between guess and answer.....
        dif=ts-tss
      if (dabs(dif).lt.TOL) go to 50
C
C.....Update TS to make difference small.....
        if (first) then
          oldt=ts
          olddif=dif
           ts=ts-dsign(50.d0,dif)
```

```
          first=.false.
        else
          newt=(oldt*dif-ts*olddif)/(dif-olddif)
          oldt=ts
          olddif=dif
          ts=newt
        endif
      if (dabs(ts-oldt).lt.1.d-5*ts) go to 50
      enddo
 50   if (iter1.gt.MAXITR) pause 'Max. iterations exceeded in LTE'
C
C.....Check to see if average electron temperature is above
C       the limit.  If so, return without altering any
C       values.....
      if (tav.ge.ts) return
C
C.....If bulk LTE is in effect, replace TAV and DL with their
C       proper LTE values.....
      tav=ts
      dl=dl1
C
C.....TEC must now be recalculated, since it depends on TEE,
C       which will change to keep the average temperature
C       above its limit.  An iteration like that in the function
C       TSC is used.....
      first=.true.
      goon=.false.
      r1=TWO*THREE*ts+TWO*tc*jcj
      dh=TWO
C
C.....Begin iteration.....
      param1=TWO*(jcj+THREE)
      param2=ONE+jcj
      do iter2=1,MAXITR
C.....Calculate collector edge electron temperature.....
        tsc=r1/(dlog((hs+HALF)/param2)+param1)
        phinc=1.7d0+.383d0*tsc/tr
        jnc=AR*tsc**2*dexp(-phinc/(XK*tsc))
        hss=jnc/j
        dif=hs-hss
      if (dabs(dif).lt.TOL) go to 60
        if (first) then
          oldh=hs
          olddif=dif
```

```
          hs=hs-dsign(dh,dif)
          hs=dmax1(hs,1.d-12)
         first=.false.
          dh=1.6d0*dh
        else
         if (.not.goon) then
           if (dif*olddif.gt.0.d0) then
             newh=hs-dsign(dh,dif)
             newh=dmax1(newh,1.d-12)
             dh=1.6d0*dh
           else
             goon=.true.
             newh=(oldh*dif-hs*olddif)/(dif-olddif)
           endif
         else
           newh=(oldh*dif-hs*olddif)/(dif-olddif)
         endif
         oldh=hs
         olddif=dif
         hs=newh
        endif
      if (dabs(hs-oldh).lt.1.d-5*hs) go to 60
      enddo
 60   if (iter2.gt.MAXITR) pause 'Max. iterations exceeded in LTE'
C.....Recompute TEC.....
      tec=tsc
C
C.....Recompute TEE.....
      tee=TWO*ts-tsc
      return
      END
C
C
      SUBROUTINE obstr2(vi,b,h,j,jcj,te,tc,tr,pd,d,ta,tee,tec,ve,vc,
     &    vd,vrad,jej)
C     IMPLICIT NONE
      INTEGER MAXITR
      REAL*8 vi,b,h,j,jcj,te,tc,tr,pd,d,ta,tee,tec,ve,vc,vd,vrad,
     &    jej,XK,HALF,TOL,AR,EMIS,ONE,TWO,THREE,TINY
      PARAMETER (XK=8.6175d-5,TWO=2.d0,HALF=.5d0,TOL=1.d-5,AR=120.d0,
     &           EMIS=.4d0,MAXITR=50,ONE=1.d0,THREE=3.d0,TINY=1.d-32)
C
C     Uses tsc2, ltec2
C
```

```
C*************************************************************
C                                                            *
C     OBSTR2 is called by TECMDL to  implement the           *
C     phenomenological equations for the obstructed          *
C     region of the ignited mode volt-ampere curve with      *
C     a negative (ion retaining) sheath at the collector.    *
C     The emitter side and collector side electron           *
C     temperatures are subject to LTE (Local Thermodynamic   *
C     Equilibrium) constraints which are implemented by      *
C     the functon TSC2 and the subroutine LTE2.  The sub-    *
C     routine is very similar to OBSTR except that the       *
C     equations for TEC and VC and the LTE routines are      *
C     different.                                             *
C                                                            *
C     Input values -                                         *
C     VI        Effective ionization energy (eV)             *
C     B         Ionizability factor                          *
C     H         Collector current factor                     *
C     J         Current density (amps/cm2)                   *
C     JCJ       Ratio of back emission to current density    *
C     TE        Emitter temperature (K)                      *
C     TC        Collector temperature (K)                    *
C     TR        Cesium reservoir temperature (K)             *
C     PD        Pressure-spacing product (torr-mm)           *
C     D         Interelectrode spacing (mm)                  *
C     TA        Average neutral and ion temperature (K)      *
C                                                            *
C     Output values -                                        *
C     TEE       Emitter side electron temperature (K)        *
C     TEC       Collector side electron temperature (K)      *
C     VE        Emitter sheath height (eV)                   *
C     VC        Collector sheath height (eV)                 *
C     VD        Arc drop (eV)                                *
C     VRAD      Plasma radiation component of arc drop (eV)* 
C     JEJ       Ratio JE/J of effective emitted current      *
C               density to working current density           *
C                                                            *
C*************************************************************
C
      INTEGER iter
      REAL*8 dlea,zetac,phinc,jnc,hs,tsc2,telect,dl,r,ans,dif,
     &   oldj,olddif,newj,param1,param2,param3,param4
      LOGICAL first,ltec
C
```

```
C.....Calculate ratio of spacing to electron-neutral
C       mean free path.....
        dlea=35.d0/((te+tc)/2000.d0)*pd
C
C.....Calculate emitter side electron temperature.....
        tee=vi/(TWO*XK*dlog(b*dlea))
C
C.....Calculate collector sheath attenuation factor.....
        zetac=(-(h+HALF)+dsqrt((h+HALF)**2+8.d0*jcj*h))/
       & (TWO*jcj)
C
C.....Calculate collector sheath height.....
        vc=XK*tc*dlog(zetac)
C
C.....Calculate collector emission factor.....
        r=jcj*dexp(vc/(XK*tc))
C
C.....Calculate collector side electron temperature.....
        tec=(THREE*tee+TWO*tc*r)/(TWO*r+THREE)
C
C.....Calculate neutralization potential and current density.....
        phinc=1.7d0+.383d0*tec/tr
        jnc=AR*tec*tec*dexp(-phinc/(XK*tec))
C
C.....Assign value to LTE limit for H and check
C       to see if LTE limits the electron temperature
C       at the collector edge - if so, calculate new
C       values for TEC, VC, and R.....
        hs=jnc/j
        if ((r+HALF).gt.hs) then
          tec=tsc2(tee,tc,tr,hs,j,jcj,vc)
           r=hs-HALF
          ltec=.true.
        else
          ltec=.false.
        endif
C
C.....Calculate average electron temperature.....
        telect=(tee+tec)/TWO
C
C.....Calculate ratio of spacing to total
C       electron mean free path, including
C       ion scattering.....
        dl=dlea+3.4d+7*j*d/(telect**2.5d0)
```

```
C
C.....If LTE has occured at the collector edge, the routine
C       LTE is called to check that the average electron
C       temperature is above the bulk LTE limit.  If not,
C       the LTE routine will calculate new emitter side,
C       collector side,  and average electron temperatures
C       The ratio of spacing to total mean free path is also
C       recalculated.....
      if (ltec) call lte2(tee,tec,telect,tc,tr,hs,j,jcj,vc,dl,dlea,
     &  d,r)
C
C.....Calculate radiation component of arc drop.....
      vrad=9.65d+5*pd/(j*ta)*dexp(-2.d0/(XK*telect))*
     &  (ONE+.069d0*dexp(.58d0/(XK*telect))
     &  *(EMIS/dsqrt(d/10.d0)+HALF))
C
C.....Guess JEJ and enter secant method iteration.....
      jej=TWO
      first=.true.
C
C.....First compute some parameters in order to save time in the
C         iteration.....
      param1=TWO*XK*(tec-te+(tec-tc)*r)+vrad
      param2=TWO*XK*(tee-te)
      param3=.75d0*dl+r
      param4=-ONE/(XK*tee)
C.....Start iteration.....
      do iter=1,MAXITR
C.....Calculate emitter sheath height.....
        ve=param2*(jej-ONE)+param1
C
C.....Calculate answer for JEJ and compute difference from
C       guess for JEJ.....
        if (ve*param4.le.dlog(TINY)) then
C.....Case where ve is so large that it would cause exp function
C           to underflow.....
          ans=ONE
        else
C.....Normal case.....
          ans=ONE+param3*dexp(ve*param4)
        endif
        dif=jej-ans
      if (dabs(dif).lt.TOL) go to 70
C
```

```
C.....Update value of JEJ until convergence.....
        if (first) then
          oldj=jej
          olddif=dif
           jej=jej-dsign(.2d0,dif)
          first=.false.
        else
          newj=(oldj*dif-jej*olddif)/(dif-olddif)
          oldj=jej
          olddif=dif
          jej=newj
         endif
      if (dabs(jej-oldj).lt.1.d-5*jej) go to 70
      enddo
 70   if (iter.gt.MAXITR) pause 'Exceeded maximum iterations in
     &OBSTR2'
C
C.....Calculate arc drop.....
      vd=ve-vc
      return
      END
C
C
      SUBROUTINE satur2(vi,b,bp,h,j,jcj,jsp,te,tc,tr,pd,d,ta,tee,tec,
     &     ve,vc,vd,vrad,jsj,jij)
C     IMPLICIT NONE
      INTEGER MAXITR
      REAL*8 vi,b,bp,h,j,jcj,jsp,te,tc,tr,pd,d,ta,tee,tec,ve,vc,vd,
     &     vrad,jsj,jij,XK,TWO,HALF,TOL1,TOL2,AR,EMIS,ONE,THREE,TINY
      PARAMETER (XK=8.6175d-5,TWO=2.d0,HALF=.5d0,TOL1=1.d-6,TOL2=1.d-5,
     &          MAXITR=100,AR=120.d0,EMIS=.4d0,ONE=1.d0,THREE=3.d0,
     &          TINY=1.d-32)
C
C     Uses tsc2, ltec2
C
C*************************************************************
C                                                  *
C     SATUR2 is called by TECMDL to implement the phenomen-*
C     ological model equations in the saturation region,   *
C     with a negative collector sheath.  The formulation   *
C     is very similar to SATUR except that the equations   *
C     for TEC and VC and the LTE routines are different.   *
C                                                  *
C     Input values -                                 *
```

```
C     VI      Effective ionization energy (eV)          *
C     B       Ionizability factor                     *
C     BP      Temperature increase parameter            *
C     H       Collector current factor                *
C     J       Current density (amps/cm2)              *
C     JCJ     Ratio of back emission to current density *
C     TE      Emitter temperature (K)                 *
C     TC      Collector temperature (K)               *
C     TR      Cesium reservoir temperature (K)          *
C     PD      Pressure-spacing product (torr-mm)        *
C     D       Interelectrode spacing (mm)             *
C     TA      Average neutral and ion temperature (K)   *
C                                                     *
C     Output values -                                 *
C     TEE     Emitter side electron temperature (K)     *
C     TEC     Collector side electron temperature (K)   *
C     VE      Emitter sheath height (eV)              *
C     VC      Collector sheath height (eV)            *
C     VD      Arc drop (eV)                           *
C     VRAD      Plasma radiation component of arc drop (eV)*
C     JSJ     Ratio JS/J of effective emitted current   *
C             density to working current density        *
C     JIJ     Ratio Ji/J of additional ion current to   *
C             the emitter to working current density    *
C                                                       *
C*************************************************************
C
      INTEGER iwhch,iter1,iter2
      REAL*8 dlea,zetac,phinc,jnc,hs,tsc2,telect,dl,r,ans,f,oldj,
     &       oldf,newj,param1,param2,param3,param4,js,g,
     &       x1,x2,x3,y1,y2,y3,ys
      LOGICAL first,ltec
C
C.....Guess ion current ratio.....
      jij=0.d0
C
C.....Set iteration counter.....
      iwhch=1
C
C.....Calculate ratio of gap to electron-neutral mean free path.....
      dlea=35.d0/((te+tc)/2000.d0)*pd
C
C.....Begin modified linear interpolation search for JIJ.....
C
```

```
      do iter1=1,MAXITR
C.....Calculate emitter side electron temperature.....
        tee=vi/(TWO*XK*dlog(b*dlea)-XK*dlog(ONE-bp*jij))
C
C.....Calculate collector sheath attenuation factor.....
        zetac=(-(h+HALF)+dsqrt((h+HALF)**2+8.d0*jcj*h))/
     &    (2.d0*jcj)
C
C.....Calculate collector sheath height.....
        vc=XK*tc*dlog(zetac)
C
C.....Calculate collector emission factor.....
        r=jcj*dexp(vc/(XK*tc))
C
C.....Calculate collector side electron temperature.....
        tec=(THREE*tee+TWO*tc*r)/(TWO*r+THREE)
C
C.....Calculate neutralization potential and current density.....
        phinc=1.7d0+.383d0*tec/tr
        jnc=AR*tec*tec*dexp(-phinc/(XK*tec))
C
C.....Assign value to LTE limit for H and check
C       to see if LTE limits the electron temperature
C       at the collector edge - if so, calculate new
C       values for TEC, VC, and R.....
        hs=jnc/j
         if ((r+HALF).gt.hs) then
          tec=tsc2(tee,tc,tr,hs,j,jcj,vc)
           r=hs-HALF
          ltec=.true.
        else
          ltec=.false.
        endif
C
C.....Calculate average electron temperature.....
        telect=(tee+tec)/TWO
C
C.....Calculate ratio of spacing to total
C       electron mean free path, including
C       ion scattering.....
        dl=dlea+3.4d+7*j*d/(telect**2.5d0)
C
C.....If LTE has occured at the collector edge, the routine
C       LTE is called to check that the average electron
```

```
C       temperature is above the bulk LTE limit.  If not,
C       the LTE routine will calculate new emitter side,
C       collector side,  and average electron temperatures
C       The ratio of spacing to total mean free path is also
C       recalculated.....
        if (ltec) call lte2(tee,tec,telect,tc,tr,hs,j,jcj,vc,dl,dlea,
     &    d,r)
C
C.....Calculate radiation component of arc drop.....
        vrad=9.65d+5*pd/(j*ta)*dexp(-2.d0/(XK*telect))*
     &    (ONE+.069d0*dexp(.58d0/(XK*telect))*
     &    (EMIS/dsqrt(d/10.d0)+HALF))
C
C.....Guess JSJ and enter secant method iteration.....
        jsj=TWO
        first=.true.
C.....First calculate some parameters in order to save time
C         in the iteration.....
        param1=TWO*XK*(tec-te+(tec-tc)*r)+vrad-jij*(
     &          3.89d0+TWO*XK*tee)
        param2=TWO*XK*(tee-te)
        param3=.75d0*dl+r
        param4=-ONE/(XK*tee)
C.....Begin iteration.....
        do iter2=1,MAXITR
C.....Calculate emitter sheath height.....
          ve=(param2*(jsj-ONE)+param1)/(ONE+jij)
C
C.....Calculate answer for JSJ and compute difference from
C       guess for JSJ.....
          if (ve*param4.le.dlog(TINY)) then
C.....Case where ve is so large that the exp function would
C             underflow.....
            ans=ONE+jij
          else
C.....Normal case.....
            ans=ONE+(param3-HALF*jij)*dexp(-ve/(XK*tee))+jij
          endif
          f=jsj-ans
        if (dabs(f).lt.TOL1) go to 80
C
C.....Update value of JSJ until convergence.....
          if (first) then
            oldj=jsj
```

```
            oldf=f
             jsj=jsj-dsign(.2d0,f)
            first=.false.
          else
            newj=(oldj*f-jsj*oldf)/(f-oldf)
            oldj=jsj
            oldf=f
            jsj=newj
          endif
        if (dabs(jsj-oldj).lt.1.d-5*jsj) go to 80
        enddo
 80     if (iter2.gt.MAXITR) pause 'Max. iterations exceeded in
    &SATUR2 for finding current ratio'
C
C.....Calculate value of JS from eqn. (35) of Massey,
C       McDaniel, and Bederson.....
        js=jsp*dexp(612.d0*dsqrt(dsqrt(-j*jij*dsqrt(ve)))/te)
C
C.....Compute error term.....
        g=js/j-jsj
      if (dabs(g).lt.TOL2) go to 90
C
C.....Update JIJ to make error small.....
        if (iwhch.eq.1) then
          if ((jij.eq.0.d0).and.(g.gt.0.d0)) then
            pause ' No solution in SATUR2'
            return
          endif
          x1=jij
          y1=g
          jij=-.1d0
          x2=jij
          iwhch=2
        else if (iwhch.eq.2) then
          x2=jij
          y2=g
          if (y1*y2.gt.0.d0) then
            x1=x2
            y1=y2
            jij=jij+dsign(.1d0,g)
C.....Prevent jij from becoming equal to -1.  Make it the
C         nearest larger number.....
            if (jij.le.-ONE) jij=-.999999d0
          else
```

```
            iwhch=3
            ys=y2
            jij=(x1*y2-x2*y1)/(y2-y1)
           endif
         else if (iwhch.eq.3) then
           x3=jij
           y3=g
           if (y3*y1.lt.0.d0) then
            x2=x3
            y2=y3
             if (y3*ys.gt.0.d0) y1=y1/TWO
           else
            x1=x3
            y1=y3
             if (y3*ys.gt.0.d0) y2=y2/TWO
           endif
           ys=y3
           jij=(x1*y2-x2*y1)/(y2-y1)
         endif
       enddo
 90    if (iter1.gt.MAXITR) then
c        write(*,'(a)') ' Maximum iterations exceeded in
c     &SATUR2 for finding ion current'
         write(6,'(a)') ' Maximum iterations exceeded in
     &SATUR2 for finding ion current'
c        write(*,*) j,tr
         write(6,*) j,tr
         stop
       endif
C
C.....Calculate arc drop.....
       vd=ve-vc
       return
       END
C
C
       REAL*8 FUNCTION tsc2(tee,tc,tr,hs,j,jcj,vc)
C      IMPLICIT NONE
       INTEGER MAXITR
       REAL*8 tee,tc,tr,hs,j,jcj,vc,XK,ONE,TWO,THREE,TOL,AR,HALF
       PARAMETER (XK=8.6175d-5,ONE=1.d0,TWO=2.d0,THREE=3.d0,TOL=
     &            1.d-5,AR=120.d0,MAXITR=50,HALF=.5d0)
C
C*************************************************************
```

```
C                                                              *
C     The function TSC2 is called by the subroutines OBSTR2*
C     and SATUR2 in order to compute the collector side    *
C     electron temperature when LTE conditions exist at the*
C     collector and the collector sheath is negative.  It  *
C     is very similar to the function TSC.  One difference *
C     is that the collector sheath is recalculated by TSC2.*
C                                                            *
C     Input values -                                          *
C     TEE          Emitter side electron temperature (K)   *
C     TC           Collector temperature (K)                 *
C     TR           Cesium reservoir temperature (K)         *
C     HS           Ratio of neutralization current Jn to   *
C                  Current density J                        *
C     J            Current density (amps/cm2)               *
C     JCJ          Ratio of back emission to current den-  *
C                  sity                                     *
c     vc           collector sheath height (ev)             *
C                                                            *
C     Output values -                                         *
C     TSC2          LTE value for electron temperature at   *
C                  collector edge of plasma (K)             *
C     VC           Recalculated value for collector         *
C                  sheath (eV)                              *
C                                                            *
C************************************************************
C
      INTEGER iter
      REAL*8 dh,phinc,jnc,hss,dif,oldh,olddif,newh
      LOGICAL first,goon
C
C.....Enter iteration (first guess for HS has already been
C       calculated by calling routine).....
C
C.....Calculate collector edge electron temperature.....
      first=.true.
      goon=.false.
      dh=ONE
      do iter=1,MAXITR
        tsc2=(THREE*tee+(TWO*hs-ONE)*tc)/(TWO*(hs+ONE))
C
C.....Calculate neutralization work function and current density.....
        phinc=1.7d0+.383d0*tsc2/tr
        jnc=AR*tsc2**2*dexp(-phinc/(XK*tsc2))
```

```
C
C.....Find answer for HS, difference between guess and answer.....
       hss=jnc/j
       dif=hs-hss
     if (dabs(dif).lt.TOL) go to 100
C
C.....Update HS to make difference small.....
       if (first) then
         oldh=hs
         olddif=dif
          hs=hs-dsign(dh,dif)
          hs=dmax1(hs,1.d-12)
         first=.false.
          dh=1.6d0*dh
       else
         if (.not.goon) then
           if (dif*olddif.gt.0.d0) then
             newh=hs-dsign(dh,dif)
             newh=dmax1(newh,1.d-12)
            dh=1.6d0*dh
          else
            goon=.true.
            newh=(oldh*dif-hs*olddif)/(dif-olddif)
           endif
         else
           newh=(oldh*dif-hs*olddif)/(dif-olddif)
          endif
         oldh=hs
         olddif=dif
         hs=newh
        endif
     if (dabs(hs-oldh).lt.1.d-5*hs) go to 100
     enddo
 100  if (iter.gt.MAXITR) pause 'Exceeded max. iterations in TSC2'
C
C.....Recalculate VC.....
     vc=XK*tc*dlog((hs-HALF)/jcj)
     return
     END
C
C
     SUBROUTINE lte2(tee,tec,tav,tc,tr,hs,j,jcj,vc,dl,dlea,d,r)
C     IMPLICIT NONE
     INTEGER MAXITR
```

```
      REAL*8 tee,tec,tav,tc,tr,hs,j,jcj,vc,dl,dlea,d,r,XK,
     &       ONE,TWO,THREE,TOL,AR,HALF
      PARAMETER (XK=8.6175d-5,ONE=1.d0,TWO=2.d0,THREE=3.d0,TOL=
     &          1.d-5,AR=120.d0,MAXITR=50,HALF=.5d0)
C
C*************************************************************
C                                                            *
C     The routine LTE2 is called by OBSTR2 and SATUR2 to     *
C     perform checking and, if necessary,recomputation of    *
C     the electron temperatures when the collector sheath    *
C     is negative.  It is very similar to the routine        *
C     LTE, however, the collector sheath is re-              *
C     calculated in LTE2.                                    *
C                                                            *
C     Input values -                                         *
C     TEE       Electron temperature at emitter edge (K)     *
C     TEC       Electron temperature at collectore edge (K)  *
C     TAV       Average electron temperature (K)             *
C     TC        Collector temperature (K)                    *
C     TR        Cesium reservoir temperature (K)             *
C     HS        Ratio of neutalization current to current    *
C               density                                      *
C     J         Current density (amps/cm2)                   *
C     JCJ       Ratio of back emission to current density    *
C     VC        Collector sheath height (eV)                 *
C     DL        Ratio of gap to total electron mean free     *
C               path                                         *
C     DLEA      Ratio of gap to electron-neutral mean        *
C               free path                                    *
C     D         Interelectrode gap (mm)                      *
C                                                            *
C     Output values (recalculated) -                         *
C     TEE                                                    *
C     TEC                                                    *
C     TAV                                                    *
C     DL                                                     *
C     VC                                                     *
C                                                            *
C*************************************************************
C
      INTEGER iter1,iter2
      REAL*8 ts,dl1,tss,dif,oldt,olddif,newt,dh,
     &       phinc,jnc,hss,oldh,newh,tsc
      LOGICAL first,goon
```

```
C
C.....First guess for TS.....
      ts=tav
C
C.....Enter secant method search for TS.....
C
      first=.true.
      dh=TWO
      do iter1=1,MAXITR
C.....Calculate new value for ratio of gap to mean free path.....
        dl1=dlea+3.4d+7*j*d/ts**2.5d0
C
C.....Calculate an answer for TS.....
        tss=1.7d0/(XK*dlog((120.d0*ts*ts)/(j*dl1))-.383d0/tr)
C
C.....Find difference between guess and answer.....
        dif=ts-tss
      if (dabs(dif).lt.TOL) go to 110
C
C.....Update TS to make difference small.....
        if (first) then
          oldt=ts
          olddif=dif
           ts=ts-dsign(50.d0,dif)
          first=.false.
        else
          newt=(oldt*dif-ts*olddif)/(dif-olddif)
          oldt=ts
          olddif=dif
          ts=newt
         endif
      if (dabs(ts-oldt).lt.1.d-5*ts) go to 110
      enddo
 110  if (iter1.gt.MAXITR) pause 'Max. iterations exceeded in LTE'
C
C.....Check to see if average electron temperature is above
C       the limit.  If so, return without altering any
C       values.....
      if (tav.ge.ts) return
C
C.....If bulk LTE is in effect, replace TAV and DL with their
C       proper LTE values.....
      tav=ts
      dl=dl1
```

```
C
C.....TEC must now be recalculated, since it depends on TEE,
C       which will change to keep the average temperature
C       above its limit.  An iteration like that in the function
C       TSC2 is used.....
      first=.true.
      goon=.false.
      dh=TWO
C
C.....Begin iteration.....
      do iter2=1,MAXITR
C.....Calculate collector edge electron temperature.....
         tsc=(TWO*THREE*ts+(TWO*hs-ONE)*tc)/(TWO*(hs+ONE)+
     &       THREE)
        phinc=1.7d0+.383d0*tsc/tr
        jnc=AR*tsc**2*dexp(-phinc/(XK*tsc))
        hss=jnc/j
       dif=hs-hss
      if (dabs(dif).lt.TOL) go to 120
       if (first) then
        oldh=hs
        olddif=dif
         hs=hs-dsign(dh,dif)
         hs=dmax1(hs,1.d-12)
         first=.false.
         dh=1.6d0*dh
       else
         if (.not.goon) then
           if (dif*olddif.gt.0.d0) then
             newh=hs-dsign(hs,dif)
             newh=dmax1(newh,1.d-12)
             dh=1.6d0*dh
           else
             newh=(oldh*dif-hs*olddif)/(dif-olddif)
             goon=.true.
           endif
         else
          newh=(oldh*dif-hs*olddif)/(dif-olddif)
         endif
        oldh=hs
        olddif=dif
        hs=newh
       endif
     if (dabs(hs-oldh).lt.1.d-5*hs) go to 120
```

```
      enddo
 120  if (iter2.gt.MAXITR) pause 'Max. iterations exceeded in LTE2'
C
C.....Recompute TEC.....
      tec=tsc
C
C.....Recompute VC.....
      vc=XK*tc*dlog((hs-HALF)/jcj)
C
C.....Recompute TEE.....
      tee=TWO*ts-tsc
      r=hs-HALF
      return
      END
C
      SUBROUTINE unig(te,tc,tr,d,phie,phic,j,ji,v,qe,sheath)
C     IMPLICIT NONE
      REAL*8 te,tc,tr,d,phie,phic,ji,v,qe,
     + j,PI,XKE,TFACT,XNFACT,TOL,XK,ME,MI,DEFAULT,AR
      INTEGER sheath,TRY,ITMAX
      PARAMETER (PI=3.141592654,XKE=8.61753d-5,TFACT=1.05d0,
     + XNFACT=0.8d0,TOL=1.d-5,XK=1.3807d-16,ME=9.1095d-28,
     + MI=2.207d-22,DEFAULT=-99.d0,TRY=4,AR=120.d0,ITMAX=30)
C
C     UNIG
C     John McVey 30 March 1990
C     DOCUMENT CONTROL #C-568-006-D-033090
C
C     Rev. C: Modifications for use in CYLCON6
C       Calculation of cesiated work functions removed.
C       Changed to double precision.
C
C     Rev. D:  Eliminated problem with divide by zero in update routine.
C
C     Unig is a subroutine package for calculating the output voltage of
C     a thermionic converter operating in the diffusion-dominated
C     unignited mode.
C
C     INPUTS:
C          te          Emitter temperature in K
C          tc          Collector temperature in K
C          tr          Cesium reservoir temperature in K.
C          d           Interelectrode gap in centimeters
C          phie        Emitter work function in eV.
```

```
C          phic          Collector work function in eV.
C          j             Net electron current density in Amps/sq. cm.
C     OUTPUTS:
C          ji            Ion current density in Amps/sq. cm.
C          v             Output voltage in volts.
C          qe            Electron cooling in Watts/sq. cm.
C          sheath        Integer indicating sheath configuration.
C                        0 = no solution
C                        1 = DU
C                        2 = DD
C                        3 = UU
C                        4 = UD
C
C     uses du,dd,uu,ud,dn1,dn2,denav,tcalc,update,coefs
C
      INTEGER izone,itert,iwhich
      REAL*8 taav,na0,na1,naav,veli,nav,r,vele,alpha,
     +  dife,difi,lambde,lambdi,e,i,zetae,zetcpr,vevc,zetac,
     +  zetepr,xiepr,nenc,psi,telans,navans,f,g,x1,x2,x3,
     +  y1,y2,y3,f1,f2,f3,g1,g2,g3,xnew,ynew,update,tel,
     &  ve,vc,vp,js,jie,jc,pcs,dlam,teff,arate,ionprob,
     &  mue,mui,emob,imob,jion,heat
      js=AR*te*te*dexp(-phie/(XKE*te))
      jc=AR*tc*tc*dexp(-phic/(XKE*tc))
      pcs=2.45d+8*dexp(-8910.d0/tr)/dsqrt(tr)
      taav=(te+tc)/2.d0
      na0=1333.2d0*pcs/(XK*te)
      na1=1333.2d0*pcs/(XK*tc)
      naav=(na0+na1)/2.d0
      veli=dsqrt(8.d0*XK*taav/(PI*MI))
      dlam=d*1.2d-14*naav
      if (dlam.lt.1.d0) then
        teff=taav
      else if ((dlam.ge.1.d0).and.(dlam.le.10.d0)) then
        teff=taav+(dlam-1.d0)*(te/tc)/18.d0
      else
        teff=te
      end if
      arate=1333.2d0*pcs/dsqrt(2.d0*PI*MI*XK*teff)
      ionprob=1.d0/(1.d0+2.d0*dexp((3.89d0-phie)/(XKE*te)))
      jie=ionprob*arate*1.6022d-19
      tel=1.1d0*te
      if (j.lt.-jc) then
        v=-DEFAULT
```

```
      sheath=0
       return
    endif
    if (j.gt.js) then
       v=DEFAULT
      sheath=0
      return
    endif
    nav=1.d+11
    izone=0
    iwhich=1
    do itert=1,ITMAX
       r=tel/taav
       vele=dsqrt(8.d0*XK*tel/(PI*ME))
       alpha=vele/veli
      mue=emob(tel,naav,nav)
      mui=imob(taav,naav,nav)
       dife=mue*tel*XKE
       difi=mui*taav*XKE
       lambde=3.d0*dife/vele
       lambdi=3.d0*difi/veli
       e=.75d0*r/(r+1.d0)*d/lambde
       i=.75d0/(r+1.d0)*d/lambdi
      call du(j,js,jc,jie,tel,te,i,e,alpha,zetae,zetcpr,ve,
     +          vc,vevc)
       if ((ve.gt.0.d0).and.(vc.gt.0.d0)) then
        sheath=1
        goto 12
      endif
      call dd(j,js,jc,jie,te,tc,i,e,alpha,zetae,zetac,ve,vc,vevc)
       if ((ve.gt.0.d0).and.(vc.gt.0.d0)) then
        sheath=2
        goto 12
      endif
      call uu(j,js,jc,jie,tel,te,i,e,alpha,zetepr,zetcpr,
     +          xiepr,ve,vc,vevc)
       if ((ve.gt.0.d0).and.(vc.gt.0.d0)) then
        sheath=3
        goto 12
      endif
      call ud(j,js,jc,jie,tel,te,tc,i,e,alpha,zetepr,zetac,xiepr,
     +          ve,vc,vevc)
       if ((ve.gt.0.d0).and.(vc.gt.0.d0)) then
        sheath=4
```

```
        goto 12
      endif
      sheath=0
      izone=izone+1
       if (izone.lt.TRY) then
         tel=TFACT*tel
        nav=XNFACT*nav
        iwhich=1
        goto 100
      else
         v=-dsign(DEFAULT,j)
        return
      endif
12     if ((sheath.eq.2).or.(sheath.eq.4)) then
         call dn1(j,jc,i,e,zetac,lambde,d,nenc,psi)
      else
         call dn2(j,jc,i,e,zetcpr,lambde,d,nenc,psi)
      endif
       vp=XKE*tel*((psi-1.d0)*dlog(nenc))
13     call tcalc(te,tc,j,js,jc,ve,vc,vp,zetae,zetac,telans,sheath)
      call denav(j,jc,zetcpr,zetac,nenc,vele,navans,sheath)
      f=tel-telans
      g=nav-navans
      if (iwhich.eq.1) then
        x1=nav
        y1=tel
        f1=f
        g1=g
        tel=tel-dsign(20.d0,f)
        iwhich=2
      elseif (iwhich.eq.2) then
        x2=nav
         y2=tel
        f2=f
        g2=g
        nav=nav-dsign(1.d+9,g)
        iwhich=3
      elseif (iwhich.eq.3) then
         if ((dabs(f/tel).lt.TOL).and.(dabs(g/nav).lt.TOL))
     +      goto 200
        x3=nav
        y3=tel
        f3=f
        g3=g
```

```
          xnew=update(x1,x2,x3,f1,f2,f3,g1,g2,g3)
           if (xnew.lt.0.d0) xnew=x3
          ynew=update(y1,y2,y3,f1,f2,f3,g1,g2,g3)
           if (dabs(ynew-y3).gt.y3/2.d0) ynew=y3*(1.d0+dsign(.5d0,ynew-
     +          y3))
          x1=x2
          y1=y2
          f1=f2
          g1=g2
          x2=x3
          y2=y3
          f2=f3
          g2=g3
          nav=xnew
          tel=ynew
         endif
  100   enddo
C
  200   if (itert.gt.ITMAX) then
         v=-dsign(DEFAULT,j)
         return
        end if
        ji=jion(j,jc,zetcpr,zetac,alpha,sheath)
        v=phie-phic+vevc-vp
        qe=heat(j,js,ji,phie,te,tel,ve,zetae,sheath)
C
        return
        END
C
C
        REAL*8 FUNCTION update(x1,x2,x3,f1,f2,f3,g1,g2,g3)
C       IMPLICIT NONE
        REAL*8 x1,x2,x3,f1,f2,f3,g1,g2,g3
C
C       Updates parameters for the two-dimensional secant method
C       iteration in UNIG used to find the average electron temperature
C       and plasma density.
C
        REAL*8 r,u
        r=x1*(f2*g3-f3*g2)+x2*(f3*g1-f1*g3)+x3*(f1*g2-f2*g1)
        u=f2*g3-f3*g2+f3*g1-f1*g3+f1*g2-f2*g1
        if (u.eq.0.d0) pause ' U is zero, chuckie!'
        update=r/u
        return
```

```
      END
C
C
      SUBROUTINE du(j,js,jc,jie,tel,te,i,e,alpha,zetae,zetcpr,ve,
     +    vc,vevc)
C     IMPLICIT NONE
      REAL*8 j,js,jc,jie,tel,te,i,e,alpha,zetae,zetcpr,ve,vc,vevc,
     + XKE,ZERO,ONE,TWO,DEFAULT
      PARAMETER (XKE=8.61753d-5,ZERO=0.d0,ONE=1.d0,TWO=2.d0,
     + DEFAULT=-99.d0)
C
C     Solves for the emitter and collector sheath heights in the
C     condition where an ion retaining sheath is at the emitter
C     and an electron retaining one is at the collector.
C
      REAL*8 a,b,c,disc
      LOGICAL bad1

      bad1 = .false.
      a=TWO*i*js
      b=TWO*js+(e-i)*j
      c=-(j*(ONE+TWO*e)+alpha*jie*(ONE+TWO*i))
      disc=b*b-4.d0*a*c
      if (disc.lt.ZERO) bad1=.true.
      disc=dmax1(disc,ZERO)
      zetae=(-b+dsqrt(disc))/(TWO*a)
      if ((zetae.le.ZERO).or.(bad1)) then
        ve=DEFAULT
      else
        ve=-XKE*te*dlog(zetae)
      endif
      zetcpr=(j+jc)*(TWO-zetae)/(j-js*zetae*zetae+alpha*jie)
      if (zetcpr.gt.ZERO) then
        vc=-XKE*tel*dlog(zetcpr)
      else
        vc=DEFAULT
      endif
      vevc=ve+vc
      return
      END
C
C
      SUBROUTINE dd(j,js,jc,jie,te,tc,i,e,alpha,zetae,zetac,ve,
     +    vc,vevc)
```

```
C     IMPLICIT NONE
      INTEGER ITMAX
      REAL*8 j,js,jc,jie,te,tc,i,e,alpha,zetae,zetac,ve,vc,vevc,
     +  XKE,ZERO,ONE,TWO,DEFAULT,FOUR,TOL
      PARAMETER (XKE=8.61753d-5,ZERO=0.d0,ONE=1.d0,TWO=2.d0,
     +  DEFAULT=-99.d0,ITMAX=40,FOUR=4.d0,TOL=1.d-5)
C
C     Solves for the emitter and collector sheath heights in the
C     condition where there are ion retaining sheaths at both
C     electrodes.
C
      INTEGER iter
      REAL*8 a,b1,c1,b,c,disc1,q,disc2,f,x1,x2,f1,f2
      LOGICAL bad1,bad2
      a=(TWO*i-ONE)*js
      b1=TWO*js+(ONE+e-i)*j
      c1=-TWO*((ONE+e)*j+i*alpha*jie)
      zetac=ZERO
      do iter=1,ITMAX
        bad1=.false.
        bad2=.false.
        b=b1+jc*zetac
         c=c1-TWO*jc*zetac
         disc1=b*b-FOUR*a*c
         if (disc1.lt.ZERO) bad1=.true.
         disc1=dmax1(disc1,ZERO)
         zetae=(-b+dsqrt(disc1))/(TWO*a)
         q=(TWO*alpha*jie-zetae*(TWO*js*zetae-j))/(TWO-zetae)
         disc2=(j+q)*(j+q)+16.d0*jc*q
         if (disc2.lt.ZERO) bad2=.true.
         disc2=dmax1(disc2,ZERO)
         f=zetac-(-(j+q)+dsqrt(disc2))/(FOUR*jc)
        if (iter.eq.1) then
          x1=zetac
          f1=f
           zetac=zetac-dsign(.1d0,f)
        else
          x2=zetac
          f2=f
          zetac=x2+f2*(x2-x1)/(f1-f2)
          x1=x2
          f1=f2
        endif
        if (dabs((zetac-x2)/zetac).lt.TOL) goto 100
```

```
      enddo
  100 if (iter.gt.itmax) bad2=.true.
      if ((zetae.le.ZERO).or.(bad1)) then
        ve=DEFAULT
      else
        ve=-XKE*te*dlog(zetae)
      endif
      if ((zetac.le.ZERO).or.(bad2)) then
        vc=DEFAULT
      else
        vc=-XKE*tc*dlog(zetac)
      endif
      vevc=ve-vc
      return
      END
C
C
      SUBROUTINE uu(j,js,jc,jie,tel,te,i,e,alpha,zetepr,zetcpr,
     +      xiepr,ve,vc,vevc)
C     IMPLICIT NONE
      INTEGER ITMAX
      REAL*8 j,js,jc,jie,tel,te,i,e,alpha,zetepr,zetcpr,xiepr,ve,
     + vc,vevc,XKE,ZERO,ONE,TWO,DEFAULT,FOUR,TOL
      PARAMETER (XKE=8.61753d-5,ZERO=0.d0,ONE=1.d0,TWO=2.d0,
     + DEFAULT=-99.d0,ITMAX=40,FOUR=4.d0,TOL=1.d-5)
C
C     Solves for the emitter and collector sheath heights in the
C     condition where there are electron retaining sheaths at both
C     electrodes.
C
      INTEGER iter
      REAL*8 a,b,c,disc,tau,zzx,fmin,f,dfdz,delta
      LOGICAL bad1
      a=(TWO*i+ONE)*alpha*jie
      b=-j*(ONE-e+i)
      c=TWO*(i+ONE)*(j-js)
      disc=b*b-FOUR*a*c
      disc=dmax1(disc,ZERO)
      zetepr=(-b+dsqrt(disc))/(TWO*a)
      zetepr=dmax1(zetepr,ZERO)
      tau=tel/te
      bad1=.false.
      if (b.le.ZERO) then
        zzx=(-b/(a*(tau+ONE)))**(ONE/tau)
```

```
        if (zzx.ge.ZERO) then
          fmin=a*zzx**(tau+ONE)+b*zzx+c
          if (fmin.gt.ZERO) then
           bad1=.true.
           zetepr=zzx
           go to 110
          endif
        endif
      endif
      do iter=1,ITMAX
        xiepr=dsign(dabs(zetepr)**tau,zetepr)
        f=a*zetepr*xiepr+b*zetepr+c
        dfdz=a*(tau+ONE)*xiepr+b
        if (dfdz.ne.ZERO) delta=-f/dfdz
        zetepr=zetepr+delta
        if (dabs(delta/zetepr).lt.TOL) goto 100
      enddo
  100 if (iter.gt.ITMAX) bad1=.true.
  110 xiepr=dsign(dabs(zetepr)**tau,zetepr)
      if ((zetepr.le.ZERO).or.(bad1)) then
        ve=DEFAULT
      else
        ve=-XKE*tel*dlog(zetepr)
      endif
      zetcpr=(j+jc)/(alpha*jie*xiepr-(js-j)/zetepr)
      if (zetcpr.gt.ZERO) then
        vc=-XKE*tel*dlog(zetcpr)
      else
        vc=DEFAULT
      endif
      vevc=vc-ve
      return
      END
C
C
      SUBROUTINE ud(j,js,jc,jie,tel,te,tc,i,e,alpha,zetepr,zetac,
     +    xiepr,ve,vc,vevc)
C     IMPLICIT NONE
      INTEGER ITMAX
      REAL*8 j,js,jc,jie,tel,te,tc,i,e,alpha,zetepr,zetac,xiepr,
     + ve,vc,vevc,XKE,ZERO,ONE,TWO,DEFAULT,TOL
      PARAMETER (XKE=8.61753d-5,ZERO=0.d0,ONE=1.d0,TWO=2.d0,
     + DEFAULT=-99.d0,ITMAX=50,TOL=1.d-5)
C
```

```
C     Solves for the emitter and collector sheath heights in the
C     condition where an electron retaining sheath is at the emitter
C     and an ion retaining one is at the collector.
C
      INTEGER iter,ii
      REAL*8 f(2),x(2),delta(2),pderiv(2,2),tau,a,b1,c,zetacg,
     +  zetepg,q,determ
      LOGICAL bad
      tau=tel/te
      a=TWO*i*alpha*jie
      b1=(e-i)*j
      c=(j-js)*(TWO*i+ONE)
      iter=1
      zetacg=ZERO
      zetepg=ONE
      x(1)=zetepg
      x(2)=zetacg
      do iter=1,ITMAX
        bad=.false.
        xiepr=dsign(dabs(x(1))**tau,x(1))
        f(1)=x(1)*(a*xiepr+b1+jc*x(2))+c
        q=TWO*alpha*jie*xiepr-TWO*(js-j)/x(1)-j
        f(2)=x(2)*(TWO*jc*x(2)+j+q)-TWO*q
        if ((dabs(f(1)).lt.TOL).and.(dabs(f(2)).lt.TOL)) goto 100
        pderiv(1,1)=(tau+ONE)*a*xiepr+b1+jc*x(2)
        pderiv(2,1)=(x(2)-TWO)*(TWO*alpha*tau*jie*(xiepr/x(1))
     +    +TWO*(js-j)/(x(1)*x(1)))
        pderiv(1,2)=jc*x(1)
        pderiv(2,2)=4.d0*jc*x(2)+j+q
        determ=pderiv(1,1)*pderiv(2,2)-pderiv(1,2)*pderiv(2,1)
        if (determ.eq.ZERO) then
          bad=.true.
          delta(1)=ZERO
          delta(2)=ZERO
        else
          bad=.false.
          delta(1)=(pderiv(1,2)*f(2)-pderiv(2,2)*f(1))/determ
          delta(2)=(pderiv(2,1)*f(1)-pderiv(1,1)*f(2))/determ
        endif
        if ((dabs(delta(1)).lt.TOL).and.(dabs(delta(2)).lt.
     +    TOL)) goto 100
        do ii=1,2
          x(ii)=x(ii)+delta(ii)
        enddo
```

```
      enddo
  100 if (iter.gt.itmax) bad=.true.
      zetepr=x(1)
      zetac=x(2)
      xiepr=dsign(dabs(zetepr)**tau,zetepr)
      if ((zetepr.le.ZERO).or.(bad)) then
        ve=DEFAULT
      else
        ve=-XKE*tel*dlog(zetepr)
      endif
      if ((zetac.le.ZERO).or.(bad)) then
        vc=DEFAULT
      else
        vc=-XKE*tc*dlog(zetac)
      endif
      vevc=-ve-vc
      return
      END
C
C
      SUBROUTINE dn1(j,jc,i,e,zetac,lambde,d,nenc,psi)
C     IMPLICIT NONE
      REAL*8 j,jc,i,e,zetac,lambde,d,nenc,psi,ONE,TWO
      PARAMETER(ONE=1.d0,TWO=2.d0)
C
C       Evaluates parameters NENC and PSI for computation of the
C       plasma drop VP in UNIG.  Used for cases in which there is an ion
C       retaining collector sheath.
C
      REAL*8 jczc
      jczc=TWO*jc*zetac+j
      nenc=ONE+TWO*e*j/jczc+TWO*zetac*i/(TWO-zetac)
      psi=j*(.75d0*d/lambde)/(e*j+jczc*i*zetac/(TWO-zetac))
      return
      END
C
C
      SUBROUTINE dn2(j,jc,i,e,zetcpr,lambde,d,nenc,psi)
C     IMPLICIT NONE
      REAL*8 j,jc,i,e,zetcpr,lambde,d,nenc,psi,ONE,TWO
      PARAMETER (ONE=1.d0,TWO=2.d0)
C
C       Evaluates parameters NENC and PSI for computation of the
C       plasma drop VP in UNIG.  Used for cases in which there is an
```

```
C     electron retaining collector sheath.
C
      REAL*8 jcj
      jcj=TWO*jc+(TWO-zetcpr)*j
      nenc=ONE+TWO*zetcpr*e*j/jcj+TWO*i
      psi=j*(.75d0*d/lambde)/(e*j+jcj*i/zetcpr)
      return
      END
C
C
      SUBROUTINE denav(j,jc,zetcpr,zetac,nenc,vele,nav,sheath)
C     IMPLICIT NONE
      INTEGER sheath
      REAL*8 j,jc,zetcpr,zetac,nenc,vele,nav,ONE,TWO,EC
      PARAMETER (ONE=1.d0,TWO=2.d0,EC=1.602d-19)
C
C     Calculates the average plasma density in the interelectrode
C     space.  This is used in UNIG to calculate the amount of
C     electron-ion scattering.
C
      if (sheath.eq.0) then
        nav=(j+TWO*jc)*(nenc+ONE)/(EC*vele)
      elseif ((sheath.eq.1).or.(sheath.eq.3)) then
        nav=((TWO-zetcpr)*j+TWO*jc)*(nenc+ONE)/(zetcpr*EC
     +      *vele)
      elseif ((sheath.eq.2).or.(sheath.eq.4)) then
        nav=(j+TWO*jc*zetac)*(nenc+ONE)/(EC*vele)
      endif
      return
      END
C
C
      SUBROUTINE tcalc(te,tc,j,js,jc,ve,vc,vp,zetae,zetac,tel,sheath)
C     IMPLICIT NONE
      INTEGER sheath
      REAL*8 te,tc,j,js,jc,ve,vc,vp,zetae,zetac,tel,TWOK
      PARAMETER (TWOK=5802.5d0)
C
C     Uses energy balance to calculate an average electron temperature
C     in the interelectrode space.
C
      REAL*8 y
      if (sheath.eq.1) then
       y=js*zetae+jc
```

```
      tel=(js*zetae*te+TWOK*j*(vp-vc)+jc*tc)/y
     elseif (sheath.eq.2) then
      y=js*zetae+jc*zetac
      tel=(js*zetae*te+TWOK*j*vp+jc*zetac*tc)/y
     elseif (sheath.eq.3) then
      y=js+jc
      tel=(js*te+TWOK*j*(ve+vp-vc)+jc*tc)/y
     elseif (sheath.eq.4) then
      y=js+jc*zetac
      tel=(js*te+TWOK*j*(ve+vp)+jc*zetac*tc)/y
     endif
     return
     END
C
C
     REAL*8 FUNCTION jion(j,jc,zetcpr,zetac,alpha,sheath)
C     IMPLICIT NONE
     REAL*8 j,jc,zetcpr,zetac,alpha,TWO
     PARAMETER(TWO=2.d0)
     INTEGER sheath
C
     if ((sheath.eq.1).or.(sheath.eq.3)) then
      jion=((TWO-zetcpr)*j+TWO*jc)/(alpha*zetcpr)
     else if ((sheath.eq.2).or.(sheath.eq.4)) then
      jion=zetac*(j+TWO*jc*zetac)/(alpha*(TWO-zetac))
     end if
     return
     END
C
C
     REAL*8 FUNCTION heat(j,js,ji,phie,te,tel,ve,zetae,sheath)
C     IMPLICIT NONE
     REAL*8 j,js,ji,phie,te,tel,ve,zetae,TK,VI
     PARAMETER(TK=2.d0/11604.5d0,VI=3.89d0)
     INTEGER sheath
C
     if ((sheath.eq.1).or.(sheath.eq.3)) then
       heat=j*(phie+ve+tel*TK)+js*(te-tel)*TK+ji*(VI-phie)
     else if ((sheath.eq.2).or.(sheath.eq.4)) then
       heat=j*(phie+tel*TK)+js*zetae*(te-tel)*TK+ji*(ve+VI-phie)
     end if
     return
     END
C
```

```
C
      REAL*8 FUNCTION emob(tel,na,n)
C     IMPLICIT NONE
      REAL*8 tel,na,n
C
      REAL*8 csecea,lnl,nuea,nuei,re,taue,muea,muei,mue
C       Evaluate electron-neutral cross-section (cm2)
      csecea=1.d-16*(535.d0+tel*(-.27d0+tel*5.2d-5))
C
C       Evaluate the Coulomb logarithm, electron-neutral collision
C       frequency, electron-ion collision frequency, and the ratio.
      if (n.gt.0.d0) then
        lnl=dlog(12390.d0*tel**1.5/dsqrt(n))
      else
        lnl=dlog(12390.d0*tel**1.5/1.d-16)
      end if
      nuea=7.319d+5*na*csecea*dsqrt(tel)
      nuei=dmax1(1.070d0*n*lnl/tel**1.5,1.d-16)
      re=nuei/nuea
C
C       Calculate the electron mobility (cm2/volt-sec).
      taue=(1.d0+re*(14.1d0+re*(30.6d0+re*16.3d0)))/(1.d0+re*
     +     (21.1d0+re*(37.4d0+re*16.3d0)))
      muea=5.167d+17/nuea
      muei=3.058d+17/nuei
      mue=muea*muei/(muea+muei)*taue
      emob=mue/299.8d0
C
      return
      END
C
C
      REAL*8 FUNCTION imob(ti,na,n)
C     IMPLICIT NONE
      REAL*8 ti,na,n
C
      REAL*8 csecia,lnl,nuia,nuii,ri,taui,muia,mui
C       Evaluate ion-neutral cross-section (cm2)
      csecia=1.d-16*(1667.d0+ti*(-.807d0+ti*(4.77d-4-1.047d-7*ti)))
C
C       Evaluate the Coulomb logarithm, ion-neutral collision frequency,
C       ion-ion collision frequency, and the ratio.
      if (n.gt.0.d0) then
        lnl=log(12390.d0*ti**1.5/dsqrt(n))
```

```
      else
        lnl=log(12390.d0*ti**1.5/1.d-16)
      end if
      nuia=1051.d0*na*csecia*dsqrt(ti)
      nuii=dmax1(1.537d-3*n*lnl/ti**1.5,1.d-16)
      ri=nuii/nuia
C
C     Calculate the ion mobility (cm2/volt-sec).
      taui=(1.d0+ri*(4.2d0+ri*2.86d0))/(1.d0+ri*(4.24d0+ri*2.91d0))
      muia=1.959d+12/nuia
      mui=muia*taui
      imob=mui/299.8d0
C
      return
      END

      REAL*8 FUNCTION ndsphi(te,tr,phi0)
C     IMPLICIT NONE
      INTEGER MAXITR
      REAL*8 te,tr,phi0,SMALL,ERRTOL
      PARAMETER(SMALL=1.d-5,ERRTOL=1.d-6,MAXITR=100)
C
C     Written by John McVey and Jean-Louis Desplat
C     Control #C-568-007-D-061290
C     This version uses a value of 1.95 eV for the cesium ion
C     adsorption energy rather than 2.04 eV (see functions f1 and f2).
C
C     uses f1,f2
C
C     ************************************************************
C
C          The function Nedsphi calculates the cesiated emitter
C          work function based on the emitter temperature,
C          cesium reservoir temperature (cesium pressure), and an
C          effective bare work function of the emitter surface.
C          The equations are based on the article "Correlation of
C          Emission Processes for Adsorbed Alkali Films on Metal
C          Surfaces" by N.S. Rasor and C. Warner, Journal of
C          Applied Physics, Vol. 35, #9, 1964.  This theory is
C          inaccurate for high bare work functions and low values of
C          T/TR (Phi0 above 5.5 and T/Tr below 2.5 simultaneously, for
C          example).  The theory does take into account the slight
C          non-uniqueness in T/Tr.
C
```

```
C         Inputs:
C           Te   Emitter temperature in K.
C           Tr   Cesium reservoir temperature in K.
C           Phi0 Effective emitter bare work function in eV.
C
C         Outputs:
C                Returns cesiated emitter work function in eV.
C     Version D is double precision.
C     ***************************************************************
C
      INTEGER itcnt,i
      REAL*8 x(2),f(2),p(2,2),cor(2),cov,dphi,dy,dx,xdx1,s1,
     *  ydy1,s2,determ,err1,err2,f1,f2
C     Initial guesses
      cov=dmax1((phi0-te/tr)/(phi0-1.d0),1.d-6)
      if (cov.le.0.5d0) then
        dphi=2.2d0*(phi0-1.5d0)*cov
      else
        dphi=1.1d0*(phi0-1.5d0)
      endif
      dphi=dmax1(dphi,1.d-6)
      do itcnt=1,MAXITR
        dy=dmax1(SMALL*dphi,1.d-6)
        dx=dmax1(SMALL*cov,1.d-6)
       x(1)=cov
       x(2)=dphi
C      Compute values of two functions which will be zero at solution.
       f(1)=f1(cov,dphi,te,tr,phi0)
        f(2)=f2(cov,dphi,te,phi0)
        if (cov.lt.0.2d0) then
         xdx1=cov-dx
          s1=1.d0
       else
         xdx1=cov+dx
          s1=-1.d0
       endif
        if (dphi.lt.0.2d0) then
         ydy1=dphi-dy
          s2=1.d0
       else
         ydy1=dphi+dy
          s2=-1.d0
       endif
C      Compute partial derivatives of both functions.
```

```
        p(1,1)=(f(1)-f1(xdx1,dphi,te,tr,phi0))/(dsign(dx,s1))
        p(1,2)=(f(1)-f1(cov,ydy1,te,tr,phi0))/(dsign(dy,s2))
        p(2,1)=(f(2)-f2(xdx1,dphi,te,phi0))/(dsign(dx,s1))
        p(2,2)=(f(2)-f2(cov,ydy1,te,phi0))/(dsign(dy,s2))
C       Perform Newton-Raphson.
       determ=p(1,1)*p(2,2)-p(2,1)*p(1,2)
        if (dabs(determ).le.1.0d-20) then
         pause 'No convergence in ndsphi'
          ndsphi=-1.0d-12
         return
       else
         cor(1)=(f(2)*p(1,2)-f(1)*p(2,2))/determ
         cor(2)=(f(1)*p(2,1)-f(2)*p(1,1))/determ
       endif
       do i=1,2
         x(i)=x(i)+cor(i)
       enddo
        err1=dabs(cor(1)/x(1))
        err2=dabs(cor(2)/x(2))
        cov=dmin1(x(1),.99)
        cov=dmax1(cov,0.)
        dphi=dmax1(x(2),0.)
        if (((err1.lt.ERRTOL).and.(err2.lt.ERRTOL)).or.((dabs(f(1))
     &    .lt.ERRTOL).and.(dabs(f(2)).lt.ERRTOL))) go to 10
      enddo
C     Return value of cesiated work function.
 10   if (itcnt.gt.MAXITR) then
        pause 'No convergence in ndsphi'
        ndsphi=-1.0d-12
      else
        ndsphi=phi0-dphi
      endif
      return
      END
C     ***************************************************************
      REAL*8 FUNCTION f1(x,y,te,tcs,phi0)
C     IMPLICIT NONE
      REAL*8 x,y,te,tcs,phi0,PI,K,ONE,TWO,HALF,TPMK,VI,PHII0
      PARAMETER (PI=3.141592654,K=1.d0/11604.5d0,ONE=1.d0,TWO=2.d0,
     & TPMK=TWO*PI*2.207d-22*1.381d-16,HALF=0.5d0,VI=3.89d0,PHII0=
     & 1.95d0)
C
C     ***********************************************
C
```

```
C       F1 is called by ndsphi.
C       The value at solution will be near zero.
C
C     ***********************************************
C
      REAL*8 phia0,e0,pcs,g,factr1,factr2,mucs,sigfcs
      phia0=.777d0*dsqrt(phi0)
      e0=phi0-phia0-VI+PHII0
      pcs=2.45d+8*dexp(-8910.d0/tcs)/dsqrt(tcs)
      mucs=1333.d0*pcs/dsqrt(TPMK*te)
      g=.18d0+.2d0*x
      factr1=ONE+HALF*dexp((e0-g*y)/(K*te))
      factr2=x/(ONE-x)*dexp(x/(ONE-x))
      sigfcs=HALF*dexp(62.d0+4.8d0*x*(ONE-HALF*x))
      f1=sigfcs*factr2*dexp(-phia0/(K*te))/(factr1*mucs)-ONE
      return
      END
C     ************************************************************
      REAL*8 FUNCTION f2(x,y,te,phi0)
C     IMPLICIT NONE
      REAL*8 x,y,te,phi0,EC,SIGCS,RCS,ALPHCS,K,PI,ONE,TWO,A,B,VI,PHII0
      PARAMETER (EC=4.8032d-10,SIGCS=3.56d+14,RCS=1.4d-8,ALPHCS=1.5d-23,
     &    K=1.d0/11604.5d0,PI=3.141592654,ONE=1.d0,TWO=2.d0,
     &
A=6.25d+11*4.*PI*EC*EC*SIGCS*RCS,B=TWO*PI*ALPHCS*SIGCS/RCS,
     &    VI=3.89d0,PHII0=1.95d0)
C
C     ***********************************************
C
C       F2 is called by ndsphi.
C       The value at solution will be near zero.
C
C     ***********************************************
C
      REAL*8 g,e0,phia0
      g=.18d0+.2d0*x
      phia0=.777d0*dsqrt(phi0)
      e0=phi0-phia0-VI+PHII0
      f2=y*(ONE+B*g*x+TWO*dexp((-e0+g*y)/(K*te)))-A*x
      return
      END
```